Akademia Górniczo-Hutnicza
im. Stanisława Staszica
w Krakowie

Praca magisterska

Klasyfikacja danych wielowymiarowych algorytmami SVM

Marcin Orchel

Kierunek: Informatyka
Specjalność: Systemy komputerowe

Nr albumu: 105846

Promotor
dr hab. Witold Dzwinel, Prof. n. AGH

Wydział Elektrotechniki, Automatyki, Informatyki i Elektrotechniki

Kraków 2005

Składam serdeczne podziękowania Panu profesorowi Witoldowi Dzwinelowi i Panu magistrowi Marcinowi Kurdzielowi za pomoc przy wykonywaniu niniejszej pracy.

Spis treści

Streszczenie

W pracy tej zostały omówione algorytmy optymalizacji maszyn wektorów wspierających. Aby była możliwa optymalizacja dużych zbiorów danych, stosuje się technikę dekompozycji, polegającą na rozpatrywaniu mniejszych podproblemów optymalizacyjnych SVM. W pracy tej została zaproponowana nowa heurystyka dekompozycji problemu SVM polegająca na poszukiwaniu kierunku najszybszego wzrostu funkcji celu przy możliwie najdokładniejszym spełnieniu warunków optymalizacyjnych.

Jednym z najbardziej efektywnych algorytmów rozwiązywania problemu optymalizacyjnego SVM jest algorytm Minimalnej Optymalizacji Sekwencyjnej. Istotą tego algorytmu jest dekompozycja problemu SVM na podproblemy dwuparametrowe, które rozwiązywane są analitycznie. Dotychczas nie zostało przedstawione rozwiązanie analityczne podproblemów SVM więcej niż dwuparametrowych [18]. W pracy tej został zaproponowany algorytm rozwiązujący analitycznie podproblemy SVM również dla więcej niż dwóch parametrów. Jasność algorytmu stanowi o tym, że jest on ciekawą alternatywą dla skomplikowanych metod numerycznych. Zarówno nowa heurystyka jak i ogólne rozwiązanie analityczne podproblemów SVM zostały zaimplementowane i porównane z kluczową implementacją SVM - BSVM [9].

Słowa kluczowe

maszyny wektorów wspierających , algorytm minimalnej optymalizacji sekwencyjnej

Abstract

In this thesis there were presented optimization algorithms of support vector machines. In order to optimise the huge amount of data, it is wise to use a decomposition technic, which relies on considering smaller optimisation subproblems. In this work, it is proposed a new SVM decomposition heuristic, which relies on searching direction of the fastest growth of the target function with the most accurate optimization constraints fulfilling.

One of the most effective algorithm of computing SVM problem is Sequential Minimal Optimisation (SMO). The key of this algorithm is a decomposition of the SVM problem into two-parameters subproblems, which have analytical solutions. Up to now, it was not presented any analytical solution of subproblems with more than two parameters [18]. In this thesis, it is proposed a new algorithm which compute analytically SVM subproblems also for more than two parameters. Simplicity of analytical solution accounts for that, it is an interesting method compared with existing, complex numerical algorithms. Both new heuristic and analytical solution of the SVM subproblems were tested and compared with crucial SVM implementation - BSVM [9].

Key words

support vector machines, SVM, sequential minimal optimization, SMO

Wstęp

Istnieje wiele problemów, których nie można opisać za pomocą modeli matematycznych. Przykładami są rozpoznawanie pisma ręcznego, rozpoznawanie mowy, rozpoznawanie twarzy, detekcja spamu. Istotną cechą systemów komputerowych, które mogłbyby rozwiązywać tego typu problemy jest możliwość uczenia się. Uczenie się systemów komputerowych polega na analizie danych treningowych i znajdowaniu na jej podstawie przewidywanego modelu systemu. Powszechnie stosowanymi metodami budowania modelu są metody oparte na drzewach decyzyjnych, sieciach neuronowych oraz na logice rozmytej.

Okazuje się, że maszyny wektorów wspierających (ang. *Support vector Machines*) w wielu przypadkach lepiej przewidują zachowanie systemu niż metody konkurencyjne [4] [8]. Metoda SVM z powodzeniem była zastosowana do rozwiązywania wielu rzeczywistych problemów, takich jak rozpoznawanie napisanych ręcznie cyfr [6][20][21][3], rozpoznawanie obiektów [1], identyfikacja głosu [22], rozpoznawanie twarzy na obrazach [16], klasyfikacja tekstu [12].

Metoda SVM posiada wiele zalet. Radzi sobie z nadmiernym dopasowaniem danych, z nieliniowymi granicami decyzyjnymi, pozwala na uniezależnienie się od wymiaru danych oraz na reprezentację zagadnienia z mniejszą liczbą wektorów wejściowych, ponadto posiada silne podstawy matematyczne.

Pomimo tych zalet metoda SVM nie jest zbyt popularna. Przyczyną takiego stanu rzeczy jest konieczność rozwiązywania problemu optymalizacyjnego z zakresu programowania kwadratowego, co dla dużej ilości danych wielowymiarowych, sięgających nawet kilkuset tysięcy wektorów wejściowych powoduje problemy z szybkością działania [18]. Aby była możliwa efektywna optymalizacja dużych zbiorów danych stosuje się technikę dekompozycji polegającą na rozpatrywaniu mniejszych podproblemów optymalizacyjnych. W tym wypadku wybór parametrów podczas dekompozycji, a tym samym ilość iteracji potrzebnych do znalezienia modelu stanowi istotny czynnik wpływający na szybkość działania metody SVM.

Kolejnym uniedogodnieniem metody SVM jest konieczność stosowania w ogólnym wypadku procedur numerycznych rozwiązujących podproblemy optymalizacyjne z warunkami ograniczającymi funkcję celu. Istnieje algorytm, zwany Minimalną Optymalizacją Sekwencyjną (SMO), który dekomponuje problem SVM na podproblemy, które są rozwiązywane analitycznie, lecz podproblemy są tylko dwuparametrowe.

Cele pracy Celem pracy jest udoskonalenie heurystyki, odpowiadającej za wybór parametrów podczas dekompozycji, co przyczyni się do wzrostu szybkości działania metody SVM.

Drugim celem jest stworzenie metody, która rozwiązuje analitycznie podproblemy SVM dla więcej niż dwóch parametrów, co pozwoli na ominięcie w ogólnym przypadku konieczności stosowania metod numerycznych rozwiązujących zagadnienie optymalizacji SVM.

Trzecim celem jest implementacja metody SVM wraz z wymienionymi wyżej udoskonaleniami.

Implementacja powinna spełniać następujące założenia:

- porównywalny czas działania z wybranymi implementacjami SVM
- generowanie wyników z żądaną dokładnością dla szerokiej gamy danych wejściowych, również danych rzeczywistych
- implementacja udoskonalonej heurystyki i metody analitycznej rozwiązywania podproblemów wieloparametrowych
- język programowania: C

W rozdziale 1 została przedstawiona teoria generalizacji na której opiera się metoda SVM, a także różne rodzaje klasyfikatorów SVM i ich podstawowe własności.

W rozdziale 2 została zaprezentowana idea dekompozycji, oraz algorytm Minimalnej Optymalizacji Sekwencyjnej wraz z heurystykami dla dwóch parametrów.

W rozdziale 3 został umieszczony opis metody analitycznej rozwiązywania podproblemów SVM, nazwanej Analityczną Optymalizacją Sekwencyjną, oraz opis istniejącej heurystyki dekompozycji wieloparametrowej wraz z autorskimi udoskonaleniami, jak również testy porównawcze nowej heurystyki z poprzednimi.

W rozdziale 4 został opisany program stworzony w ramach pracy magisterskiej, nazwany ASVM rozwiązujący problem SVM, natomiast w rozdziale 5 program ASVM został porównany z kluczowymi implementacjami SVM.

Rozdział 1

Metoda wektorów wspierających

1.1. Klasyfikatory

Maszynowe uczenie się polega na analizie doświadczenia i wyznaczaniu na tej podstawie przewidywanego modelu systemu. Model systemu reprezentowany jest przez *funkcję celu*, przekształcającą dane wejściowe w dane wyjściowe.

Podstawowym podziałem algorytmów maszynowego uczenia się jest podział na klasę algorytmów zwaną uczeniem z nadzorowaniem, i drugą klasę zwaną uczeniem bez nadzoru.

Uczenie z nadzorowaniem charakteryzuje się tym, że algorytm poszukujący funkcji celu korzysta z danych treningowych szczególnej postaci – par składających się z danych wejściowych i wyjściowych. Natomiast w uczeniu bez nadzoru algorytm korzysta z danych treningowych składających się jedynie z danych wejściowych. Jego zadaniem jest znalezienie modelu systemu dopasowującego się w optymalny sposób do danych wejściowych.

Uczenie z nadzorowaniem prowadzi do modelu systemu za pomocą którego można przewidywać wartości wyjściowe dla danych testowych. Wartości wyjściowe mogą być liczbami rzeczywistymi, wtedy uczenie z nadzorowaniem nazywane jest *regresją*, bądź mogą być symbolami pochodzącymi ze skończonego zbioru klas, wtedy uczenie z nadzorowaniem nazywane jest *klasyfikacją*. W przypadku klasyfikacji funkcja celu nazywana jest *funkcją decyzyjną*. Szczególnym przypadkiem klasyfikacji jest przypadek, gdy wartości wyjściowe pochodzą jedynie z dwuelementowego zbioru klas. Taki klasyfikator nazywany jest *klasyfikatorem binarnym* w odróżnieniu od *klasyfikatora wieloklasowego*.

Istotnym zagadnieniem dotyczącym maszyn uczących się jest sposób mierzenia skuteczności systemu uczącego się. W przypadku uczenia z nadzorowaniem, bo o nim będzie mowa w dalszej części pracy dobrym kryterium jest poprawność klasyfikacji danych testowych, czyli tzw. *generalizacja*. Poniżej została przedstawiona istota teorii statystycznej uczenia się maszyn, na której opierają się algorytmy SVM [23].

Zakłada się, że dane treningowe mają ten sam rozkład prawdopodobieństwa i są wzajemnie niezależne (ang. *independent and identically distributed*, i.i.d.)[3], a dane testowe mają ten sam rozkład prawodpodobieństwa co dane treningowe.

Dany jest zbiór możliwych funkcji celu f_i postaci: $x \rightarrow f_i(x) = y$, gdzie x jest wektorem wejściowym, a y jest klasą w przypadku klasyfikacji lub liczbą rzeczywistą w przypadku regresji.

Funkcja straty L, która reprezentuje błąd testowania, będzie w tym wypadku połową różnicy między prawidłową wartością wyjściową, a wynikiem zwróconym przez funkcję f_i:

$$L(x) = \frac{1}{2} |y - f_i(x)| \tag{1.1}$$

Wartość oczekiwana błędu testowania dla funkcji f_i, zwana *ryzykiem* wynosi:

$$R(f_i) = \int L(x)\, dP(x,y) = \int \frac{1}{2} |y - f_i|\, dP(x,y), \tag{1.2}$$

gdzie $P(x,y)$ jest rozkładem prawdpodobieństwa danych.

Dla danych treningowych, czyli dla skończonej liczby przykładów x_i, gdzie $i \in \{1..l\}$, znane jest zachowanie funkcji f, a więc można zdefiniować *ryzyko doświadczalne* dla tych przykładów:

$$R_{emp}(f_i) = \frac{1}{l} \sum_{j=1}^{l} \frac{1}{2} |y_j - f_i(x_j)| = \frac{1}{2l} \sum_{j=1}^{l} |y_j - f_i(x_j)| \tag{1.3}$$

Zadaniem maszyny uczącej się może być znalezienie funkcji f minimalizującej ryzyko R.

Teoria rozwinięta przez Vapnika i Chervonenkisa, zwana *teorią VC* [23] daje odpowiedź na pytanie, które funkcje f minimalizują ryzyko R, dla przypadku klasyfikacji binarnej. W tym celu zostało wyprowadzone ograniczenie górne ryzyka R występującego z prawdopodobieństwem $1 - \eta$, w postaci:

$$R(f_i) \leq R_{emp}(f_i) + \sqrt{\frac{h\left(\log\left(\frac{2l}{h}\right) + 1\right) - \log\left(\frac{\eta}{4}\right)}{l}}, \tag{1.4}$$

gdzie $\eta \in [0,1]$, h jest właściwością zbioru funkcji f_i, tzw. wymiarem Vapnika Chervonenkisa, VC [20].

A zatem mając dane jedynie ryzyko doświadczalne i wymiar VC zbioru funkcji f_i można oszacować błąd maszyny uczącej się dla przyszłych danych.

Różne zbiory funkcji celu, a zatem różne klasyfikatory posiadają odmienne górne granice ryzyka. *Minimalizacja strukturalna ryzyka* (ang. *Structural Risk Minimization*) jest metodą, która porównuje różne klasyfikatory za pomocą wyznaczonej górnej granicy błędu klasyfikacji.

Algorytmy SVM minimalizują górną granicę błędu postaci 1.4.

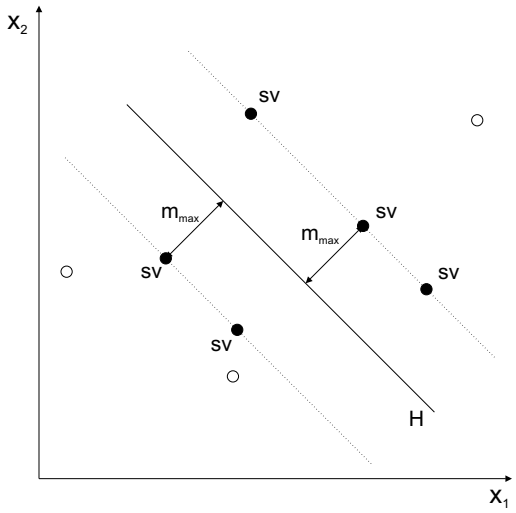
W następnym punkcie zostanie przedstawiony sposób działania klasyfikatorów SVM.

1.2. Klasyfikacja za pomocą wektorów wspierających

Maszyny z wektorami wspierającymi (ang. *support vector machines*) zostały po raz pierwszy zaproponowane przez Vapnika w 1963 [25]. Były to *klasyfikatory liniowe*, czyli takie dla których argumentem funkcji decyzyjnej jest liniowa kombinacja współrzędnych wektora wejściowego. W 1992 został zaprezentowany klasyfikator nieliniowy SVM [2]. W 1995 roku został przedstawiony klasyfikator SVM, który bierze pod uwagę możliwość występowania zaszumionych danych [6].

W roku 1997 została przedstawiona metoda regresji z wektorami wspierającymi. (ang. *Support Vector Regression*, SVR) [24]. W pracy zostanie pominięty problem regresji, uwaga zostanie skoncetrowana na problemie klasyfikacji.

Najprostszym klasyfikatorem SVM jest klasyfikator *maksymalnego marginesu* (ang. *maximal margin classifier*). Klasyfikator ten znajduje hiperpłaszczyznę rozdzielającą dane treningowe na dwie klasy w ten sposób, że maksymalizuje wartość marginesu geometrycznego dla wszystkich punktów treningowych. Marginesem geometrycznym hiperpłaszczyzny H jest jej odległość do najbliższych punktów. Punkty położone najbliżej hiperpłaszczyzny nazywane są *wektorami wspierającymi* (ang. *support vectors*) (Rys. 1.1).



Rysunek 1.1. Rysunek przedstawia wektory wspierające wśród punktów trenowanych za pomocą klasyfikatora maksymalnego marginesu dla przypadku dwuwymiarowego, oznaczenie $m_{\max}$ - maksymalny margines geometryczny.

Po dokładniejszym omówieniu tego klasyfikatora zostanie przedstawiony klasyfikator słabego marginesu (ang. *soft margin classifier*), taki, który bierze pod uwagę możliwość występowania zaszumionych danych. Zostanie również pokazana metoda, za pomocą której jest możliwe tworzenie klasyfikatorów nieliniowych.

1.2.1. Klasyfikator maksymalnego marginesu

Klasyfikator maksymalnego marginesu został zaproponowany przez Vapnika w 1963. Klasyfikator ten znajduje hiperpłaszczyznę rozdzielającą dane treningowe na dwie klasy w ten sposób, że maksymalizuje wartość marginesu geometrycznego dla wszystkich punktów treningowych. Jest to klasyfikator liniowy, i działa na danych które są li-

niowo separowalne. Rozpatrywany klasyfikator jest ponadto klasyfikatorem binarnym. Dana jest n-wymiarowa hiperpłaszczyzna H określona wzorem:

$$(H)\ g(\vec{x}) = 0,$$

gdzie $g(\vec{x}) = w^t\vec{x} + b$, w – wektor wagowy, b – wyraz wolny, odchylenie.

Hiperpłaszczyzna H jest *granicą decyzyjną* przyporządkowującą punkty obu klasom w następujący sposób. Gdy $g(\vec{x_1}) > 0$ dla pewnego punktu x_1 to punkt ten należy do dodatniej strony hiperpłaszczyzny H i jest przyporządkowany do pierwszej klasy, gdy zaś $g(\vec{x_1}) < 0$ to punkt x_1 należy do ujemnej strony hiperpłaszczyzny H i jest przyporządkowany do drugiej klasy. Wartość funkcji decyzyjnej $g(\vec{x})$ dla wektorów wspierających jest nazywana *marginesem funkcyjnym.*

Po pomnożeniu obu stron równania hiperpłaszczyzny H przez dowolną liczbę dodatnią otrzymuje się nie zmienioną hiperpłaszczyznę w tym sensie, że przyporządkowania punktów do klas nie zostaną zmienione, jak również położenie punktów względem siebie pozostanie bez zmian, w szczególności zbiór wektorów wspierających pozostanie nienaruszony. Po takiej operacji może zmienić się jedynie margines funkcyjny, a zatem można przyjąć, że $g(x) = 1$ lub $g(x) = -1$ dla punktów wspierających w zależności od położenia odpowiednio po dodatniej lub ujemnej stronie hiperpłaszczyzny H.

Odległość punktu x do danej hiperpłaszczyzny H wyraża się wzorem [10]:

$$d = \frac{|g(x)|}{\|w\|}. \tag{1.5}$$

Margines geometryczny jest odległością punktów wspierających do hiperpłaszczyzny. Zawsze gdy jest mowa o *marginesie*, to w domyśle chodzi o margines geometryczny. Jako, że zostało przyjęte, iż $g(x) = 1$ lub$g(x) = -1$dla punktów wspierających, to

$$\gamma = \frac{|g(x)|}{\|w\|} = \frac{1}{\|w\|}. \tag{1.6}$$

Klasyfikator maksymalnego marginesu znajduje hiperpłaszczyznę taką, która maksymalizuje wartość marginesu geometrycznego, a zatem taką dla której wartość $\|w\|$ jest minimalna. Problem można zatem zapisać następująco:

Minimalizacja $\langle w \cdot w \rangle$

przy warunkach:

$$y_i(\langle w \cdot x_i \rangle + b) \geq 1,$$

gdzie $i \in \{1..l\}$,

przy założeniu, że wektory treningowe są liniowo separowalne.

$y_i \in \{1, -1\}$ stanowi informację o przynależności i-tego wektora treningowego do klasy pierwszej lub drugiej.

Dla powyższego problemu można skonstruować lagranżjan postaci:

$$L(w, b, \vec{\alpha}) = \frac{1}{2}\langle w \cdot w \rangle - \sum_{i=1}^{l} \alpha_i (y_i(\langle w \cdot x_i \rangle + b) - 1),$$

gdzie $\vec{\alpha}$ to wektor mnożników Lagrange'a, o rozmiarze l.

Następnie korzystając z teorii Wolfa można wyprowadzić postać dualną problemu wyjściowego tylko ze zmiennymi α_i otrzymując:

Problem optymalizacyjny 1.2.1. *Maksymalizacja funkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle x_i \cdot x_j \rangle$$

przy następujących ograniczeniach:
$\sum_{i=1}^{l} y_i \alpha_i = 0$ *oraz*
$\alpha_i \geq 0$ *dla* $i \in \{1..l\}$.

Wektor wagowy problemu pierwotnego otrzymuje się ze wzoru:

$$\vec{w} = \sum_{i=1}^{l} y_i \alpha_i \vec{x_i}. \tag{1.7}$$

Problem dualny jest łatwiejszy do obliczenia, ponieważ występuje w nim funkcja, którą łatwiej optymalizować oraz prostszy warunek nierównościowy.

1.2.2. Klasyfikator nieliniowy

W 1992 Boser i Guyon oraz Vapnik zaprezentowali modyfikację klasyfikatora maksymalnego marginesu w celu utworzenia klasyfikatora nieliniowego. Podstawą modyfikacji jest wprowadzenie do problemu optymalizacji triku jądra. *Trik jądra* polega na zastąpieniu iloczynu skalarnego dwóch wektorów, tzw. jądrem. *Jądro* to funkcja postaci:

$K(x, y) = \langle x, y \rangle$. Aby funkcja była jądrem musi spełniać *warunek Mercera*:

Niech X będzie skończoną przestrzenią wejściową, gdzie $K(x, z)$ jest symetryczną funkcją na X. $K(x, z)$ jest funkcją jądra, wtedy i tylko wtedy, gdy macierz

$\mathrm{K} = (\mathrm{K}(\mathrm{x_i}, \mathrm{x_j}))_{\mathrm{i,j=1}}^{\mathrm{l}}$jest dodatnio określona, czyli gdy nie ma ujemnych wartości własnych.

A zatem problem optymalizacyjny (PO 1.2.1) może zostać zapisany w postaci:

Problem optymalizacyjny 1.2.2. *Maksymalizacja funkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j)$$

przy następujących ograniczeniach:
$\sum_{i=1}^{l} y_i \alpha_i = 0$ *oraz*
$\alpha_i \geq 0$ *dla* $i \in \{1..l\}$.

Zaletą takiego rozwiązania jest transformacja nieliniowa danych wejściowych za pomocą jądra do nowej przestrzeni wielowymiarowej. Granica decyzyjna w nowej przestrzeni cech jest liniowa, ale może być nieliniowa w wejściowej przestrzeni cech. Kolejną zaletą stosowania jądra jest uniezależnienie problemu optymalizacyjnego od wymiaru danych. Poniżej zostały przedstawione najbardziej popularne jądra:

Jądro liniowe

$$K(x, y) = xy + a \tag{1.8}$$

Jądro wielomianowe

$$K(x, y) = (xy + a)^d \tag{1.9}$$

Jądro potencjalnych funkcji bazowych (RBF)

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{1.10}$$

Jądro sigmoidalne

$$K(x, y) = \tanh(\kappa xy - \delta) \tag{1.11}$$

1.2.3. Klasyfikator słabego marginesu

W 1995 roku Corinna Cortes i Vapnik zaproponowali *klasyfikator słabego marginesu*, modyfikację klasyfikatora SVM, taką która klasyfikuje punkty podobnie jak klasyfikator maksymalnego marginesu, ale który bierze pod uwagę również możliwość występowania zaszumionych danych. Wiele rzeczywistych problemów posiada zaszumione dane, tzn. takie, w których występują pewne czynniki, który nie powinny być brane pod uwagę przy wyznaczaniu funkcji decyzyjnej. Tworzenie modelu z niechcianymi czynnikami może powodować pogorszenie jakości systemu i możliwą komplikację granicy decyzyjnej. Taki proces nazywany jest nadmiarowym dopasowaniem danych (ang. *overfitting data*)[17]. Aby zapobiec takiej sytuacji powstał klasyfikator, który bierze pod uwagę możliwe wahania wartości danych. Punkty wspierające w tym wypadku to punkty nie tylko najbliższe hiperpłaszczyźnie, ale również dalsze (Rys. 1.2). Definicja problemu dla klasyfikatora słabego marginesu:

Minimalizacja $\langle w \cdot w \rangle + C \sum_{i=1}^{l} \xi_i$

przy warunkach:

$$y_i \left(\langle w \cdot x_i \rangle + b \right) \geq 1 - \xi_i,$$

$$\xi_i \geq 0$$

gdzie $i \in \{1..l\}$,

ξ_i - są to tzw. *zmienne słabości* (ang. *slack variables*), takie, że $1 - \xi_i$ określa odległość punktu od hiperpłaszczyzny.

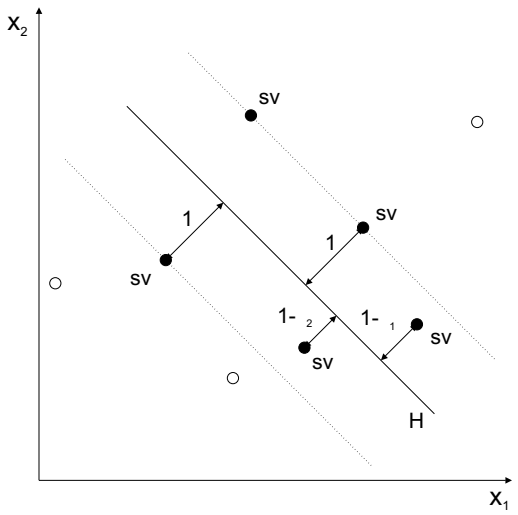
Problem dualny w stosunku do tego problemu wygląda następująco:

Problem optymalizacyjny 1.2.3. *Maksymalizacja f unkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j),$$

przy następujących ograniczeniach:

(O1) $\sum_{i=1}^{l} y_i \alpha_i = 0$ *oraz*



Rysunek 1.2. Rysunek przedstawia wektory wspierające wśród punktów trenowanych za pomocą klasyfikatora słabego marginesu dla przypadku dwuwymiarowego.

(O2) $0 \leq \alpha_i \leq C$ *dla* $i \in \{1..l\}$,
gdzie $C < \infty$.

Powyższy problem różni się od problemu maksymalnego marginesu tym, że parametry α dodatkowo ograniczone są z góry.

