www.MatlabKar.com
متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/276255530

مروری بر ماشین های بردار پشتیبان

Research · May 2015

DOI: 10.13140/RG.2.1.4981.6487

CITATIONS
0

READS
2,446

1 author:

Mazdak Fatahi
Razi University

9 PUBLICATIONS **3** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project

Neuromorphic Hardware View project

All content following this page was uploaded by Mazdak Fatahi on 13 May 2015.

The user has requested enhancement of the downloaded file.

www.MatlabKar.com
متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

دانشکده فنی و مهندسی

گروه مهندسی کامپیوتر

عنوان :

مروری بر ماشین‌های بردار پشتیبان

Support Vector Machines: A survey

استاد مربوطه:

دکتر جهانشاه کبودیان

ارائه دهنده:

مزدک فتاحی

چکیده

استفاده از ماشین‌های بردار پشتیبان (SVM) که توسط Vapnik ارائه شد، بعنوان یکی از راه حل‌ها در یادگیری ماشین و تشخیص الگو گسترش یافته است. SVM پیشگویی‌های خود را با استفاده از ترکیبی خطی از تابع Kernel که بر روی مجموعه‌ای از داده‌های آموزشی با نام بردارهای پشتیبان عمل می‌کند، انجام می‌دهد. روش ارائه شده توسط SVM با روش‌های قابل مقایسه مانند Neural Network متفاوت می‌باشد، تعلیم SVM همواره می‌نیمم سراسری را پیدا می‌کند. ویژگی‌های یک SVM به مقدار زیادی به انتخاب kernel آن مربوط می‌شود. تعلیم SVM منجر به یک مسئله‌ی برنامه‌ریزی درجه‌ی دوم (QP) از نوع مقید می‌گردد که حل آن برای حجم زیادی از نمونه‌ها می‌تواند با روش‌های عددی بسیار مشکل باشد، لذا برای ساده کردن حل این مسئله‌ی بهینه‌سازی، روش‌های متعددی ارائه گردیده که متناسب با نیاز قابل استفاده و پیاده‌سازی می‌باشند. در این مقاله ابتدا راجع مسئله‌ی یادگیری ماشین، طبقه‌بندی و معرفی SVM بحث شده و سپس معادله‌ی مربوط به یادگیری در SVM بدست می‌آید. در پایان نیز تعدادی از روش‌های حل این معادله ذکر می‌گردند. در این میان به کاربردها و بسته‌های کاربردی برای کار با SVM نیز اشاره گردیده است.

کلمات کلیدی:SVM، بردار پشتیبان، بهینه‌سازی مقید، طبقه‌بندی،Classification

فهرست مطالب

۱- مقدمه ۱

۲- مروری بر دسته‌بندی کننده‌ی SVM ٤

۲-۱- حالت جدایی‌پذیر ٦

۳- SVM در حالت‌های غیرخطی و جدایی‌ناپذیر ۱۳

۳-۱- داده‌های جدایی ناپذیر ۱۳

۳-۲- ماشین‌های بردار پشتیبان غیر خطی ۱۵

۳-۲- ماشین‌های بردار پشتیبان به عنوان جداکننده‌ی چند کلاسه ۱۷

٤- راه‌حل‌های بهینه جهت حل معادله‌ی QP برای SVM ۱۸

٤-۱- Decomposition Method : ۲۰

٤-۲- SVM^{light} : ۲٤

٤-۳- Chunking ۲۵

٤-٤- SMO(Sequential Minimal Optimization) ۲۵

٤-٥- Core Vector Machines ۳۰

٤-٦- LSVM (Lagrangian SVM) ۳۲

٤-۷- Reduced SVM ۳۲

٤-۸- Relevance Vector Machine ۳۲

٤-۹- انتخاب مجموعه‌ی کاری ۳۳

٥- کاربردهای SVM ۳٥

٥-۱- تشخیص و شناسایی چهره ۳٥

٥-۲- تشخیص و شناسایی اشیاء ۳٥

٥-۳- تشخیص دست‌نوشته و ارقام ۳٦

۵-٤- تشخیص صحبت و گوینده ۳٦
۵-۵- بازیابی تصاویر و اطلاعات ۳٦
۵-٦- پیش‌بینی ۳۷
۵-۷- سایر کاربردها ۳۷
٦- ابزارهای پیاده‌سازی SVM ۳۸
٦-۱-MATLAB ۳۸
٦-۲-LIBSVM ٤۱
٦-۳- svm-struct-matlab ٤۲
۷- جمع‌بندی و نتیجه‌گیری ٤۲
پیوست ۱ ..... ٤۳
پیوست ۲: ٤٤
مراجع ٤۷

فهرست تصاویر

شکل ۱- مثالی از طبقه‌بندی تصاویر..... ۱

شکل ۲ - SVM به عنوان یک ابر صفحه برای جداسازی خطی نمونه‌ها در فضای داده‌ها..... ٤

شکل ۳- تفکیک داده‌ها در فضا توسط ابرصفحه‌های مختلف..... ۵

شکل ٤- با کاهش دقت نقاط آموزش، تفکیک‌کننده‌ای که از هر دو دسته بیشترین فاصله را دارد مناسب‌تر به نظر می‌رسد ۵

شکل ۵- جداسازی خطی در حالت دوبعدی..... ۷

شکل ٦- جداسازی خطی در حالت سه‌بعدی..... ۸

شکل ۷..... ۹

شکل ۸ - نمایش بردارهای پشتیبان..... ۱۲

شکل ۹- داده‌های آموزشی در حالت جدایی‌ناپذیر..... ۱۵

شکل ۱۰ – انتقال مسئله از فضای دو بعدی به سه بعدی..... ۱٦

شکل ۱۱- الف) یک زیر مسئله‌ی بهینه‌سازی که نقاط شرایط بهینه بودن را نقض کرده و داخل ناحیه‌ی ۱± قرار می‌گیرند. ب) ناحیه‌ی تصمیم‌گیری مجدداً تعریف می‌شود. از آنجایی که هیچ نقطه‌ای با $\alpha=0$ در فاصله‌ی ۱± قرار ندارد، لذا راه حل بهینه است. همانطور که دیده می‌شود حاشیه نیز کاهش یافته و شکل سطح تصمیم‌گیری متفاوت خواهد بود.[١٤] ۲۱

شکل ۱۲ - دو ضریب لاگرانژ باید تمامی قیدهای مطرح شده در مسئله را مرتفع نمایند. قید نامساوی در مسئله سبب می‌شود که ضرایب در چهارچوب محدود شده و شرط تساوی سبب واقع شدن آنها روی خط قطری می‌گردد. بنابراین SMO باید مقدار بهینه را روی این قطعه خط قطری بدست آورد..... ۲۷

شکل ۱۳- مسئله‌یMEB..... ۳۱

شکل 14- داده‌های آزمایشی تولید شده..... ۳۹

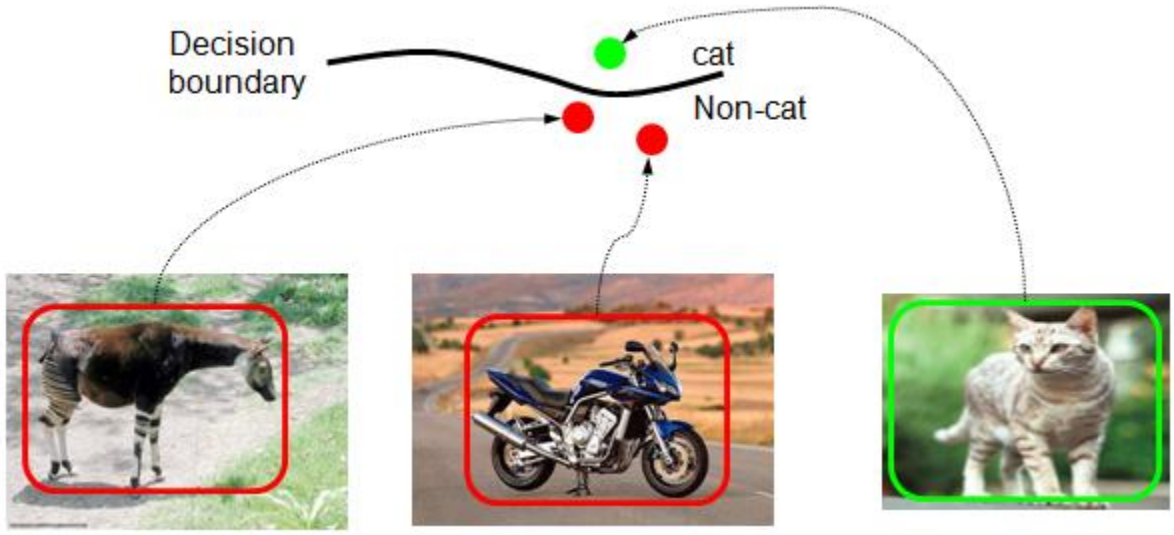
شکل ۱۵- داداه‌های دسته‌بندی شده با svm..... ٤۰

۱- مقدمه

در حالت کلی یادگیری ماشین به دو شکل نظارت شده و نظارت نشده انجام می‌شود. بسیاری از روش‌های یادگیری ماشین که به صورت نظارت شده عمل می‌نمایند، به این صورت کار می‌کنند که مجموعه‌ای از بردارهای ورودی مانند $X=\{x_n\}$ و بردارهای خروجی متناظر با آنها $T=\{t_n\}$ داده می‌شود. هدف این است که ماشین قادر باشد با استفاده از این داده‌های آموزشی برای ورودی x جدید، t را پیش‌بینی نماید.[6]

در این راستا می‌توان دو حالت متمایز را در نظر گرفت: *رگرسیون(Regression)*، که در آن t یک متغیر پیوسته است و *طبقه‌بندی کردن(Classification)*، که در آن t متعلق به یک مجموعه‌ی گسسته می‌باشد. بسیاری از مسائل یادگیری ماشین اساساً در دسته دوم واقع می‌شوند و هدف تعلیم ماشین به گونه‌ایست که ماشین بتواند بخوبی این دسته‌بندی را انجام دهد.

بعنوان مثال فرض کنید که هدف طراحی و تعلیم ماشینی است که قادر به تشخیص تصویر چهره(face) و غیر چهره (non-face) باشد. این سیستم عملاً کاری جز دسته‌بندی انجام نمی‌دهد. اگر برای هر ورودی (در این مثال تصویر ورودی) یک بردار n بعدی x فرض کنیم (مثلاً بردار ویژگی خاصی که از یک تصویر استخراج شده است) ماشین باید قادر باشد بردارهای x که بعنوان داده‌های تعلیمی هستند و در فضای n-بعدی به شکل نقاطی دیده می‌شوند را، دسته‌بندی نمایند. شکل زیر این دسته‌بندی را برای ماشینی نمایش می‌دهد که قادر به تشخیص تصویر گربه از سایر تصاویر می‌باشد.



شکل ۱- مثالی از طبقه‌بندی تصاویر

در فرآیند یادگیری نیاز است که ابتدا سیستم تعلیم یافته و سپس به‌ازای مقادیر ورودی جدید تست شود. به شکل ریاضی مسئله‌ی یادگیری ماشین را می‌توان به شکل یک نگاشت در نظر گرفت که طی آن $x_i \rightarrow y_i$. در واقع ماشین با مجموعه‌ای از نگاشت‌های ممکن به شکل $x \rightarrow f(x,\alpha)$ تعریف می‌شود که در آن خود توابع $f(x,\alpha)$ توسط برچسب α قابل تنظیم می‌باشند. فرض می‌شود که سیستم deterministic بوده و به‌ازای یک ورودی خاص x و انتخاب α ماشین همواره یک خروجی مشخص برابر با $f(x,\alpha)$ بدهد. انتخاب αی مناسب همان عملی‌است که یک ماشین تعلیم دیده (Trained Machine) انجام می‌دهد. بعنوان مثال یک شبکه‌ی عصبی که α متناظر با وزن‌ها و مقدار بایاس در آن باشد، یک ماشین یادگیری می‌باشد.

در اینجا بر حالتی تمرکز خواهیم داشت که پیشگویی $y(x,w)$ توسط ترکیب خطی از تابع پایه‌ی $\Phi_m(x)$ به فرم زیر بیان می‌شود:

$$y(x,w) = \sum_{m=0}^{M} w_m \phi_m(x) = w^T \phi$$

(۱)

که در آن w_m پارامترهای مدل می‌باشند که وزن نامیده می‌شوند. در SVM توابع پایه به عنوان توابع Kernel استفاده می‌شوند که به‌ازای هر x_m در مجموعه‌ی آموزشی داریم $\Phi_m(x)=K(x,x_m)$ که در آن $K(.,.)$ تابع Kernel می‌باشد.

تخمین وزن‌ها در SVM بوسیله‌ی بهینه‌سازی ضوابطی حاصل می‌شود که به صورت همزمان سعی در می‌نیمم کردن تابع $y(x,w)$ دارند. در نتیجه تعدادی از وزن‌ها صفر می‌شوند، که سبب می‌شود مدل خلوتی (Sparse Model) حاصل شود که مدیریت پیش‌گویی آن توسط رابطه‌ی (۱) و فقط وابسته به زیرمجموعه‌ای از تابع Kernel باشد.[6]

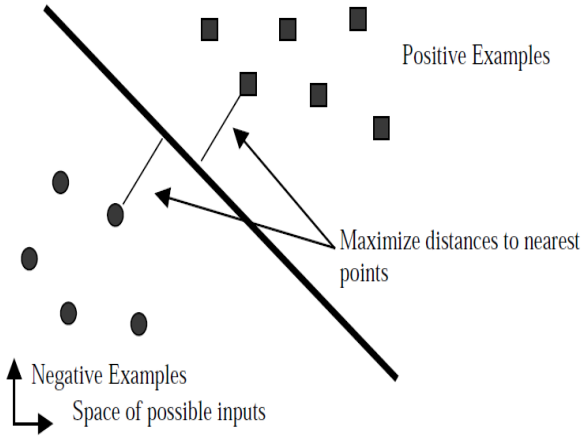
در سال‌های اخیر استفاده از ماشین‌های بردار پشتیبان(SVM)، بسیار مورد توجه قرار گرفته است. بطور تجربی نشان داده شده است که استفاده از SVM در کاربردهایی مانند تشخیص دست‌نوشته‌ها، تشخیص چهره و... نتایج خوبی را حاصل نموده است. تا زمان ارائه‌ی [3] استفاده از ماشین‌های بردار پشتیبان تنها به گروه خاصی از محققین محدود می‌شد. علت این محدودیت در استفاده از SVM، کندی الگوریتم آموزش برای این روش می‌باشد، خصوصاً در مورد مجموعه‌های آموزش بزرگ. به بیان دیگر الگوریتم آموزش SVM برای بسیاری از

مهندسین، پیچیده و دشوار بوده است و به همین دلیل راه‌حل‌ها و شیوه‌های جدیدی برای مواجه با این مسئله ارائه شده است.