Przedstawiony w tej pracy algorytm optymalizacji SVM przystosowany jest do rozwiązywania powyższego problemu, jako bardziej odpowiadającemu rzeczywistości niż problem maksymalnego marginesu ze względu na występowanie zaszumienia danych. Wybierając jednak dostatecznie duży parametr C otrzymuje się problem maksymalnego marginesu. W dalszym ciągu jeśli będzie mowa o problemie SVM to w kontekście problemu (PO 1.2.3).

1.2.4. Warunek komplementarności Karush-Kuhn-Tuckera

Warunek komplementarności Karush-Kuhn-Tuckera dla klasyfikatora maksymalnego marginesu jest następujący:

Warunkiem koniecznym na to aby $\vec{\alpha}$ był rozwiązaniem (PO 1.2.1) jest spełnianie:

$$\alpha_i \left[y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 \right] = 0 \tag{1.12}$$

dla $i \in \{1..l\}$

Z powyższego warunku wynika, że

$$\alpha_i = 0 \Rightarrow y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 \geq 0$$

$$\alpha_i > 0 \Rightarrow y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 = 0$$

Gdy $\alpha_i > 0$, to i-ty wektor jest wektorem wspierającym. Jeśli $\alpha_i = 0$ to i-ty wektor może być wektorem wspierającym, lecz nie musi. Jako, że parametr $\alpha_i = 0$ nie ma wpływu na maksymalizowaną funkcję i można go pominąć, niektórzy autorzy [3] przez wektory wspierające rozumieją wektory, takie, że $\alpha_i > 0$. Wektory dla których $\alpha_i > 0$

są wystarczające do budowy modelu, tym samym model jest reprezentowany przez wektory danych odpowiadające tym parametrom.

Dla klasyfikatora słabego marginesu warunek ten ma postać:

$$\alpha_i \left[y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 + \xi_i \right] = 0, \tag{1.13}$$

$$\xi_i \left(\alpha_i - C \right) = 0 \tag{1.14}$$

dla $i \in \{1..l\}$

Podobnie jak dla klasyfikatora maksymalnego marginesu, gdy $\alpha_i > 0$, to i-ty wektor jest wektorem wspierającym. Jeśli $\alpha_i = 0$ to i-ty wektor może być wektorem wspierającym. Ponadto z (1.14) wynika, że niezerowe zmienne słabości mogą wystąpić tylko gdy $\alpha_i = C$.

Powyższy warunek komplementarności można zapisać w następujący sposób:

$$\begin{aligned} &\alpha_i = 0 \Rightarrow y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 \geq 0 \\ &\alpha_i = C \Rightarrow y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 \leq 0 \\ &0 < \alpha_i < C \Rightarrow y_i \left(\langle \vec{w} \vec{x_i} \rangle + b \right) - 1 = 0 \end{aligned}$$

Wprowadzając oznaczenie na funkcję decyzyjną:

$$f\left(\alpha_i\right) = \langle \vec{w} \vec{x_i} \rangle + b \tag{1.15}$$

przybiera on postać:

$$\begin{aligned} &\alpha_i = 0 \Rightarrow y_i f\left(x_i\right) - 1 \geq 0 \\ &\alpha_i = C \Rightarrow y_i f\left(x_i\right) - 1 \leq 0 \\ &0 < \alpha_i < C \Rightarrow y_i f\left(x_i\right) - 1 = 0 \end{aligned} \tag{1.16}$$

Z powyższego warunku można wyznaczyć wartość b biorąc pod uwagę dowolny parametr niegraniczny. Jednakże ze względu na dokładność numeryczną rozsądniej jest brać pod uwagę wszystkie parametry. Sposób wyznaczania wartości b został podany w artykule [13].

Rozdział 2

Techniki implementacyjne

Klasyfikacja za pomocą SVM sprowadza się do maksymalizacji kwadratowej funkcji wklęsłej z dodatkowymi warunkami. Taki problem nie posiada lokalnych ekstremów, co ułatwia znalezienie globalnego rozwiązania. Warunki nierównościowe w problemie SVM powodują jednak, że rozwiązanie analityczne nie jest możliwe do bezpośredniego wyznaczenia dla więcej niż dwóch parametrów. Dlatego stosuje się różne metody numeryczne. Podstawową koncepcją tych metod jest eliminacja warunku nierównościowego, a następnie rozwiązywanie pokrewnego problemu optymalizacyjnego - maksymalizacji funkcji wklęsłej z jednym warunkiem liniowym dla którego istnieje przybliżona metoda rozwiązywania, a mianowicie metoda Newtona.

Wadą stosowania tych metod jest konieczność przechowywania w pamięci całej macierzy jądra. A więc złożoność pamięciowa tych algorytmów jest kwadratowa w stosunku do ilości wektorów wejściowych. Już dla kilkudziesięciu tysięcy wektorów wymagana pamięć przekracza możliwości standardowych komputerów PC. Powstała jednak technika, która umożliwia ominięcie tego problemu, a mianowicie technika dekompozycji [15]. Istotą techniki dekompozycji jest podział problemu na mniejsze podproblemy. Technika ta umożliwiła powstanie nowoczesnego algorytmu SMO, który opiera się na dekompozycji problemu na podproblemy najmniejsze z możliwych, dwuparametrowe. Takie podproblemy posiadają rozwiązanie analityczne, i nie jest konieczne stosowanie wspomnianych wcześniej metod numerycznych. Duże zbiory danych wymuszają stosowanie metody dekompozycji również w przypadku metod numerycznych nie tylko ze względu na złożoność pamięciową, ale również obliczeniową.

Poniżej zostanie przedstawiona idea dekompozycji i algorytm SMO. W kolejnym rozdziale będzie przedstawiony nowy algorytm optymalizacji ASO.

2.1. Idea dekompozycji

Dekompozycja problemu optymalizacyjnego SVM została przedstawiona po raz pierwszy przez Osunę [15]. Metoda ta polega na wyróżnieniu dwóch podzbiorów para-

metrów, tzw. *zbioru aktywnego*, zwanego inaczej roboczym i *zbioru pasywnego*. W każdym kroku iteracyjnym parametry pasywne nie są optymalizowane, w przeciwieństwie do parametrów aktywnych. Algorytmy opierające się na dekompozycji znajdują zbiór parametrów aktywnych taki, że optymalizacja tych parametrów powoduje możliwie jak największe przesunięcie aktualnego rozwiązania w kierunku maksimum globalnego.

Podproblemy SVM wyznaczone w procesie dekompozycji przybierają postać:

Problem optymalizacyjny 2.1.1. *Maksymalizacja funkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} y_i y_j \alpha_i \alpha_j K(x_i, x_j),$$

przy ograniczeniach:

$$\sum_{i=1}^{p} y_i \alpha_i = A$$

$$0 \leq \alpha_i \leq C$$

dla $i \in \{1..p\}$,

gdzie p jest liczbą wybranych parametrów do kroku optymalizacyjnego metody dekompozycji, $p \in [2, l]$,

zaś A jest tak dobrane aby zadość uczynić warunkowi równościowemu problemu (PO 1.2.1), czyli tak aby $\sum_{i=1}^{l} y_i \alpha_i = 0$.

Sposób wyboru aktywnego zbioru parametrów opiera się na heurystykach, najprostszą z nich jest *heurystyka porcji* (ang. *chunking*). Heurystyka ta polega na aribtralnym wyborze w korku początkowym podzbioru parametrów aktywnych, a następnie wyznaczeniu rozwiązania optymalnego dla tych parametrów. Kolejno tworzony jest nowy zbiór parametrów aktywnych, w skład którego wchodzą wszystkie wektory wspierające otrzymane w poprzednim rozwiązaniu, oraz M wektorów ze zbioru parametrów pasywnych najgorzej spełniających kryterium KKT, gdzie M jest parametrem systemu. Dla nowo powstałego zbioru parametrów obliczane jest ponownie optymalne rozwiązanie.

Proces iteracyjny zostaje zatrzymany, gdy jest spełnione wybrane kryterium stopu.

Algorithm 1 Chunking

Dany zbiór treningowy S

$$\alpha \leftarrow 0$$

Procedure Chunking;
  Wybierz arbitralnie $\widehat{S} \subset S$;
  repeat
    Rozwiąż podproblem optymalizacyjny na $\widehat{S}$;
    Wybierz nowy zbiór aktywny;
  until (jest spełnione wybrane kryterium stopu);

Zbiór parametrów aktywnych musi być dostatecznie mały tak, aby optymalizacja tego zbioru za pomocą pakietu optymalizującego była dostatecznie szybka.

W następnym punkcie zostanie opisany algorytm dla którego zbiór aktywnych parametrów jest wielkości dwa.

2.2. Minimalna optymalizacja sekwencyjna

Algorytm Minimalnej Optymalizacji Sekwencyjnej (ang. *Sequential Minimal Optimization*, SMO) [18] oparty jest na metodzie dekompozycji Osuny [15]. Zbiór aktywny jest najmniejszym z możliwych, zbiorem dwuelementowym. Korzyścią płynącą z tego faktu jest możliwość analitycznego rozwiązania podproblemu dwuparametrowego. A zatem nie ma konieczności stosowania skomplikowanych metod numerycznych rozwiązujących podproblem dla dowolnej liczby parametrów.

Oprócz analitycznego rozwiązania podproblemu SVM, częścią algorytmu jest heurystyka wyboru parametrów zbioru aktywnego oraz warunek stopu.

2.2.1. Rozwiązanie analitycznie dla dwóch punktów.

Rozwiązanie analityczne dotyczy nieliniowej klasyfikacji słabego marginesu podproblemu SVM. Na początku zostanie przyjęte założenie takie, że dwoma wybranymi parametrami do zbioru aktywnego są parametry α_1 i α_2.

Zakłada się ponadto, że wejściowe parametry $\vec{\alpha_i}$ spełniają warunek równościowy. Mogą na przykład wszystkie być równe zero.

$$\sum_{i=1}^{l} \alpha_i^{\text{old}} y_i = 0,$$

$$\alpha_1^{\text{old}} y_1 + \alpha_2^{\text{old}} y_2 + \sum_{i=3}^{l} \alpha_i^{\text{old}} y_i = 0$$

Tylko parametry α_1^{old} i α_2^{old} zostaną poddane optymalizacji, a zatem nowe wartości tych parametrów również muszą spełniać warunek równościowy problemu SVM. Z tego wynika, że

$$\alpha_1 y_1 + \alpha_2 y_2 = \alpha_1^{\text{old}} y_1 + \alpha_2^{\text{old}} y_2.$$

Ponadto nowe wartości parametrów α_1 i α_2 muszą spełniać warunek nierównościowy:

$$0 \leq \alpha_1,\ \alpha_2 \leq C.$$

Dla wyliczonych wartości parametrów α_1 i α_2, α_2 spełnia powyższe warunki, wtedy i tylko wtedy, gdy

$$U \leq\ \alpha_2 \leq V, \tag{2.1}$$

gdzie
dla $y_1 \neq y_2$:

$$U = \max\left(0,\ \alpha_2^{\text{old}} - \alpha_1^{\text{old}}\right),$$

$$V = \min\left(C,\ C - \alpha_1^{\text{old}} + \alpha_2^{\text{old}}\right),$$

dla $y_1 = y_2$:

$$U = \max\left(0,\ \alpha_1^{\text{old}} + \alpha_2^{\text{old}} - C\right),$$

$$V = \min\left(C,\ \alpha_1^{\text{old}} + \alpha_2^{\text{old}}\right).$$

Wyprowadzenie powyższego faktu zostało zamieszczone w dodatku A.1
Dalej zostanie wprowadzone oznaczenie:

$$E_i = \sum_{j=1}^{l} \alpha_j y_j K\left(x_j, x_i\right) - y_i \tag{2.2}$$

oraz

$$\kappa = K\left(x_1, x_1\right) + K\left(x_2, x_2\right) - 2K\left(x_1, x_2\right) \tag{2.3}$$

Nowe wartości parametrów są nastepujące:
Najpierw jest wyliczane α_2^{unc}

$$\alpha_2^{\text{unc}} = \alpha_2^{\text{old}} + \frac{y_2\left(E_1 - E_2\right)}{\kappa}$$

a następnie

$$\alpha_2 = \begin{cases} V, & jesli\ \alpha_2^{\text{new,unc}} > V, \\ \alpha_2^{\text{new,unc}}, & jesli\ U \le \alpha_2^{\text{new,unc}} \le V, \\ U, & jesli\ \alpha_2^{\text{new,unc}} < U, \end{cases} \tag{2.4}$$

$$\alpha_1^{\text{new}} = \alpha_1^{\text{old}} + y_1 y_2\left(\alpha_2^{\text{old}} - \alpha_2^{\text{new}}\right).$$

Wyprowadzenie powyższych wzorów zostało zamieszczone w Dodatku A.

2.2.2. Heurystyka

Wybór heurystyki stanowi isotny element algorytmów opartych na dekompozycji.

Wybór parametrów do zbioru aktywnego w oryginalnym algorytmie SMO zaproponowanym przez Platta [18] odbywa się w ten sposób, że:

pierwszy parametr jest wybierany spośród wszystkich parametrów, ponadto pierwszeństwo mają parametry, które spełniają nierówność $0 < \alpha_1 < C$,

drugi parametr jest wybierany taki, dla którego wartość $E_1 - E_2$ jest maksymalna, tak aby relatywnie niskim kosztem wybrać taki parametr, który maksymalizuje zmianę wartości parametru α_2:

Optymalizacja heurystyki

W [13] została zaproponowana optymalizacja heurystyki Platta.

Algorithm 2 Heurystyka Platta

```
Procedure ChooseFirstParameter;
begin
    for i: = 1 to l do
    begin
        if (chosenParameter > 0 and chosenParameter < C)
        begin
            ChooseSecondParameter(i);
            if (not changed)
                break;
        end;
    end;
    for i: = 1 to l do
        ChooseSecondParameter(i);
    if (changed)
        ChooseFirstParameter;
end;
Function ChooseSecondParameter(parameter : Integer);
begin
    Znajdź drugi parametr taki, który maksymalizuje wartość E1 − E2;
    Optymalizuj;
    if (not changed)
    begin
        for i: = 1 to l do
            if (chosenParameter > 0 and chosenParameter < C)
            begin
                Optymalizuj;
                if (warunek stopu spełniony) exit;
                if (changed) return;
            end;
        for i: = 1 to l do
        begin
            Optymalizuj;
            if (warunek stopu spełniony) exit;
            if (changed)
            return;
        end;
    end;
end;
```

Parametr pierwszy jest wybierany spośród parametrów, które nie spełniają warunku KKT (1.16). Parametr drugi jest wybierany w taki sposób, aby wartość $E_1 - E_2$ była maksymalna.

W aplikacji LibSVM [5] została użyta podobna heurystyka, a mianowicie znajdująca pierwszy parametr najgorzej spełniający warunek KKT, a drugi taki, aby wartość $E_1 - E_2$ była maksymalna.

Algorithm 3 Heurystyka LibSVM

```
Procedure ChooseFirstParameter;
begin
    for i: = 1 to l do
    begin
        if (parametr i nie spełnia KKT)
        begin
            ChooseSecondParameter(i);
            if (not changed)
                break;
        end;
    end;
    for i: = 1 to l do
        ChooseSecondParameter(i);
    if (changed)
        ChooseFirstParameter;
end;
Function ChooseSecondParameter(parameter : Integer);
begin
    Znajdz drugi parametr taki, który maksymalizuje wartość E1 – E2.
    Optymalizuj;
    if (warunek stopu spełniony) exit;
end;
```

Warunek stopu

Warunek stopu zastosowany w oryginalnym algorytmie SMO polega na zakończeniu obliczeń, jeśli wszystkie parametry spełniają warunek KKT z pewną ustaloną dokładnością, który jest warunkiem koniecznym i wystarczającym poprawnego rozwiązania problemu optymalizacyjnego SVM. Praktyczne testy pokazują, że wystarczającą dokładnością jest dokładność 0.001. Algorytmy SVM mają to do siebie, iż są wolno zbieżne dla dużych dokładności.

Rozdział 3

Proponowane algorytmy optymalizacji

W punkcie 3.1 zostanie przedstawiona nowa metoda optymalizacji, tzw. metoda *Analitycznej Optymalizacji Sekwencyjnej* (ang. *Analytical Sequential Optimization*, ASO) rozwiązująca analitycznie wieloparametrowe podproblemy SVM. Ograniczeniem zarówno tej metody, jak również istniejących metod numerycznych jest słaba skalowalność na dużą liczbę parametrów.

Dlatego istotną kwestią rozwiązywania problemu SVM o dużych rozmiarach jest algorytm dekompozycji. W punkcie 3.3 zostanie przedstawiona nowa heurystyka dekompozycji polegająca na poszukiwaniu kierunku najszybszego wzrostu funkcji celu, przy jak najdokładniejszym spełnieniu warunku równościowego problemu SVM.

3.1. Metoda Analitycznej Optymalizacji Sekwencyjnej

Algorytm SMO opiera się na optymalizacji podproblemów dwuparametrowych, które rozwiązywane są analitycznie. Proponowany w tej pracy algorytm Analitycznej Optymalizacji Sekwencyjnej rozwiązuje analitycznie podproblemy wieloparametrowe. Rozwiązanie analityczne podproblemów w przeciwieństwie do metod numerycznych jest rozwiązaniem mniej skomplikowanym i łatwiejszym do implementacji.

W dalszej części pracy idea dekompozycji problemu SVM na podproblemy SVM będzie nazywana *dekompozycją zewnętrzną*. Rozwiązanie podproblemów również opiera się na dekompozycji, dla odróżnienia będzie ona nazywana *dekompozycją wewnętrzną*. Polega ona na podziale podproblemu na jeszcze mniejsze podproblemy, które z kolei dzielone są na kolejne podproblemy, itd. Ideą ASO jest minimalizacja koniecznych do rozpatrzenia podproblemów. Podproblemy dekompozycji wewnętrznej będą nazywane w dalszej części *podproblemami ASO*. Mają one następującą postać:

Problem optymalizacyjny 3.1.1. *Maksymalizacja funkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{q} \alpha_i - \frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{q} y_i y_j \alpha_i \alpha_j K(x_i, x_j),$$

przy ograniczeniach:

$$\sum_{i=1}^{q} y_i \alpha_i = B$$

$$0 \leq \alpha_i \leq C$$

dla $i \in [1, q]$,

gdzie p jest liczbą parametrów aktywnych dekompozycji zewnętrznej, $p \in [2, l]$, *q jest liczbą parametrów dekompozycji wewnętrznej,* $q \in [2, p]$,
zaś B jest tak dobrane aby zadość uczynić warunkowi równościowemu podproblemu SVM (PO 2.1.1), czyli tak aby $\sum_{i=1}^{p} y_i \alpha_i = A$.

Jeśli będzie mowa w dalszej części pracy o podproblemie, w domyśle będzie chodziło o podproblem ASO.

W punkcie 3.1.1 zostanie przedstawiony model geometryczny-analityczny metody ASO, a w kolejnym punkcie algorytmy metody Analitycznej Optymalizacji Sekwencyjnej.

3.1.1. Model geometryczno-analityczny Analitycznej Optymalizacji Sekwencyjnej

Model geometryczny polega na reprezenetacji podproblemu ASO za pomocą obiektów geometrycznych. Model analityczny uzyskuje się natomiast za pomocą sprowadzania tworów geometrycznych n-wymiarowych do ujęcia analitycznego.

Istotną częścią modelu geometrycznego podproblemu ASO jest punkt geometryczny reprezentujący rozwiązanie następującego podproblemu optymalizacyjnego:

Problem optymalizacyjny 3.1.2. *Maksymalizacja funkcji*

$$W(\vec{\alpha}) = \sum_{i=1}^{q} \alpha_i - \frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{q} y_i y_j \alpha_i \alpha_j K(x_i, x_j)$$

przy ograniczeniu:

$$\sum_{i=1}^{q} y_i \alpha_i = B$$

Powyższy problem tym różni się od podproblemu ASO, że nie ma w nim warunku nierównościowego.

Jest to problem maksymalizacji funkcji kwadratowej wklęsłej, dwukrotnie różniczkowalnej, z jednym warunkiem równościowym, a zatem posiada rozwiązanie analityczne. Wyprowadzenie rozwiązania opiera się na włączeniu warunku równościowego do funkcji celu W, a następnie wyznaczeniu pochodnych cząstkowych funkcji celu z włączonymi warunkiem równościowym.

Przyrównując pochodne cząstkowe do zera otrzymuje się układ równań liniowych. Znalezienie współczynników tego układu równań zostało po raz pierwszy zaproponowane w tej pracy. W dodatku B.1 zostało zamieszczone wyprowadzenie tych współczynników. Poniżej zostaną przedstawione jedynie wyprowadzone wzory.

Rozwiązanie analityczne problemu (PO 3.1.2) jest rozwiązaniem następującego układu równań:

$$\sum_{i=1}^{q-1} y_i \alpha_i^{\text{new}} \kappa_{iqk} = \sum_{i=1}^{q-1} y_i \alpha_i \kappa_{iqk} + E_k - E_q \tag{3.1}$$

gdzie $\kappa_{iqk} = K_{iq} + K_{kq} - K_{qq} - K_{ik}$,
$k \in [1, q-1]$.
Parametr α_q^{new} wyliczany jest ze wzoru:

$$\alpha_q^{\text{new}} = \gamma_q - \sum_{i=1}^{q-1} s_{iq} \alpha_i^{\text{new}}, \tag{3.2}$$

gdzie $\gamma_q = \sum_{i=1}^{q} s_{iq} \alpha_i$,

$s_{iq} = y_i y_q$.

Wyrażenia κ_{iqk} tworzą macierz symetryczną.

Układ równań liniowych może zostać rozwiązany metodą dokładną np. metodą eliminacji Gaussa z częściowym wyborem elementu podstawowego, złożoność obliczeniowa tej metody wynosi $O(n^3)$, dla niewielu parametrów będzie to odpowiednia metoda ze względu na jej stabilność.

W modelu geometrycznym metody ASO oprócz punktu reprezentującego rozwiązanie (PO 3.1.2) występują struktury geometryczne, wielowymiarowe odpowiadające warunkom podproblemu ASO.

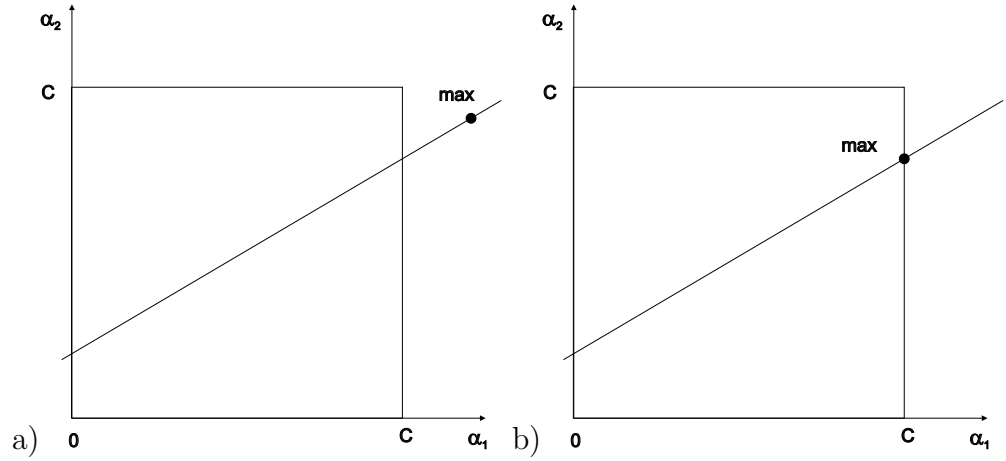
W dalszym ciągu liczba wymiarów będzie odpowiadała liczbie parametrów q. A zatem zgodnie z tym założeniem funkcja W będzie $q+1$ wymiarowa.

Dla przypadku dwuwymiarowego warunek liniowy podproblemu ASO reprezentowany jest graficznie przez prostą, zaś warunek nierównościowy przez kwadrat.

Można zauważyć, że rozwiązanie problemu (PO 3.1.2) leży na prostej, która reprezentuje warunek równościowy problemu (PO 3.1.2), natomiast rozwiązanie nie leży w obrębie kwadratu, reprezentującego warunek nierównościowy podproblemu ASO, ponieważ problem (PO 3.1.2) nie musi spełniać tego warunku (Rys. 3.1), punkt a). Jeśli jednak zdarzy się sytuacja, że rozwiązanie problemu (PO 3.1.2) spełnia ten warunek, to graficznie punkt reprezentujący rozwiązanie (PO 3.1.2) będzie leżał w obrębie kwadratu (Rys. 3.1), punkt b). Wtenczas jest ono równocześnie rozwiązaniem podproblemu ASO.

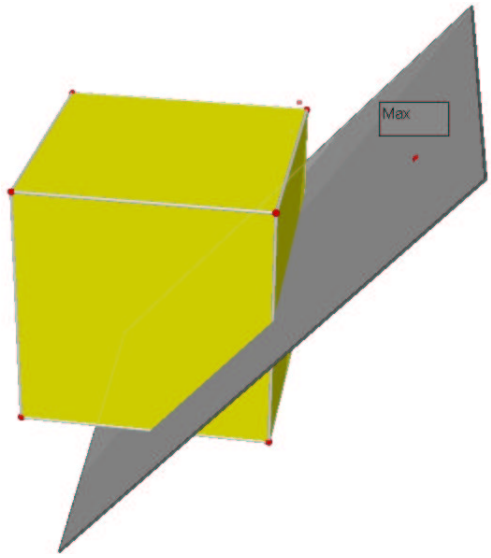
W tej sytuacji nie jest konieczne dalsze postępowanie postępowanie mające na celu znalezienie rozwiązania podproblemu ASO.

W geometrii wielowymiarowej tworem geometrycznym odpowiadającym warunkowi liniowemu jest hiperpłaszczyzna, zaś warunkowi nierównościowemu odpowiada



Rysunek 3.1. Rysunek przedstawia położenie optymalnego rozwiązania problemu (PO 3.1.2) w przypadku a) poza kwadratem, w przypadku b) w obrębie kwadratu dla dwóch wymiarów.

hipersześcian. Na rysunku (Rys. 3.2) został przedstawiony ten sam problem dla trzech wymiarów.



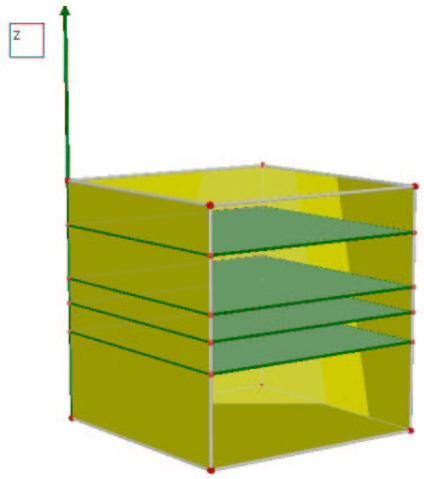
Rysunek 3.2. Rysunek przedstawia położenie poza kwadratem optymalnego rozwiązania problemu (PO 3.1.2) dla przypadku trójwymiarowego.

Uogólniając wnioski wysunięte dla przypadku dwuwymiarowego i trójwymiarowego do wielu wymiarów, jeśli punkt rozwiązania problemu (PO 3.1.2) leży w obrębie hipersześcianu, to rozwiązanie jest równocześnie rozwiązaniem podproblemu ASO dla q wybranych parametrów. Dalej zostanie rozpatrzony przypadek kiedy punkt rozwiązania (PO 3.1.2) leży poza hipersześcianem, a więc kiedy rozwiązanie (PO 3.1.2) nie spełnia warunku nierównościowego podproblemu ASO dla q wybranych parametrów. Wtenczas konieczne jest dalsze postępowanie mające na celu znalezienie optymalnego rozwiązania SVM.

Podproblem $q-1$ *wymiarowy* podproblemu q wymiarowego powstaje przez zastąpienie jednej zmiennej podproblemu ASO konkretną wartością. Zmienna ta zostanie nazwana *zmienną kierunkową* podproblemu $q-1$ wymiarowego. Każda zmienna kierunkowa przyjmuje wartości rzeczywiste z zakresu $\alpha \in [0, C]$, a zatem możliwych podstawień jest nieskończenie wiele, a więc możliwych podproblemów q-1 wymiarowych podproblemu ASO jest nieskończenie wiele.

Graficznie warunek równościowy podproblemu $q-1$ wymiarowego jest reprezentowany przez hiperpłaszczyznę $q-1$ wymiarową, zaś warunek nierównościowy będzie hipersześcianem $q-1$ wymiarowym.

Dla przypadku trójwymiarowego można sobie wyobrazić, że warunkiem nierównościowym podproblemu $q-1$ wymiarowego jest kwadrat leżący w obrębie dowolnej płaszczyzny przecinającej sześcian, takiej, że wektorem kierunkowym tej płaszczyzny jest wektor jednostkowy osi wybranej wcześniej zmiennej kierunkowej. O konkretnym położeniu decyduje wartość zmiennej kierunkowej. Na rysunku (Rys. 3.3) założono, że zmienną kierunkową jest parametr z.



Rysunek 3.3. Rysunek przedstawia kwadraty reprezentujące warunki nierównościowe podproblemów dwuwymiarowych podproblemu trójwymiarowego.

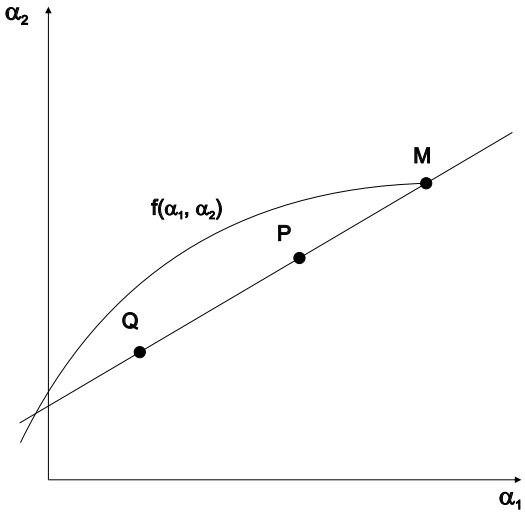
Podstawowy problem dotyczy możliwości ograniczenia liczby tych płaszczyzn, w ogólności hiperpłaszczyzn, a zatem podproblemów $q-1$ wymiarowych. Analitycznie zagadnienie dotyczy ograniczenia możliwości wyboru wartości zmiennej kierunkowej wybranego podproblemu $q-1$ wymiarowego. Drugim zagadnieniem jest ograniczenie możliwości wyboru podproblemów $q-1$ wymiarowych, a tym samym zmiennej kierunkowej.

Korzystając z pewnych właściwości funkcji celu podproblemu ASO oraz z faktu znalezienia położenia rozwiązania problemu (PO 3.1.2) jest możliwe poczynienie tych ograniczeń.

W tym celu zostanie przedstawione następujące twierdzenie dotyczące funkcji wklęsłych:

Twierdzenie 3.1.1. *Jeśli zostanie poprowadzona półprosta od maksimum globalnego (M) danej funkcji wklęsłej, przechodząca przez dowolny inny punkt (P), to dla wszystkich punktów leżących na półprostej PQ bez początku (Q to punkt leżący na półprostej MP taki, że jego odległość do M jest większa niż punktu P do M) wartość funkcji będzie mniejsza niż w punkcie P, czyli, że $f(Q) < f(P)$ (Rys. 3.4).*

Powyższe twierdzenie jest spełnione niezależnie od wymiaru funkcji celu.



Rysunek 3.4. Rysunek pomocniczy do twierdzenia (Tw. 3.1.1)

Dowód. Definicja funkcji wklęsłej w przestrzeni n-wymiarowej brzmi następująco:

$$f(\theta\alpha_1 + (1-\theta)\alpha_2) \geq \theta f(\alpha_1) + (1-\theta) f(\alpha_2),$$

gdzie α_1 i α_2 są wektorami n-wymiarowymi należącymi do dziedziny funkcji f, a $\theta \in (0,1)$.

Dowód nie wprost.

Zakłada się, że dla punktu Q leżącego na półprostej MP zachodzi:

$f(Q) \geq f(P)$,czyli że wartość funkcji w punkcie Q jest większa lub równa wartości w punkcie P.

Jako, że funkcja f jest wklęsła można zapisać warunek wklęsłości dla punktów Q i M:

$$f(\theta Q + (1-\theta) M) \geq \theta f(Q) + (1-\theta) f(M) \tag{3.3}$$

Z faktu, że punkt P leży między punktami Q i M (lub jest punktem Q) wynika, że:

$$\exists_{\theta_1 \in (0,1)} (\theta_1 Q + (1-\theta_1) M = P). \tag{3.4}$$

Dla θ_1 zdefiniowanego w (3.4) można zapisać (3.3) w postaci:

$$f(\theta_1 Q + (1-\theta_1) M) \geq \theta_1 f(Q) + (1-\theta_1) f(M).$$

Podstawiając (3.4) do powyższego:

$$f(P) \geq \theta_1 f(Q) + (1-\theta_1) f(M)$$

$f(P) \geq \theta_1 (f(P) + \beta) + (1-\theta_1) f(M)$,gdzie $\beta > 0$

$$f(P) - \theta_1 f(P) \geq \theta_1 \beta + (1-\theta_1) f(M)$$

$$(1-\theta_1) f(P) \geq \theta_1 \beta + (1-\theta_1) f(M)$$

Jako, że $\theta_1 \beta \geq 0$ z faktu prawdziwości powyższej nierówności wynika, że

$$(1-\theta_1) f(P) \geq (1-\theta_1) f(M)$$

Jako, że $(1-\theta_1) > 0$
otrzymuje się

$$f(P) \geq f(M)$$

Co jest sprzeczne ponieważ M jest jedynym maksimum globalnym, a P jest różne od M.

A więc warunek początkowy $f(Q) \geq f(P)$ jest nieprawdziwy, cbdo. □

Jako, że powyższe twierdzenie jest spełnione dla każdej półprostej wychodzącej od maksimum można wziąść pod uwagę *pęk takich półprostych* w przestrzeni q-wymiarowej. Każda półprosta wychodząca od maksimum zostanie nazwana *promieniem*.

Definicja 3.1.1. Zbiór punktów przecięcia promieni z hipersześcianem ograniczony do podzbioru punktów najbliższych maksimum dla każdej półprostej z tego pęku zostanie nazwany *widokiem*.

Dla przypadku dwuwymiarowego widok został przedstawiony na rysunku (Rys. 3.5).

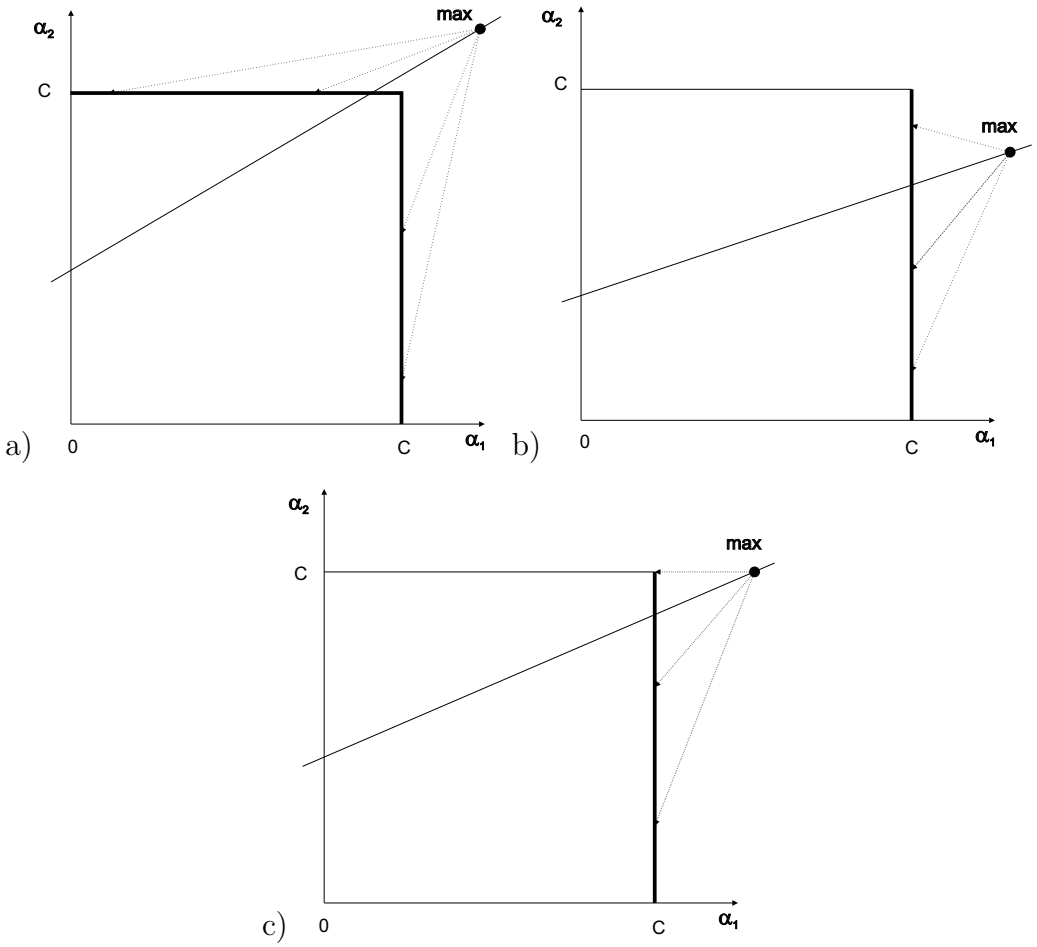
Można zauważyć, że zgodnie z (Def. 3.1.1) widok jest determinowany przez punkty przecięcia promieni najbliższe maksimum dla każdego promienia (Rys. 3.5). W zależności od położenia maksimum (PO 3.1.2) otrzymuje się różne obiekty geometryczne reprezentujące widok. W przypadku a) widokiem jest zbiór dwóch sąsiadujących ze sobą boków kwadratu. W przypadku b) i c) widokiem jest jeden bok kwadratu.

W celu wyznaczenia możliwych obiektów geometrycznych reprezentujących widok dla przypadku q-wymiarowego zdefiniowana zostanie ściana hipersześcianu:

Definicja 3.1.2 (Definicja ściany hipersześcianu)**.** Dla hipersześcianu danego warunkiem nierównościowym podproblemu ASO *ścianą hipersześcianu* q wymiarowego nazywany jest hipersześcian $q-1$ wymiarowy powstały z hipersześcianu q wymiarowego przez zastąpienie jednej ze zmiennych w warunku nierównościowym wartością graniczną (0 lub C).

Twierdzenie 3.1.2. *Widok jest podzbiorem ścian* $q-1$ *wymiarowych hiperszescianu, których jest dokładnie 2p.*

Dowód. Najpierw zostanie udowodniona teza, że do widoku mogą należeć tylko i wyłącznie punkty należące do ścian hipersześcianu.



Rysunek 3.5. Rysunek przedstawia widok oznaczony gruba linią, linią przerywaną zostały zaznaczone promienie wychodzące z punktu maksymalnego problemu (PO 3.1.2) w stronę kwadratu.

Gdyby widok zawierał punkt wewnętrzny hipersześcianu, to na odcinku łączącym ten punkt z dowolnym punktem poza obrębem hipersześcianu leżałby punkt zewnętrzny (należący do ściany hipersześcianu). A więc odległość tego punktu zewnętrznego do punktu leżącego poza hipersześcianem byłaby w tym wypadku mniejsza od odległości od punktu wewnętrznego, a zatem zgodnie z definicją widoku nie mógłby należeć do widoku.

Liczba zmiennych problemu q wymiarowego wynosi q, każdą z nich można zastąpić wartością 0 lub C i otrzyma się istotnie różnie ściany, a więc ścian jest $2q$, cbdo. □

Wyżej zostało pokazane jak można ograniczyć liczbę hiperpłaszczyzn $q-1$ wymiarowych, a tym samym problemów $q-1$ wymiarowych. Na początku ilość możliwych hiperpłaszczyzn była nieskończona, okazało się jednak, żę można ją ograniczyć do skończonej wartości $2q$. Powyższe rozumowanie nie wskazuje jednak na dokładną liczbę ścian widoku, która jak się okaże będzie jeszcze mniejsza.

W powyższym rozumowaniu została wykorzystana informacja o tym, że funkcja celu jest wklęsła, oraz o położeniu maksimum (PO 3.1.2) poza hipersześcianem.

Jednak położenie maksimum (PO 3.1.2) jest dane konkretnie, co stanowi przesłankę do dalszego zmniejszenia liczby możliwych podproblemów i wyznaczeniu prawidłowej liczby ścian widoku.

Jak położenie maksimum (PO 3.1.2) wpływa na typ widoku?

Dla przypadku dwuwymiarowego (Rys. 3.5) można zauważyć następującą prawidłowość: W przypadku a) konieczna była zmiana zarówno współrzędnej x jak i współrzędnej y w celu otrzymania punktu najbliższego maksimum leżącego na kwadracie. Dlatego do widoku należą dwie ściany $y = \mathrm{C}$ oraz $x = \mathrm{C}$. W przypadku b) i c) konieczna była zmiana jedynie współrzędnej x w celu otrzymania współrzędnych punktu najbliższego maksimum leżącego na kwadracie. Dlatego do widoku należy jedna ściana $x = \mathrm{C}$.

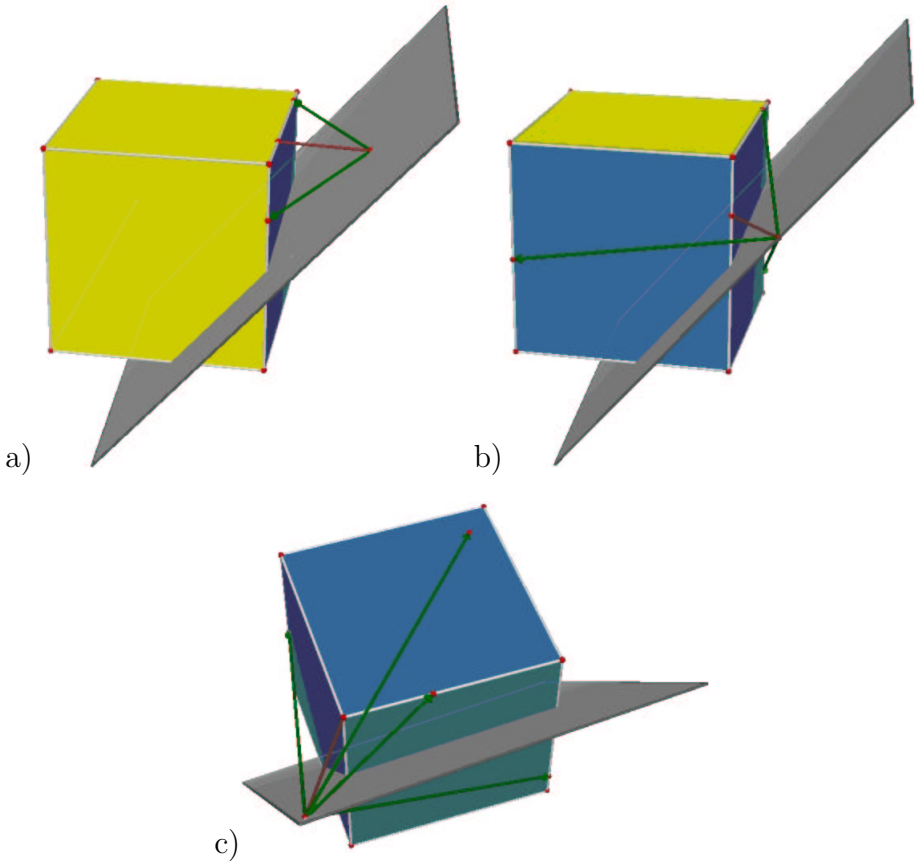
Powyższe rozumowanie z powodzeniem można przeprowadzić dla przypadku trójwymiarowego.

W przypadku trójwymiarowym ścianami sześcianu są kwadraty (Rys. 3.6).

Zależnie od położenia maksimum widokiem może być zbiór składający się z jednej ściany a), z dwóch sąsiadujących ścian b), lub z trzech sąsiadujących ścian c). Podobnie jak dla przypadku dwuwymiarowego ów podział determinowany jest przez zmianę współrzędnych punktu maksymalnego w celu otrzymania punktu najbliższego maksimum leżącego na sześcianie. Dla przypadku a) konieczna była zmiana jednego parametru w celu otrzymania punktu najbliższego, z kolei dla przypadku b) konieczna była zmiana dwóch parametrów, dla c) wszystkich trzech parametrów. Liczbie zmienionych parametrów odpowiada liczba ścian należących do widoku. Zaś parametr graniczny na który została zmieniona współrzędna determinuje czy jest to ściana, której odpowiada parametr C czy też 0.

Uogólniając powyższe dla przypadku q wymiarowego można otrzymać typ widoku rozpatrując najbliższy punkt do maksimum leżący na hipersześcianie, który zostanie nazwany *punktem rzutowym widoku*. Zależnie od ilości parametrów, takich które wychodzą poza zakres $\alpha \in [0, C]$, otrzymuje się różne widoki, każdy z nich składający się z odpowiedniej ilości ścian $q-1$ wymiarowych.

Operacja wyznaczenia punktu rzutowego sprowadza się do obcięcia wszystkich



Rysunek 3.6. Rysunek przedstawia oznaczony niebieskim kolorem widok, składający się z jednej, dwu lub trzech ścian, promienie wychodzące z punktu maksymalnego funkcji W spełniającej warunek równościowy zaznaczone zielonymi wektorami oraz punkt najbliższy maksimum leżący w obrębie sześcianu do którego prowadzi czerwony wektor.

współrzędnych punktu maksimum wychodzących poza zakres $\alpha \in [0, C]$ do najbliższej wartości granicznej. Parametry które zostały obcięte zostaną nazwane *parametrami rzutowanymi*. Zbiór tych parametrów oznaczony jest symbolem C_q.

Twierdzenie 3.1.3. *Punkt należy do widoku, wtedy i tylko wtedy, gdy należy do jednej ze ścian punktu rzutowego odpowiadającej parametrowi rzutowanemu.*

Dowód. Tożsama postać twierdzenia (Tw. 3.1.3):

Punkt nie należy do widoku wtedy i tylko wtedy, gdy nie należy do żadnej ze ścian punktu rzutowego odpowiadającej dowolnemu parametrowi rzutowanemu.

Punkt maksimum zostanie oznaczony jako $P_m = [x_{1m}, x_{2m}, ..., x_{pm}]$.

oraz punkt hipersześcianu należący do widoku$P_q = [x_{1q}, x_{2q}, ..., x_{pq}]$.

Prosta przechodząca przez punkty P_m i P_q ma wzór:

$$\forall_i \left(x_i = x_{im} + (x_{im} - x_{iq})\, t\right)$$

Prosta przechodzi przez punkt P_q, a więc można wyznaczyć parametr t dla tego punktu:

$$x_{1q} = x_{1m} + (x_{1m} - x_{1q})\, t_q$$

$$t_q = 1$$

A zatem zachodzi:

$$\forall_i \left(x_{iq} = x_{im} + (x_{im} - x_{iq})\, t_q\right)$$

Punkt nie należy do widoku, a więc dla dowolnego punktu leżącego na prostej p_{qm} w kierunku maksimum istnieje punkt należący do hipersześcianu.

Nowy punkt leżący na prostej p_{qm} w kierunku maksimum będzie oznaczony jako $x_{iq\varepsilon}$

A zatem zachodzi $x_{iq\varepsilon} < C \vee x_{iq\varepsilon} > 0$

Wtenczas

$$x_{iq\varepsilon} = x_{im} + (x_{im} - x_{iq})\, (t_q + \varepsilon)\,,$$

gdzie $\varepsilon > 0$ i jest dowolnie małe.

$$x_{iq\varepsilon} = x_{im} + (x_{im} - x_{iq})\, t_q + (x_{im} - x_{iq})\, \varepsilon$$

$$x_{iq\varepsilon} = x_{iq} + (x_{im} - x_{iq})\, \varepsilon$$

Aby było spełnione $x_{iq\varepsilon} < C \vee x_{iq\varepsilon} > 0$ dla dowolnie małego ε, musi zachodzić odpowiednio:

$x_{iq} = C \wedge (x_{im} - x_{iq}) < 0$ lub $x_{iq} = 0 \wedge (x_{im} - x_{iq}) > 0$

Po przekształceniu:

$$\begin{aligned} x_{im} < C \quad &\text{dla} \quad \mathrm{x_{iq}} = \mathrm{C} \\ x_{im} > 0 \quad &\text{dla} \quad \mathrm{x_{iq}} = 0 \end{aligned} \tag{3.5}$$

Parametr rzutowany ma taką właściwość, że:
$x_{im} > C$dla $x_{iq} = C$ oraz $x_{im} < 0$ dla $x_{iq} = 0$.
Jako, że (3.5) musi zachodzić dla każdego parametru, a dowolny punkt ściany punktu rzutowego zawiera parametr rzutowany, dla którego nie zachodzi (3.5), a więc punkt nie należący do widoku nie może należeć do takiej ściany, cbdo. □

Twierdzenie 3.1.4. *Maksymalna liczba ścian widoku wynosi q.*

Dowód. Punkt leżący na hipersześcianie należy do ścian określonych analitycznie współrzędnymi granicznymi tego punktu. Jeśli każda współrzędna maksimum została obcięta podczas wyznaczania punktu rzutowego widoku to, z faktu, że dla każdego parametru rzutowanego otrzymuje się jedną scianę widoku do której ten punkt należy wynika, że w sumie może ich być maksymalnie q, tyle co współrzędnych rzutowanych tego punktu, cbdo. □

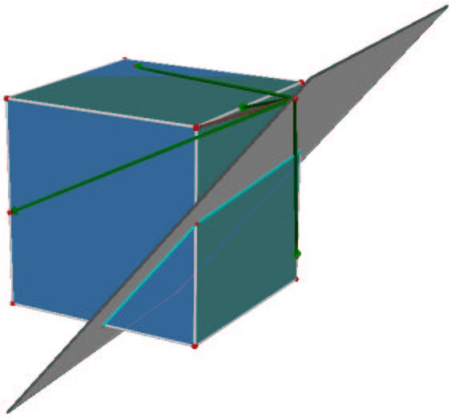
A zatem można ograniczyć liczbę ścian $q-1$ wymiarowych, a tym samym podproblemów $q-1$ wymiarowych do ilości ścian widoku, która maksymalnie wynosi q.

Na tym etapie jest wyznaczony widok, a tym samym zestaw podproblemów $q-1$ wymiarowych odpowiadających poszczególnym ścianom widoku.

Włączenie warunku równościowego

Jako że zarówno warunek nierównościowy jak i warunek równościowy mają być równocześnie spełnione w podproblemie ASO, a zatem można ograniczyć warunkiem równościowym zbiór ścian widoku wyznaczonych wcześniej. Dlatego zbiór ścian widoku zostanie ograniczony do zbioru ścian przecinających się z hiperpłaszczyzną, określoną warunkiem równościowym.

Rysunek (Rys. 3.7) przedstawia istotę dołączenia warunku równościowego do poprzednich rozważań dla przypadku trójwymiarowego.



Rysunek 3.7. Na rysunku została zaznaczona kolorem jasnoniebieskim część widoku składającego się z trzech ścian i przeciętego płaszczyzną.

Widok hipersześcianu jest dodatkowo ograniczony hiperpłaszczyzną. Dla przypadku trójwymiarowego zbiór ścian widoku, które równocześnie przecinają się z płaszczyzną może być mniejszy od mocy zbioru ścian widoku. Na rysunku (Rys. 3.7) pokazany jest przypadek gdy zbiór ścian widoku przecinających się z płaszczyzną wynosi dwa.