در ادامه‌ی این مقاله و در بخش دوم مروری بر مفهوم دسته‌بندی که در SVM انجام می‌گیرد خواهیم داشت و مثال‌هایی از کاربردهای این دسته‌بندی کننده را ذکر می‌نماییم، سپس در بخش سوم ابتدا راجع مسائل بهینه‌سازی مقید و حل آنها به روش ضرائب لاگرانژ پرداخته و در ادامه مسئله مربوط به یادگیری در SVM را در ساده‌ترین حالت به شکل یک مسئله‌ی برنامه‌ریزی درجه‌ی دوم فرموله می‌نماییم و در بخش سوم راجع به سایر حالت‌های ممکن صحبت خواهیم کرد. در بخش چهارم تعدادی از راه‌حل‌ها که برای محاسبه‌ی کارآمد مسئله‌ی QP در SVM ارائه شده‌اند معرفی می‌شوند و در بخش پنجم به ابزارهایی که برای پیاده‌سازی SVM طراحی گردیده‌اند اشاره می‌گردد. در بخش ششم به کاربردهای SVM پرداخته و در پایان نیز در بخش هفتم به جمع‌بندی و نتیجه‌گیری خواهیم رسید.

۲- مروری بر دسته‌بندی کننده‌ی SVM

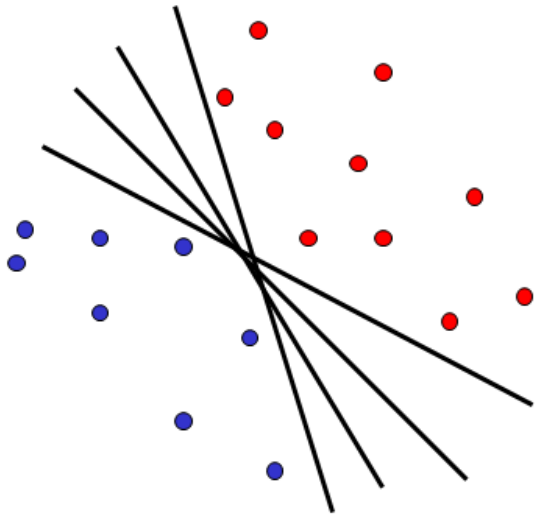
در ۱۹۷۹ ماشین‌های بردار پشتیبان(Support Vector Machines) توسط Vapnik ارائه گردید. در ساده‌ترین فرم آن یعنی SVM خطی، SVM عبارتست از *یک ابر صفحه که مجموعه‌ی نمونه‌های مثبت و منفی را با حداکثر فاصله(maximum margin) از هم جدا نموده است.* (شکل ۲)



شکل ۲ – SVM به عنوان یک ابر صفحه برای جداسازی خطی نمونه‌ها در فضای داده‌ها

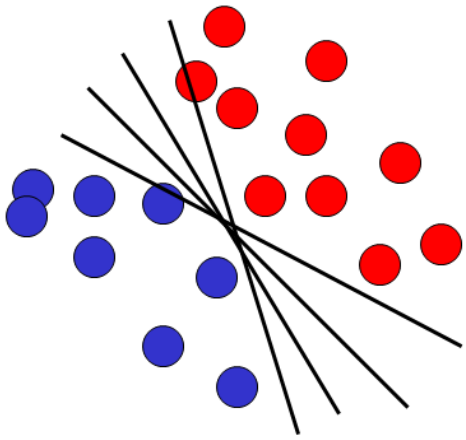
در شکل (۳) نمونه‌ای از داده‌های تفکیک‌پذیر (جدایی‌پذیر) در فضای دوبعدی دیده می‌شوند.(در ادامه گفته خواهد شد که این حالت ساده‌ترین حالت در دسته‌بندی (طبقه‌بندی) داده‌ها می‌باشد که از کمترین پیچیدگی برخوردار می‌باشد و دادها در آن با استفاده از یک ابرصفحه به صورت خطی به دو دسته‌ی کاملاً مجزا تفکیک می‌شوند) همانطور که دیده می شود این داده‌ها را می‌توان توسط چندین خط جداکننده (یا ابرصفحه در فضای n-بعدی) تفکیک و طبقه‌بندی نمود و در عین حال تمام این دسته‌بندی‌ها نیز صحیح می‌باشند.

در حالت کلی این مسئله را می‌توان در فضای n-بعدی در نظر گرفت که در آن داده‌ها در دو دسته واقع شده‌اند که این دو دسته به خوبی قابل جداسازی می‌باشند. در این حالت بجای خطوط جداکننده از ابر صفحات جداکننده (Hyperplane) استفاده خواهد شد. در اینجا سئوال این است که در میان این جداکننده‌ها کدامیک بهترین می‌باشد؟ برای این منظور باید به مفهوم ریسک‌پذیری دقت نمود. در واقع در اینجا حالتی را می‌خواهیم انتخاب نماییم که بیشترین دقت را در تفکیک‌کنندگی ارائه دهد، یعنی با وجود مقداری خطا و جابجایی در داده‌ها تفکیک‌کنندگی همچنان صحیح باشد. مثلاً حالتی را درنظر بگیرید که هرکدام از داده‌های دارای حاشیه‌ای از خطا باشند، در این



شکل ۳- تفکیک داده‌ها در فضا توسط ابرصفحه‌های مختلف

حالت اگر ابرصفحه‌ی جداکننده به اندازه‌ی کافی از داده‌های هر دسته دور نباشد، سبب خواهد شد که با اندکی خطا دادها از مرز جدا کنندگی عبور نمایند. شکل ٤ این حالت را نمایش می‌دهد.



شکل ٤- با کاهش دقت نقاط آموزش، تفکیک‌کننده‌ای که از هر دو دسته بیشترین فاصله را دارد مناسب‌تر به نظر می‌رسد

همانطور که دیده می‌شود، جدا کننده‌ای مناسب است که بیشترین فاصله‌را از هر دو دسته از داده‌ها داشته باشد. به عبارت دیگر سبب شود بیشترین حاشیه‌ی ریسک ایجاد شود. بنابراین در مورد این ماشین که خروجی آن

برچسب گروه‌ها (در اینجا ۱- و ۱+) می‌باشد، هدف بدست آوردن ماکزیمم فاصله بین دسته‌های داده می‌باشد. این حالت به داشتن *ماکزیمم حاشیه* تعبیر می‌شود.

در ارتباط با SVM می‌توان مسئله‌ی طبقه‌بندی داده‌ها را در چند حالت مختلف بررسی نمود:

۱- استفاده از ماشین بردار پشتبان خطی و وجود دو دسته داده:

الف - داده‌ها دقیقا در دو دسته قرار گرفته باشند، یعنی جدایی‌پذیر باشند. (Separable Data)

ب - داده‌های دو دسته قابل تفکیک به دو دسته‌ی جدا از هم نباشند. (Nonseparable Data)

۲-ماشین بردار پشتبان غیرخطی:

۳- ماشین بردار پشتیبان برای تفکیک‌کننده‌های چند کلاسه

۲-۱- حالت جدایی‌پذیر

ساده‌ترین حالت حالتی است که ماشین به صورت خطی روی داده‌های جدایی‌پذیر آموزش دیده باشد. جالب توجه می‌باشد که معادلات حاصل از این حالت قابل تعمیم به حالت غیرخطی و جدایی‌ناپذیر نیز می‌باشد، لذا این حالت اساس تعریف سایر حالات می‌باشد. لازم به ذکر است که در این وضعیت فرض شده که داده‌ها دقیقا در دو دسته قرار دارند.

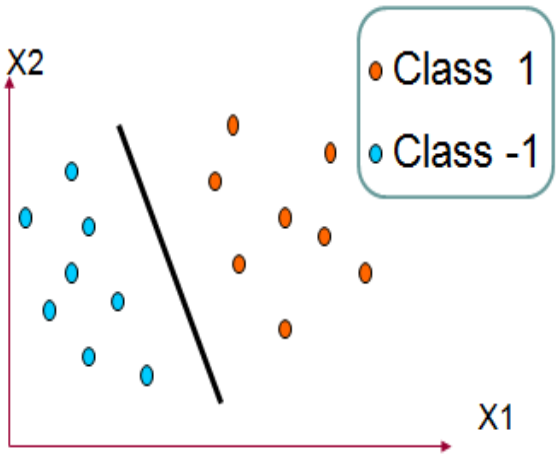
در این حالت فرض می‌شود که مجموعه‌ای از نمونه‌های آموزشی تفکیک‌پذیر وجود دارند که می‌توان آنها را با y_i برچسب زد. در این حالت نمونه‌ها به شکل زوج‌های مرتب (x_i,y_i) بیان می‌شوند که در آن $i=1,...,l$ و x_i متعلق به R^n و y_i دارای مقادیر $\{-1,1\}$ می‌باشند.[1]

همانطور که در شکل ۳ دیده شد داده‌ها را می‌توان توسط چندین خط جداکننده (یا ابرصفحه در فضای n-بعدی) تفکیک و طبقه‌بندی نمود. برای انتخاب بهترین ابرصفحه‌ی جداکننده با توجه به مفهوم تعریف شده در مورد *ماکزیمم حاشیه* باید عمل نمود. در حالت خطی "حاشیه" به صورت فاصله‌ی ابر صفحه تا نزدیک‌ترین نمونه‌ی منفی و مثبت تعریف می‌شود.[4] هر خط تفکیک کننده در فضای ۲ بعدی به شکل معادله‌ی خط زیر نوشته می‌شود:

$$w_1x_1 + w_2x_2 + b = 0$$

(۲)

شکل ۵ این موضوع را برای x_i ها که در دو کلاس با برچسب‌های ۱- و ۱+ واقع شده‌اند، در فضای دو بعدی نمایش می‌دهد:



شکل ۵- جداسازی خطی در حالت دوبعدی

خط جداکننده با فرمول ۲ نمایش داده می‌شود.

و در فضای ۳ بعدی به شکل زیر خواهد بود:

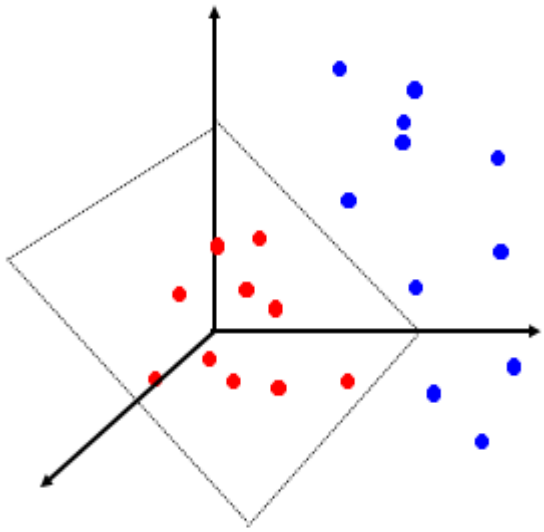
$$w_1x_1 + w_2x_2 + w_3x_3 + b = 0$$

(۳)

که در این حالت فرمول ۳ نمایش‌دهنده‌ی یک صفحه خواهد بود (شکل ٦). در حالت کلی و برای یک جداکننده از نوع ابرصفحه خواهیم داشت:

$$\sum_i w_i x_i + b = 0$$

(٤)



شکل ٦- جداسازی خطی در حالت سه‌بعدی

که در نمایش برداری به شکل زیر قابل بیان می‌باشد:

$$u = \vec{w}.\vec{x} + b$$

(٥)

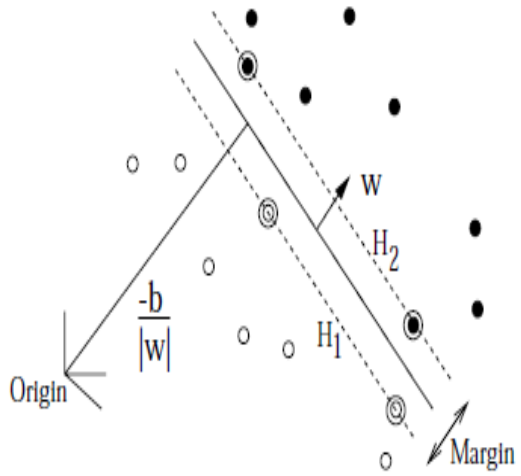
که در آن w بردار وزن بردار عمود بر ابرصفحه و b مقدار bias می‌باشد. در این نمایش u=0 اشاره به خود ابرصفحه‌ی جداکننده داشته و نزدیک‌ترین نقاط بر روی صفحه‌ی u=±1 قرار دارند. در واقع به فرض تفکیک‌پذیر بودن دو کلاس داده‌ی مثبت و منفی، بردارهای مرزی روی ابر صفحه‌های زیر قرار خواهند گرفت:

$$\vec{w}.\vec{x} + b = \pm 1$$

(٦)

ناحیه‌ی بین این دو ابرصفحه، حاشیه (Margin) گفته می‌شود.