Uogólniając do przypadku q wymiarowego zbiór ścian widoku, które równocześnie przecinają się z hiperpłaszczyzną może być mniejszy od mocy zbioru ścian widoku. Z

tego względu znalezienie tego podzbioru może dodatkowo zmniejszyć liczbę podproblemów $q-1$ wymiarowych.

Z drugiej strony istnieje conajmniej jedna ściana widoku przecinająca się z hiperpłaszczyzną.

Twierdzenie 3.1.5. *Istnieje co najmniej jedna ściana widoku przecinająca się z hiperpłaszczyzną.*

Dowód. Jako, że punkt maksymalny należy do hiperpłaszczyzny, a hiperpłaszczyzna przecina hipersześcian conajmniej w jednym punkcie, to poprowadzony promień z punktu maksymalnego do punktu przecięcia z hipersześcianem wyznacza ścianę widoku, która zawiera ten punkt przecięcia, cbdo. □

Zbiór parametrów kierunkowych odpowiadających ścianom widoku zostanie oznaczony jako: C_q. Zaś zbiór parametrów kierunkowych odpowiadających ścianom widoku przecinających dodatkowo hiperpłaszczyznę jako: C'_q.

Konsekwencją twierdzenia (Tw. 3.1.5) jest fakt, iż jeśli zbiór C_q jest niepusty, to zbiór C'_q jest również niepusty.

Twierdzenie 3.1.6. *Istnieje problem SVM, taki, że istnieje podproblem ASO taki, że maksymalna liczba ścian widoku przecinająca hiperpłaszczyznę wynosi k, gdzie k to liczba ścian widoku.*

Dowód. Gdy do hiperpłaszczyzny należy punkt rzutowy widoku, wtedy jako, że wszystkie ściany do których należy punkt rzutowy widoku należą do niego, a te ściany również należą do hiperpłaszczyzny, a zatem liczba ścian widoku przecinająca hiperpłaszczyznę jest równa liczbie ścian widoku w tym przypadku.

Pozostaje jeszcze pokazać, że do hiperpłaszczyzny może należeć punkt rzutowy widoku. Jeśli podczas wyznaczania analitycznego punktu rzutowego podczas obcinania parametrów, które wychodzą poza dozwolony zakres, tyle samo zostanie odjęte co dodane, wtedy punkt rzutowy również będzie spełniał równanie równościowe, a zatem jest możliwy taki przypadek, cbdo □

Wniosek Gdy liczba ścian widoku wynosi q, to maksymalna liczba ścian przecinających hiperpłaszczyznę może wynieść q.

A więc w pesymistycznej wersji pozostaną do sprawdzenia wszystkie ściany widoku, których maksymalnie może być q.

Sprawdzenie przecięcia wybranej ściany hipersześcianu odpowiadającego warunkowi nierównościowemu z hiperpłaszczyzną odpowiadającą warunkowi równościowemu problemu (PO 3.1.1). Wzór na ścianę odpowiadającą parametrowi granicznemu wygląda następująco:

$$\alpha_g = A, \tag{3.6}$$

gdzie $A \in \{0, C\}$,
α_g jest wybranym parametrem granicznym punktu widokowego,
$g \in \{1..q\}$,

oraz na hiperpłaszczyznę:

$$\sum_{i=1}^{q} y_i \alpha_i = A. \tag{3.7}$$

Zadanie polega na stwierdzeniu czy hiperpłaszczyzna (3.7) przecina daną ścianę g. Włączając pierwszy warunek do drugiego otrzymuje się:

$$\sum_{\substack{i=1,\\ i\neq g}}^{q} y_i \alpha_i = A - y_g \alpha_g$$

Należy sprawdzić czy powyższe równanie ma rozwiązanie.
Jako, że y_i może być rowne albo 1 albo -1 można wyróżnić dwa zbiory indeksów:
I_1 to zbiór indeksów dla których $y_i = 1$, bez indeksu g,
I_2 to zbiór indeksów dla których y_i = -1, bez indeksu g.
Zapisując zgodnie z powyższym:

$$\sum_{\substack{i=1,\\ i\in I_1,\\ i\neq g}}^{p} \alpha_i - \sum_{\substack{i=1,\\ i\in I_2,\\ i\neq g}}^{p} \alpha_i = A - y_g \alpha_g \tag{3.8}$$

Gdy $A - y_g\alpha_g > 0$ to istnieje rozwiązanie (3.8), gdy

$$\max \left(\sum_{\substack{i=1,\\ i\in I_1,\\ i\neq g}}^{p} \alpha_i - \sum_{\substack{i=1,\\ i\in I_2,\\ i\neq g}}^{p} \alpha_i \right) > A - y_g \alpha_g$$

$$\max \sum_{\substack{i=1,\\ i\in I_1,\\ i\neq g}}^{p} C > A - y_g \alpha_g$$

$$|I_1|\, C > A - y_g \alpha_g \tag{3.9}$$

Dla przypadku, gdy $A - y_g\alpha_g < 0$ to istnieje rozwiązanie (3.8), gdy

$$\min \left(\sum_{\substack{i=1,\\ i\in I_1,\\ i\neq g}}^{p} \alpha_i - \sum_{\substack{i=1,\\ i\in I_2,\\ i\neq g}}^{p} \alpha_i \right) < A - y_g \alpha_g$$

Po przekształceniach

$$-|I_2|\, C < A - y_g \alpha_g \tag{3.10}$$

Dla przypadku, gdy $A - y_g\alpha_g = 0$ istnieje rozwiązanie (3.8), tym rozwiązaniem jest wyzerowanie wszystkich parametrów z grupy I_1 i I_2.

Stosując powyższe postępowanie dla każdej ściany widoku otrzymuje się zbiór ścian widoku przecinających hiperpłaszczyznę.

Można zastosować zamiast powyższego analitycznego algorytmu metodę geometryczną:

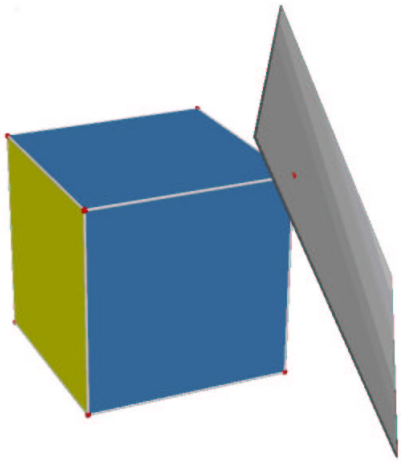
Jeśli wszystkie wierzchołki należące do ściany leżą po jednej stronie hiperpłaszczyzny to nie przecina ona tej ściany, jeśli wierzchołki leżą po obu stronach hiperpłaszczyzny, lub jeśli któryś z wierzchołków należy do hiperpłaszczyzny to ściana przecina hiperpłaszczyznę.

Jednakże metoda geometryczna nadaje się dla niewielkich podproblemów ASO, ponieważ: liczba wierzchołków ściany $q-1$ wymiarowej wynosi 2^{q-1}, maksymalna liczba ścian widoku wynosi q, a zatem liczba nierówności koniecznych do sprawdzenia może wynosić $q2^{q-1}$, co dla duzych q staje się znaczącym kosztem.

Możliwości zmniejszenia liczby ścian widoku

Udoskonaleniem mającym na celu zmniejszenie liczby ścian koniecznych do sprawdzenia, czy posiadają rozwiązanie optymalne jest rozpatrzenie przypadku gdy hiperpłaszczyzna przecina daną scianę, ale wszystkie punkty wspólne tego przecięcia należą również do innej ściany, która została już wcześniej rozpatrzona.

Dla przypadku trójwymiarowego sytuacja została przedstawiona na rysunku (Rys. 3.8).



Rysunek 3.8. Rysunek przedstawia sytuację, gdy płaszczyzna ma tylko jeden punkt wspólny z sześcianem.

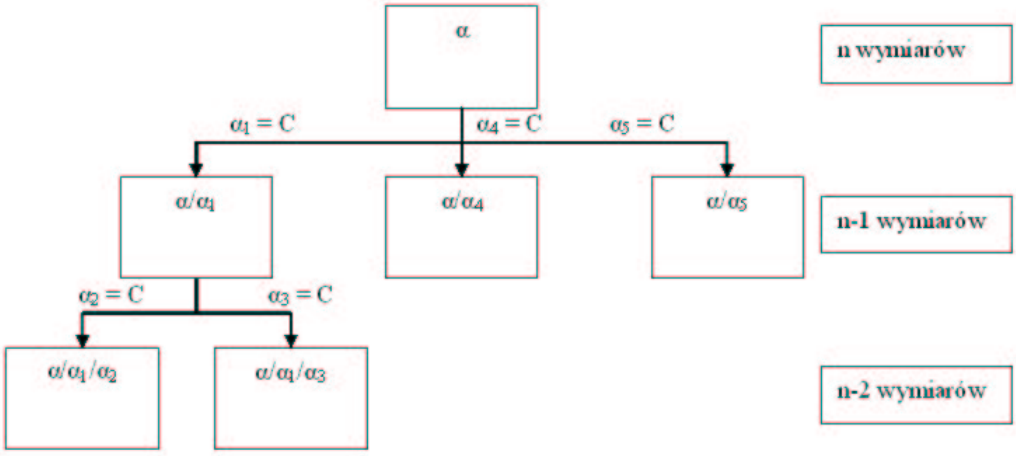
W tym wypadku rozpatrzenie jednego podproblemu, odpowiadającego jednej ze ścian widoku jest wystarczające i nie jest konieczne rozpatrywanie pozostałych ścian.

Pewną grupą udoskonaleń mogących przyczynić się znacząco do zmniejszenia liczby ścian widoku koniecznych do rozpatrzenia są udoskonalenia, które dają wynik przybliżony. Polegałyby one na tym, iż w kazdym kroku dekompozycji wewnętrznej wybierana byłaby mała liczba podproblemów, dla których z największym prawdopodobieństwiem istnieje rozwiązanie optymalne dla jednego z nich. Wtenczas znalezione rozwiązanie, jeśli byłoby lepsze od poprzedniego byłoby akceptowane, jednakże nie byłoby w każdym przypadku optymalne. Jako, że metoda ASO zakłada rozwiązanie dokładne podproblemu (PO 2.1.1) powyższe udoskonalenia nie będą rozwijane w tej pracy.

3.1.2. Model drzewiasty podziału podproblemu SVM na podproblemy dekompozycji wewnętrznej

W modelu geometrycznym widać było, że twory geometryczne q wymiarowe zostają sprowadzone do tworów $q-1$ wymiarowych, co odpowiada sprowadzaniu problemów q wymiarowych do problemów $q-1$ wymiarowych. Model drzewiasty ukazuje istotę sprowadzania podproblemów do mniejszych podproblemów. Został on przedstawiony na rysunku (Rys. 3.9). Model drzewiasty pokazuje, iż metoda ASO polega na przeszukiwaniu drzewa potencjalnych rozwiązań i wyborze najlepszego znalezionego. Każdemu potencjalnemu rozwiązaniu odpowiada ścieżka od korzenia do liścia w drzewie, w którym węzłom odpowiadają podproblemy ASO, zaś krawędziom odpowiadają ściany hipersześcianu, które jednocześnie należą do widoku i przecinają hiperpłaszczyznę odpowiadającą warunkowi równościowemu podproblemu ASO.

Liście mają minimalny wymiar dwa, wtenczas istnieje bezpośrednie rozwiązanie analityczne podproblemu za pomocą wzorów SMO.



Rysunek 3.9. Rysunek przedstawia model drzewiasty algorytmu dekompozycji wewnętrznej ASO.

3.1.3. Algorytm podziału podproblemu SVM na podproblemy dekompozycji wewnętrznej

Po wybraniu parametrów za pomocą algorytmu dekompozycji zewnętrznej wyznaczane jest optymalne rozwiązanie problemu (PO 3.1.2) dla q wybranych parametrów, zgodnie ze wzorami 3.1.

Następnie wyznaczany jest zbiór parametrów spośród wybranych parametrów takich, które nie spełniają warunku nierównościowego podproblemu ASO dla q wybranych parametrów. Jeśli nie ma takowych to znalezione rozwiązanie problemu (PO 3.1.2) jest jednocześnie rozwiązaniem podproblemu ASO. Jeśli istnieją takowe, to każdy taki parametr jest zaokrąglany do najbliższej liczby ze zbioru $G = \{0, C\}$. Zbiór tych zaokrąglanych parametrów zostanie oznaczony jako: C_q. Następnie zbiór C_q ograniczany jest do zbioru parametrów C'_q dla których istnieje co najmniej jeden zestaw pozostałych $q-1$ parametrów, tak aby był spełniony warunek równościowy podproblemu ASO. Co najmniej jeden z tych parametrów ma wartość optymalną. Z tego względu dla

każdego parametru ze zbioru C'_q poszukiwane są optymalne wartości pozostałych $q-1$ parametrów.

Sposób poszukiwania optymalnych wartości pozostałych $q-1$ parametrów.

Każdemu wybranemu parametrowi ze zbioru C'_q zostaje przyporządkowany dokładnie jeden podproblem składający się z pozostałych $q-1$ parametrów. Rozwiązanie tego podproblemu stanowi zbiór poszukiwanych optymalnych wartości parametrów.

Moc zbioru C'_q zostanie oznaczona jako $\left|C'_q\right|$. Podproblemy odpowiadające parametrom ze zbioru C'_q zostaną oznaczone jako P_q^i dla $i \in \left\{1..\left|C'_q\right|\right\}$. W celu rozwiązania podproblemu P_q^i rozwiązywany jest problem optymalizacyjny (PO 3.1.2) dla tego podproblemu ASO. A następnie stosowana jest analogiczna procedura co poprzednio. To znaczy jeśli wszystkie znalezione parametry spełniają warunek nierównościowy to znalezione rozwiązanie (PO 3.1.2) jest rozwiązaniem podproblemu P_q^i. Jeśli zaś niektóre parametry nie spełniają warunku nierównościowego podproblemu ASO dla $q-1$ wybranych parametrów, zbiór tych parametrów będzie oznaczony jako C_{q-1}, to zbiór C_{q-1} ograniczany jest do zbioru parametrów C'_{q-1} dla których istnieje co najmniej jeden zestaw pozostałych $q-2$ parametrów, tak aby był spełniony warunek równościowy podproblemu ASO dla $q-1$ parametrów. Dla każdego parametru ze zbioru C'_{q-1} wyznaczana jest grupa podproblemów P_{q-1}^i. Jeden z tych parametrów ma wartość optymalną. Rozwiązanie podproblemów P_{q-1}^i posłuży do znalezienia pozostałych parametrów i tym samym zbioru możliwych parametrów. Rozwiązaniem podproblemu P_q^i jest najlepszy zbiór parametrów z wcześniej wyznaczonych.

Podproblemy P_{q-1}^i rozwiązywane są w analogiczny sposób, a w kolejnych krokach podproblemy P_{q-2}^i,..,.P_{q-j}^i... itd..

Warunek stopu Podproblem nie jest rozwiązywany dalej za pomocą podproblemów, jeśli wszystkie znalezione parametry problemu optymalizacyjnego (PO 3.1.2) dla tego podproblemu będą spełniały warunek nierównościowy, czyli gdy $|C_{q-j}| = 0$, bądź gdy podproblem będzie składał się z dwóch parametrów. Wtenczas taki podproblem rozwiązywany jest za pomocą znanej metody SMO. Po spełnieniu jednego z dwóch powyższych warunków, co jest równoważne wyznaczeniu możliwego zestawu parametrów, a więc jednego z możliwych rozwiązań, rozwiązanie to jest porównywane z najlepszym dotychczas znalezionym. Kryterium porównawczym jest wartość funkcji celu.

Powyższy schemat postępowania został przedstawiony w algorytmie 4.

Liczba podproblemów algorytmu rekurencyjnego dekompozycji wewnętrznej stanowi kluczowy czynnik decydujący o szybkości algorytmu. W następnym punkcie zostanie przedstawione oszacowanie liczby podproblemów ASO.

3.1.4. Oszacowanie liczby podproblemów dekompozycji wewnętrznej

Dany jest podproblem SVM p wymiarowy. Liczba podproblemów $p-1$ wymiarowych wynosi maksymalnie p. Dla każdego podproblemu postępowanie jest podobne, a więc występuje rekurencja, która kończy się w wersji pesymistycznej na przypadku dwuwymiarowym, dla którego znane jest rozwiązanie analityczne.

Algorithm 4 Algorytm dekompozycji wewnętrznej metody ASO.

```
Procedure Decomposition();
begin
    ChooseActiveParameters;
    InteriorDecomposing(allActiveParameters);
end.
InteriorDecomposing(chosenParameters: array of Integer)
begin
    if (size(chosenParameters) = 2)
    begin
        ComputeSMO(chosenParameters);
        CompareActualResultWithBest(actualResult);
        return;
    end;
    else
    begin
        actualResult := ComputePO4problem(chosenParameters);
        CiSet := FindCiSet(chosenParameters, actualResult);
        if (CiSet is empty)
        begin
            CompareActualResultWithBest(actualResult);
            return;
        end;
        else
        begin
            CiPriSet := FindCiPriSet(chosenParameters, actualResult);
            for i:=1 to size(CiPriSet) do
                InteriorDecomposing(chosenParametersWithout_ith);
        end;
    end;
end;
```

A zatem w sumie w wypadku pesymistycznym liczba podproblemów będzie równa p!/2.

Razem z podproblemem dla p parametrów: $p!/2 + 1$

Dla małych p liczba podproblemów jest do zaakceptowania jednak, dla większych p liczba podproblemów powoduje, że ta metoda nie może konkurować z metodami numerycznymi.

Przykładowo dla $p = 3$ otrzymuje się maksymalnie 4 podproblemy do rozpatrzenia.

Dla $p = 4$ otrzymuje się w najgorszym razie 13 podproblemów.

Jak się okaże w praktyce wersja pesymistyczna nie jest osiągana, i otrzymuje się dużo mniej podproblemów niż we wzorze $p!/2 + 1$.

3.1.5. Testy liczby podproblemów dekompozycji wewnętrznej

Liczba podproblemów algorytmu ASO w wersji pesymistycznej wynosi $p!/2 + 1$.

Poniżej zostaną zrobione testy, które wykażą ile w praktyce wynosi liczba podproblemów.

Test 3.1.1. Testy zostały wykonane z założeniami takimi jak w testach: (Test 5.1.1), (Test 5.1.2) i (Test 5.1.3) z rozdziału Rezultaty. Wyniki zostały przedstawione w tabeli 3.1.

Test/Liczba parametrów/(p!/2 + 1)	*3/4*	*4/13*	*5/61*
jądro RBF	2.92 (73%)	10.2 (78%)	38.99 (63.92%)
jądro liniowe	3.14 (78,5%)	8.37 (64,38%)	24.24 (39.73%)
jądro wielomianowe	2.73 (68,25%)	8,63 (66.38%)	34,53 (56.61%)

Tablica 3.1. Stosunek liczby wszystkich podproblemów ASO do liczby podproblemów SVM - średnia liczba podproblemów ASO dla jednego podproblemu SVM.

Z powyższego testu wynika, że rzeczywista liczba problemów może być nawet o połowę mniejsza niż pesymistycznie zakładana. Zauważyć również można dobrą skalowalność liczby podproblemów na większą liczbę parametrów. Im więcej parametrów tym procentowa liczba podproblemów w stosunku do pesymistycznej jest mniejsza.

3.2. Porównanie metody Analitycznej Optymalizacji Sekwencyjnej z metodami numerycznymi

3.2.1. Metoda punktu wewnętrznego

Metoda punktu wewnętrznego zastosowana do rozwiązywania podproblemów SVM składa się z dwóch etapów. W etapie pierwszym zostaje włączony warunek nierównościowy do funkcji celu W za pomocą metody barierowej. W drugim etapie powstały problem optymalizacyjny maksymalizacji funkcji celu z jednym warunkiem równościowym rozwiązywany jest za pomocą metody Newtona.

Metoda barierowa Włączenie warunku nierównościowego do maksymalizowanej funkcji realizowane jest za pomocą funkcji $I_- : R \to R$ takiej, że:

$$I_{-}(u) = \begin{cases} 0 & u \leq 0 \\ \infty & u > 0 \end{cases}$$

.

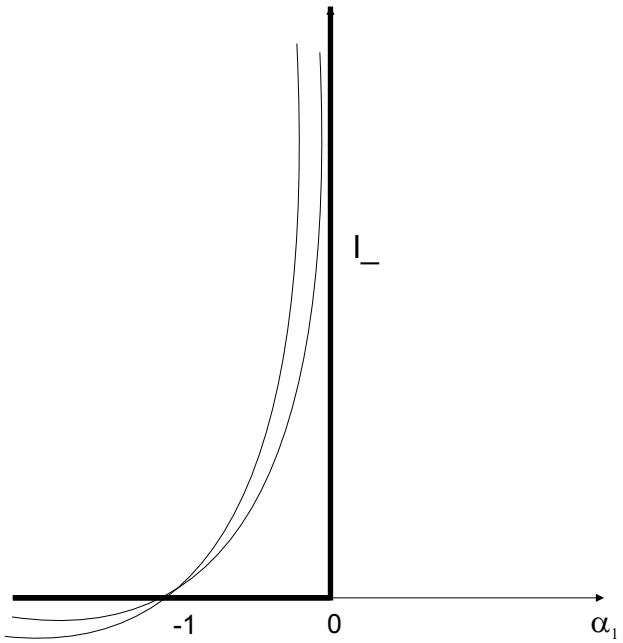
Funkcja celu przybiera postać:

$$W_2(\vec{\alpha}) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} y_i y_j \alpha_i \alpha_j K(x_i, x_j) + \sum_{i=1}^{l} I_{-}(f_i(x)),$$

Funkcja W_2 nie jest w ogólnym przypadku różniczkowalna, a więc nie można zastosować bezpośrednio metody Newtona. Rozwiązaniem jest aproksymacja włączonego członu funkcji W_2. Podstawową ideą metody barierowej jest aproksymacja funkcji I_{-} funkcją logarytmiczną:

$$\widehat{I_{-}}(u) = -\left(\frac{1}{t}\right) \log(-u),$$

gdzie $t > 0$ jest parametrem ustalającym dokładność aproksymacji (Rys. 3.10). Wtenczas taka bariera zwie się *barierą logarytmiczną*.



Rysunek 3.10. Na wykresie została przedstawiona funkcja I_{-} oraz jej dwa przybliżenia dla różnych t.

Problem optymalizacyjny przybiera postać:

Problem optymalizacyjny 3.2.1. *Maksymalizacja funkcji*

$$\widehat{W_2}(\vec{\alpha}) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} y_i y_j \alpha_i \alpha_j K(x_i, x_j) + \sum_{i=1}^{l} - \left(\frac{1}{t}\right) \log(-\alpha_i),$$

przy ograniczeniu:

$$\sum_{i=1}^{p} y_i \alpha_i = A$$

dla $i \in \{1..p\}$,

W drugim kroku problem (PO 3.2.1) rozwiązywany jest za pomocą przybliżonej metody Newtona, a więc funkcja $\widehat{W_2}$ najpierw aproksymowana jest za pomocą wzoru Taylora z drugimi pochodnymi, a następnie znajdowane jest rozwiązanie analityczne problemu optymalizacji funkcji kwadratowej wklęsłej z jednym warunkiem liniowym.

Dla dużego t, a więc gdy przybliżenie jest dokładne, rozwiązanie problemu (PO 3.2.1) jest trudne numerycznie do wyliczenia za pomocą metody Newtona, ponieważ Hesjan zmienia się w szybkim tempie blisko końców dziedziny. Dlatego stosuje się sekwencję problemów (PO 3.2.1), w każdym kroku zwiększając wartość parametru t, a tym samym dokładność aproksymacji, i startując metodę Newtona z rozwiązania znalezionego w poprzednim kroku.

Zarówno metoda ASO jak i metoda punktu wewnętrznego wykorzystują rozwiązanie analityczne pewnego problemu optymalizacyjnego z jedym warunkiem równościowym. Metoda ASO tak wykorzystuje warunek nierównościowy, że w dalszym ciągu wyznaczane jest rozwiązanie dokładne, natomiast w metodzie punktu wewnętrznego warunek nierównościowy powoduje pogorszenie dokładności rozwiązania, z dwojakiego powodu: aproksymacji włączonego członu do funkcji W, oraz aproksymacji funkcji W_2 za pomocą wzoru Taylora z drugimi pochodnymi. I dlatego w celu otrzymania rozwiązania dokładnego z ustalonym błędem konieczne jest postępowanie iteracyjne zewnętrzne ze względu na parametr t odpowiadający za dokładność przybliżenia funkcji $I\_$, oraz wewnętrzne ze względu na niedokładność aproksymacji funkcji W_2. W metodzie ASO nie jest konieczne postępowanie iteracyjne ponieważ algorytm wyznacza teoretycznie rozwiązanie dokładne.

Dla małej ilości parametrów aktywnych porównywanie złożoności obliczeniowej powinno opierać się na dokładnym wyliczeniu liczby operacji koniecznych do wyznaczenia. Zadanie jest utrudnione ze względu na to, że metoda punktu wewnętrznego jest metodą iteracyjną z dwoma iteratorami.

Koszty stałe w każdej iteracji wewnętrznej są kosztami związanymi z aproksymacją funkcji za pomocą wzoru Taylora, a zatem obliczenia pierwszych i drugich pochodnych, a także rozwiązania układu równań liniowych ($O(n^3)$). W metodzie ASO dla każdego podproblemu ASO kosztem stałym jest rozwiązanie problemu (PO 3.1.2), o złożoności ($O(n^3)$). Liczba koniecznych do rozpatrzenia podproblemów jest wyznaczona teoretycznie tylko dla przypadku pesymistycznego, który w praktyce się nie zdarza. Dlatego porównanie liczby podproblemów ASO z liczbami iteracji metody punktu wewnętrznego jest trudne do wykonania. Dla małej liczby parametrów, gdzie liczba podproblemów ASO jest mała, metoda ASO ma szansę być szybszą od metody punktu wewnętrznego. Natomiast dla większej liczby parametrów metoda ASO ze względu na wysoką liczbę podproblemów staje się niepraktyczna.

3.2.2. Metody gradientowe

Metody gradientowe zastosowane do rozwiązywania podproblemów SVM polegają na przybliżaniu rozwiązania według rosnącego gradientu. Jedną z podstawowych metod z tej grupy jest metoda największego wzrostu, polegająca na modyfikacji jednego tylko parametru w danym kroku optymalizacyjnym. Metody gradientowe bazują na analizie lokalnej, gradientu w aktualnie rozpatrywanym punkcie. Podczas procesu iteracyjnego kolejne przybliżenia leżą w obrębie zarówno hipersześcianu, jak również hiperpłaszczyzny. Natomiast metoda Analitycznej Optymalizacji Sekwencyjnej podczas wyznaczania rozwiązania dokładnego bazuje na analizie globalnej. Wykorzystywane są informacje o maksimum funkcji celu W z włączonym warunkiem równościowym, które to może leżeć poza hipersześcianem.

3.3. Heurystyka

Heurystyka SVM powinna polegać na takim wyborze parametrów do zbioru aktywnego podczas każdej iteracji, aby ilość iteracji była możliwie jak najmniejsza.

Podstawowym podejściem na którym opiera się większość heurystyk SVM jest możliwie jak największa maksymalizacja funkcji celu w każdym kroku iteracyjnym.

W tym celu wyznaczany jest gradient funkcji celu i wybierana jest grupa parametrów, dla których gradient jest największy, ale tak aby zmiana wartości parametrów była możliwa ze względu na warunek nierównościowy, oraz tak aby warunek równościowy był spełniony.

Aby móc rozpatrywać warunki nierównościowe i związane z nimi warunki możliwości optymalizacji, konieczne jest włączenie warunku równościowego do funkcji celu. Funkcja z włączonym warunkiem równościowym zostanie oznaczona jako W_2.

Pochodna takiej funkcji została wyprowadzona w dodatku B.1 i dla wszystkich wektorów wynosi:

$$\frac{\partial W_2\left(\alpha_1^{\text{new}}, \alpha_2^{\text{new}}, ..., \alpha_{l-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(\sum_{i=1}^{l-1} y_i \alpha_i^{\text{new}} \kappa_{ilk} - \sum_{i=1}^{l-1} y_i \alpha_i \kappa_{ilk} - E_k + E_l \right) \tag{3.11}$$

Pochodna w aktualnie rozpatrywanym punkcie wynosi:

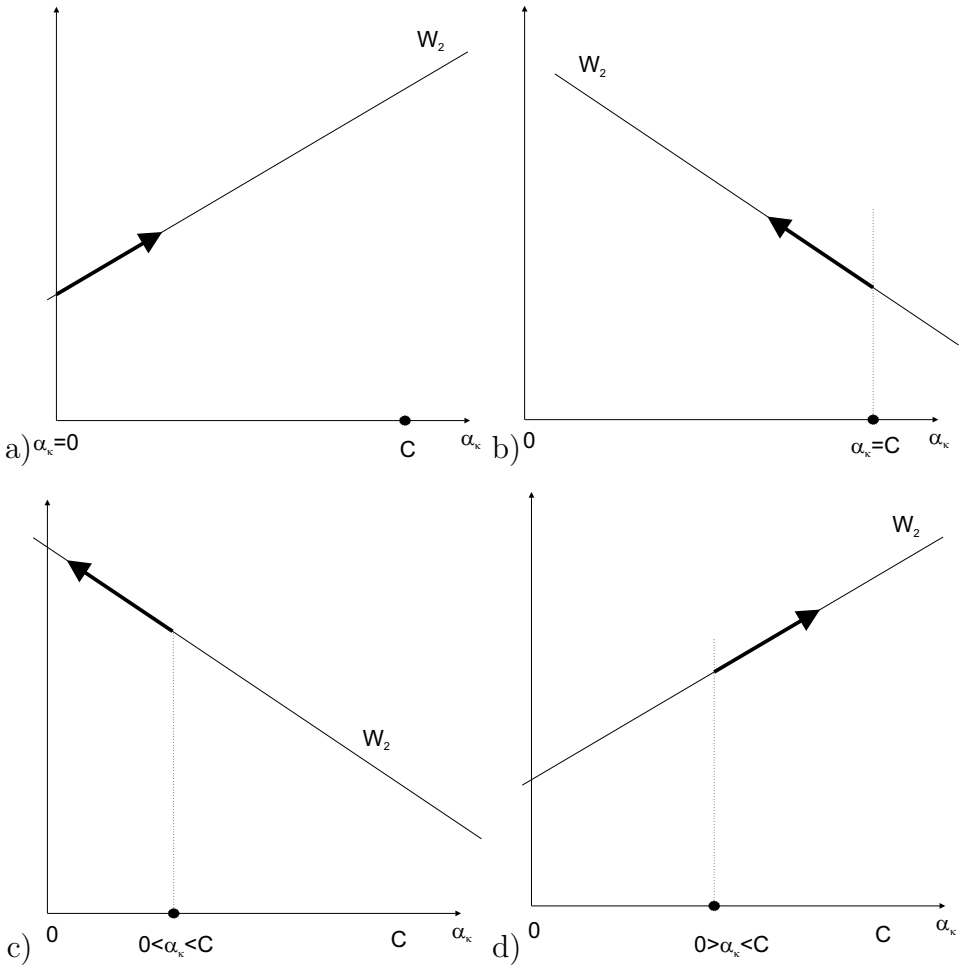
$$\frac{\partial W_2\left(\alpha_1, \alpha_2, ..., \alpha_{l-1}\right)}{\alpha_k} = y_k \left(E_l - E_k\right). \tag{3.12}$$

Aby długość wektora gradientu była maksymalna, wybierane są parametry, dla których wartość pochodnych cząstkowych funkcji celu problemu SVM z włączonym warunkiem równościowym jest maksymalna. Dotyczy to nie tylko l-1 wybranych parametrów, lecz wszystkich l parametrów, ponieważ wybór l-tego parametru jest dowolny, dlatego, że kolejność rozpatrywania parametrów jest dowolna.

Kolejnym aspektem jest możliwość zmiany wartości parametrów w kierunku wzrostu funkcji celu W_2, bowiem parametry obłożone są warunkiem nierównościowym. Aby było możliwe zwiększenie wartości funkcji W_2 przy zmianie parametru musi być spełniony następujący warunek optymalizacyjny:

$$\text{(W1)} \begin{cases} W_2'(\alpha_k) > 0 \; dla \; \alpha_k = 0 \\ W_2'(\alpha_k) < 0 \; dla \; \alpha_k = C \\ W_2'(\alpha_k) \neq 0 \; dla \; 0 < \alpha_k < C \end{cases}$$

Sens powyższego spostrzeżenia przedstawia rysunek (Rys. 3.11):



Rysunek 3.11. Rysunek przedstawia sytuację, gdy w przypadku a) parametr ma wartość 0 i przy zwiększeniu jego wartości rośnie wartość funkcji W_2, w przypadku b) parametr ma wartość C i przy zmniejszeniu jego wartośći rośnie wartość funkcji W_2, w przypadku c) parametr nie ma wartości granicznej i przy zmniejszeniu jego wartości rośnie wartość funkcji W_2, w przypadku d) parametr nie ma wartości granicznej i przy zwiększeniu jego wartości rośnie wartość funkcji W_2.

Po podstawieniu wartości pochodnej do warunku W1 otrzymuje się:

$$\begin{cases} y_k(E_l - E_k) > 0 \text{ dla } \alpha_\text{k} = 0 \\ y_k(E_l - E_k) < 0 \text{ dla } \alpha_\text{k} = \text{C} \\ y_k(E_l - E_k) \neq 0 \text{ dla } 0 < \alpha_\text{k} < \text{C} \end{cases}$$

Z drugiej strony warunek, który powinien być spełniony przez rozwiązanie optymalne, odpowiada warunkowi dla którego nie jest możliwa żadna optymalizacja, a zatem:

$$\text{KKT}\begin{cases} y_k\left(E_l - E_k\right) \leq 0 \text{ dla } \alpha_\text{k} = 0 \\ y_k\left(E_l - E_k\right) \geq 0 \text{ dla } \alpha_\text{k} = \text{C} \\ y_k\left(E_l - E_k\right) = 0 \text{ dla } 0 < \alpha_\text{k} < \text{C} \end{cases} \tag{3.13}$$

Reasumując heurystyka na tym etapie polega na wyborze parametrów takich, że gradient funkcji W_2 jest maksymalny, ale z zastrzeżeniem warunku W1, który jest warunkiem koniecznym możliwości zmiany wartości parametru w kierunku wzrostu funkcji celu:

$$\begin{cases} \max W_2'\left(\alpha_k\right) \text{ dla } \alpha_\text{k} = 0 \\ \min W_2'\left(\alpha_k\right) \text{ dla } \alpha_\text{k} = \text{C} \\ \max\left(\left|W_2'\left(\alpha_k\right)\right|\right) \text{ dla } 0 < \alpha_\text{k} < \text{C} \end{cases}$$

Włączając ostatni człon warunku do wcześniejszych otrzymuje się:

$$\text{H1}\begin{cases} \max W_2'\left(\alpha_k\right), \; gdy \; \left(\alpha_k = 0\right) \vee \left(0 < \alpha_k < C \wedge W_2'\left(\alpha_k\right) > 0\right) \\ \min W_2'\left(\alpha_k\right), gdy \; \left(\alpha_k = C\right) \vee \left(0 < \alpha_k < C \wedge W_2'\left(\alpha_k\right) < 0\right) \end{cases}$$

W tej heurystyce parametry niegraniczne zostały przyporządkowane do grupy parametrów zerowych lub równych C.

W powyższej heurystyce została wzięta pod uwagę możliwość, że liczba parametrów, które spełniają W1 jest niedostateczna, wtedy brakujące parametry wybierane są z grupy parametrów nie spełniających W1, co oznacza praktyce, że nie będą one optymalizowane.

Warunkiem, który musi być spełniony przez dotychczasowe wartości parametrów jest warunek równościowy:

$$\text{Z: } \sum_{i=1}^{p} y_i \alpha_i = A$$

Po optymalizacji również musi być spełniony, a zatem:

$$\sum_{i=1}^{p} y_i\left(\alpha_i + \Delta\alpha_i\right) = A$$

$$\sum_{i=1}^{p} y_i \alpha_i + y_i \Delta\alpha_i = A$$

Podstawiając założenie Z otrzymuje się:

$$\sum_{i=1}^{p} y_i \Delta\alpha_i = 0 \tag{3.14}$$

Poniżej zostanie przeanalizowana kwestia możliwości spełnienia tego warunku. Rzeczywiste zmiany parametrów, które nastąpią po optymalizacji nie są znane. Dlatego muszą zostać powzięte pewne założenia dotyczące zmiany parametrów. Założeniem może być to, aby moduł zmiany każdego parametru był ten sam, czyli że parametry zmieniają się równomiernie. Wtenczas:

$$|\Delta\alpha| \sum_{i=1}^{p} y_i sgn\left(\Delta\alpha_i\right) = 0$$

Aby powyższe równanie było spełnione przy założeniu, że nastąpiła zmiana parametrów to:

$$\sum_{i=1}^{p} y_i sgn\left(\Delta\alpha_i\right) = 0$$

$$\sum_{i=1}^{p} sgn\left(y_i \Delta\alpha_i\right) = 0 \tag{3.15}$$

Dalej należy przeanalizować znak wyrażenia $y_i\Delta\alpha_i$. Powyższa równość będzie spełniona, gdy będzie tyle samo wyrażeń $y_i\Delta\alpha_i$ o znaku ujemnym jak i dodatnim.

$$y_i\Delta\alpha_i > 0 \Leftrightarrow (y_i = 1 \wedge \Delta\alpha_i > 0) \vee (y_i = -1 \wedge \Delta\alpha_i < 0)$$

$$\Delta\alpha_i > 0 \Leftrightarrow \alpha_i < C$$

$$\Delta\alpha_i < 0 \Leftrightarrow \alpha_i > 0$$

A zatem, gdy

$$\text{G1: } (y_i = 1 \wedge \alpha_i < C) \vee (y_i = -1 \wedge \alpha_i > 0)$$

oraz

$$y_i\Delta\alpha_i < 0 \Leftrightarrow (y_i = 1 \wedge \Delta\alpha_i < 0) \vee (y_i = -1 \wedge \Delta\alpha_i > 0)$$

Czyli gdy:

$$\text{G2: } (y_i = 1 \wedge \alpha_i > 0) \vee (y_i = -1 \wedge \alpha_i < C)$$

Powyżej został wyznaczony podział parametrów na dwie grupy G1 i G2, dla pierwszej $y_i\Delta\alpha_i > 0$, a dla drugiej $y_i\Delta\alpha_i < 0$. Warunkiem, aby była spełniona równość (3.15) jest równoliczność zbiorów G1 i G2:

$$|G1| = |G2|\,. \tag{3.16}$$

Można również zauważyć, że parametry niegraniczne należą do obu z tych grup.

Warunek (3.15) należy następnie połączyć z wyznaczoną wcześniej heurystyką H1. Dla heurystyki H1 parametry niegraniczne zostały jednoznacznie przyporządkowane do grupy parametrów zerowych lub równych C. A zatem ten podział można włączyć do powyższych rozważań na temat warunku równościowego, otrzymując:

$$\begin{aligned} &\text{G1}: (\mathrm{y_i} = 1 \wedge \alpha_\mathrm{i} = 0) \vee (\mathrm{y_i} = -1 \wedge \alpha_\mathrm{i} = \mathrm{C}) \\ &\text{G2}: (\mathrm{y_i} = 1 \wedge \alpha_\mathrm{i} = \mathrm{C}) \vee (\mathrm{y_i} = -1 \wedge \alpha_\mathrm{i} = 0) \end{aligned} \tag{3.17}$$

Dalej łącząc heurystykę H1 z warunkiem (3.15) i przyjmując jego absolutne spełnienie otrzymuje się następującą heurystykę:

Heurystyka H2:

Wyznacz $|G1|$ maksymalnych parametrów z grupy G1, a następnie $|G1|$ maksymalnych parametrów z grupy G2.

Z (3.16) wynika, że ta heurystyka jest poprawna dla parzystej liczby parametrów.

Heurystyka H2 została zaproponowana przez Joachima w [11] i zaimplementowana w oprogramowaniu SVM Light. Heurystyka ta znana jest pod nazwą metody dopasowanego kierunku (ang. *feasible solution*) Zoutendijks'a [26].

W rozważaniu na temat warunku równościowego zostało założone, iż moduł wszystkich parametrów jest taki sam. W niektórych przypadkach jest to niemożliwe do spełnienia. Np. gdy jeden z parametrów jest niegraniczny i ma wartość bliską C, wtedy zmiana w kierunku rosnącego gradientu jest zmianą parametru na wartość C. Wtenczas zmiana parametru jest bardzo mała i aby założenie o tym, że wszystkie zmiany mają mieć tą samą wartość bezwzględną znaczyłoby, że muszą być bardzo małe, podobnie jak ta zmiana. Bardzo mała zmiana parametrów zazwyczaj nie prowadziłaby do znalezienia optymalnego rozwiązania.

Jako, że zmiana parametrów ma prowadzić do optymalnego rozwiązania, założeniem lepiej opisującym sytuację jest założenie, że wszystkie parametry zmienią się maksymalnie w kierunku gradientu, jednakże tak aby był spełniony warunek równościowy.

Gdy nie występują parametry graniczne sytuacja ta sprowadza się do wcześniej przedstawionej. Wszystkie parametry mają wartość bezwzględną zmiany taką samą, równą C.

Gdy zaś występują parametry niegraniczne spełnienie (3.14) już nie jest takie proste.

Warunek równościowy przyjmuje następującą postać:

$$\sum_{l=1}^{l_1-g_1} C + \sum_{l=1}^{g_1} \Delta\alpha_l - \sum_{k=1}^{k_1-g_2} C - \sum_{k=1}^{g_2} \Delta\alpha_k = 0 \tag{3.18}$$

gdzie

$$l_1 = |G1|$$

$$k_1 = |G2|$$

g_1 to liczba parametrów granicznych z grupy G1, a g_2 to liczba parametrów granicznych z grupy G2.

Przy założeniu, że zmiany parametrów niegranicznych przyjmują tą samą wartość, równanie (3.18) jest spełnione, np. gdy liczba parametrów niegranicznych w grupie G1 jest równa liczbie parametrów niegranicznych w grupie G2. Istnieje również możliwość spełnienia (3.18) dla różnej liczby parametrów niegranicznych w grupie G1 i G2, jednakże przez założenie początkowe, że parametry niegraniczne przyjmują tą samą wartość nie będzie rozpatrywany. Reasumując spełnienie (3.18) polega na tym aby w grupie parametrów G1 i G2, wybranych wg (3.16), pojawił się warunek równoliczności parametrów niegranicznych. Przy absolutnym spełnieniu tego warunku, kwestia włączenia warunku maksymalizacji gradientu może być rozwiązana w następujący sposób:

Heurystyka H3 przypadek parzysty

1. Wyznacz $|G1|$ maksymalnych parametrów z grupy G1, a następnie wyznacz tyle samo maksymalnych elementów niegranicznych z grupy G2, co w grupie G1. Jeśli nie jest to możliwe przejdź do punkt 2, jeśli jest, wyznacz pozostałą ilość parametrów granicznych.
2. Wyznacz $|G1|$ maksymalnych parametrów z grupy G2, a następnie wyznacz tyle samo maksymalnych elementów niegranicznych z grupy G1, co w grupie G2. Wyznacz pozostałą ilość parametrów granicznych.

Warto zaznaczyć, że jeśli nie uda się znaleźć zbioru parametrów w punkcie 1, to na pewno zostanie znaleziony w punkcie 2.

Zostały zrobione testy, w których jest pokazane w ilu przypadkach nowy wybór parametrów spowodował docelowy efekt, jakim jest zmniejszenie różnicy między zbiorami G1 i G2.

Test 3.3.1. Dla przypadku jądra RBF i zbioru treningowego (Rys. 5.1). Liczba wektorów 6000. Wyniki zostały przedstawione w tabeli 3.2.

	$\sum_{i=1}^{p} y_i \Delta\alpha_i$ *dla przypadku heurystyki H3*	$\sum_{i=1}^{p} y_i \Delta\alpha_i$ *dla przypadku heurystyki Joachima*
2	517	47
4	291	91
6	313	91

Tablica 3.2. Test różnicy między zbiorami G1 i G2 dla heurystyki H3

Z powyższego testu wynika, że w większości wypadków nowa heurystyka działa tak jak zakładano, czyli zmniejsza wartość sumy $\sum_{i=1}^{p} y_i \Delta\alpha_i$. Aby przekonać się czy połączenie z heurystyką H1 przyniesie dobre efekty zostały wykonane następujące testy:

Test 3.3.2. Testy heurystyki H3 dla zbioru treningowego (Rys. 5.1). Jądro RBF. Liczba wektorów 6000. Wyniki zostały przedstawione w tabeli 3.3.

	Heurystyka H3	*Heurystyka H2 (Joachima)*
2	2639	2744
4	1406	1656
6	1014	1256

Tablica 3.3. Test heurystyki H3 dla jądra RBF.

Test 3.3.3. Dla zbioru treningowego (Rys. 5.3). Dla jądra wielomianowego. Liczba wektorów 11000. Wyniki zostały przedstawione w tabeli 3.4.

	Heurystyka H3	*Heurystyka H2 (Joachima)*
2	8464	9730
4	5280	5400

Tablica 3.4. Test heurystyki H3 dla jądra wielomianowego

Testy pokazują, iż nowa heurystyka jest lepsza od heurystyki H2.

Powyższa heurystyka posiada tą cechę, że liczba parametrów niegranicznych utrzymywana jest na mniejszym poziomie, niż w heurystyce Joachima. Liczba parametrów niegranicznych stanowi istotny czynnik zbieżności heurystyki, ponieważ wpływa bezpośrednio na dokładność obliczeń.

Nieparzysta liczba parametrów w zbiorze aktywnym

Dla przypadku gdy liczba parametrów jest nieparzysta konieczna jest pewna modyfikacja heurystyki H3.

Konieczne jest aby równość (3.18) była spełniona, przy założeniu, że $l_1 \neq k_1$. Zakłada się, iż parametry niegraniczne przyjmują wartość C/2 oraz, że grupy G1 i G2 różnią się ilościowo tylko jednym parametrem.

Oznaczenia:

$$l_1 - g_1 = c_1$$

$$k_1 - g_2 = c_2$$

Przy tych założeniach konieczne jest aby było spełnione:

$$Cc_1 + \frac{C}{2}g_1 = Cc_2 + \frac{C}{2}g_2 \tag{3.19}$$

przy założeniu, że

$$c_1 + g_1 = c_2 + g_2 - 1$$

co po przekształceniu przyjmuje postać:

$$c_1 - c_2 = g_2 - g_1 - 1 \tag{3.20}$$

(3.19) można zapisać w postaci:

$$C\left(c_1 - c_2\right) + \frac{C}{2}\left(g_1 - g_2\right) = 0$$

Podstawiając (3.20) do powyższego równania otrzymuje się:

$$C\left(g_2 - g_1 - 1\right) + \frac{C}{2}\left(g_1 - g_2\right) = 0$$

Po przekształceniach:

$$g_2 = g_1 + 2 \tag{3.21}$$

Aby było spełnione (3.18) dla przypadku nieparzystego zostanie zaprezentowana następująca heurystyka:

Heurystyka H3, przypadek nieparzysty (H3NP)

1. Wyznacz $|G1|$ maksymalnych parametrów z grupy G1, a następnie wyznacz tyle samo maksymalnych elementów niegranicznych z grupy G2, co w grupie G1 plus dwa. Jeśli nie jest to możliwe, to wyznacz pozostałą ilość parametrów za pomocą parametrów granicznych i uruchom punkt 2.

2. Wyznacz $|G1|$ maksymalnych parametrów z grupy G2, a następnie wyznacz tyle samo maksymalnych elementów niegranicznych z grupy G1, co w grupie G2 plus dwa. Jeśli nie jest to możliwe, to wyznacz pozostałą ilość parametrów za pomocą parametrów granicznych.

Warto zaznaczyć, iż jeśli w obu krokach heurystyka nie będzie w stanie wyznaczyć parametrów zgodnie z (3.20), to brane są pod uwagę parametry z punktu 1.

W heurystyce (H3NP) nacisk kładziony jest na spełnienie warunku równościowego. Aby zadość uczynić heurystyce H1 wybierane są maksymalne elementy zarówno spośród elementów niegranicznych jak i granicznych.

Heurystyka Joachima jakkolwiek nie jest przystosowana do nieparzystej liczby parametrów, ale dla celów porównawczych można założyć, że zbiór G1 jest o jeden element mniejszy od zbioru G2 i wybierać elementy zgodnie z tą heurystyką.

Test 3.3.4. Testy heurystyki (H3NP) dla zbioru treningowego (Rys. 5.1). Jądro RBF. Liczba wektorów 6000. Wyniki zostały przedstawione w tabeli 3.5.

	Heurystyka (H3NP)	*Heurystyka H2 (Joachima)*
3	2365	2584
5	1155	1477

Tablica 3.5. Test heurystyki H3 (przypadek nieparzysty) dla jądra RBF.

Test 3.3.5. Dla zbioru treningowego (Rys. 5.3). Dla jądra wielomianowego. Liczba wektorów 11000. Wyniki zostały przedstawione w tabeli 3.6.

	Heurystyka (H3NP)	*Heurystyka H2 (Joachima)*
3	2547	9730
5	988	4836

Tablica 3.6. Test heurystyki H3 (przypadek nieparzysty) dla jądra wielomianowego

Można zauważyć, że heurystyka (H3NP) jest lepsza od heurystyki H2 we wszystkich testowanych przypadkach, ciekawą rzeczą jest drastyczne zmniejszenie liczby parametrów dla jądra wielomianowego.

Rozszerzenie heurystyki H3

W wielu rzeczywistych przypadkach, gdy ilość parametrów niegranicznych jest duża, pojawiają się problemy z dokładnością obliczeń. Wtenczas może się zdarzyć ze wybór parametrów zgodnie z heurystyką nie prowadzi do lepszych rozwiązań, a warunek stopu nie jest dalej spełniony. Rozwiązaniem problemu jest rozpatrzenie parametrów, które nie spełniają warunków KKT oraz parametrów niegranicznych. W obu tych grupach mogą występować parametry jeszcze niezoptymalizowane. Proponowana heurystyka jest następująca:

Wybierz pierwszy parametr taki, który nie spełnia warunku KKT lub jest parametrem niegranicznym, wybierany jest on ze zbioru G1, lub G2. Wybierz pozostałe parametry odpowiednio z grupy G2 lub G1 takie, które maksymalizują gradient.

Możliwe są udoskonalenia rozszerzenia heurystyki H3 zgodnie z istotą heurystyki H3, jak również rozpatrywana jest możliwość ponownego, dokładnego obliczenia warunków KKT.

Heurystyka programu BSVM Program BSVM, z którym będzie porównywana implementacja algorytmów zaproponowanych w tej pracy korzysta jeszcze z innej heurystyki. Program BSVM rozwiązuje inny problem niż ogólnie przyjęty problem SVM, a mianowicie do funkcji celu problemu pierwotnego SVM dodany został pewien składnik, co pozwoliło na sformułowanie problemu dualnego bez warunku równościowego [9]. Jako, że nie ma warunku równościowego, nie jest konieczne definiowanie grup G1 i G2 jak w przypadku heurystyki problemu SVM rozpatrywanej w tej pracy. A zatem heurystyka BSVM nie bierze pod uwagę warunku równościowego i wygląda nastepująco:

Heurystyka BSVM

r- liczba parametrów niegranicznych w danym kroku dekompozycji

Jeśli $r > 0$

Do zbioru parametrów aktywnych wybierz:

min (q/2, r) parametrów niegranicznych najlepiej spełniających kryterium KKT dla funkcji celu problemu SVM zdefiniowanego w [9].

q - min (q/2, r) parametrów najgorzej spełniających kryterium KKT dla funkcji celu problemu SVM zdefiniowanego w [9].