شکل ٧ بیانگر حالت دو بعدی با فرض منفی بودن مقدار bias می‌باشد:



شکل ۷

همانطور که در شکل دیده می‌شود فضا به دو دسته از نمونه‌های با ویژگی‌های زیر تقسیم می‌شود:

$$\vec{w}.\vec{x} + b \geq +1 \qquad \text{for } y_i = 1$$

(۷)

$$\vec{w}.\vec{x} + b \leq -1 \qquad \text{for } y_i = -1$$

(۸)

می‌توان فرمول فوق را به شکل زیر ترکیب نمود:

$$y_i(\vec{w}.\vec{x} + b) - 1 \geq 0 \qquad \forall i$$

(۹)

در این حالت فاصله تا مبداء به صورت عمودی (نزدیکترین فاصله) برای نقاطی که روی ابر صفحه‌ی x.w+b=+1 قرار دارند برابر با :

$$\frac{|1 - b|}{||w||}$$

(۱۰)

و به شکل مشابه فاصله تا مبداء به صورت عمودی برای نقاطی که روی ابر صفحه‌ی x.w+b=-1 قرار دارند برابر با :

$$\frac{|-1-b|}{\|w\|}$$

(۱۱)

خواهد بود.[5] از طرفی فاصله‌ی مبداء تا ابرصفحه‌ی جداکننده برابر با :

$$\frac{|b|}{\|w\|}$$

(۱۲)

می‌باشد. بنابراین کمترین فاصله‌ی بین این ابرصفحه تا هر یک از صفحات مذکور به شکل زیر خواهد بود:

$$d_+ = d_- = \frac{1}{\|w\|}$$

(۱۳)

بنابراین حاشیه‌ی مذکور یعنی فاصله‌ی بین دو ابر صفحه‌ی مورد نظر به شکل زیر بدست می‌آید:

$$d_+ + d_- = \frac{2}{\|w\|}$$

(۱٤)

به این ترتیب حداکثر شدن حاشیه می‌تواند به شکل معادله‌ی بهینه‌سازی مقید زیر بیان شود:

$$\min \frac{1}{2}\|w\|^2 \qquad \text{subject to: } y_i(\vec{w}.\vec{x}+b)-1 \geq 0 \qquad \forall i$$

(۱۵)

برای حل این مساله ساده‌تر آن است که مساله بصورت دوگان(Dual) مطرح و حل شود. برای بدست آوردن شکل دوگان مسئله ضرایب لاگرانژ مثبت $\alpha_i \geq 0$ در قیود ضرب شده و از تابع هدف کسر می‌شود، در نتیجه به معادله‌ی زیر می‌رسیم که Primal Problem گفته می‌شود:

$$L_p = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i(y_i(\vec{w}.\vec{x}+b)-1) \qquad (۱٦)$$

شرایط Karush-Kuhn-Tucker (KKT) نقش مهمی را در مسائل بهینه‌سازی مقید، بازی می‌کنند. این شرایط، شرایط لازم و کافی برای داشتن پاسخ بهینه برای معادلات مقید را بیان می‌دارند و باید مشتق تابع نسبت به متغیرها برابر صفر باشد. با اعمال شرایط KKT روی L_p و اگر از L_p نسبت به w و b مشتق گرفته شده و برابر صفر قرار داده شود، خواهیم داشت:

$$w = \sum_i \alpha_i y_i x_i$$

(۱۷)

$$0 = \sum_i \alpha_i y_i$$

(۱۸)

در واقع در اینجا و برای Primal Problem (L_p)، شرایط KKT به شکل زیر خواهند بود:

$$\frac{\partial}{\partial w_v} L_p = w_v - \sum_i \alpha_i y_i x_{iv} = 0 \qquad v = 1, \dots, d$$

(۱۹)

$$\frac{\partial}{\partial b} L_p = -\sum_i \alpha_i y_i = 0$$

(۲۰)

$$y_i(\vec{w}.\vec{x} + b) - 1 \geq 0 \qquad i = 1, \dots, l$$
$$\alpha_i \geq 0 \quad \forall i$$

(۲۱)

$$\alpha_i(y_i(\vec{w}.\vec{x} + b) - 1) \geq 0 \qquad \forall i$$

(۲۲)

که از نتیجه‌ی آن برای بدست آوردن دوگان مسئله (L_D) استفاده شده است. شرایط KKT هر گونه مسئله‌ی بهینه‌سازی مقیدی را (چه محدب و چه غیر محدب) باهر نوع قیدی، برآورده می‌سازد. مسئله‌ی SVM یک مسئله‌ی محدب می‌باشد و برای مسائل محدب شرایط KKT شرایط لازم و کافی می‌باشند تا w,b,α بتوانند راه‌حل مسئله باشند. بنابراین حل مسئله‌ی SVM معادل است با یافتن یک راه حل برای شرایط KKT، که از آن در

روش‌های مختلفی برای یافتن راه حل SVM استفاده شده است. یک کاربرد از این شرایط این است که اگر w در روال آموزش تعیین شده و b مشخص نشده باشد، می‌توان b را بدست آورد.

با جایگذاری در معادله‌ی ۱۶ خواهیم داشت :

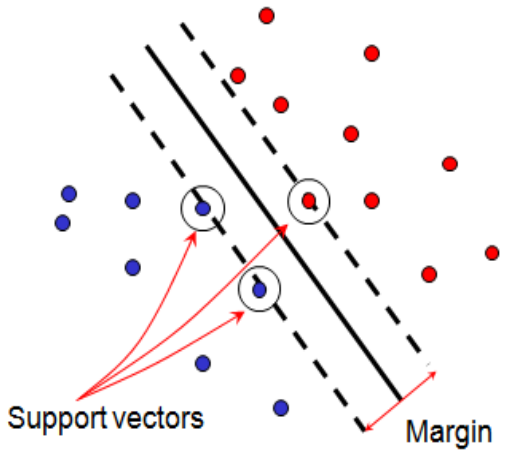
$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j (\vec{x_i}.\vec{x_j})$$

(۲۳)

که معادله‌ی دوگان[1] نامیده می‌شود. L_p و L_D هر دو از یک شرایط بدست آمده‌اند، لذا با محاسبه‌ی می‌نیمم L_p و یا ماکزیمم L_D (که دوگان Lp می‌باشد) با شرط $\alpha_i \geq 0$ می‌توان به جواب رسید.

برای هر نمونه‌ی آموزشی (x_i) یک ضریب لاگرانژ $\alpha_i \geq 0$ وجود دارد. در حل مساله به نمونه‌هایی که به ازای آنها $\alpha_i > 0$ می‌باشد، بردارهای پشتیبان یا "Support Vectors" گفته می‌شوند که بر روی ابر صفحات معادله‌ی (٦) قرار می‌گیرند (شکل ۸).

مسئله‌ی به دست آمده برای SVM یک مسئله‌ی بهینه‌سازی مقید درجه‌ی دوم و محدب می‌باشد.



شکل ۸ – نمایش بردارهای پشتیبان

[1] Dual Problem

قابل توجه می‌باشد که تمامی کارهایی که انجام می‌شود برای آن است که مسئله به یک مسئله‌ی بهینه‌سازی تبدیل شود که قیود آن قابل مدیریت‌تر باشند نسبت به (۷)و(۸). یافتن راه حل برای مسئله‌ی مطرح شده مستلزم روش‌های عددی می‌باشد که به‌ازای حجم زیاد داده‌ها بسیار مشکل و زمانبر خواهد بود. در قسمت‌های بعدی به راه‌حل‌هایی که برای کاهش پیچیدگی این محاسبات ارائه شده است اشاره خواهد گردید.

جهت تکمیل این بحث روش ضرائب لاگرانژ و شرایط KKT برای حل مسائل بهینه‌سازی مقید در ضمیمه‌ی ۲ آمده است.

۳- SVM در حالت‌های غیرخطی و جدایی‌ناپذیر

در این بخش سایر حالت‌های ممکن برای دسته‌بندی داده‌ها را بررسی می‌نماییم.

۳-۱- داده‌های جدایی ناپذیر

در این مورد فرض بر این است که دادهای موجود به سادگی قابل تفکیک به دو دسته نباشند.در واقع در این حالت که به واقعیت نیز نزدیک‌تر است، داده‌ها با نویز همراه بوده و لازم است SVM برای غلبه بر این حالت مقداری عمومی‌سازی (generalization) گردد. این وضعیت با تعریف یک حاشیه‌ی نرم(Soft Margin) قابل حل می‌باشد[9]، زیرا اگر الگوریتم فوق برای حالت جدایی‌پذیر به مجموعه‌ای از عناصر جدایی‌ناپذیر اعمال شود راه حل ممکنی پیدا نخواهد شد. برای گسترش ایده‌ی مطرح شده در بخش قبلی به این حالت، متغیری تحت عنوان متغیر مثبت کاهشی (کمبود)[۲] در قیود مسئله تعریف می‌شود و به این ترتیب خواهیم داشت:

$$x_i . w + b \geq +1 - \xi_i \quad \text{for } y_i = 1$$

(۲٤)

$$x_i . w + b \leq -1 + \xi_i \quad \text{for } y_i = -1$$

(۲۵)

$$\xi_i \geq 0 \quad \forall i$$

(۲٦)

بنابراین برای هر خطایی که رخ می‌دهد، هر ξ_i باید صورت جداگانه افزایش می‌یابد. بنابراین $\sum_i \xi_i$ به عنوان یک کران بالا برای تعداد خطاهای آموزش می‌باشد. در نتیجه بهترین راه برای نسبت دادن یک هزینه‌ی اضافی

[۲] Slack variable

برای خطاها، تغییر تابع هدف برای به حداقل رسانیدن تابع $\frac{\|w\|^2}{2} + C(\sum_i \xi_i)^k$ بجای $\frac{\|w\|^2}{2}$ می‌باشد. مقدار C توسط کاربر انتخاب می‌گردد. یک مقدار C بزرگ به معنی اعمال جریمه‌ی بیشتر برای خطاها می‌باشد. تابع مطرح شده به ازای هر مقدار مثبت از k تابعی محدب می‌باشد، خصوصاً به ازای k=1 و k=2 به یک مسئله‌ی برنامه‌ریزی درجه‌ی دوم تبدیل می‌شود. با محاسبه‌ی دوگان تابع مشابه با حالت جدایی‌پذیر، به ۲۷ می‌رسیم:

$$Maximize:\ L_D = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j (\vec{x_i}.\vec{x_j})$$

(۲۷)

$$subject\ to:$$

$$0 \le \alpha_i \le C,$$

$$\sum_i \alpha_i y_i = 0$$

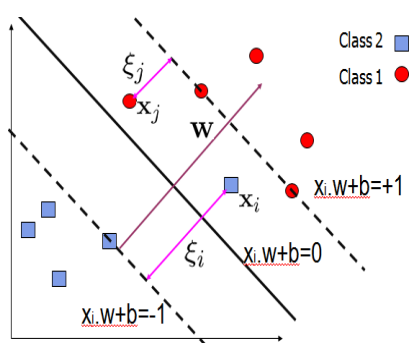
(۲۸)

در این شرایط راه حل از طریق ۲۹ ارائه می‌گردد:

$$w = \sum_{i=1}^{N_s} \alpha_i y_i x_i$$

(۲۹)

در ۲۹، N_S اشاره به تعداد بردارهای پشتیبان دارد. بنابراین تنها اختلاف با حالت ابرصفحه‌ی بهینه در این است که در این حالت α_i دارای کران بالای C می‌باشد.[5]



شکل ۹- داده‌های آموزشی در حالت جدایی‌ناپذیر

مسئله‌ی primal لاگرانژ به شکل ۳۰ خواهد بود:

$$L_p = \frac{\|w\|^2}{2} + C\left(\sum_i \xi_i\right)^k - \sum_i \alpha_i\{y_i(x_i.w + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i$$

(۳۰)

که در آن μ_i ضرائب لاگرانژ می‌باشند که سبب مثبت شدن ξ_i می‌گردند.

با اعمال شرایط KKT بر مسئله‌ی primal حاصل (L_p) خواهیم داشت:

$$\frac{\partial}{\partial w_v} L_p = w_v - \sum_i \alpha_i y_i x_{iv} = 0 \qquad v = 1, \dots, d$$

(۳۱)

$$\frac{\partial}{\partial b} L_p = -\sum_i \alpha_i y_i = 0$$

(۳۲)

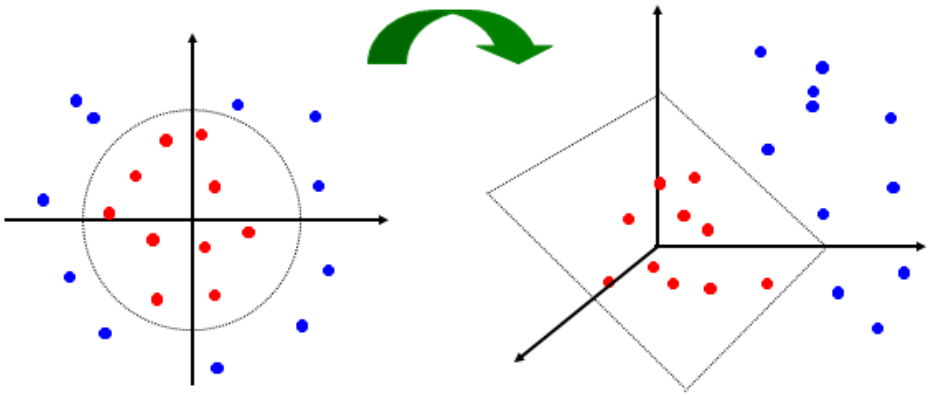
$$\frac{\partial}{\partial \xi_i} L_p = C - \alpha_i - \mu_i = 0$$

(۳۳)

که با اعمال این شرایط در L_p می‌توان به L_D (۲۷)رسید. پس از محاسبه‌ی ضرائب لاگرانژ، $0 \le \alpha_i \le C$ بردارهای پشتیبان برای این حالت می‌باشند.

۳-۲- ماشین‌های بردار پشتیبان غیر خطی

در حالتی که داده‌ها به سادگی از هم تفکیک نمی‌شوند، یک تفکیک‌کننده‌ی خطی نمی‌تواند موثر باشد. اما اگر داده‌ها را به فضایی با ابعاد بیشتر انتقال دهیم، می‌توان راه حلی برای تفکیک آنها به دست آورد. به عنوان مثال شکل ۱۰ مثالی از این انتقال را نمایش می‌دهد.



شکل ۱۰ – انتقال مسئله از فضای دو بعدی به سه بعدی

همانطور که دیده می‌شود پس این انتقال می‌توان کلاسه‌بندی را انجام داد. در حالت‌های ذکر شده در بخش‌های گذشته، در واقع از حاصلضرب داخلی داده‌های آموزشی ورودی در بردارهای پشتیبان (۲۷) برای تشکیل جداکننده‌ی خطی به شکل ابرصفحه استفاده شده است. در اینجا ابتدا داده‌ها را به یک فضای اقلیدسی با ابعاد بالاتر نگاشت نموده و سپس از ضرب داخلی عناصر بدست آمده استفاده می‌نماییم.[5]

$$\Phi: R^d \longrightarrow \mathcal{H}$$

(۳٤)

این فضای با ابعاد بالاتر و کاملاً برداری فضای هیلبرت خوانده می‌شود.[12] بنابراین الگوریتم آموزش وابسته به این ضرب داخلی در فضای H به شکل $\Phi(x_i).\Phi(x_j)$ خواهد شد. اگر یک تابع هسته[۳] به شکل زیر تعریف کینم:

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j)$$

(۳۵)

آنگاه در الگوریتم یادگیری تنها کافیست که بدون اینکه بطور صریح به Φ اشاره کنیم، فقط از K استفاده نماییم. شرایط فوق بیانگر یک جداسازی خطی ولی در ابعاد بالاتر می‌باشد.Boser, Guyon and Vapnik, 1992 نشان داده‌اند با این نگاشت و در صورتی که تابع هسته شرایط قضیه‌ی Mercer[٤] را دارا باشد، هسته‌ی مورد استفاده مناسب بوده و داده‌های

[۳] Kernel Function

[٤] Mercer's Condition

جدایی‌ناپذیر در فضای اصلی در فضای نگاشته شده، جدایی‌پذیر خواهند بود و لذا جایگزینی هسته می‌تواند الگوریتمی غیرخطی را از الگوریتم خطی ذکر شده ایجاد نماید.[5,9]

در زیر برخی از توابع Kernel پرکاربرد آمده‌اند:

$$K(x_i, x_j) = (x_i . x_j + 1)^p$$

(۳۶)

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$

(۳۷)

$$K(x_i, x_j) = \tanh(\beta x_i . x_j + \delta)$$

(۳۸)

به طور مشابه با داشتن هسته‌های ذکر شده دوگان مسئله‌ی بهینه‌سازی به شکل (٤٨) خواهد بود:

$$Maximize: \mathrm{L_D} = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

(۳۹)

$$subject\ to:$$
$$0 \le \alpha_i \le C,$$
$$\sum_i \alpha_i y_i = 0$$

(٤٠)

و همچنین به شکل مشابه بردارهای پشتیبان نمونه‌هایی هستند که ضرایب لاگرانژ متناظر آنها در رابطه‌ی $0 < \alpha_i \le C,$ صادق باشند.

۳-۲- ماشین‌های بردار پشتیبان به عنوان جداکننده‌ی چند کلاسه

در دنیای واقعی بسیاری از مسائل شامل دسته‌بندی‌های چند کلاسه می‌باشد و فقط به دسته‌بندی‌های دودویی محدود نمی‌شود. راه‌حل‌های مختلفی برای این دسته ارائه شده است. شکل ۱۰ یکی از ساده‌ترین راه‌حل‌ها را که در آن از یک دسته‌بندی دودویی استفاده می‌شود نمایش می‌دهد.[9] عموماً برای این دسته از کلاسه‌بندی‌ها از ترکیبی از روش‌های

دودویی استفاده می‌شود. یک روش ساده‌ی دیگر روش یکی در مقابل بقیه[۵] می‌باشد، در این روش یک دسته به عنوان داده‌های مثبت و بقیه منفی فرض می‌شوند. [5]

٤- راه‌حل‌های بهینه جهت حل معادله‌ی QP برای SVM

همانطور که دیده شد، آموزش ماشین بردارهای پشتیبان، منجر به حل مسئله‌ی بهینه‌سازی درجه‌ی دوم زیر می‌شود:

$$\text{Minimize: } W(\alpha) = -\sum_i \alpha_i + \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j K(x_i, x_j) \quad i,j \in \{1,\dots,l\}$$

(٤١)

$$\text{subject to :}$$
$$0 \le \alpha_i \le C,$$
$$\sum_i \alpha_i y_i = 0$$

(٤٢)

تعداد نمونه‌های آموزشی،l می‌باشد. αها برداری با l متغیر می‌باشند و α_iها نظیر به داده‌های آموزشی (x_i, y_i) می‌باشند و نیز $K: R^n \times R^n \to R$ تابع هسته و C ثابتی مثبت است. $\alpha^* = [\alpha_1, \dots, \alpha_l]^T$ با توجه به قیود ذکر شده بعنوان جواب معادله‌ی فوق می‌باشد. با تعریف ماتریس Q به شکل $(Q)_{ij} = y_i y_j k(x_i, x_j)$ می‌توان مسئله‌ی فوق را به شکل زیر بازنویسی نمود:[١٦]

$$\text{minimize } W(\alpha) = -\alpha^T 1 + \frac{1}{2}\alpha^T Q \alpha$$
$$\text{Subject to : } \alpha^T y = 0$$
$$0 \le \alpha \le C1$$

(٤٣)

سختی حل مسئله‌ی QP مطرح شده در اندازه ماتریس ایجاد شده می‌باشد(ماتریس Q) که عناصر آن عموماً غیر صفر می‌باشند، بنابراین حجم زیادی از حافظه نیاز است که بتوان این ماتریس و محاسبات مربوطه را در آن گنجانید. همچنین روش‌های سنتی مانند روش‌های نیوتنی و یا شبه- نیوتنی[٦] و ... قادر به حل این مسئله به صورت مستقیم نخواهند بود.

---

[۵] one-versus-rest classifiers

[٦] Quasi-Newton

مسئله‌ی بهینه‌سازی بردارهای پشتیبان فقط در صورتی می‌تواند به شکل تحلیلی حل شود که مجموعه‌ی آموزش عناصر بسیار کمی داشته باشد و یا در مورد تفکیک‌پذیر، زمانی که از قبل بدانیم که کدام عناصر از داده‌های آموزشی Support Vector هستند. SVM از توابع Kernel استفاده می‌نماید. بسیاری از توابع Kernel به شکل برنامه‌ریزی درجه‌ی دوم فرموله می‌شوند. اگر m تعداد داده‌های آموزشی باشد، پیچیدگی مرتبه‌ی زمانی برای یک مسئله‌ی برنامه‌ریزی درجه‌ی دوم، از مرتبه‌ی $O(m^3)$ و پیچیدگی فضایی آن از مرتبه‌ی حداقل درجه‌ی دوم می‌باشد. در مثال دنیای واقعی معادله‌ی بهینه‌سازی SVM از نوع برنامه‌ریزی درجه‌ی دوم، باید به صورت عددی محاسبه گردد. در مواردی با اندازه کوچک، هر راه حلی که برای برنامه‌ریزی خطی درجه‌ی دوم از نوع محدب وجود دارد، قابل استفاده می‌باشد.
با توجه به ابعاد بسیار بزرگ مسئله‌ی QP ارائه شده در حالت‌های مختلف در بخش قبل(L_D) که از SVM بدست می‌آید، می‌توان فهمید که این مسئله به سادگی و با روش‌های حل بهینه‌سازی درجه‌ی دوم قابل حل نمی‌باشد. مسئله‌ی درجه‌ی دوم ارائه شده شامل ماتریسی می‌باشد که تعداد عناصر آن مربع تعداد عناصر مجموعه‌ی آموزشی می‌باشد..

برای مسائل بزرگ روش‌های و تکنیک‌های مختلفی وجود دارد که در آنها بر خلاف روش نیوتن که حل ماتریس Hessian مربوط به کل مسئله را درمحاسبات دخالت می‌دهد، فقط قسمتی از ماتریس که در هر تکرار نیاز می‌باشد، مورد استفاده قرار می‌گیرد.[۱۵]

اساسا این راه‌حل‌ها به صورت تکراری عمل نموده و در هر تکرار مجموعه‌ای کوچک از متغیرها که به مجموعه‌ی کاری[۷] معروف است را انتخاب نموده و با توجه به این مجموعه مسئله‌ی QP را حل می‌نمایند.

ایده‌ی کلی به شکل مراحل اساسی زیر قابل بیان می‌باشند:[5]

۱- توجه به شرایط بهینه بودن(KKT) که راه حل باید آنرا دارا باشد.

۲- تعریف یک استراتژی برای رسیدن به حالت بهینه با کاهش یکنواخت دوگان تابع هدف با در نظر گرفتن قیود مسئله.

۳- تصمیم‌گیری برای یک الگوریتم تجزیه بطوریکه در هر زمان تنها بخش‌هایی از داده‌های آموزش نیاز به مدیریت داشته باشند.

برای کاهش پیچیدگی زمانی و فضایی، روش‌های مختلفی برای تخمین ماتریس Kernel کاهش‌یافته با استفاده از روش Nystrom (Williams and Seeger, 2001) ، تخمین حریصانه (Smola and Scholkopf, 2000) ، نمونه‌برداری (Achlioptas et al., 2002) و تجزیه‌ی ماتریس(Fine and Scheinberg, 2001) ابداع شده است. از میان کاره‌های انجام

[۷] Working Set

شده، روش Decomposition Method [١٤]، از همه مهمتر می‌باشد. به همین سبب در قسمت بعد، ابتدا این روش بررسی شده و سپس به تعدادی دیگر از روش‌ها به صورت گذرا اشاره خواهد شد.:

٤-١- Decomposition Method :

در ١٩٩٧ Osuna اثبات نمود که یک مسئله‌ی بزرگ QP می‌تواند به یک سری از زیر مسئله‌های QP کوچکتر شکسته شود. تا زمانی که حداقل یک نمونه که شرایط KKT را محقق نمی‌نماید(نقض می‌نماید)، به نمونه‌های زیر مسئله‌ی قبلی اضافه شده، هر مرحله کل تابع هدف را کاهش می‌دهد و نقطه‌ای را که قیود را رعایت می‌کند نگهداری می‌نماید. بنابراین دنباله‌ای از زیر مسئله‌های QP که همیشه حداقل یک عنصر را اضافه می‌کنند همگرایی را گارانتی می‌نماید. [4]

الگوریتم کلی که در این روش استفاده می‌شود، اساس بسیاری از روش‌های دیگر می‌باشد. هر روش مبتنی بر تجزیه[٨]، ٣ ویژگی اصلی دارد:[٢١]

- نحوه‌ی انتخاب مجموعه‌ی کاری
- نحوه‌ی حل زیر مسئله‌ی ایجاد شده
- شرایط پایان الگوریتم

این روش که توسط Osuna در [١٤] مطرح شده است، مسئله‌ی بهینه‌سازی بدست آمده را به دو قسمت inactive و active (یا همان working set) تقسیم می‌نماید. [١٦]

در [١٤] بهینه‌سازی سراسری را تضمین نموده و می‌تواند در زمانی که حجم داده‌های آموزشی بسیار بالا می‌باشد، کارآمد باشد. ایده‌ی اصلی در این الگوریتم حل زیر مسئله‌ها و بررسی شرایط بهینه بودن در هر تکرار می‌باشد. همچنین در این مقاله این الگوریتم برای کاربرد تشخیص چهره بکار رفته که تعداد نمونه‌ها در آن ٥٠٠٠٠ نمونه بوده است.

حافظه‌ی مورد نیاز برای حل یک مسئله‌ی QP با ابعاد مثال ارائه شده در این مقاله (ماتریسی با ابعاد ٥٠٠٠٠*٥٠٠٠٠) و با فرض اینکه برای هر نمونه تعداد ٨ بایت اختصاص داده شده باشد، 20GB خواهد بود.

برای ارائه‌ی الگوریتم مطرح شده ابتدا ٢ شرط مطرح می‌شود:

١- شرایط بهینه بودن[٩] : این شرایط به ما اجازه می‌دهد که به صورت محاسباتی در مورد بهینه شدن راه حل در یک تکرار خاص تصمیم‌گیری نماییم.

---

[٨] Decomposition Method

[٩] Optimality Conditions

۲- استراتژی بهبود[۱۰] : اگر یک راه‌حل خاص بهینه نباشد، این استراتژی راهی برای بهبود تابع هزینه ارائه می‌دهد. این استراتژی عمدتاً در ارتباط با متغیرهایی می‌باشد که شرایط بهینه بودن را نقض می‌نمایند.

الگوریتم Decomposition :

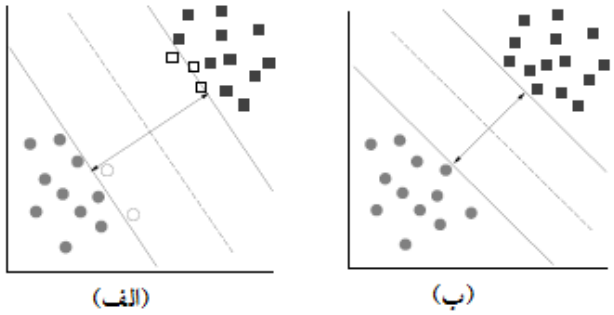
در هر تکرار α_iهای مرتبط به مسئله، به دو دسته تقسیم می‌شوند(B,N). فرض کنید بتوان مجموعه‌ی کاری B را طوری انتخاب نمود که $|B|\leq l$ و همچنین به اندازه‌ی کافی بزرگ باشد که بتواند شامل بردارهای پشتیبان باشد($\alpha_i > 0$)و نیز به اندازه‌ای نیز کوچک باشد که انجام محاسبات روی آن مقدور باشد(توسط کامپیوتر) . عناصر مجموعه‌ی B شرایط بهینه بودن را دارا هستند و سایر عناصر که این شرایط را نقض می‌کنند در مجموعه‌ی N دسته‌بندی می‌گردند. با این شرایط الگوریتم به شکل زیر خواهد بود :

۱- به صورت دلخواه |B| نقطه را از مجموعه‌ی داده‌ها انتخاب کنید.

۲- زیر مسئله‌ای را که توسط متغیرهای B تعریف می‌شود حل کنید.