Jesli r = 0

Wybierz q/2 elementów najgorzej spełniających KKT dla których $y_i = 1$ oraz q/2 elementów najgorzej spełniających KKT dla których y_i = -1.

Koncepcja heurystyki BSVM tym różni się od koncepcji heurystyki H1, iż w każdej iteracji brane są pod uwagę parametry niegraniczne, co służy temu aby liczba parametrów niegranicznych pozostawała na niskim poziomie.

Możliwości dalszych udoskonaleń

Heurystyka opiera się na poszukiwaniu grupy parametrów takich, których optymalizacja maksymalizuje funkcję celu w danym kroku iteracyjnym. Maksymalizacja funkcji celu SVM różni się jednak metodologicznie od zwykłej maksymalizacji funkcji, ponieważ włączona jest do niej metoda dekompozycji. Bez metody dekompozycji w kolejnych iteracjach rozpartywana byłaby funkcja dla wszystkich zmiennych i byłby wyznaczany punkt bliższy maksymalnego niż dotychczasowy. Z włączoną metodą dekompozycji w każdym kroku iteracyjnym wybierana jest skończona ilość parametrów dla których poszukiwane jest lepsze rozwiązanie. A więc ważne jest to, aby zmiana grupy parametrów aktywnych powodowała zmianę globalną, wg wybranego kryterium globalnego. W przypadku maksymalizacji funkcji celu za pomocą wyboru parametrów, dla których gradient jest największy, takie przełożenie ma miejsce. Bowiem długość wektora gradientu wybranych parametrów o możliwie maksymalnej wartości bezwzględnej pochodnych cząstkowych jest wtedy maksymalna.

Jednakże poszukiwanie parametrów o maksymalnym gradiencie jest jedynie przybliżonym sposobem maksymalizacji funkcji celu w każdym kroku iteracyjnym. Bowiem brane jest pod uwagę jedynie nachylenie funkcji w punkcie początkowym. Nachylenie zaś w kolejnych punktach może się istotnie różnić od nachylenia w punkcie początkowym. Dlatego maksymalne nachylenie punktu początkowego niekoniecznie przekłada się na maksymalną zmianę funkcji celu. Możliwą poprawą tego stanu rzeczy, mogłoby być rozpatrywanie nachylenia nie tylko w punkcie początkowym, ale również w dalszych punktach. Druga pochodna jest stała, a zatem możliwe jest dokładne oszacowanie możliwej zmiany funkcji celu na danym odcinku. Takie oszacowanie biorące pod

uwagę pochodne pierwsze i drugie łatwo zrobić korzystając ze wzoru Taylora. Pozostaje kwestia odcinka dla którego przeprowadzane jest oszacowanie możliwego wzrostu funkcji celu. Rozsądnym wydaje się wybór odcinka o długości C w przypadku parametrów granicznych, choć może zdarzyć się, że optymalne rozwiązanie będzie miało wartość niegraniczną. Zastosowanie drugiej pochodnej daje więc dokładną wartość maksymalizacji funkcji celu dla wybranego parametru. Koncepcja zastosowania drugich pochodnych została zbadana w 2005 roku w artykule [19].

Kolejną rzeczą jest przyjęte uproszczenie, że maksymalizacja funkcji celu w każdej iteracji prowadzi do minimalnej liczby iteracji. Łatwo wyobrazić sobie funkcję która rośnie początkowo szybko, a następnie bardzo powoli, i w konsekwencji droga ta będzie wolniejsza od drogi umiarkowanego ciągłego wzrostu.

Metodą która mogłaby rozwiązać ten problem mogłoby być jakiekolwiek przybliżenie rozwiązania problemu SVM. Jednakże taka znajomość jest trudna do uzyskania, więc pozostaje analiza zachowania algorytmu podczas poprzednich kroków iteracyjnych.

Można wyobrazić sobie problem *dolin wymiarowych*, i zdefiniować tzw. *zygzakowanie wymiarowe*. Doliny wymiarowe pojawiają się wtedy, gdy jakiś parametr zostaje wybierany do optymalizacji w kolejnych iteracjach. Algorytm „wpada"w dolinę tego parametru, inaczej mówiąc zygzakuje pomiędzy kierunkiem odpowiadającym temu parametrowi, a pozostałymi. Zygzakowanie w przypadku zwykłej maksymalizacji rozwiązywane jest zwykle za pomocą metody kierunków sprzężonych. Wybierany jest kierunek pośredni między dwoma poprzednimi. Jednakże tutaj w grę wchodzi zygzakowanie wymiarowe. Nałożone ograniczenie na algorytm rozpatrywania ustalonej liczby parametrów za każdym razem, nie pozwala na wyjście z doliny za pomocą łączenia ze sobą parametrów z poprzednich kroków optymalizacyjncych. Rozwiązaniem jest wykluczenie z dalszych iteracji powtarzającego się parametru. Możliwa jest dalsza analiza zachowania procesu iteracji dla danej heurystyki.

3.4. Warunek stopu

Warunek stopu zastosowany w programie Analytical SVM (ASVM) implementującym metodę ASO oraz opisane wyżej heurystyki jest podobny jak we wszystkich popularnych programach SVM, a mianowicie polega na sprawdzaniu czy jest spełniony warunek KKT dla wszystkich parametrów. Obliczanie warunków KKT jest konieczne przy wyznaczaniu heurystyki, dlatego nie ma potrzeby ponownego obliczania tych warunków przy warunku stopu.

Rozdział 4

Implementacja metody Analitycznej Optymalizacji Sekwencyjnej i heurystyki

W ramach pracy magisterskiej powstał program ASVM (ang. *analytical SVM*, ASVM) implementujący metodę ASO oraz nową heurystykę. Podstawowymi celami, które realizuje program ASVM są odpowiednia szybkość działania programu oraz poprawność i dokładność generowanych wyników.

W celu osiągnięcia odpowiedniej szybkości działania programu, analizowane były „wąskie gardła"programu, szczególnie dla przypadku dużej liczby parametrów. W praktyce takie właśnie dane stanowią trudność dla istniejących algorytmów SVM. Podstawowym wąskim gardłem jeśli chodzi o szybkość działania programu jest obliczanie wartości funkcji jądra. Funkcja jądra wywoływana jest m.in. przy obliczaniu wartości funkcji decyzyjnej. Wartości funkcji decyzyjnej wykorzystywane są: przy obliczeniach równań analitycznego rozwiązania problemów (PO 3.1.2), podczas wyznaczania funkcji celu przy porównywaniu jakości znalezionego rozwiązania w dekompozycji wewnętrznej ASO, przy obliczaniu heurystyki oraz przy obliczaniu warunków KKT wykorzystywanych jako kryterium stopu.

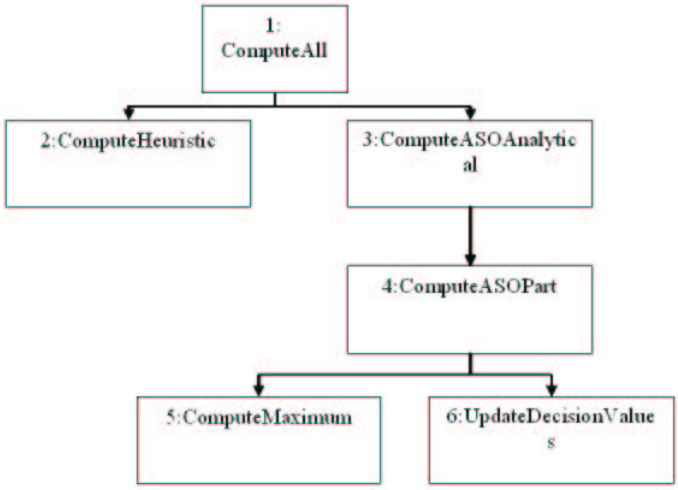
4.1. Struktura programu ASVM

Program ASVM został napisany w języku C. Zastosowaną metodą programowania było programowanie funkcjonalne. Schemat wywołań głównych funkcji został zamieszczony na rysunku (Rys. 4.1).

ComputeAll: Funkcja ta odpowiada za zarządzanie wywoływaniem podproblemów dekompozycji zewnętrznej

ComputeASOAnalytical: Funkcja odpowiada za wywoływanie rekurencyjnej funkcji ComputeASOPart i za aktualizację listy z wartościami E_i.

ComputeASOPart: Jest to funkcja rekurencyjna odpowiadająca za zarządzanie podproblemami dekompozycji wewnętrznej.



Rysunek 4.1. Rysunek przedstawia strukturę funkcjonalną programu ASVM.

ComputeMaximum: Funkcja wyznaczająca współczynniki układu równań ASO oraz obliczająca jego rozwiązanie. Zastosowaną metodą wyznaczania rozwiązania układu równań liniowych jest metoda Gaussa z pełnym wyborem elementu podstawowego.

Do każdej funkcji w programie zostały napisane szczegółowe testy.

4.2. Wejście/Wyjście

Program ASVM czyta plik konfiguracyjny z parametrami metody SVM, w linii poleceń można podać ścieżkę do pliku konfiguracyjnego. Przykładowy plik konfiguracyjny wygląda następująco:

smo.cfg:

```
//File with training data
Dane\a4a.txt
//File with testing data
Dane\a4aTest.txt
//Data file format
//dense - 0; sparse - 1
1
//Output file name
output.txt
//Model file name
model.txt
//SMO parameters
//Linear Equation Solver
//0 - gaussj
//1 - lubksb
0
//Max box value
```

```
1
//number of parameters changes through one iteration for ASO part
// 2 - SMO
2
//Kernel parameters
//Which kernel
//1 - Polynomial Kernel (xy +C) D
//2 - RBF Kernel -gamma*||a-b|| 2
//3 - Sigmoid Kernel tanh(Cxy+D)
2
//Polynomial/Sigmoid kernel parameters
//C
1
//D
3
//RBF kernel parameters
//gamma
1.0
//Kkt error
0.001
//cache size (MB) integer
30
```

Odpowiedzią programu są komunikaty ASVM pojawiające się na standardowym wyjściu, jak również komunikaty zapisywane do pliku podawanego w parametrach. Do pliku modelowego zapisywane są znalezione parametry SVM, które mogą posłużyć do testowania danych.

4.3. Struktury danych

W programie ASVM znajdują się dwie globalne struktury danych:

```
struct Data
{
int numberOfVectors;
int vectorsDimension;
int *membership;
double **vectors;
} myData;
```

Zmienna myData zawiera podstawowe informacje o danych pochodzących z pliku treningowego.

```
struct Parameters
{
enum LinearEquationSolver linearEquationSolver;
double maxBoxValue;
int ASOParametersCount;
//kernel parameters
double polynomialKernelParameterC;
```

```
double polynomialKernelParameterD;
double RBFKernelParameterGamma;
//stopping criterion parameter
double kktError;
enum DataFileFormat dataFileFormat;
char *dataFileName;
char *outputFileName;
FILE *outputFile;
double *alfa;
double threshold;
double *heuristicKKTerror;
double *kernelMatrixDiagonal;
int *heuristicSign;
int heuristicIsChange;
int *nonBoundParametersClass;
int allIterations;
int interiorIterations;
int cacheSize; //in MBs
int cacheRows;
int *cacheParameters;
long *parametersFrequency;
long nonBounds;
double **globalCache;
enum KernelType kernelType;
} myParameters;
```

Z kolei zmienna myParameters zawiera parametry SVM zarówno te czytane z pliku konfiguracyjnego jak również obliczanie w trakcie działania programu często wykorzystywane przez różne funkcje SVM, jak choćby parametry α.

Zbiory wektorów wejściowych przechowywane są w całości,a więc traktowane są zawsze jako dane gęste. W przyszłości planuje się zaimplementować struktury danych przystosowane do przechowywania danych rzadkich, co pozwoli na obniżenie zapotrzebowania na pamięć ogromnych zbiorów danych o stukturze rzadkiej.

4.4. Obliczanie wartości funkcji decyzyjnej

Funkcję decyzyjną (1.15) można zapisać w postaci:

$$f(\alpha_i) = \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) + b \tag{4.1}$$

Funkcja decyzyjna (4.1), o której będzie mowa w tym punkcie nie jest wyznaczana w ogólności w całości, nie jest konieczna bowiem za każdym razem znajomość parametru b, dlatego będzie rozpatrywana następująca funkcja zamiast (4.1):

$$g(\alpha_i) = \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) \tag{4.2}$$

Obliczanie funkcji (4.2) wiąże się z dużymi kosztami obliczeniowymi.

Obliczenie wartości funkcji $g(\cdot)$ dla wszystkich parametrów ma złożoność rzędu $O(l^2)$. W każdym obliczeniu podstawowym kosztem stałym jest wyznaczenie wartości jądra dla parametrów i oraz j.

Poniżej zostaną przedstawione miejsca w programie, gdzie są używane wartości $g(\cdot)$.

- Przy każdym obliczeniu układu równań, konieczne jest obliczenie parametrów E_i dla $i \in \{1..p\}$

$$\sum_{i=1}^{q-1} y_i \alpha_i^{new} \kappa_{ipk} = \sum_{i=1}^{q-1} y_i \alpha_i \kappa_{ipk} + E_k - E_p,$$

  gdzie wartość E_i wynosi:

$$E_i = g(\alpha_i) - y_i.$$

- Wewnątrz algorytmu dekompozycji ASO, po znalezieniu możliwego rozwiązania porównywane jest ono z najlepszym dotychczas znalezionym rozwiązaniem. Porównywanie odbywa się przy pomocy obliczonej wartości funkcji celu. Funkcja celu jest wyznaczana przy wykorzystaniu wartości $g(\cdot)$ w następujący sposób:

$$W(\vec{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \alpha_i y_i g(\alpha_i), \quad (4.3)$$

  Takie wyznaczenie wartości funkcji celu ma złożoność $O(l)$. Istnieje możliwość dalszego zmniejszenia tej złożoności poprzez wprowadzenie aktualizacji funkcji dualnej. Wtenczas złożoność wynosiłaby $O(q)$, gdzie q to liczba aktualizowanych parametrów w danym kroku optymalizacyjnym.
- Przy wyznaczaniu heurystyki konieczne jest wyznaczanie pochodnych funkcji W_2 według wzoru (3.12):

$$\frac{\partial W_2(\alpha_1, \alpha_2, ..., \alpha_{l-1})}{\alpha_k} = y_k (E_l - E_k),$$

  gdzie $E_i = g(\alpha_i) - y_i$
- Przy wyznaczaniu warunku KKT (1.16) po znalezieniu rozwiązania podproblemu dekompozycji zewnętrznej w celu sprawdzenia warunku stopu.

Należy zwrócić uwagę na to, że wartości funkcji $g(\cdot)$ w dwóch ostatnich przypadkach nie muszą być bezpośrednio wyznaczane, bowiem zostają zachowane podczas obliczania przypadku drugiego.

Metody zmniejszenia czasu obliczania funkcji $g(\cdot)$.

Podstawową metodą zmniejszenia czasu obliczania funkcji $g(\cdot)$ jest wykorzystanie wcześniej obliczonych wartości funkcji $g(\cdot)$ dla każdego parametru. Zaletą metody ASO nad algorytmami numerycznymi jest wykorzystanie przez nią wcześniej obliczonych wartości funkcji $g(\cdot)$ przy obliczaniu podproblemu ASO, oraz zwrócenie nowych wartości funkcji $g(\cdot)$ do dalszego wykorzystania. Wadą aktualizacji wartości funkcji

$g(\cdot)$ jest zmniejszenie dokładności tej wartości w kolejnych krokach algorytmu. Jednakże zaleta zwiększonej szybkości działania przeważa nad wadą dokładności. Celem osiągnięcia większej dokładności możliwe jest obliczanie na nowo wartości funkcji $g(\cdot)$ co wybraną liczbę iteracji algorytmu.

Przed procesem aktualizacji funkcji $g(\cdot)$ konieczne jest obliczenie wartości tejże funkcji. W celu wyeliminowania tego kroku ze względu na złożoność, rozpatrywany jest zerowy punkt początkowy. Wtenczas wartości funkcji $g(\cdot)$ są równe 0 dla każdego parametru.

Metoda aktualizacji wartości funkcji $g(\cdot)$

Metoda aktualizacji funkcji $g(\cdot)$ przebiega następująco:

a) Na początku aktualizowane są częściowo wartości funkcji $g(\cdot)$ dla zbioru q wybranych parametrów. Częściowo oznacza, że brana jest pod uwagę wartość j we wzorze (4.2) odpowiadająca wybranemu parametrowi, a więc taka, że $i = j$. Ten krok ma złożoność $\mathrm{O}(q)$.

b) W kolejnym kroku aktualizowane są całkowicie wszystkie wybrane parametry. Wartości j pochodzą od wszystkich pozostałych wybranych parametrów. Ten krok ma złożoność $\mathrm{O}(q^2)$.

c) W ostatnim kroku aktualizowane są wszystkie nie wybrane parametry, gdzie wartości j są indeksami wybranych parametrów. Ten krok ma złożoność $\mathrm{O}(pl)$.

W przypadku b) wystarczy obliczać jedynie połowę parametrów ze względu na symetrię.

Przypadek c) nie jest konieczny do obliczania funkcji decyzyjnej w przypadku obliczania wartości E_i (punkt 1).

Istotnym udoskonaleniem przejawiającym się w zmniejszeniu liczby parametrów, które należy aktualizować, jest sprawdzanie, które parametry uległy zmianie i tylko takie biorą udział w aktualizacji.

4.5. Cache wartości funkcji jądra

Kolejną grupą udoskonaleń jest wprowadzenie cache'a dla wartości funkcji jądra. Wartości te wykorzystywane są przede wszystkim podczas czasochłonnego obliczania funkcji decyzyjnej. Możliwość utrzymywania cache'a dla wszystkich możliwych wartości funkcji jądra jest praktycznie niemożliwa, ze względu na złożoność pamięciową $\mathrm{O}(l^2)$. Dlatego stosuje się cache o mniejszej pojemności, nie zawierający wszystkich możliwych wartości.

W programie ASVM zostało zastosowanych wiele rodzajów cache'a.

Gdy jest obliczany po raz pierwszy układ równań analitycznego rozwiązania ASO, zapamiętuje się wszystkie wyznaczane wartości jądra. Dla kolejnych podproblemów ASO rozwiązywane są układy równań o mniejszej liczbie parametrów ale wykorzystujące wyznaczone wcześniej wartości funkcji jądra.

Drugim rodzajem cache'u zastosowanym w ASVM jest cache globalny przechowujący wartości funkcji jądra dla tych samych parametrów, a więc wartości diagonalne tablicy jądra.

Trzecim najbardziej ogólnym cache'em jest cache globalny który przechowuje wartości ostatnio używane. Dla danego parametru i przechowywany jest cały wiersz wartości K_{ij}. Podczas dodawania nowego wiersza do cache'a, jeśli nie ma wolnego miejsca

usuwany jest wiersz odpowiadający parametrowi najrzadziej wybieranemu podczas dekompozycji zewnętrznej. Jeśli nie ma w cache'u parametru, który był rzadziej wybierany od wstawianego to nowy wiersz nie jest zapisywany w cache'u. Cache globalny wykorzystywany podczas obliczania funkcji decyzyjnej daje znaczący przyrost prędkości programu.

4.6. Implementacja heurystyki

Wyznaczanie wartości E_l:

Wartość E_l konieczna jest do wyznaczenia zarówno przy heurystyce, jak również przy obliczaniu warunku stopu.

Jako, że E_l jest obecne przy obliczaniu wszystkich pochodnych pozostałych parametrów (3.12) ważna jest dokładność wyznaczenia E_l.

Dla $0 < \alpha_k < C$

$$y_k \left(E_l - E_k\right) = 0$$

$$E_l = E_k$$

Dla $\alpha_k = 0$

$$\begin{cases} E_l \leq E_k \; dla \; y_k = 1 \\ E_l \geq E_k \; dla \; y_k = -1 \end{cases}$$

Dla $\alpha_k = C$

$$\begin{cases} E_l \geq E_k \; dla \; y_k = 1 \\ E_l \leq E_k \; dla \; y_k = -1 \end{cases}$$

A więc
$E_l \leq E_k$,gdy

$$G1 : (y_i = 1 \wedge \alpha_i = 0) \vee (y_i = -1 \wedge \alpha_i = C)$$

$E_l \geq E_k$,gdy

$$G2 : (y_i = 1 \wedge \alpha_i = C) \vee (y_i = -1 \wedge \alpha_i = 0)$$

$E_l = E_k$,gdy

$$G3 : 0 < \alpha_k < C$$

Z grupy G1 oraz G3 poszukiwany jest element minimalny, zaś z grupy G2 i G3 poszukiwany jest element maksymalny. Następnie brana jest średnia arytmetyczna wartości maksymalnej i minimalnej.

Można pokazać, że element E_l odpowiada parametrowi b problemu pierwotnego SVM. Ten sposób aktualizacji parametru b został wykorzystany m.in. w aplikacji SVM Light.

Wyznaczanie warunku KKT

Przy obliczaniu warunku stopu konieczne jest sprawdzenie czy jest spełniony warunek KKT, zaś przy heurystyce H1 konieczne jest aby nie był on spełniony maksymalnie. W celu łatwego sprawdzania warunku KKT następujące wartości zostają zapisane w liście, której elementy w_k odpowiadają poszczególnym parametrom:

Gdy $\alpha_k = 0$

$$w_k = -y_k \left(E_l - E_k\right)$$

Gdy $\alpha_k = C$

$$w_k = y_k \left(E_l - E_k\right)$$

Gdy $0 < \alpha_k < C$

$$w_k = -\left|y_k \left(E_l - E_k\right)\right|$$

W liście elementy w_k, które są większe od zera spełniają warunek KKT, a które nie, to nie spełniają, elementy w liście najmniejsze spełniają warunek KKT najgorzej.

Jako, że warunek KKT powinien być spełniony z pewną tolerancją $\varepsilon > 0$, lista w po wprowadzeniu modyfikacji wygląda następująco:

Gdy $\alpha_k = 0$

$$w_k = -y_k \left(E_l - E_k\right) + \varepsilon$$

Gdy $\alpha_k = C$

$$w_k = y_k \left(E_l - E_k\right) + \varepsilon$$

Gdy $0 < \alpha_k < C$

$$w_k = -\left|y_k \left(E_l - E_k\right)\right| + \varepsilon$$

Dodanie tolerancji błędu nie zmienia wyników porównań elementów listy w ze sobą, dlatego lista w może w dalszym ciągu być używana na potrzeby heurystyki.

Rozdział 5

Rezultaty

5.1. Testy heurystyki i szybkości działania programu ASVM

Dla potrzeb testów został napisany program, które na podstawie dowolnego rysunku czarno-białego generuje losowe punkty i zapisuje je wraz z przyporządkowaniem klasowym. Rysunki zostały przetestowane z różnymi jądrami. Jądra RBF oraz wielomianowy radzą sobie z nieliniowymi granicami decyzyjnymi, w odróżnieniu od jądra liniowego. Program ASVM jest porównany z najlepszym dostępnym programem rozwiązującym problem SVM, a mianowicie BSVM. Program BSVM okazał się w większości wypadków lepszy od programu SVM Light co zostało pokazane w artykule [9]. W programie BSVM znajduje się optymalizator numeryczny TRON [14] rozwiązujący problemy optymalizacyjne z warunkami równościowymi i nierównościowymi za pomocą metody „zaufanych obszarów"(ang. *trust region method*). Natomiast we wspomnianym programie SVM Light jest używany optymalizator LOQO oparty na wspomnianej w tej pracy metodzie punktu wewnętrznego.

Jako, że w programie BSVM funkcja celu jest inna niż funkcja celu problemu SVM wartości funkcji dualnej mogą się nieznacznie różnić od wartości funkcji dualnej programu ASVM, mimo ustawionej tej samej dokładności warunku stopu równej 0.001.

Parametr C został ustawiony na 1. Cache globalny we wszystkich implementacjach został ustawiony na 30MB.

Testy zostały wykonane na komputerze z 512MB pamięci RAM i procesorem Pentium IV 1,5Ghz.

Test 5.1.1. Test dla danych z rysunku (Rys. 5.1). Liczba wektorów: 6000. Jądro RBF z parametrem gamma = 1. Liczba parametrów niegranicznych: 3. Wyniki zostały zebrane w tabeli 5.1.

W tym teście liczba iteracji programu ASVM okazała się lepsza od liczby iteracji BSVM, istotna jest również dobra skalowalność heurystyki ASVM na większą liczbę

Rysunek 5.1. Rysunek przedstawia dwie klasy zaznaczone kolorem białym i czarnym z których pochodzą punkty treningowe, granica decyzyjna nieliniowa.

Wielkość podproblemów	*Liczba iteracji ASVM (H3)*	*Funkcja dualna (ASVM)*	*Czas wykonania ASVM [s]*	*Liczba iteracji BSVM*	*Funkcja dualna (BSVM)*	*Czas wykonania BSVM [s]*
2	2639	2984,60	11	4202	2989,95	8
3	2365	2984,60	13	2771	2989,95	7
4	1406	2984,60	13	2264	2989,95	8
5	1155	2984,60	25	2025	2989,95	8

Tablica 5.1. Liczba iteracji i czasy obliczeń dla jądra RBF

parametrów. Czas wykonania programu ASVM okazał się nieznacznie gorszy. Widać również, że czas wykonania dla 5 parametrów jest już dużo większy od czasów wykonania dla 2,3 i 4 parametrów. Z drugiej strony BSVM nie wykazuje lepszych wyników czasowych dla większej liczby parametrów.