۳- تا زمانی که $j\in N$ وجود دارد بگونه‌ای که $\alpha_i = 0, i \in B، g(x_j)y_j < 1$ را با $\alpha_j = 0$ جایگزین نموده و زیر مسئله‌ی جدید را حل کنید. بعبارت دیگر یعنی تا زمانی که شرایط بهینه بودن نقض می‌شود $g(x_i) = (\sum_{i=1}^{l} \alpha_j u_j K(x_i, x_j) + b$

باید توجه نمود که از آنجایی که این الگوریتم به صورت تکه‌ای تابع هدف را در هر تکرار بهبود می‌بخشد، لذا دارای



شکل ۱۱- الف) یک زیر مسئله‌ی بهینه‌سازی که نقاط شرایط بهینه بودن را نقض کرده و داخل ناحیه‌ی ۱± قرار می‌گیرند. ب) ناحیه‌ی تصمیم‌گیری مجدداً تعریف می‌شود. از آنجایی که هیچ نقطه‌ای با α=0 در فاصله‌ی ۱± قرار ندارد، لذا راه حل بهینه است. همانطور که دیده می‌شود حاشیه نیز کاهش یافته و شکل سطح تصمیم‌گیری متفاوت خواهد بود.[۱٤]

[۱۰] Strategy for Improvement

دور[۱۱] نبوده و تابع هدف دارای کران می‌باشد (محدب بوده و توسط ناحیه‌ی امکان‌پذیر، محدود می‌باشد) و الگوریتم در تعداد محدودی تکرار به مقدار بهینه‌ی سراسری همگرا خواهد شد. شکل ۱۱، تفسیر هندسی از روشی را نشان می‌دهد که الگوریتم برای تعریف مجدد سطح جدا کننده با افزودن نقاطی که شروط بهینه را نقض می‌نمایند، بکار می‌برد.

در [۱٦] روش فوق به شکل الگوریتم زیر آورده شده است :

While (the optimality conditions are violated)
Select q variables for the working set B. The remaining l-q variables are fixed at their current value.
Decompose problem and **solve** QP-subproblem: **optimize** W(α) on B.
Terminate and **return** α.

از آنجایی که ماتریس Q، positive-semidefinite می‌باشد و تمام قیود ذکر شده خطی می‌باشند، لذا مسئله‌ی مورد نظر یک مسئله‌ی بهینه‌سازی محدب خواهد بود. برای این دسته از مسائل شرایط KKT، شرایط لازم و کافی برای بهینه بودن می‌باشند. با توجه به قیود مسئله، یعنی :

$$\text{Subject to}: \alpha^T y = 0$$
$$0 \le \alpha \le C1$$
(٤٤)

ضرائب لاگرانژ را به این ترتیب علامت‌گذاری می‌نماییم: برای شرایط تساوی با λ^{eq} و برای کران بالا λ^{up}و پایین با λ^{lo}. به این ترتیب α بهینه خواهد بود اگر λ^{eq}، λ^{up}و λ^{lo} وجود داشته باشند، بنابراین با اعمال شرایط KKT خواهیم داشت:

$$g(\alpha) + \left(\lambda^{eq} y - \lambda^{lo} + \lambda^{up}\right) = 0$$
(٤٥)

$$\forall i \in [1..l]: \qquad \lambda_i^{lo}(-\alpha_i) = 0$$
$$\forall i \in [1..l]: \qquad \lambda_i^{up}(\alpha_i - C) = 0$$
$$\lambda^{lo} \ge 0$$
$$\lambda^{up} \ge 0$$
$$\alpha^T y = 0$$
$$0 \le \alpha \le C1$$
(٤٦)

[۱۱] cycle

$g(\alpha)$ بردار مشتق‌های جزئی در α است. برای این مسئله داریم:

$$g(\alpha) = -1 + Q\alpha$$

(٤٧)

همانطور که در الگوریتم مشخص است، اگر شرایط بهینه بودن وجود نداشته باشد، الگوریتم مسئله را می‌شکند و به حل مسئله‌ی کوچکتر می‌پردازد. با توجه به تقسیم نمونه‌ها در دو دسته‌ی B و N، می‌توان ماتریس Q و متغیرهای αو y را نیز متناظر با Bو N اندیس‌گذاری نمود:

$$\alpha = \begin{vmatrix} \alpha_B \\ \alpha_N \end{vmatrix} \quad y = \begin{vmatrix} y_B \\ y_N \end{vmatrix} \quad Q = \begin{vmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{vmatrix}$$

(٤٨)

با توجه به تقارن Q داریم که $Q_{NB} = Q_{BN}^T$ ، در نتیجه می‌توان مسئله را به شکل زیر باز نویسی نمود:

$$W(\alpha) = -\boldsymbol{\alpha}_B^T(\mathbf{1} - Q_{BN}\boldsymbol{\alpha}_N) + \frac{1}{2}\boldsymbol{\alpha}_B^T Q_{BB}\boldsymbol{\alpha}_B + \frac{1}{2}\boldsymbol{\alpha}_N^T Q_{NN}\boldsymbol{\alpha}_N - \boldsymbol{\alpha}_N^T\mathbf{1}$$

(٤٩)

$$subject\ to: \quad \boldsymbol{\alpha}_B^T\boldsymbol{y}_B + \boldsymbol{\alpha}_N^T\boldsymbol{y}_N = 0$$
$$\mathbf{0} \le \boldsymbol{\alpha} \le C\mathbf{1}$$

(٥٠)

با توجه به اینکه متغیرهای مجموعه‌ی N به صورت ثابت فرض می‌شوند، لذا عبارت $\frac{1}{2}\boldsymbol{\alpha}_N^T Q_{NN}\boldsymbol{\alpha}_N - \boldsymbol{\alpha}_N^T\mathbf{1}$ مقداری ثابت خواهد بود. مسئله‌ی به دست آمده یک مسئله‌ی بهینه‌سازی درجه دوم از نوع positive semidefinite می‌باشد که با توجه به کوچک بودن نسبی ابعاد مسئله، با استفاده از روش‌های مرسوم قابل حل می‌باشد. در این روال مهمترین فرایند *انتخاب αiها* برای تشکیل B می‌باشد، توجه داریم که B در هر تکرار به‌روز می‌شود. بنابراین سرعت روش حل می‌تواند وابسته به *انتخاب صحیح مجموعه‌ی کاری* باشد.

در خصوص همگرایی این روش در [٢٣] یک حالت ساده از شرایط بهینه بودن برای مسئله فرض شده و سپس نشان داده شده است که این روش می‌تواند با رعایت شرایط بهینه بودن مفروض، پس از تعداد محدودی تکرار به جواب بهینه همگرا شود.

٤-٢- SVM^{light}:

در [١٦] روشی ارائه شده با عنوان SVM^{light} که در آن تعداد ثابتی، که می‌تواند هر عدد زوج کمتر از تعداد نمونه‌های آموزشی باشد، از متغیرها با استفاده از روش Steepest Descent به صورت متقارن، برای مجموعه‌ی کاری انتخاب می‌شوند و زیر مسئله‌ی ایجاد شده با روش‌های سنتی قابل حل می‌باشد.

پس از انتخاب مجموعه‌ی کاری، باید از بین متغیرهای به گونه‌ای انتخاب انجام شود که در تکرار فعلی بیشترین پیشرفت در بهبود تابع هدف ایجاد گردد. در [١٦] از روش Steepest Descent برای یافتن بهترین جهت نزولی با q عنصر غیر صفر استفاده می‌نماید. برای این منظور مسئله‌ی بهینه‌سازی زیر حل می‌شود:

$$minimize \quad v(\boldsymbol{d}) = g(\boldsymbol{\alpha}^t)^T \boldsymbol{d}$$

(51)

$$subject\ to: \quad \boldsymbol{y}^T \boldsymbol{d} = \boldsymbol{0}$$

$$d_i \geq 0 \qquad for\ i: \alpha_i = 0$$

$$d_i \leq 0 \qquad for\ i: \alpha_i = C$$

(52)

$$-\boldsymbol{1} \leq \boldsymbol{d} \leq \boldsymbol{1}$$

$$|\{d_i : d_i \neq 0\}| = q$$

(53)

همانطور که قبلاً اشاره شد، g(α) بردار مشتق‌های جزئی در α است. مسئله‌ی بیان شده نشان می‌دهد که جهت نزولی در نقطه‌ی $\boldsymbol{\alpha}^t$ مورد نظر می‌باشد. این روش مجموعه‌ی کاری را با بیشترین شیب نزولی انتخاب می‌نماید.

در [٢٢] در خصوص همگرایی این روش بحث شده و آمده است که تعداد اعضای مجموعه‌ی کاری هر عدد زوجی می‌تواند باشد.

در این میان برخی روش‌های دیگر نیز وجود دارند که با ایجاد تغییراتی در روش‌های ذکر شده ایجاد شده‌اند. بعنوان مثال، در [٢١] روشی برای انتخاب مجموعه‌ی کاری ارائه شده است که مبتنی بر روش Conjugate Gradient می‌باشد. همچنین در [٢١] نشان داده شده است که روش تجزیه با توجه به مجموعه‌ی کاری انتخاب شده، همیشه در پس از تعداد محدودی تکرار پایان می‌یابد و جواب بهینه را پیدا خواهد نمود. تفاوت این روش با روش SVM^{light} در روش انتخاب متغیرهای مجموعه‌ی کاری می‌باشد. در SVM^{light} از با استفاده از روش Steepest Decent عناصر مجموعه‌ی کاری انتخاب می‌شوند ولی در این روش از Conjugate Gradient استفاده می‌نماید.روش GC روشی کارا در حل مسائل

درجه‌ی دوم نامقید است. در این روش فرض می‌شود که $h(\alpha)$ تابع هدف باشد، بناب این جهت $d(k)$ در تکرار k–ام خواهد بود:

$$d(k) = -\nabla h\big(\alpha(k)\big) + \beta(k)d(k-1), \quad k = 0,1,2,\dots$$

(٥٤)

$$\beta(0) = 0, \qquad \beta(k) = \frac{\left\|\nabla h\big(\alpha(k)\big)\right\|^2}{\left\|\nabla h\big(\alpha(k-1)\big)\right\|^2}, k = 1,2,\dots$$

(٥٥)

در [٢١] نشان داده شده است که این الگوریتم در تعداد محدودی تکرار متوقف خواهد شد.

٤-٣- Chunking

Vapnik روشی را برای حل QP SVM مطرح نمود که به الگوریتم *chunking* معروف می‌باشد و اولین الگوریتم ارائه شده از این دسته می‌باشد. این روش براساس این واقعیت استوار است که مقدار در درجه‌ی دوم، با حالتی که از حذف سطرها و ستون‌های ماتریس متناظر با ضرائب لاگرانژ برابر صفر می‌باشند، حاصل می‌شود، معادل است. [4] بنابراین مسئله‌ی QP اصلی می‌تواند به یک سری از مسائل QP با اندازه‌ی کوچکتر شکسته شود، که هدف نهایی آن شناسایی تمام ضرائب لاگرانژ غیر صفر و حذف ضرائب لاگرانژ صفر می‌باشد. این روش سبب می‌شود که سایز ماتریس از توان دوم تعداد نمونه‌های آموزشی به حدود توان دوم تعداد ضرائب لاگرانژ غیر صفر، کاهش یابد، ولی باز هم این روش نمی‌تواند برای مقادیر داده‌ی بسیار زیاد مناسب باشد، زیرا گنجاندن این ماتریس کاهش یافته نیز در حافظه کار مشکلی می‌باشد. این روش نیاز به بهینه کردن تمام ضرائب لاگرانژ غیر صفر دارد، در عین حال ممکن است ماتریس Kernel ساخته شده، به اندازه‌ای بزرگ باشد که در حافظه گنجانده نشود.

٤-٤- SMO(Sequential Minimal Optimization)

Sequential minimal optimization (SMO) الگوریتمی برای حل برنامه‌ریزی درجه‌ی دوم منتج شده در ضمن فرآیند آموزش ماشین بردارهای پشتیبان(SVM)، می‌باشد. این روش توسط John Platt (از Microsoft Research) در ١٩٩٨ ابداع گردید. SMO به طور وسیعی برای تعلیم ماشین بردارهای پشتیبان، بکار گرفته شده و توسط بسته‌ی بسیار محبوب LIBSVM پیاده‌سازی شده است. انتشار SMO در ١٩٩٨ سبب موج استقبال وسیعی توسط انجمن SVM گردید، زیرا روش‌های موجود برای حل SVM بسیار پرهزینه و پیچیده بوده‌اند.[13]

در حقیقت SMO، الگوریتم ساده‌ایست که به سرعت مسئله‌ی QP SVM را بدون هیچگونه ذخیره‌سازی ماتریس و بدون استفاده از راه حل عددی بهینه‌سازی برنامه ریزی درجه‌ی دوم، حل می‌کند. SMO با استفاده از تئوریOsuna اقدام به تجزیه‌ی مسئله‌ی اصلی به تعدادی زیر مسئله می‌نماید و به این ترتیب همگرایی روش را نیز تضمین می‌نماید. برخلاف سایر روش‌های پیشین، SMO در هر مرحله کوچکترین مسئله را برای بهینه‌سازی انتخاب می‌نماید. برای مسئله‌ی QP، SVM استاندارد کوچکترین مسئله‌ی بهینه‌سازی ممکن شامل ۲ ضریب لاگرانژ می‌باشد، زیرا ضرائب لاگرانژ باید از قیود خطی تساوی تبعیت نمایند. در هر مرحله، SMO دو ضریب لاگرانژ را برای بهینه کردن انتخاب می‌نماید و مقدار بهینه را برای این ضرائب پیدا نموده و SVM را برای یافتن مقدار بهینه به‌روز می‌نماید.

مزیت روش SMO در این است که راه حل مسئله برای دو ضریب لاگرانژ می‌تواند به صورت تحلیلی حاصل شود. بنابراین در سراسر راه حل از بهینه‌سازی QP به صورت عددی پرهیز می‌گردد. بعلاوه همانطور که گفته شد این روش نیازی به حافظه جهت ذخیره‌سازی ماتریس ندارد، بنابراین مجموعه‌های آموزشی بسیار بزرگ را نیز می‌توان توسط این روش مدیریت نمود.

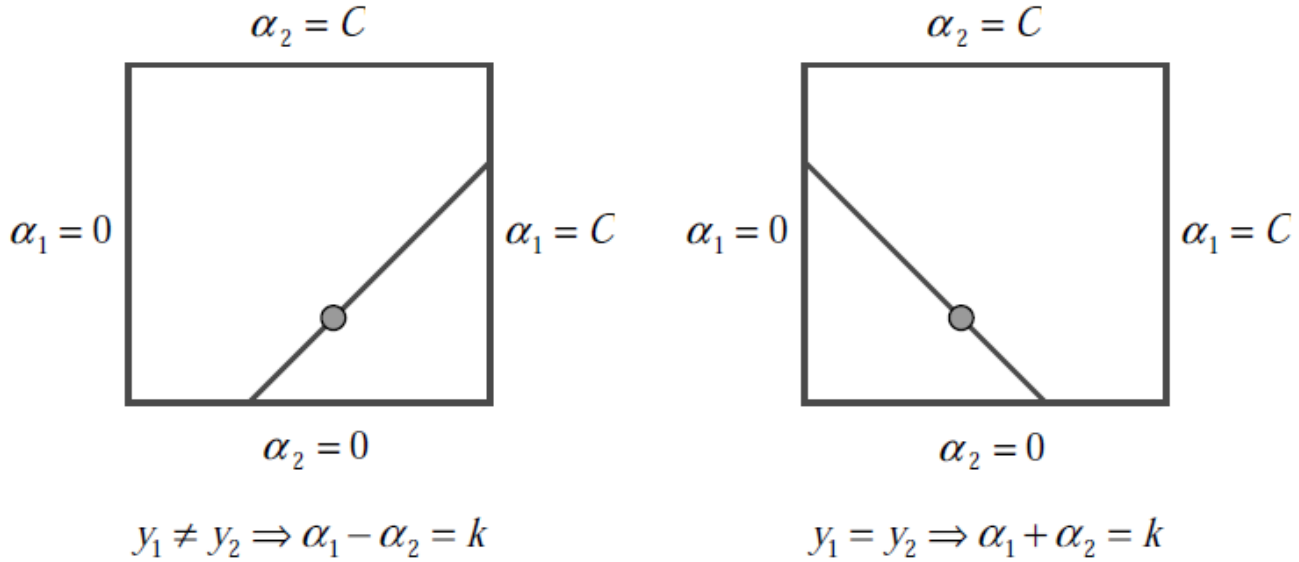
الگوریتم SMO در [4] بری آموزش ماشین‌های بردار پشتیبان، به طور کامل ارائه شده است. آموزش یک ماشین بردار پشتیبان مستلزم حل یک معادله‌ی بهینه‌سازی ازنوع برنامه‌ریزی درجه‌ی دوم بسیار بزرگ می‌باشد. میزان فضایی مورد نیاز برای SMO نسبت به اندازه‌ی مجموعه‌ی آموزش خطی می‌باشد که این خود به SMO اجازه‌ی مدیریت مجموعه‌های آموزشی بزرگ را می‌دهد. با توجه به اینکه در این روش از محاسبات ماتریسی اجتناب شده است، لذا SMO بین مرتبه‌های خطی و دوم نسبت به اندازه‌ی مجموعه‌ی آموزشی قرار می‌گیرد، در حالیکه در SVM استاندارد در میان مرتبه‌های دوم و سوم قرار می‌گیرد. در دنیای واقعی SMO می‌تواند بیش از ۱۰۰۰ بار از الگوریتم قطعه‌بندی استاندارد برایSVM سریعتر باشد.[4]

در روش SMO [4] ، در هر تکرار دو متغیر انتخاب شده و مسئله به صورت تحلیلی حل می‌شود، بنابراین دو عنصر در SMO وجود دارد[۱۷]، یک روش آنالیزی برای حل مسئله با دو ضریب لاگرانژ، و یک روال اکتشافی برای انتخاب اینکه کدام ضرائب برای بهینه‌سازی انتخاب شوند.

٤-٤-١- حل مسئله برای دو ضریب لاگرانژ

برای محاسبه‌ی این مسئله به ازای دو ضریب لاگرانژ، ابتدا باید محدودیت‌های روی ضرائب را در نظر گرفت و سپس به محاسبه‌ی می‌نیمم مقید شده پرداخت. با توجه به اینکه در این حالت فقط دو متغیر وجود دارد، می‌توان آنها را در دو بعد نمایش داد. قیدها نامساوی در مسئله QP SVM سبب قرار گرفت ضرایب در یک چهارچوب و قید مساوی مسئله سبب واقع شدن آنها روی یک خط (قطری از چهارچوب) می گردد(شکل ۱۲). بنابراین می‌نیمم مقید، برای تابع هدف،

روی این قطر واقع خواهد شد. قیدهای مسئله علت این نکته را که چرا دو ضریب لاگرانژ، حداقل تعداد ضریب برای بهینه‌سازی می‌باشد: انتخاب تنها یک ضریب، سبب عدم برآورده شدن قیدهای مسئله می‌شود. انتهای قطعه خط قطری را برای دومین ضریب (بدون از دست دادن کلیت مسئله) برحسب ضریب $\boldsymbol{\alpha_2}$، نوشته می‌شود.



شکل ۱۲ – دو ضریب لاگرانژ باید تمامی قیدهای مطرح شده در مسئله را مرتفع نمایند. قید نامساوی در مسئله سبب می‌شود که ضرایب در چهارچوب محدود شده و شرط تساوی سبب واقع شدن آنها روی خط قطری می‌گردد. بنابراین SMO باید مقدار بهینه را روی این قطعه خط قطری بدست آورد.

برای نوشتن حدود خطها خواهیم داشت:

$$y_1 \neq y_2: \qquad L = \max(0, \alpha_2 - \alpha_1), \qquad H = \min(C, C + \alpha_2 - \alpha_1)$$

(۵۶)

$$y_1 = y_2: \qquad L = \max(0, \alpha_2 + \alpha_1 - C), \qquad H = \min(C, \alpha_2 + \alpha_1)$$

(۵۷)

همانطور که در [25] نشان داده شده است پس از جایگذاری و گرفتن مشتق دوم از دوگان مسئله‌ی بهینه‌سازی خواهیم داشت:

$$\frac{d^2 L_D}{d\alpha^2} = K(x_1, x_1) + K(x_2, x_2) - 2K(x_1, x_2)$$

(۵۸)

که تعریف می‌کنیم:

$$\eta = K(x_1, x_1) + K(x_2, x_2) - 2K(x_1, x_2)$$

(۵۹)

که مثبت یا منفی بودن η بستگی به تابع Kernel مورد استفاده دارد. در اینجا می‌خواهیم با داشتن مقادیر قبلی برای α_1 و α_2 و سایر α_iها، مقدار جدید α_2 را محاسبه نماییم. با تعریف u به شکل زیر :

$$u = \sum_{j=1}^{N} y_j \alpha_j K(x_j, x) - b$$

(٦٠)

و تعریف E_i به عنوان خطا در نمونه‌ی i–ام آموزشی:

$$E_i = u_i - y_i$$

(٦١)

با جایگذاری در مشتق مرتبه اول خواهیم داشت:

$$\frac{dL_D}{d\alpha_2} = \eta\alpha_2 + (y_2(E_1 - E_2) - \eta\alpha_2)$$

(٦٢)

از مساوی صفر قرار دادن رابطه‌ی فوق داریم:

$$\alpha_2^{new} = \alpha_2 + \frac{y_2(E_1 - E_2)}{\eta}$$

(٦٣)

مقدار بهینه‌ی محاسبه شده برای α_2 باید توسط قیدهای مسئله محدود گردد، لذا خواهیم داشت:

$$\alpha_2^{new,clipped} = \begin{cases} H & if\ \alpha_2^{new} \geq H; \\ \alpha_2^{new} & if\ L < \alpha_2^{new} < H \\ L & if\ \ \alpha_2^{new} \leq L \end{cases}$$

(٦٤)

با تعریف $s=y_1y_2$، α_1 جدید با توجه به α_2 جدید محاسبه می‌شود:

$$\alpha_1^{new} = \alpha_1 + s(\alpha_2 - \alpha_2^{new,clipped})$$

(٦٥)

٤-٤-٢- روش اکتشافی برای انتخاب ضرایب لاگرانژ

برای افزایش سرعت همگرایی، در *SMO* از روش اکتشافی برای انتخاب اینکه کدام ضرائب لاگرانژ جهت بهینه‌سازی انتخاب شوند، استفاده می‌کند. برای این کار از دو انتخاب اکتشافی مجزا برای ضرائب استفاده می‌شود. در الگوریتم ارائه شده [4]، از دو حلقه‌ی متداخل استفاده شده است. در حلقه‌ی خارجی در اولین اجرا، تمام مجموعه‌ی داده‌های آموزشی را برای یافتن نمونه‌ای که شرایط *KKT* را نقض نماید، پویش می‌نماید. اگر نمونه‌ای شرایط *KKT* را نقض نماید، مناسب برای بهینه‌سازی خواهد بود. در[26] این موضوع به شکل واضح‌تری اشاره شده است و بیان نموده که، حلقه خارجی به دنبال ضریبی می‌گردد که شرایط زیر را نقض نماید:

$$\alpha_i = 0 \quad => y_i f(x_i) \geq 1$$

(٦٦)

$$0 < \alpha_i < C \quad => y_i f(x_i) = 1$$

(٦٧)

$$\alpha_i = C \quad => y_i f(x) \leq 1$$

(٦٨)

در واقع این شرایط به همان شرایط *KKT* اشاره دارند. بعد از اولین تکرار حلقه‌ی خارجی تکرار را برروی نمونه‌هایی از ضرایب لاگرانژ که بین *0* و *C* قرار دارند(*C* و*0* نباشند، مثلاً $0 < \alpha_i < C$)، ادامه می‌دهد. مجدداً، هر نمونه با شرایط *KKT* مقایسه شده و نمونه‌ای که آنرا نقض نماید، مناسب برای بهینه‌سازی خواهد بود. این بهینه‌سازی در حلقه‌ی خارجی تا زمانی اجرا می‌شود که تمام این نمونه‌های مذکور شرایط *KKT* را با خطای ε برآورده سازند. در ادامه حلقه خارجی بازگشت نموده و تکرار را روی تمام مجموعه‌ی آموزشی انجام می‌دهد. حلقه خارجی به طور مداوم تا زمانی که تمامی نمونه‌های آزمایشی شرایط *KKT* را با خطای ε برآورده سازند، بین این دو انتخاب این جابجا می‌شود.

بعد از انتخاب اولین ضریب توسط حلقه‌ی خارجی، دومین ضریب توسط حلقه داخلی و با تکرار بر روی ضرائب باقیمانده، برای انجام بهینه‌سازی انتخاب می‌شود. ارزیابی تابع *Kernel* زمانبر می‌باشد، لذا *SMO* از قدرمطلق اندازه‌ی

گام $(E_1 - E_2)$ برای تخمین استفاده می‌نماید. *SMO* از یک *cache* برای نگهداری مقدار خطای نمونه‌های غیرمرزی($0 < \alpha_i < C$) استفاده نموده و سپس از یک خطا برای تخمین اندازه‌ی گام ماکزیمم، استفاده می‌نماید. در صورتی که E_1 مثبت باشد، نمونه‌ای با حداقل خطا E_2 انتخاب می‌گردد. در صورتی که E_1 منفی باشد، نمونه‌ای با حداکثر خطا E_2 انتخاب می‌گردد.

در الگوریتم ارائه شده در *[4]*، یک مرحله‌ی به‌روزرسانی برای b در نظر گرفته شده‌است که در *[27]* روشی بهینه برای الگوریتم *SMO* آمده است که طی آن نیازی با به‌روزرسانی b نمی‌باشد.

الگوریتم ارائه شده برای *SMO* در *[4]* به صورت شبه کد آمده است. در *[26]* نیز این الگوریتم به صورت ساده‌تری ذکر شده است.

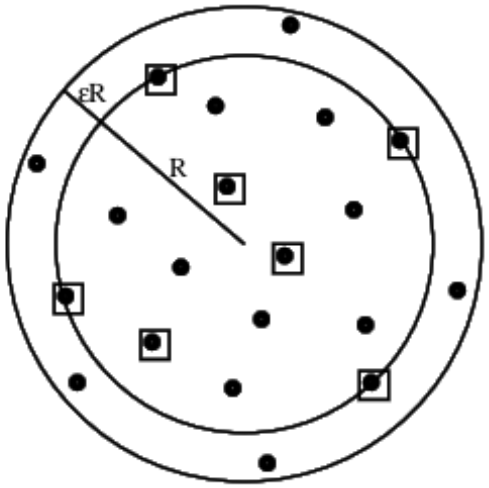
Core Vector Machines-٥-٤

در [2] روش CVM به عنوان روشی سریع برای آموزش SVM با داده‌های با حجم بسیار بالا مطرح شده است.SVM استاندارد دارای زمان آموزشی از مرتبه‌ی زمانی $O(m^3)$ و مرتبه پیچیدگی فضایی $O(m^2)$ می‌باشد. (m اندازه‌ی مجموعه‌ی آموزش می‌باشد). با مشاهده‌ی اینکه پیاده‌سازی‌های تجربی SVM فقط تخمینی از راه حل بهینه را با روش‌های تکرار ارائه می‌دهند،در [2] روش‌های kernel با توجه به چنین تخمین‌هایی، گسترش داده شده است. ابتدا نشان داده شده است که هر متد Kernel به طور مشابه می‌تواند با مسائله‌ی MEB در هندسه‌ی محاسباتی، فرموله شود. سپس با انطباق یک الگوریتم موثر در تخمین MEB، مقدار تخمینی بهینه با توجه به ایده‌ی core set به دست می‌آید. روش Core Vector Machine ارائه شده در [2]، می‌تواند با kernelهای غیر خطی بکار رود و دارای پیچیدگی زمانی خطی از m و پیچیدگی فضایی مستقل از m می‌باشد. نتایج تجربی نشان داده است که در مقادیر زیاد داده‌ها، دقت CVM به خوبی SVM می‌باشد و در عین حال می‌تواند با سرعت بالاتر حجم زیادی از داده‌ها را پاسخ‌گو باشد.