Rysunek 5.2. Rysunek przedstawia dwie klasy zaznaczone kolorem białym i czarnym z których pochodzą punkty treningowe, granica decyzyjna liniowa.

Test 5.1.2. Test dla danych z rysunku (Rys. 5.2). Liczba wektorów 10100. Jądro liniowe: $(xy+1)$. Liczba parametrów niegranicznych: 3. Wyniki zostały zebrane w tabeli 5.2.

Dla jądra liniowego liczba iteracji programu ASVM okazała się lepsza od liczby iteracji BSVM. Podobnie jak w przypadku jądra RBF można zauważyć dobrą skalowalność heurystyki ASVM na większą liczbę parametrów. Widać również, że czas wykonania pozostaje na tym samym poziomie dla parametrów 2, 3, 4. Czas wykonania BSVM jest porównywalny z ASVM. BSVM nie wykazuje przyrostu prędkości dla większej liczby parametrów.

Wielkość podproblemów	*Liczba iteracji ASVM (H3)*	*Funkcja dualna (ASVM)*	*Czas wykonania ASVM [s]*	*Liczba iteracji BSVM*	*Funkcja dualna (BSVM)*	*Czas wykonania BSVM [s]*
2	1163	1052,19	8	1898	1053,19	9
3	928	1052,19	9	1108	1053,19	9
4	632	1052,19	11	910	1053,19	9
5	435	1052,19	18	874	1053,19	9

Tablica 5.2. Liczba iteracji i czasy obliczeń dla jądra liniowego.

Rysunek 5.3. Rysunek przedstawia dwie klasy zaznaczone kolorem białym i czarnym z których pochodzą punkty treningowe, granica deycyzjna wielomianowa.

Test 5.1.3. Test dla danych z rysunku (Rys. 5.3). Liczba wektorów 6000. Jądro wielomianowe o wykładniku 3: $(xy+1)^3$. Liczba parametrów niegranicznych: 6. Wyniki zostały zebrane w tabeli 5.3.

Wielkość podproblemów	*Liczba iteracji ASVM (H3)*	*Funkcja dualna (ASVM)*	*Czas wykonania ASVM [s]*	*Liczba iteracji BSVM*	*Funkcja dualna (BSVM)*	*Czas wykonania BSVM [s]*
2	3908	1060,24	10	2166	1061,44	5
3	1250	1060,24	6	1493	1061,44	5
4	1849	1060,24	13	907	1061,44	5
5	566	1060,24	13	849	1061,44	5

Tablica 5.3. Liczba iteracji i czasy obliczeń dla jądra wielomianowego.

Liczba iteracji programu ASVM okazała się nieco większa dla parzystej liczby parametrów od liczby iteracji BSVM, widać również dobrą skalowalność w obrębie heurystyki zarówno dla parzystej liczby parametrów jak i nieparzystej. W praktyce jądro wielomianowe jest mniej przydatne dla danych rzeczywistych niż jądro RBF, ponieważ granica decyzyjna jest bardziej ograniczona.

W powyższych testach została pokazana dobra skalowalność implementacji heurystyki ASVM oraz jej lepsze działanie w porównaniu z heurystyką zaimplementowaną w

BSVM, jednakże nie została osiągnięta lepsza skalowalność czasowa programu ASVM na większa liczbę parametrów i wyniki czasowe są porównywalne z wynikami programu BSVM, czasami nieznacznie gorsze.

Testy na danych rzeczywistych

Dla danych rzeczywistych wielowymiarowych liczba parametrów niegranicznych jest dużo większa niż dla nieskomplikowanych danych dwuwymiarowych, z tego względu spodziewana jest lepsza skalowalność czasowa algorytmu ASVM, co wynika z większej szansy na zakończenie algorytmu rekurencyjnego ASO przed dojściem do przypadku dwuparametrowego.

Test 5.1.4. Źródło: UCI Machine Learning Repository

Pliki z danymi pochodzą ze strony: [7].

Plik a4a.txt.

Plik zawiera dane statystyczne o rodzinach podzielonych na dwie klasy, informujący o tym, czy dana rodzina osiąga dochody powyżej 50k czy też nie.

Jądro RBF.

gamma = 1.

Liczba wektorów = 4781.

Wymiar: 119.

Liczba parametrów niegranicznych = około 3400.

Wyniki zostały zebrane w tabeli 5.4.

Wielkość podproblemów	*Liczba iteracji ASVM (H3)*	*Funkcja dualna (ASVM)*	*Czas wykonania ASVM [s]*	*Liczba iteracji BSVM*	*Funkcja dualna (BSVM)*	*Czas wykonania BSVM [s]*
2	9718	1500,58	39	15767	1500,80	46
3	5987	1500,58	33	10388	1500,80	20
4	5968	1500,58	36	8709	1500,80	48
5	3527	1500,58	36	6865	1500,80	52
6	5456	1500,58	58	6205	1500,80	55
7	2432	1500,58	95	5215	1500,80	51

Tablica 5.4. Liczba iteracji i czasy obliczeń dla danych rzeczywistych o dochodach.

Testy pokazują, iż dla danych rzeczywistych program ASVM radzi sobie lepiej od programu BSVM pod względem liczby iteracji, co przekłada się na lepsze wyniki czasowe dla niektórych przypadków. Heurystyka ASVM dla parzystej liczby parametrów jest nieco gorsza od heurystyki dla nieparzystej liczby parametrów.

Widać również lepszą skalowalność czasową algorytmu ASO na większą liczbę parametrów, niż w poprzednich testach dwuwymiarowych, na co ma wpływ przede wszystkim liczba parametrów niegranicznych, w powyższym teście do pięciu parametrów aktywnych algorytm ASO zwraca porównywalne między sobą wyniki czasowe, dla więcej niż dwóch parametrów wyniki okazują się lepsze niż dla dwóch parametrów.

Słabe i mocne strony programu ASVM

Z powyższych testów wynika bardzo dobre działanie heurystyki ASVM, i jej bardzo dobra skalowalność. W praktycznie wszystkich przypadkach heurystyka okazała się

lepsza od heurystyki programu BSVM, czasami nawet trzykrotnie. Testy pokazują również, że heurystyka dla parzystej liczby parametrów jest nieco gorsza od tej dla nieparzystej liczby parametrów aktywnych.

Wąskim gardłem programu ASVM jest jego skalowalność czasowa na większą liczbę parametrów aktywnych. Jakkolwiek dla danych rzeczywistych, wielowymiarowych z dużą ilością parametrów niegranicznych wyniki okazały się lepsze niż dla danych z małą liczbą parametrów niegranicznych, to i tak, stosowanie algorytmu analitycznego rozwiązania podproblemu SVM dla więcej niż 7 parametrów aktywnych może być kłopotliwe.

5.2. Podsumowanie

Możliwości rozwoju

Możliwości udoskonalenia heurystyki są duże, przy wyprowadzeniu wzorów na spełnienie warunku równościowego zostało poczynionych kilka założeń upraszczających, można te założenia rozszerzyć i tym samym otrzymać dokładniejsze przybliżenie celów heurystyki. Ważnym aspektem jest dalsze badanie relacji między heurystyką maksymalnego gradientu ze spełnieniem warunków KKT z ograniczeniem równościowym problemu SVM. Pozwoliłoby to na zbadanie możliwości osłabienia założenia absolutnego spełnienia ograniczenia równościowego. Istotną kwestią dla dokładności obliczeń jest ograniczenie liczby parametrów niegranicznych podczas procesu optymalizacji. Konieczna jest więc dalsza analiza sposobu aktualizacji parametrów i utworzenie kolejnych udkoskonaleń heurystyki minimalizującej szansę powstawania nowych parametrów niegranicznych. Ponadto wykorzystanie drugich pochodnych oraz innych możliwości opisanych w punkcie 3.3 mogłoby dodatkowo poprawić heurystykę.

Rozwój algorytmu analitycznego ASO powinien przebiegać w kierunku zmniejszenia liczby podproblemów dekompozycji wewnętrznej, wprowadzenie udoskonaleń przedstawiownych w punkcie 3.1.1 może tą liczbę poprawić. Możliwości zrównoleglenia ASO są bardzo dobre, ze względu na podział podproblemu SVM na podproblemy dekompozycji wewnętrznej, które mogą być wykonywane niezależnie od siebie w przeciwieństwie do trudności napotykanych przy zrównoleglaniu metod numerycznych z zakresu programowania kwadratowego.

Ponadto algorytm ASO może być stosowany również do ogólniejszych problemów optymalizacyjnych. Uogólnienie polegałoby na rozpatrywaniu również innych funkcji wklęsłych, nie tylko kwadratowych, oraz ogólniejszych warunków nierównościowych, które odpowiadałyby bardziej skomplikowanym tworom geometrycznym.

Z powyższego wynika, że istnieje spory potencjał dalszych udoskonaleń heurystyki ASVM, oraz algorytmu analitycznego ASO zarówno od strony teoretycznej jak i implementacyjnej.

Rozwój aplikacji ASVM będzie polegał przede wszystkich na poprawieniu jej szybkości działania, oraz dodaniu pewnych funkcjonalności, takich jak rozpatrywanie pokrewnych problemów SVM, implementacja regresji SVM, algorytmu wyboru optymalnych parametrów SVM za pomocą metody walidacji krosowej.

Realizacja celów

Wszystkie zakładane cele zostały zrealizowane.

Pierwszym celem pracy było udoskonalenie heurystyki wyboru parametrów aktywnych. W pracy została wyprowadzona w nowy sposób istniejąca heurystyka dekompozycji zaproponowana po raz pierwszy w artykule [11]. Zostały przy tym zdefiniowane konieczne założenia tej heurystyki. Udoskonalenie polega na pokazaniu, iż można te założenia osłabić. Usprawnienie heurystyki dotyczy wyboru parametrów niegranicznych do zbioru aktywnego. Testy wykazały, iż rzeczywiście nowa heurystyka zachowuje się lepiej od porównywanej herystyki.

Drugim celem było stworzenie analitycznej metody rozwiązującej podproblemy SVM. Proces wyprowadzenia tej metody był długotrwały. Pierwszym napotkanym problemem było rozwiązanie analityczne podproblemu SVM bez warunku nierównościowego dla ogólnego przypadku wielu parametrów. W wyprowadzeniu tych wzorów

istotnym elementem było posłużenie się analitycznym rozwiązaniem dla przypadku dwóch parametrów [18]. Kłopoty w uogólnieniu sprawiało mnogość rachunków, które w wersji rozbudowanej zajmowały kilkanaście stron.

Kolejnym etapem w wyprowadzeniu analitycznej metody było posłużenie się modelem geometrycznym podproblemów SVM. Metoda polegała na wizualizacji pewnych właściwości funkcji celu i sprowadzaniu tych właściwości do postaci analitycznej.

Następną kwestią było skonstruowanie dowodów wszystkich zauważonych prawidłowości i tym samym udowodnienie poprawności prezentowanej metody. Oprócz głównego algorytmu dekompozycji wewnętrznej konieczne było stworzenie dodatkowych algorytmów odpowiadających poszczególnym etapom metody analitycznej ASO. Jakkolwiek nowa metoda jest efektywna tylko dla niewielkiej liczby parametrów aktywnych, to i tak stanowi ona ważny krok, świadczący o tym, że możliwe jest analityczne rozwiązanie zagadnienia SVM dla wielu parametrów.

Trzecim celem była implementacja metody SVM wraz z wyprowadzonymi udoskonaleniami. Implementacja miała spełniać zaprezentowane we wstępie założenia.

Pierwszym założeniem była szybkość programu porównywalna z innymi wybranymi implementacjami SVM. Program ASVM stworzony w ramach tej pracy magisterskiej, został porównany z programem BSVM [9]. Szybkość programu ASVM okazała się porównywalna, w niektórych przypadkach lepsza od szybkości BSVM. Istnieje jednak w dalszym ciągu spory zakres możliwości poprawy szybkości działania aplikacji z czysto implementacyjnego punktu widzenia.

Drugim założeniem była poprawna generacja wyników dla różnych danych treningowych. Do każdej funkcji rozwiązującej problem SVM zostały napisane testy sprawdzające poprawność działania. Program został przetestowany zarówno na danych specjalnych, jak również na danych rzeczywistych problemów. Testowanie odbyło się z różnymi rodzajami jąder: wielomianowym, liniowym, funkcji potencjałowych (RBF).

Udało się zaimplementować metodę analityczną rozwiązywania podproblemów SVM, jak również udoskonaloną heurystykę. Program ASVM tym samym stanowi samodzielną implementację nie korzystającą z żadnych bibliotek numerycznych.

Osiągnięcia

1. Metoda analityczna rozwiązywania podproblemów SVM dla dowolnej liczby parametrów
   wyprowadzenie rozwiązania analitycznego podproblemu SVM bez warunku nierównościowego,
   algorytm dekompozycji wewnętrznej,
   algorytm sprawdzania przecięcia ściany hipersześcianu z hiperpłaszczyzną,
   stworzenie teorii projekcji maksimum funkcji celu do hipersześcianu,
   dowody wszystkich twierdzeń teorii projekcji.
2. Nowa heurystyka ASVM
   wyprowadzenie warunku KKT dla problemu dualnego SVM,
   wyprowadzenie heurystyki zaprezentowanej przez Joachima [11],
   zaprezentowanie nowej heurystyki dla nieparzystej liczby parametrów,
   poprawa heurystyki [11] dla przypadku parzystej liczby parametrów,
3. Program ASVM rozwiązujący podproblemy SVM dla zbioru aktywnego wieloparametrowego bez korzystania z bibliotek numerycznych rozwiązujących podproblemy SVM.

Dodatek A

Wyprowadzenia wzorów SMO

A.1. Wyprowadzenie wzorów na ograniczenia parametrów SMO

Poniżej zostały zamieszczone wyprowadzenia wzorów na zmienne U i V ograniczające parametr α_2 (2.1):

$U \leq \alpha_2 \leq V$,gdzie

dla $y_1 \neq y_2$

$$U = \max\left(0,\ \alpha_2^{old} - \alpha_1^{old}\right),$$

$$V = \min\left(C,\ C - \alpha_1^{old} + \alpha_2^{old}\right)$$

dla $y_1 = y_2$.

$$U = \max\left(0,\ \alpha_1^{old} + \alpha_2^{old} - C\right),$$

$$V = \min\left(C,\ \alpha_1^{old} + \alpha_2^{old}\right)$$

Poniżej zostały przedstawione dwa sposoby wyprowadzenia powyższych wzorów. Pierwszy oparty na analizie geometrycznej, a drugi na analitycznej.

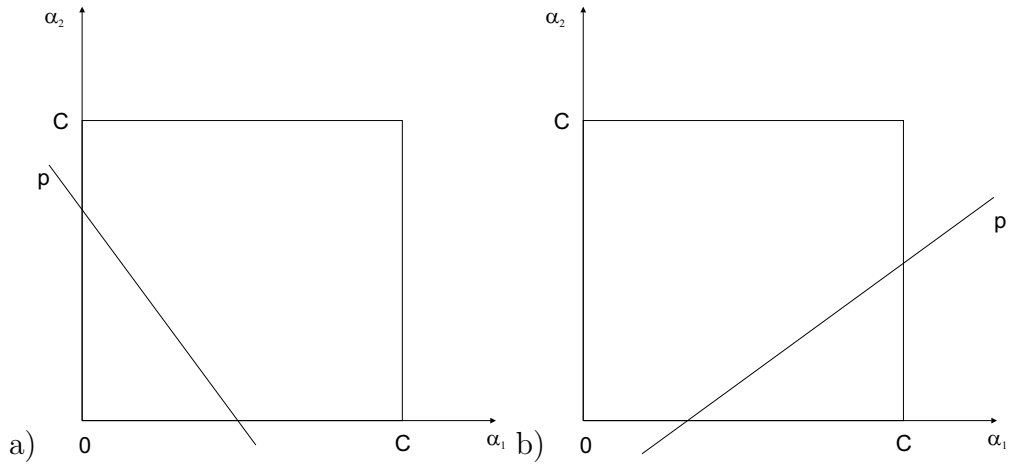
Dowód geometryczny. Równanie przedstawiające warunek równościowy problemu SVM wygląda następująco:

$$\alpha_2 = \alpha_1^{old} y_1 y_2 + \alpha_2^{old} - \alpha_1 y_1 y_2 \tag{A.1}$$

Prosta przecina lewy bok kwadratu, gdy $\alpha_1 = 0$;

Prosta przecina prawy bok kwadratu, gdy $\alpha_1 = C$;

Gdy $y_1 = y_2$, to $y_1 y_2 = 1$, podstawiając powyższe do równania (A.1) otrzymuje się:
(p) $\alpha_2 = \alpha_1^{old} + \alpha_2^{old} - \alpha_1$
Prosta p ma współczynnik kierunkowy ujemny, równy -1 (Rys. A.1), punkt a).



Rysunek A.1. Rysunek przedstawia prostą p w przypadku a) o współczynniku ujemnym, w przypadku b) dodatnim.

Następnie po podstawieniu za α_1 0 oraz C otrzymuje się wartości punktów przecięcia prostej p z prostymi $\alpha_1 = 0$ oraz $\alpha_1 = C$:
$\alpha_2 = \alpha_1^{old} + \alpha_2^{old}$oraz $\alpha_2 = \alpha_1^{old} + \alpha_2^{old} - C$
Biorąc pod uwagę to, że te punkty przecięcia muszą leżeć w obrębie kwadratu otrzymuje się następujące ograniczenia dla wartości parametru α_2:

$$U = \max\left(0,\ \alpha_1^{old} + \alpha_2^{old} - C\right),$$

$$V = \min\left(C,\ \alpha_1^{old} + \alpha_2^{old}\right)$$

Gdy$y_1 \neq y_2$, to $y_1 y_2 = -1$, podstawiając powyższe do równania prostej p otrzymuje się:

$$\alpha_2 = -\alpha_1^{old} + \alpha_2^{old} + \alpha_1.$$

Prosta p ma współczynnik kierunkowy dodatni, równy 1 (Rys. A.1), punkt b).
Następnie po postawieniu za α_1 0 oraz C otrzymuje się wartości punktów przecięcia prostej p z prostymi $\alpha_1 = 0$ oraz $\alpha_1 = C$:
$\alpha_2 = -\alpha_1^{old} + \alpha_2^{old}$oraz $\alpha_2 = -\alpha_1^{old} + \alpha_2^{old} + C$
Biorąc pod uwagę to, że punkty przecięcia muszą leżeć w obrębie kwadratu otrzymuje się następujące ograniczenia dla wartości parametru α_2:

$$U = \max\left(0,\ \alpha_2^{old} - \alpha_1^{old}\right),$$

$$V = \min\left(C,\ C - \alpha_1^{old} + \alpha_2^{old}\right)$$

□

Dowód analityczny. Dana jest nierówność ograniczająca parametr α_1:

$$0 \leq \alpha_1 \leq C$$

oraz prosta p

$$\alpha_1 y_1 + \alpha_2 y_2 = \alpha_1^{old} y_1 + \alpha_2^{old} y_2$$

po przekształceniu:

$$\alpha_1 = \alpha_1^{old} + y_1 y_2 \alpha_2^{old} - y_1 y_2 \alpha_2$$

Po podstawieniu powyższego równania do pierwszej nierówności otrzymuje się:

$$0 \leq \alpha_1^{old} + y_1 y_2 \alpha_2^{old} - y_1 y_2 \alpha_2 \leq C$$

Gdy $y_1 = y_2$, to $y_1 y_2 = 1$
Po uwzględnieniu powyższego:

$$0 \leq \alpha_1^{old} + \alpha_2^{old} - \alpha_2 \leq C$$

Następnie biorąc pod uwagę pierwszy człon nierówności:

$$\alpha_1^{old} + \alpha_2^{old} - \alpha_2 \geq 0$$

$$\alpha_2 \leq \alpha_1^{old} + \alpha_2^{old}$$

Jako, że parametr α_2 musi spełniać również nierówność $\alpha_2 \leq C$
to ograniczenie górne tego parametru wynosi $V = \min\left(C,\ \alpha_1^{old} + \alpha_2^{old}\right)$ dla $y_1 = y_2$
Biorąc pod uwagę drugi człon nierówności:

$$\alpha_1^{old} + \alpha_2^{old} - \alpha_2 \leq C$$

$$\alpha_2 \geq \alpha_1^{old} + \alpha_2^{old} - C$$

Jako, że parametr α_2 musi spełniać również nierówność $\alpha_2 \geq 0$ to ograniczenie dolne dla tego parametru wynosi $U = \max\left(0,\ \alpha_1^{old} + \alpha_2^{old} - C\right)$ dla $y_1 = y_2$.
Gdy$y_1 \neq y_2$, to $y_1 y_2 = -1$
Po uwzględnieniu powyższego:

$$0 \leq \alpha_1^{old} - \alpha_2^{old} + \alpha_2 \leq C$$

Następnie biorąc pod uwagę pierwszy człon nierówności:

$$\alpha_1^{old} - \alpha_2^{old} + \alpha_2 \geq 0$$

$$\alpha_2 \geq \alpha_2^{old} - \alpha_1^{old}$$

Jako, że parametr α_2 musi spełniać również nierówność $\alpha_2 \geq 0$, to ograniczenie dolne dla tego parametru wynosi $U = \max\left(0,\ \alpha_2^{old} - \alpha_1^{old}\right)$ dla $y_1 \neq y_2$.

Biorąc pod uwagę drugi człon nierówności:

$$\alpha_1^{old} - \alpha_2^{old} + \alpha_2 \leq C$$

$$\alpha_2 \leq C + \alpha_2^{old} - \alpha_1^{old}$$

Jako, że parametr α_2 musi spełniać również nierówność $\alpha_2 \leq C$
to ograniczenie górne tego parametru wynosi $V = \min\left(C,\ C - \alpha_1^{old} + \alpha_2^{old}\right)$ dla $y_1 \neq y_2$. □

A.2. Wyprowadzenie wzorów na rozwiązanie analityczne SMO

Nowe wartości parametrów są nastepujące:
Najpierw jest wyliczane α_2^{unc}

$$\alpha_2^{\text{unc}} = \alpha_2^{\text{old}} + \frac{y_2\left(E_1 - E_2\right)}{\kappa}$$

a następnie

$$\alpha_2 = \begin{cases} V, & jesli\ \alpha_2^{\text{new,unc}} > V, \\ \alpha_2^{\text{new,unc}}, & jesli\ U \leq \alpha_2^{\text{new,unc}} \leq V, \\ U, & jesli\ \alpha_2^{\text{new,unc}} < U, \end{cases}$$

$$\alpha_1^{\text{new}} = \alpha_1^{\text{old}} + y_1 y_2\left(\alpha_2^{\text{old}} - \alpha_2^{\text{new}}\right).$$

Dla uproszczenia dowodu wprowadza się oznaczenie:
$K\left(x_i, x_j\right) \equiv K_{ij}$,gdzie i, j =1, 2
i definiuje się:

$$g\left(\alpha_i\right) = \sum_{j=1}^{l} y_j \alpha_j K_{ij}$$

$$E_i = g\left(\alpha_i\right) - y_i = \sum_{j=1}^{l} y_j \alpha_j K_{ij} - y_i,$$

$$v_i = \sum_{j=3}^{l} y_j \alpha_j K_{ij} = g\left(\alpha_i\right) - \sum_{j=1}^{2} y_j \alpha_j K_{ij},$$

dla $i = 1$ lub $i = 2$
gdzie l jest liczbą wszystkich wektorów.
Funkcja celu ma postać:

$$W\left(\alpha_1, \alpha_2\right) = \alpha_1 + \alpha_2 - \tfrac{1}{2} K_{11}\alpha_1^2 - \tfrac{1}{2} K_{22}\alpha_2^2 - y_1 y_2 K_{12}\alpha_1\alpha_2 - y_1\alpha_1 v_1 - y_2\alpha_2 v_2 + const \quad .$$

Po dokonaniu podstawienia $s_{ij} = y_i y_j$ dla $i,\ j = 1,\ 2$ w celu uproszczenia zapisu zachodzi:

$$W(\alpha_1, \alpha_2) = \alpha_1 + \alpha_2 - \tfrac{1}{2}K_{11}\alpha_1^2 - \tfrac{1}{2}K_{22}\alpha_2^2 - s_{12}K_{12}\alpha_1\alpha_2 - y_1\alpha_1 v_1 - y_2\alpha_2 v_2 + const$$

Warunek liniowy ma postać: $\sum_{i=1}^{l} y_i\alpha_i = 0$.
Aby był prawdziwy, nowe wartości zmiennych muszą spełniać:

$$y_1\alpha_1 + y_2\alpha_2 = y_1\alpha_1^{old} + y_2\alpha_2^{old} = \text{const}$$

Dzieląc powyższe przez y_1 i korzystając z tego, że $y_1 y_2 = \frac{y_1}{y_2}$ otrzymuje się:

$$\alpha_1 + y_1 y_2\alpha_2 = \alpha_1^{old} + y_1 y_2\alpha_2^{old}$$

upraszczając zapis:

$$\alpha_1 + s_{12}\alpha_2 = \alpha_1^{old} + s_{12}\alpha_2^{old}$$

Przyjmując oznaczenie:

$$\gamma = \alpha_1^{old} + s_{12}\alpha_2^{old}$$

Zatem:

$$\alpha_1 + s_{12}\alpha_2 = \gamma$$

$$\alpha_1 = \gamma - s_{12}\alpha_2 \tag{A.2}$$

Powyższy wzór pokazuje jak z parametru α_2 otrzymać parametr α_1.
Po podstawieniu powyższego do funkcji celu otrzymuje się:

$$W(\alpha_1, \alpha_2) = \gamma - s_{12}\alpha_2 + \alpha_2 - \tfrac{1}{2}K_{11}(\gamma - s_{12}\alpha_2)^2 - \tfrac{1}{2}K_{22}\alpha_2^2 - s_{12}K_{12}(\gamma - s_{12}\alpha_2)\alpha_2 - y_1(\gamma - s_{12}\alpha_2)v_1 - y_2\alpha_2 v_2 + const$$

Po rozpisaniu kwadratu:

$$W(\alpha_1, \alpha_2) = \gamma - s_{12}\alpha_2 + \alpha_2 - \tfrac{1}{2}K_{11}(\gamma^2 - 2\gamma s_{12}\alpha_2 + s_{12}^2\alpha_2^2) - \tfrac{1}{2}K_{22}\alpha_2^2 - s_{12}K_{12}(\gamma - s_{12}\alpha_2)\alpha_2 - y_1(\gamma - s_{12}\alpha_2)v_1 - y_2\alpha_2 v_2 + const$$

Upraszczając:

$$W(\alpha_1, \alpha_2) = \gamma - s_{12}\alpha_2 + \alpha_2 - \tfrac{1}{2}K_{11}\gamma^2 + K_{11}\gamma s_{12}\alpha_2 - \tfrac{1}{2}K_{11}\alpha_2^2 - \tfrac{1}{2}K_{22}\alpha_2^2 - s_{12}K_{12}(\gamma - s_{12}\alpha_2)\alpha_2 - y_1(\gamma - s_{12}\alpha_2)v_1 - y_2\alpha_2 v_2 + const$$

Likwidując kolejne nawiasy:

$$W(\alpha_1,\alpha_2) = \gamma - s_{12}\alpha_2 + \alpha_2 - \tfrac{1}{2}K_{11}\gamma^2 + K_{11}\gamma s_{12}\alpha_2 - \tfrac{1}{2}K_{11}\alpha_2^2 - \tfrac{1}{2}K_{22}\alpha_2^2$$
$$-s_{12}K_{12}\gamma\alpha_2 + K_{12}\alpha_2^2 - y_1\gamma v_1 + y_2\alpha_2 v_1 - y_2\alpha_2 v_2 + const$$

Następnie licząc pochodną powyższej funkcji po α_2:

$$\frac{\partial W(\alpha_2)}{\partial \alpha_2} = 1 - s_{12} + K_{11}\gamma s_{12} - K_{11}\alpha_2 - K_{22}\alpha_2$$
$$-s_{12}K_{12}\gamma + 2K_{12}\alpha_2 + y_2 v_1 - y_2 v_2$$

Punkty stacjonarne zostaną znalezione, gdy pochodna W po α_2 zostanie przyrównana do zera:

$$\frac{\partial W(\alpha_2)}{\partial \alpha_2} = 1 - s_{12} + K_{11}\gamma s_{12} - K_{11}\alpha_2 - K_{22}\alpha_2 -$$
$$s_{12}K_{12}\gamma + 2K_{12}\alpha_2 + y_2 v_1 - y_2 v_2 = 0$$

$$1 - s_{12} + K_{11}\gamma s_{12} - K_{11}\alpha_2 - K_{22}\alpha_2$$
$$-s_{12}K_{12}\gamma + 2K_{12}\alpha_2 + y_2 v_1 - y_2 v_2 = 0$$

Dzieląc powyższe rownanie przez y_2:

$$y_2 - y_1 + K_{11}\gamma y_1 - K_{11}y_2\alpha_2 - K_{22}y_2\alpha_2 -$$
$$y_1 K_{12}\gamma + 2K_{12}y_2\alpha_2 + v_1 - v_2 = 0$$

Wprowadzając dodatkowe oznaczenie new dla parametru α_2:

$$y_2 - y_1 + K_{11}\gamma y_1 - K_{11}y_2\alpha_2^{\text{new}} - K_{22}y_2\alpha_2^{\text{new}} -$$
$$y_1 K_{12}\gamma + 2K_{12}y_2\alpha_2^{\text{new}} + v_1 - v_2 = 0$$

Podstawiając za γ, v_1 i v_2:

$$y_2 - y_1 + K_{11}(\alpha_1 + s_{12}\alpha_2)y_1 - K_{11}y_2\alpha_2^{\text{new}} - K_{22}y_2\alpha_2^{\text{new}}$$
$$-y_1 K_{12}(\alpha_1 + s_{12}\alpha_2) + 2K_{12}y_2\alpha_2^{\text{new}} + g(x_1) - y_1\alpha_1 K_{11} - y_2\alpha_2 K_{12}$$
$$-g(x_2) + y_1\alpha_1 K_{12} + y_2\alpha_2 K_{22} = 0$$

$$y_2 - y_1 + K_{11}\alpha_1 y_1 + K_{11}y_2\alpha_2 - K_{11}y_2\alpha_2^{\text{new}} - K_{22}y_2\alpha_2^{\text{new}}$$
$$-y_1 K_{12}\alpha_1 - K_{12}y_2\alpha_2 + 2K_{12}y_2\alpha_2^{\text{new}} + g(x_1) - y_1\alpha_1 K_{11} - y_2\alpha_2 K_{12}$$
$$-g(x_2) + y_1\alpha_1 K_{12} + y_2\alpha_2 K_{22} = 0$$

$$y_2 - y_1 + K_{11}y_2\alpha_2 - K_{11}y_2\alpha_2^{\text{new}} - K_{22}y_2\alpha_2^{\text{new}}$$
$$-K_{12}y_2\alpha_2 + 2K_{12}y_2\alpha_2^{\text{new}} + g(x_1) - y_2\alpha_2 K_{12}$$
$$-g(x_2) + y_2\alpha_2 K_{22} = 0$$

$$y_2 - y_1 - y_2\alpha_2^{\text{new}}(K_{11} + K_{22} - 2K_{12}) + y_2\alpha_2(K_{11} + K_{22} - 2K_{12})$$
$$+g(x_1) - g(x_2) = 0$$

Wprowadzając oznaczenie$\kappa = K_{11} + K_{22} - 2K_{12}$ otrzymuje się:

$$y_2 - y_1 - y_2\alpha_2^{\text{new}}\kappa + y_2\alpha_2\kappa$$
$$+g(x_1) - g(x_2) = 0$$

Dzieląc obie strony przez y_2 i przez κ:

$$\alpha_2^{\text{new}} = \alpha_2 + \frac{y_2 \left(E_1 - E_2\right)}{\kappa}$$

Na końcu konieczne jest ograniczenie α_2^{new}, tak aby parametr α_2^{new} pozostawał w zakresie $[U, V]$.

Dodatek B

Wyprowadzenia wzorów ASO

B.1. Wyprowadzenie wzoru na rozwiązanie analityczne ASO

Rozwiązanie analityczne problemu (PO 3.1.2) jest rozwiązaniem układu równań (3.1).

Dowód. Założenia:

Zbiór aktywny składa się z p parametrów $\alpha_1 .. \alpha_p$, gdzie $2 \leq p \leq l$, l to liczba wszystkich punktów.

Aby równość $\sum_{i=1}^{l} \alpha_i y_i = 0$ była spełniona, nowe wartości muszą spełniać:

$$\sum_{i=1}^{p} y_i \alpha_i = \sum_{i=1}^{p} y_i \alpha_i^{\text{old}} = \text{const}$$

oraz nierówności $0 \leq \alpha_1 \ldots \alpha_p \leq C$.

Przyjmując oznaczenie:

$$\gamma = \sum_{i=1}^{p} y_i \alpha_i^{\text{old}}$$

Z równania $\sum_{i=1}^{p} y_i \alpha_i = \sum_{i=1}^{p} y_i \alpha_i^{\text{old}}$ otrzymuje się wzór na dowolne α_k dla $k \in [1, p]$, najpierw dzieląc równanie linii przez y_k:

$$\sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik} \alpha_i + \alpha_k = \sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik} \alpha_i^{\text{old}} + \alpha_k^{\text{old}}$$

gdzie $s_{ij} = y_i y_j$

Przyjmując oznaczenie:

$$\gamma_k = \sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik}\alpha_i^{\text{old}} + \alpha_k^{\text{old}}$$

$$\gamma_k = \frac{\gamma}{y_k}$$

A zatem:

$$\sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik}\alpha_i + \alpha_k = \gamma_k$$

$$\alpha_k = \gamma_k - \sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik}\alpha_i \tag{B.1}$$

Kolejnym etapem jest wyprowadzenie wzoru:

$$\sum_{i=1}^{p-1} y_i \alpha_i^{\text{new}} \kappa_{ipk} = \sum_{i=1}^{p-1} y_i \alpha_i \kappa_{ipk} + E_k - E_p$$

gdzie $\kappa_{ipk} = K_{ip} + K_{kp} - K_{pp} - K_{ik}$.
Rozwiązaniem powyższego układu równań są wartości parametrów $\alpha_1^{new} ... \alpha_{p-1}^{new}$
Znaczenie wartości E_k i E_p jest takie same jak w przypadku algorytmu SMO.
Pozostały parametr α_p^{new} wyliczany jest ze wzoru:

$$\alpha_p = \gamma_p - \sum_{i=1}^{p-1} s_{ip}\alpha_i.$$

Główny dowód
Na początku zostaną wprowadzone oznaczenia:

$$g(\alpha_i) = \sum_{j=1}^{l} y_j \alpha_j K_{ij}$$

$$E_i = g(\alpha_i) - y_i = \sum_{j=1}^{l} y_j \alpha_j K_{ij} - y_i$$

$$v_i = \sum_{j=p+1}^{l} y_j \alpha_j K_{ij} = g(\alpha_i) - \sum_{j=1}^{p} y_j \alpha_j K_{ij}.$$

Funkcja celu jest postaci:

$$W(\alpha_1, \alpha_2, ..., \alpha_p) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2}\sum_{i=1}^{p} K_{ii}\alpha_i^2 - \sum_{\substack{i,j=1 \\ i \neq j}}^{p} y_i y_j K_{ij} \alpha_i \alpha_j - \sum_{i=1}^{p} y_i \alpha_i v_i + \text{const}$$

Po dokonaniu podstawienia $s_{ij} = y_i y_j$ w celu uproszczenia zapisu otrzymuje się:

$$W(\alpha_1, \alpha_2, ..., \alpha_p) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} K_{ii} \alpha_i^2 - \sum_{\substack{i,j=1 \\ i \neq j}}^{p} s_{ij} K_{ij} \alpha_i \alpha_j - \sum_{i=1}^{p} y_i \alpha_i v_i + \text{const}$$

Wykorzystując wzór (B.1): $\alpha_k = \gamma_k - \sum_{\substack{i=1 \\ i \neq k}}^{p} s_{ik} \alpha_i$ dla $k = p$ otrzymuje się:

$$\alpha_p = \gamma_p - \sum_{i=1}^{p-1} s_{ip} \alpha_i$$

Podstawiając α_p do funkcji celu:

$$W(\alpha_1, \alpha_2, ..., \alpha_{p-1}) = \sum_{i=1}^{p-1} \alpha_i + \gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{p-1} K_{ii} \alpha_i^2 - \tfrac{1}{2} K_{pp} \left(\gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i \right)^2$$
$$- \sum_{i=1}^{p-2} \sum_{j=i+1}^{p-1} s_{ij} K_{ij} \alpha_i \alpha_j - \sum_{i=1}^{p-1} s_{ip} K_{ip} \alpha_i \left(\gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i \right) - \sum_{i=1}^{p-1} y_i \alpha_i v_i - y_p \alpha_p v_p + \text{const}$$

Rozpisując kwadrat:

$$\begin{aligned} &W(\alpha_1, \alpha_2, ..., \alpha_{p-1}) = \sum_{i=1}^{p-1} \alpha_i + \gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{p-1} K_{ii} \alpha_i^2 \\ &- \tfrac{1}{2} K_{pp} \left(\gamma^2 + \sum_{i=1}^{p-1} \alpha_i^2 - 2\gamma \sum_{i=1}^{p-1} s_{in} \alpha_i + 2 \sum_{i=1}^{p-2} \sum_{j=i+1}^{p-1} s_{ij} \alpha_i \alpha_j \right) \\ &- \sum_{i=1}^{p-2} \sum_{j=i+1}^{p-1} s_{ij} K_{ij} \alpha_i \alpha_j - \sum_{i=1}^{p-1} s_{ip} K_{ip} \alpha_i \left(\gamma - \sum_{j=1}^{p-1} s_{jp} \alpha_j \right) - \sum_{i=1}^{p-1} y_i \alpha_i v_i \\ &- y_p \left(\gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i \right) v_p + \text{const} \end{aligned}$$

Upraszczając wyrażenia w nawiasach:

$$\begin{aligned} &W(\alpha_1, \alpha_2, ..., \alpha_{p-1}) = \sum_{i=1}^{p-1} \alpha_i + \gamma - \sum_{i=1}^{p-1} s_{ip} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{p-1} K_{ii} \alpha_i^2 \\ &- \tfrac{1}{2} K_{pp} \gamma^2 - \tfrac{1}{2} K_{pp} \sum_{i=1}^{p-1} \alpha_i^2 + K_{pp} \gamma \sum_{i=1}^{p-1} s_{ip} \alpha_i - K_{pp} \sum_{i=1}^{p-2} \sum_{j=i+1}^{p-1} s_{ij} \alpha_i \alpha_j \\ &- \sum_{i=1}^{p-2} \sum_{j=i+1}^{p-1} s_{ij} K_{ij} \alpha_i \alpha_j - \sum_{i=1}^{p-1} s_{ip} K_{ip} \alpha_i \gamma + \sum_{i=1}^{p-1} \sum_{j=1}^{p-1} s_{ip} K_{ip} \alpha_i s_{jp} \alpha_j \\ &- \sum_{i=1}^{p-1} y_i \alpha_i v_i - y_p v_p \gamma + y_p v_p \sum_{i=1}^{p-1} s_{ip} \alpha_i + \text{const} \end{aligned}$$

Rozpisując sumę:

$$W\left(\alpha_1,\alpha_2,...,\alpha_{p-1}\right)=\sum_{i=1}^{p-1}\alpha_i+\gamma-\sum_{i=1}^{p-1}s_{ip}\alpha_i-\tfrac{1}{2}\sum_{i=1}^{p-1}K_{ii}\alpha_i^2$$
$$-\tfrac{1}{2}K_{pp}\gamma^2-\tfrac{1}{2}K_{pp}\sum_{i=1}^{p-1}\alpha_i^2+K_{pp}\gamma\sum_{i=1}^{p-1}s_{ip}\alpha_i-K_{pp}\sum_{i=1}^{p-2}\sum_{j=i+1}^{p-1}s_{ij}\alpha_i\alpha_j$$
$$-\sum_{i=1}^{p-2}\sum_{j=i+1}^{p-1}s_{ij}K_{ij}\alpha_i\alpha_j-\sum_{i=1}^{p-1}s_{ip}K_{ip}\alpha_i\gamma+\sum_{i=1}^{p-1}s_{ip}K_{ip}\alpha_i s_{ip}\alpha_i+$$
$$\sum_{i=1}^{p-1}\sum_{\substack{j=1\\i\neq j}}^{p-1}s_{ip}K_{ip}\alpha_i s_{jp}\alpha_j-\sum_{i=1}^{p-1}y_i\alpha_i v_i-y_p v_p\gamma+y_p v_p\sum_{i=1}^{p-1}s_{ip}\alpha_i+\text{const}$$

Upraszczając dalej:

$$W\left(\alpha_1,\alpha_2,...,\alpha_{p-1}\right)=\sum_{i=1}^{p-1}\alpha_i+\gamma-\sum_{i=1}^{p-1}s_{ip}\alpha_i-\tfrac{1}{2}\sum_{i=1}^{p-1}K_{ii}\alpha_i^2$$
$$-\tfrac{1}{2}K_{pp}\gamma^2-\tfrac{1}{2}K_{pp}\sum_{i=1}^{p-1}\alpha_i^2+K_{pp}\gamma\sum_{i=1}^{p-1}s_{ip}\alpha_i-K_{pp}\sum_{i=1}^{p-2}\sum_{j=i+1}^{p-1}s_{ij}\alpha_i\alpha_j$$
$$-\sum_{i=1}^{p-2}\sum_{j=i+1}^{p-1}s_{ij}K_{ij}\alpha_i\alpha_j-\sum_{i=1}^{p-1}s_{ip}K_{ip}\alpha_i\gamma+\sum_{i=1}^{p-1}K_{ip}\alpha_i^2+$$
$$\sum_{i=1}^{p-1}\sum_{\substack{j=1\\i\neq j}}^{p-1}s_{ip}K_{ip}\alpha_i s_{jp}\alpha_j-\sum_{i=1}^{p-1}y_i\alpha_i v_i-y_p v_p\gamma+y_p v_p\sum_{i=1}^{p-1}s_{ip}\alpha_i+\text{const}$$

Następnie liczona jest pochodna powyższej funkcji W po α_k, gdzie $1\leq k\leq p-1$.

$$\frac{\partial W(\alpha_1,\alpha_2,...,\alpha_{p-1})}{\alpha_k}=1-s_{kp}-K_{kk}\alpha_k-K_{pp}\alpha_k+K_{pp}\gamma s_{kp}-K_{pp}\sum_{\substack{i=1\\i\neq k}}^{p-1}s_{ik}\alpha_i$$
$$-\sum_{\substack{i=1\\i\neq k}}^{p-1}s_{ik}K_{ik}\alpha_i-s_{kp}K_{kp}\gamma+2K_{kp}\alpha_k+\sum_{\substack{i=1\\i\neq k}}^{p-1}s_{ik}K_{in}\alpha_i+\sum_{\substack{i=1\\i\neq k}}^{p-1}s_{ki}K_{kp}\alpha_i-y_k v_k+y_k v_p$$

Wyłączając y_k przed nawias otrzymuje się:

$$\frac{\partial W(\alpha_1,\alpha_2,...,\alpha_{p-1})}{\alpha_k}=y_k\left(y_k-y_p-y_kK_{kk}\alpha_k-y_kK_{pp}\alpha_k+y_pK_{pp}\gamma-K_{pp}\sum_{\substack{i=1\\i\neq k}}^{p-1}y_i\alpha_i\right.$$
$$\left.-\sum_{\substack{i=1\\i\neq k}}^{p-1}y_iK_{ik}\alpha_i-y_pK_{kp}\gamma+2y_kK_{kp}\alpha_k+\sum_{\substack{i=1\\i\neq k}}^{p-1}y_iK_{ip}\alpha_i+\sum_{\substack{i=1\\i\neq k}}^{p-1}y_iK_{kp}\alpha_i-v_k+v_p\right)$$

Podstawiając za γ, v_1 i v_2:

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(y_k - y_p - y_k K_{kk}\alpha_k^{\text{new}} - y_k K_{pp}\alpha_k^{\text{new}} + y_p K_{pp}\left(\sum_{i=1}^{p-1} s_{ip}\alpha_i + \alpha_p\right) \right.$$
$$-K_{pp}\sum_{\substack{i=1\\i\neq k}}^{p-1} y_i\alpha_i^{\text{new}} - \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ik}\alpha_i^{\text{new}} - y_p K_{kp}\left(\sum_{i=1}^{p-1} s_{ip}\alpha_i + \alpha_p\right) + 2y_k K_{kp}\alpha_k^{\text{new}}$$
$$\left. + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ip}\alpha_i^{\text{new}} + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{kp}\alpha_i^{\text{new}} - g\left(x_k\right) + \sum_{i=1}^{p} y_i\alpha_i K_{ki} + g\left(x_p\right) - \sum_{i=1}^{p} y_i\alpha_i K_{pi} \right)$$

Upraszczając nawiasy:

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(y_k - y_p - y_k K_{kk}\alpha_k^{\text{new}} - y_k K_{pp}\alpha_k^{\text{new}} + y_p K_{pp}\sum_{i=1}^{p-1} s_{ip}\alpha_i + y_p K_{pp}\alpha_p \right.$$
$$-K_{pp}\sum_{\substack{i=1\\i\neq k}}^{p-1} y_i\alpha_i^{\text{new}} - \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ik}\alpha_i^{\text{new}} - y_p K_{kp}\sum_{i=1}^{p-1} s_{ip}\alpha_i - y_p K_{kp}\alpha_p + 2y_k K_{kp}\alpha_k^{\text{new}} + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ip}\alpha_i^{\text{new}}$$
$$\left. + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{kp}\alpha_i^{\text{new}} - g\left(x_k\right) + \sum_{i=1}^{p-1} y_i\alpha_i K_{ki} + y_p\alpha_p K_{kp} + g\left(x_p\right) - \sum_{i=1}^{p-1} y_i\alpha_i K_{pi} - y_p\alpha_p K_{pp} \right)$$

Likwidując podstawienie za s:

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(y_k - y_p - y_k K_{kk}\alpha_k^{\text{new}} - y_k K_{pp}\alpha_k^{\text{new}} + K_{pp}\sum_{i=1}^{p-1} y_i\alpha_i \right.$$
$$-K_{pp}\sum_{\substack{i=1\\i\neq k}}^{p-1} y_i\alpha_i^{\text{new}} - \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ik}\alpha_i^{\text{new}} - K_{kp}\sum_{i=1}^{p-1} y_i\alpha_i + 2y_k K_{kp}\alpha_k^{\text{new}} + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{ip}\alpha_i^{\text{new}}$$
$$\left. + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i K_{kp}\alpha_i^{\text{new}} - g\left(x_k\right) + \sum_{i=1}^{p-1} y_i\alpha_i K_{ki} + g\left(x_p\right) - \sum_{i=1}^{p-1} y_i\alpha_i K_{pi} \right)$$

Następnie wyłączając przed nawias:

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(y_k - y_p - y_k\alpha_k^{\text{new}}\left(K_{kk} + K_{pp} - 2K_{kp}\right) \right.$$
$$\left. - \sum_{i=1}^{p-1} y_i\alpha_i\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) + \sum_{\substack{i=1\\i\neq k}}^{p-1} y_i\alpha_i^{\text{new}}\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) - g\left(x_k\right) + g\left(x_p\right) \right)$$

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(y_k - y_p - \sum_{i=1}^{p-1} y_i\alpha_i\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) \right.$$
$$\left. + \sum_{i=1}^{p-1} y_i\alpha_i^{\text{new}}\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) - g\left(x_k\right) + g\left(x_p\right) \right)$$

$$\frac{\partial W\left(\alpha_1^{\text{new}},\alpha_2^{\text{new}},\ldots,\alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(\sum_{i=1}^{p-1} y_i\alpha_i^{\text{new}}\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) \right.$$
$$\left. - \sum_{i=1}^{p-1} y_i\alpha_i\left(K_{ip} + K_{kp} - K_{pp} - K_{ik}\right) - E_k + E_p \right)$$

Wprowadzając oznaczenie:

$$\kappa_{ipk} = K_{ip} + K_{kp} - K_{pp} - K_{ik}$$

otrzymuje się:

$$\frac{\partial W\left(\alpha_1^{\text{new}}, \alpha_2^{\text{new}}, ..., \alpha_{p-1}^{\text{new}}\right)}{\alpha_k^{\text{new}}} = y_k \left(\sum_{i=1}^{p-1} y_i \alpha_i^{\text{new}} \kappa_{ipk} - \sum_{i=1}^{p-1} y_i \alpha_i \kappa_{ipk} - E_k + E_p \right)$$

Wyznaczany jest następnie punkt stacjonarny poprzez przyrównanie pochodnej do zera:

$$y_k \left(\sum_{i=1}^{p-1} y_i \alpha_i^{\text{new}} \kappa_{ipk} - \sum_{i=1}^{p-1} y_i \alpha_i \kappa_{ipk} - E_k + E_p \right) = 0$$

$$\sum_{i=1}^{p-1} y_i \alpha_i^{\text{new}} \kappa_{ipk} = \sum_{i=1}^{p-1} y_i \alpha_i \kappa_{ipk} + E_k - E_p$$

□

Bibliografia

[1] Volker Blanz, Bernhard Schölkopf, Heinrich H. Bülthoff, Chris Burges, Vladimir Vapnik, and Thomas Vetter. Comparison of view-based object recognition algorithms using realistic 3d models. In *ICANN 96: Proceedings of the 1996 International Conference on Artificial Neural Networks*, pages 251–256, London, UK, 1996. Springer-Verlag.

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pages 144–152, 1992.

[3] Chris J. C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector machines. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 375. The MIT Press, 1997.

[4] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[5] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines (version 2.31).

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[7] http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html.

[8] http://neuron.tuke.sk/competition/.

[9] Chih-Wei Hsu and Chih-Jen Lin. A simple decomposition method for support vector machines. *Machine Learning*, 46:291–314, 2002.

[10] Witold Janowski. *Matematyka*. Państwowe Wydawnictwo Naukowe, Warszawa, 1965.

[11] T. Joachims. Making large-scale support vector machine learning practical. In A. Smola B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[12] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[13] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to platt's smo algorithm for svm classifier design, 1999.

[14] http://www-unix.mcs.anl.gov/ more/tron/.

[15] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In J. Principe, L. Gile, N. Morgan, and E. Wilson, editors, *Neural Networks for Signal Processing VII — Proceedings of the 1997 IEEE Workshop*, pages 276 – 285, New York, 1997. IEEE.

[16] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 130, Washington, DC, USA, 1997. IEEE Computer Society.

[17] http://en.wikipedia.org/wiki/Overfitting.

[18] John C. Platt. Fast training of support vector machines using sequential minimal optimization, 1999.

[19] Pai-Hsuen Chen Rong-En Fan and Chih-Jen Lin. Working set selection using the second order information for training svm, 2005.

[20] B. Sch, o Burges, and V. Vapnik. Extracting support data for a given task, 1995.

[21] B. Sch, o Burges, and V. Vapnik. Incorporating invariances in support vector learning machines, 1996.

[22] M. Schmidt and H. Gish. Speaker identification via support vector classifiers. In *Proc. ICASSP '96*, pages 105–108, Atlanta, GA, May 1996.

[23] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.

[24] V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation.

[25] V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.

[26] G. Zoutendijk. *Methods of Feasible Directions: a Study in Linear and Non-line ar Programming*. Elsevier, 1970.

Spis rysunków

Spis tablic