نکته کلیدی در پیاده‌سازی‌های تجربی SVM این است که مانند بسیاری از روش‌های عددی، فقط *تقریبی* از راه حل بهینه با استفاده از روش‌های تکراری حاصل می‌شود. این روش‌های تقریبی در بسیاری از مسائل محاسباتی مشکل، در کامپیوتر کاربرد داشته و به طور وسیعی مورد استفاده قرار می‌گیرند. در این الگوریتم از مفهوم minimum enclosing ball (MEB) برای تخمین استفاده می‌شود. مسئله‌ی MEB کوچکترین شعاعی را که می‌تواند نقاط یک مجموعه را پوشش دهد، محاسبه می‌کند. BÌadoiuوClarkson نشان دادند که تقریبی از MEB می‌تواند با استفاده از *Core Sets* بدست آید.Core Set زیر مجموعه‌ای از نقاط ورودی است که می‌تواند تقریب خوبی از جواب بهینه را از حل مسئله‌ی بهینه‌سازی روی این مجموعه حاصل نماید در واقع Core Set در CVM نقشی مشابه با Working Set در الگوریتم Decomposition را بازی می‌کند.

برای محاسبه‌ی تقریبی از MEB به صورت زیر عمل می‌شود

برای مجموعه‌ی مفروض $S=\{x_1,x_2,...,x_m\}$ که x_i متعلق به R^d می‌باشد، minimum enclosing ball که با MEB(S) نشان داده می‌شود، عبارتست از کوچکترین حجمی که بتواند تمام نقاط S را شامل شود. اگر فرض شود که B(c,R) حجمی به مرکز c و شعاع R باشد، آنگاه برای $\varepsilon>0$، $B(c,(1+\varepsilon)R)$ تخمینی از MEB(S) خواهد بود اگر $R\leq r_{MEB(S)}$ بوده و $S\subset B(c,(1+\varepsilon)R)$ باشد.(شکل)



شکل ۱۳- مسئله‌یMEB

الگوریتم ارائه شده توسط CVM برای بهینه‌سازی و یافتن بهترین جواب تقریبی برای مسئله‌ی بهینه‌سازی، با بدست آوردن مجموعه‌ای از بردارها با نام Core Vectorها پیاده‌سازی می‌شود. در این الگوریتم مرکز حجم با در تکرار t–ام برای مجموعه‌ی S_t با c_t و شعاع آن با R_t نمایش داده می‌شود. برای $\varepsilon>0$ ، CVM به شکل زیر عمل می‌کند:

1.**Initialize** S_0, c_0 and R_0.

2.**Terminate** if there is no training point **z** such that $\phi(z)$ falls outside the $(1 + \varepsilon)$-ball $B(c_t,(1 +\varepsilon)R_t)$.

3.**Find** z such that $\phi(z)$ is furthest away from ct. Set $S_t+1 = S_t \cup \{z\}$.

4.**Find** the new $MEB(S_t+1)$ from (5) and set $\mathbf{c}_t+1 = \mathbf{c}_{MEB(S_{t+1})}$ and $\mathbf{R}_{t+1} = \mathbf{r}_{MEB(S_{t+1})}$ using (3).

5.**Increment** t by 1 and go back to Step 2.

$\phi(z)$ تابع نگاشت می‌باشد. نقاطی که به مجموعه‌ی Core Set اضافه می‌شوند، Core Vector گفته می‌شود. جزئیات مراحل فوق در [۱۵] آمده است. الگوریتم ارائه شده برای CVM برخلاف سادگی‌اش، ضمن گارانتی جواب تقریبی، دارای زمان کوتاه و پیچیدگی فضایی کم نیز می‌باشد.

۴-۶- LSVM (Lagrangian SVM)

الگوریتم LSVM (Fung andMangasarian, 2003;Mangasarian and Musicant 2001)، راه حل را با تکرارهای سریع محاسبه می‌نماید، ولی این روش نیز در مواردی با Kernel غیرخطی، نیاز به محاسبه‌ی ماتریس معکوس با ابعاد m*m می‌باشد.[2]

۴-۷-Reduced SVM

Lee و Mangasarian در ۲۰۰۱ اقدام به ابداع روش Reduced SVM یا RSVM نمودند. این روش از یک زیرمجموعه‌ی مستطیل تصادفی از ماتریس Kernel استفاده می‌نماید.[2] RSVM یکی از جایگزین‌های SVM استاندارد می‌باشد. انگیزه‌ی پیدایش این روش مشکلات SVM استاندارد در مدیریت مجموعه‌های بزرگ داده‌ها با SVM با Kernel غیرخطی می‌باشد. این روش زیر مجموعه‌ای از داده‌ها را بعنوان بردارهای پشتیبان انتخاب نموده و مسئله‌ی بهینه‌سازی کوچکتری را حل می‌کند. در [1] نشان داده شده است که RSVM نیز به شکلی فرمول SVM خطی می‌باشد و همچنین ٤ پیاده‌سازی از RSVM مورد بحث قرار گرفته است. نتایج تجربی نشان می‌دهد که دقت RSVM اندکی از SVM استاندارد کمتر است ولی در مسائلی با حجم ۱۰ها هزار داده و یا با تعداد زیاد بردار پشتیبان، RSVM می‌تواند مفید باشد. در [1] دو مقایسه صورت گرفته است:

۱- پیاده‌سازی‌های مختلف برای SVM خطی

۲- SVM استاندارد با استفاده از تابع هزینه‌ی درجه‌ی دوم و خطی.

در مواجه شدن با مجموعه‌ای بزرگ از داده‌ها،RSVM بعنوان راه‌حلی برای کاهش پیچیدگی محاسباتی و کاهش پیچیدگی‌های مدل، ارائه شده است. در [3] RSVM از دیدگاه نمونه‌برداری و آماری مورد مطالعه قرار گرفته است.

در پایان این بخش به Relevance Vector Machine به عنوان توسعه‌ای از SVM و نه روشی برای حل معادله‌ی QP و سپس به مسئله‌ی انتخاب مجموعه‌ی کاری، اشاره می‌گردد.

۴-۸- Relevance Vector Machine

اساساً روش SVM جهت بدست آوردن کارائی بهتر در کاربردهای عملی طراحی شده است. ولی این روش مشکلاتی نیز دارد. در [6] از Non-Bayesian بودن این روش بعنوان بزرگ‌ترین مشکل آن اشاره شده است. در SVM به ازای

ورودی جدید خروجی مشخصی وجود دارد. مزایای زیادی در دانستن احتمال وقوع اعضای یک کلاس وجود دارد. روش RVM (Relevance Vector Machine)که توسط Tipping (از Microsoft) ارائه شده، به صورت احتمالاتی پیش‌بینی را انجام می‌دهد و همچنان کارائی بالا و خلوت بودن SVM را حفظ می‌نماید.[6]

RVM روشی است در یادگیری ماشین که با استفاده از استنباط Bayesian راه‌حل‌های کم هزینه‌ای را برای رگرسیون و طبقه‌بندی احتمالاتی (Probabilistic Classification)، ارائه می‌دهد. RVM فرم تابعی مشابه با SVM دارد ولی طبقه‌بندی احتمالاتی را ارائه می‌کند.[11]

٤-٩- انتخاب مجموعه‌ی کاری[١٢]

در [10] به مسئله‌ی انتخاب مجموعه‌ی کاری به عنوان یک مسئله‌ی مهم در روش‌های مبتنی بر تجزیه[١٣]، پرداخته شده است. در این مقاله روشی برای انتخاب مجموعه‌ی کاری در روش تجزیه‌ی مبتی برSMO ارائه شده است. در بسیاری از مسائل برای سادگی نمایش، مسئله‌ی بهینه‌سازی مطرح شده در آموزش SVM به شکل زیر بیان می‌شود:

$$\min f(\alpha) = \frac{1}{2}\alpha^T Q\alpha - e^T\alpha$$

(٦٩)

$$subject\ to: 0 \le \alpha_i \le C, i = 1, \dots, l, y^T\alpha = 0$$

(٧٠)

که در آن e آرائه‌ای از ١ها، C کران‌بالای متغیرها،Q یک ماتریس l*l متقارن که در آن $Q_{ij}=y_iy_jK(x_i,x_j)$ و $K(x_i,x_j)$ عبارتست از تابع هسته‌ی مورد استفاده. ماتریس Q عموماً بسیار پر (خلوت نمی‌باشد) بوده و بسیار بزرگ می‌باشد، به طوری که در حافظه قرار گرفتن آن مشکل می‌باشد. روش‌های تجزیه، مانند روش‌های ذکر شده در این بخش اساساً در پی مدیریت این پیچیدگی می‌باشند. برخلاف سایر روش‌های بهینه‌سازی که در آنها تمام بردارα به صورت یکجا و در هر تکرار، به‌روز می‌شود، روش‌های مبتنی بر تجزیه فقط زیر مجموعه‌ای از α را در هر تکرار به‌روز رسانی می‌نمایند. این زیر مجموعه به *مجموعه‌ی کاری* معروف می‌باشد و با B نمایش داده می‌شود. به این ترتیب به‌روزرسانی B در هر تکرار سبب می‌شود که یک زیرمسئله‌ی کوچک در هر تکرار می‌نیمایز شود. همانطور که پیشتر اشاره شد، اوج این روش در SMO می‌باشد که در هر مرحله B، تنها دارای ٢ عنصر می‌باشد. بنابراین در هر مرحله این مسئله‌ی دو متغیره به سادگی حل می‌شود. الگوریتم زیر روش تجزیه در SMO را نمایش می‌دهد:

---

[١٢] Working Set

[١٣] Decomposition Methods

۱- α^1 را به عنوان راه حل ممکن اولیه پیدا کن. K=1

۲- اگر α^k راه حلی بهینه برای (٤١) می‌باشد متوقف شو، در غیر اینصورت مجموعه‌ی دوعنصری کاری $B=\{i,j\}\subset\{1,...,l\}$ را انتخاب کن. $N\equiv\{1,...,l)/B$ و $\alpha^k{}_B$ و $\alpha^k{}_N$ را به عنوان زیربرداری از α^k ، مناظر با B و N تعریف کن.

۳- زیر مسئله‌ی زیر را با متغیر α_B حل کن:

$$\min_{\alpha_B} \frac{1}{2}[\alpha_B^T\ (\alpha_N^k)^T]\begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN}\end{bmatrix}\begin{bmatrix}\alpha_B \\ \alpha_N^k\end{bmatrix} - [e_B^T\ e_N^T]\begin{bmatrix}\alpha_B \\ \alpha_N^k\end{bmatrix}$$

$$= \frac{1}{2}\alpha_B^T Q_{BB}\alpha_B + \left(-e_B + Q_{BN}\alpha_N^k\right)^T \alpha_B + constant$$

$$= \frac{1}{2}[\alpha_i\ \alpha_j]\begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ij} & Q_{jj}\end{bmatrix}\begin{bmatrix}\alpha_i \\ \alpha_j\end{bmatrix} + \left(-e_B + Q_{BN}\alpha_N^k\right)^T + constant$$

(٧١)

$$subject\ to:\ 0 \le \alpha_i, \alpha_j \le C, y_i\alpha_i + y_j\alpha_j = -\boldsymbol{y_N^T \alpha_N^T}$$

(٧٢)

٤- $\alpha^{k+1}{}_B$ را به عنوان راه حل بهینه برای (٧١) قرار بده، $\alpha^{k+1}{}_N=\alpha^k{}_N$. k=k+1 . برو به ۲.

باید توجه داشت که B در هر تکرار تغییرمی‌نمایدو در اینجا برای سادگی بجای B^k از B استفاده شده است. از آنجایی که در هر تکرار تنها تعداد کمی از عناصر به‌روز می‌شوند، لذا در مسائل بزرگ، روش تجزیه به کندی همگرا می‌شود. راه‌حل‌های مناسب‌تر می‌توانند از طریق کاهش تعداد مراحل تکرار، سریعتر عمل نمایند.

روش‌های موجود عموماً بر اساس نقض شرایط بهینه عمل می‌کنند که وابسته به دانستن اطلاعات مرتبه‌ی اول (مانند گرادیان)، از تابع هدف می‌باشند. مطالعات بر روی بهینه‌سازی نشان داده است که استفاده از اطلاعات مرتبه‌ی دوم، عموماً سبب سرعت بیشتر در همگرایی می‌شوند. از آنجایی که معادله‌ی (٦٩) یک مسئله‌ی درجه‌ی دوم می‌باشد، لذا اطلاعات مرتبه‌ی دوم مستقیماً به کاهش تابع هدف مربوط می‌شود. تلاش‌های بسیاری برای یافتن مجموعه‌ی کاری براساس کاهش تابع هدف انجام شده است. اما این روش‌ها اکتشافی بوده و تضمینی بر همگرایی آنها نمی‌باشد. در [10] روش ساده‌ای برای انتخاب مجموعه‌ی کاری با استفاده از اطلاعات مرتبه‌ی دوم ارائه شده است که سبب زمان تعلیم کوتاه‌تری می‌گردد.

روش‌های مختلفی برای انتخاب این مجموعه در [10] معرفی شده و مقایسه شده‌اند و نشان داده شده است که روش معرفی شده، از کارائی بیشتری نسبت به تمام آنها برخوردار می‌باشد.

۵- کاربردهای SVM

در این بخش مروری خواهیم داشته به کاربردهای تشخیص الگو[۱۴] با استفاده از SVM. با توجه به هدف کاربردها در [8] هفت دسته کاربرد مختلف برای SVM دسته‌بندی شده است که در اینجا به آنها اشاره می‌نماییم. تعدادی از کاربردها که در دسته‌بندی خاصی گنجانده نشده است در گروه سایر کاربردها، ذکر شده‌اند.

۵-۱- تشخیص و شناسایی چهره[۱۵]

این دسته یکی از محبوب‌ترین زمینه‌های در بایومتریک، تایید هویت، کنترل دسترسی و نظارت ویدئویی می‌باشد. زمینه‌های تحقیقاتی فعالی در این دسته وجود دارند که برای این کاربردها از روش‌های مختلفی استفاده می‌نمایند. به هرحال رسیدن به کارائی مطلوب در این زمینه بسیار مشکل خواهد بود. شناسایی چهره‌های افرادی که ویژگی‌های ظاهری نزدیک به هم دارند بسیار مشکل خواهد بود. همچین حالات مختلف برای یک چهره‌ی خاص نظیر آرایش‌های متفاوت و ... تشخیص را بسیار مشکل می‌نماید. همچنین استفاده از عینک و یا ریش و سیبیل می‌تواند تشخیص را مشکل و پیچیده نماید. در سال‌های اخیر تحقیقات زیادی برای استفاده از SVM در کاربردهای تشخیص چهره، شناسایی و بازشناسی چهره و تشخیص حالات چهره صورت گرفته است. هر کدام از روش‌های فوق از ورودی‌های، پایگاه‌داده‌ها و هسته‌های مختلفی برای طبقه‌بندی‌کننده‌ی SVM استفاده کرده‌اند. اولین بار Osuna از SVM برای تشخیص چهره استفاده نمود. در این کاربرد وی از یک تصویر چهره‌ی ۱۹*۱۹ و تابع هسته‌ی چند جمله‌ای درجه‌ی دوم استفاده نمود. شناسایی ماشین در مورد چهره به دو دسته‌ی شناسایی چهره و تایید چهره تقسیم می‌شود. Guo از SVM چندکلاسه و یک درخت باینری برای شناسایی چهره استفاده نمود. ویژگی‌های نرمال شده که توسط روش PCA استخراج شده بودند، به عنوان ورودی SVM بکار گرفته شدند.

۵-۲- تشخیص و شناسایی اشیاء

هدف از تشخیص یا شناسایی اشیا یافتن و تعقیب افراد در حال حرکت و یا بررسی ترافیک برای کاربردهای کنترل ترافیک می‌باشد. شناسایی اشیا ۳-بعدی نیز یکی از این زمینه‌ها می‌باشد. COIL یک پایگاه‌داده‌ی معروف می‌باشد که شامل ۷۲۰۰ تصویر از ۱۰۰ شیء و از ۷۲ زاویه دید مختلف می‌باشد. این پایگاه داده در بسیاری از کاربردهای تحقیقاتی در این زمینه مورد استفاده قرار گرفته است. Roobaert تشخیص اشیاء ۳ بعدی را توسط SVM انجام داد و توانایی

---

[۱۴] Pattern Recognition

[۱۵] Face Detection and Recognition

SVM را در این شناسایی از زاویه‌های مختلف، بررسی کرد.M.Pontil و A.Verri با آمیختن تصاویر پایگاه داده‌ی COIL با نویز و شیف دادن تصاویر، کارائی بالایی را نتیجه گرفتند.

۵-۳- تشخیص دست‌نوشته و ارقام

در میان کاربردهای مختلفی که بر پایه‌ی SVM انجام شده است، تشخیص ارقام دست‌نویس توسط SVM از تمامی الگوریتم‌های یادگیری دیگر، کارائی بالاتری داشته است. مشکل عمده در مسئله‌ی تشخیص دست‌نوشته، وجود الگوهای مختلف و متنوع نوشتاری بسیار زیاد می‌باشد. مدل‌های Elastic که برپایه‌ی مشاهدات محلی و برنامه‌نویسی پویا شکل گرفته‌اند مانند HMM، برای تحلیل این تنوع کارا و مناسب هستند. Choisy برای ترکیب این توانایی محلی با ویژگی‌های سراسری، از NSPH-HMM برای مشاهدات محلی و نرمالسازی و از SVM برای مشاهدات سراسری بر روی خروجی نرمال شده توسط NSPH-HMM، استفاده نمود. کارهای مختلفی در این زمینه انجام شده است که در مجموع برتری این روش را به سایر روش‌ها در شناسایی دست‌نوشته‌ها نشان می‌دهد.

۵-٤- تشخیص صحبت و گوینده

در مسئله‌ی تشخیص صحبت و یا شناسایی گوینده، دو روش بسیار محبوب discriminative classifierها و generative model clasifierها می‌باشند. روش‌هایی که از discriminative classifierها استفاده می‌کنند، شامل درخت‌های تصمیم‌گیری،شبکه‌های عصبی و SVM می‌باشند. معروفترین روش generative model clasifier شامل مدل Hidden Markov(HMM) و مدل ترکیبی Gaussian (GMM) می‌باشد. Bengio و Wan از SVM برای شناسایی گوینده با پایگاه‌داده‌های مختلف، استفاده نمودند. آنها بر روی داده‌های وابسته به متن و غیر وابسته به متن کار کرده و روش SVM را با آستانه‌گذاری کلاسیک برای تصمیم‌گیری در رد یا پذیرش تایید گوینده‌ی، جایگزین نمودند. استفاده‌های گوناگونی از SVM در این زمینه نیز صورت گرفته است.

۵-۵- بازیابی تصاویر و اطلاعات[۱۶]

Content-based image retrival نتیجه‌ی زمنیه‌ی تحقیقات مهمی در زمینه‌ی کتابخانه‌های دیجیتال و پایگاه‌داده‌های چندرسانه‌ای می‌باشد. Guo معیاری جدیدی را تحت عنوان فاصله از مرز[۱۷]، را برای بازیابی بافت تصاویر ابداع نمود که در آن مرز بین دسته‌ها توسط SVM بدست آمده است. برای بازیابی تصاویر مرتبط بیشتر با تصویر ورودی، طبقه‌بندی کننده‌ی SVM برای جدا کردن تصاویر به دودسته‌ی مرتبط و غیرمرتبط، استفاده شده است.Drucker,Tian وZhang

[۱۶] Information and Image Retrival

[۱۷] Distance-from-boundray

روش اتوماتیکی را با استفاده از SVM برای بازیابی تصاویر ابداع نمودند که در آن وزن‌هابا فاصله از ابر صفحه تعیین می‌شوند و داده‌ها به دو دسته‌ی داده‌های مثبت و منفی تقسیم می‌شوند.

۵-٦- پیش‌بینی

هدف اصلی در بسیاری از روش‌های پیش‌گویی غیرخطی، پیش‌بینی نقطه‌ی بعدی در یک سری زمانی[۱۸] می‌باشد[7]. Tay و Cao روش C-ascending SVM را با کاهش C ارائه دادند. این ایده بر این فرض استوار است که بهتر است که وزن بیشتری را به داده‌های اخیر بدهیم تا به داده‌های دورتر. نتایج آنها نشان داد که روش C-ascending SVM کارائی بالاتری را نسبت به SVM استاندارد در پیش‌گویی سری‌های زمانی اقتصادی، ارائه می‌دهد. Fan روش SVM را بر مسئله‌ی پیش‌بینی مشکلات شرکت‌ها با توجه به وضعیت اقصادی آنها، منطبق نمود. برای این مسئله انتخاب متغیرهای ورودی (شاخص‌های اقتصادی)، روی کارائی سیستم تاثیرگذار می‌باشد. در مقاله‌ی ارائه شده، پیشنهاد داده که از داده‌های مناسب استفاده شود که فاصله‌ی بین دسته‌های بیشترین مقدار و فاصله‌ی بین داده‌های مشابه را کمترین مقدار قرار می‌دهند.

۵-۷- سایر کاربردها

کاربردهای بسیار زیاد دیگری برای SVM در زمینه‌ی تشخیص الگو وجود دارد. بعنوان مثال Yang از SVM برای دسته‌بندی بصری بر اساس جنسیت، با استفاده از تصاویر کوچک (۱۲*۲۱) و با کیفیت پایین استفاده نمود. او برای این کار از پایگاه داده‌ی FERET استفاده نمود و از ۱۷۵۵ تصویر استفاده نمود.پس تعلیم نشان داده شد که کارائی این روش در مقابل روش‌های قبلی مانند FLD,RBF و ... بسیار بالاتر می‌باشد.(٪۳.٤ خطا)

Gutta از SVM برای دسته‌بندی حالات چهره بروی پایگاه داده‌ی FERET استفاده نمود و به ۱۰۰٪ دقت رسید. Yao پنج دسته اثر انگشت را فرض نمود و از SVM برای این دسته‌بندی استفاده نمود و کارائی خوبی را نتیجه گرفت.

علاوه بر کاربردهای ذکر شده، بسیاری از کاربردهای دیگر مانند خلاصه کردن داده‌ها[۱۹]، شناسایی هدف و... وجود دارد که SVM به خوبی از عهده‌ی آنها برمی‌آید.

در کاربرد خلاصه‌سازی داده‌ها، زیرمجموعه‌ای کوچک از پایگاه‌داده‌ای بزرگ انتخاب می‌شود و دقت تفکیک کننده که بر روی این داده‌های کاهش یافته عمل می‌کند، با نتیجه‌ی آموزش روی تمام داده‌ها، مقایسه می‌شود.

---

[۱۸] Time Series

[۱۹] Data Consentation

٦- ابزارهای پیاده‌سازی SVM

برای استفاده از طبقه‌بندی‌کننده‌ی SVM ابزارها و زیرساخت‌های مختلفی ایجاد شده است که هر کدام ویژگی‌ها و امکانات خاصی را ارائه می‌دهند. در این بخش به تعدادی از معروفترین این ابزارها پرداخته‌ها خواهد شد.

MATLAB-١-٦

یکی از ابزارهای مهم در این زمینه دستورات تعبیه شده در نرم افزار MATLAB می‌باشد. در MATLAB علاوه بر اینکه راهنمای خوبی برای آشنایی با مفهوم ماشین‌های بردار پشتیبان، ارائه شده است دستوراتی نیز برای کار با SVM طراحی شده است.

مانند هر مدل یادگیری نظارت شده‌ای در MATLAB نیز ابتدا ماشین بردار پشتیبان توسط داده‌های تعلیمی آموزش داده شده، سپس از ماشین تعلیم دیده برای دسته‌بندی(پیش‌بینی) داده‌های جدید استفاده می‌گردد. بعلاوه برای رسیدن به دقت پیش‌بینی مناسب، می‌توان از توابع هسته‌ی مختلف استفاده نمود که پارامترهای این توابع هسته، قابل تنظیم می‌باشند.

آموزش دسته‌بندی کننده‌ی SVM در MATLAB با استفاده از دستور svmtrain انجام می‌شود. الگوی این دستور به شکل کلی به این ترتیب می‌باشد:

```
SVMstruct=svmtrain(data,groups,'Kernel_Function','rbf');
```

که در آن:

data : ماتریسی از نقاط داده است. هر سطر یک مشاهده و هر ستون یک ویژگی می‌باشد.

groups : بردارهای ستونی که هر سطر متناظر با مقداری متناظر در سطری در data می‌باشد. groups می‌تواند تنها دو نوع از ورودی‌ها را شامل شود.

Kernel_Function : مقدار پیش‌فرض آن 'linear' داده‌ها را توسط ابرصفحه جدا می‌نماید. مقدار 'rbf' از تابع Gaussian radial base استفاده می نماید.

نتیجه‌ی این تابع در ساختار SVMstruct قرار می‌گیرد که شامل مقادیر بهینه برای الگوریتم SVM می‌باشد. یعنی در این مرحله ماشین SVM تعلیم دیده است و می‌توان از آن در پیش‌بینی مقادیر جدید استفاده نمود.

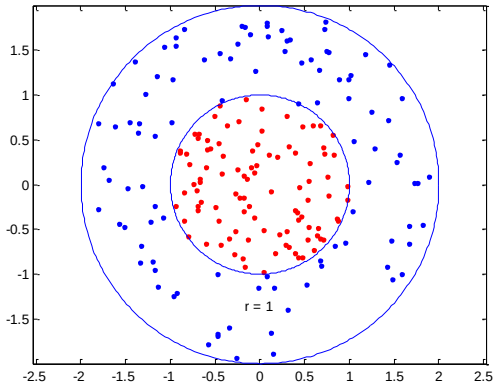
برای دسته‌بندی داده‌های جدید از تابع svmclassify استفادده می‌شود. از این تابع از ساختار SVMstruct که در دستور قبلی تکمیل شده است استفاده می‌نماید:

```
newClasses=svmclassify(SVMstruct,newData)
```

بردار خروجی در newClasses شامل نمایش دسته‌بندی برای داده‌ی جدیدnewData می‌باشد.

مثال:

در اینجا تعداد ۱۰۰ نقطه با توزیع نرمال به طور دایره‌ای تولید می‌شوند، سپس تعداد ۱۰۰ نقطه‌ی دیگر با توزیع نرمال به طور محاط بر داده‌های قبلی تولید می‌گردد. (شکل14)



شکل 14– داده‌های آزمایشی تولید شده

در پیوست ۱ کد مربوط به تولید این داده‌ها آمده است.

با فرض داشتن داده‌های فوق، می‌خواهیم با استفاده از دستورات MATLAB ضمن دسته‌بندی داده‌ها بردارهای پشتیبان را نیز مشخص نماییم. برای این کار نیاز است که علاوه بر داده‌ها(x_i) خروجی متناظر با آنها نیز (y_i) نیز مشخص باشد، تا ماشین بتواند با خواندن داده‌های ورودی و خروجی متناظر با آنها یادگیری خود را تکمیل نماید.

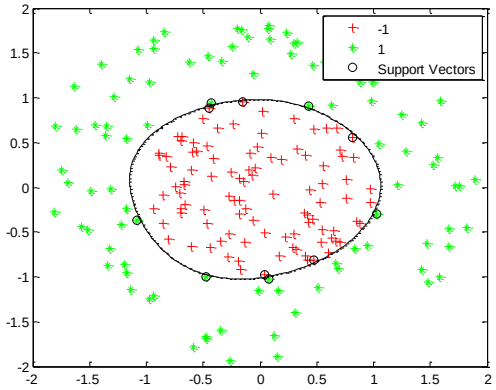
اگر فرض کنیم که داده‌ها دیسک داخلی در متغیر data1 و دیسک خارجی در data2 قرار گرفته باشند، آنگاه با دستورات زیر متغیر data3 شامل هر دو مجموعه از داده‌ها شده و خروجی متناظر آنها نیز تولید می‌گردد که در دو دسته‌ی ۱– و ۱+ قرار گرفته اند:

```
data3 = [data1;data2];
theclass = ones(200,1);
theclass(1:100) = -1;
```

در مرحله‌ی بعد نیاز به آموزش ماشین می‌باشد:

```
cl = svmtrain(data3,theclass,'Kernel_Function','rbf',...
   'boxconstraint',Inf,'showplot',true);
```

که این دستور سبب می‌شود ضمن آنکه یادگیری انجام می‌شود، دسته‌بندی و بردارهای پشتیبان نیز نمایش داده شوند (شکل ۱۵) :



شکل ۱۵– داداه‌های دسته‌بندی شده با svm

در شکل ۱۵، بردارهای پشتیبان با دایره‌های کوچکی بر روی منحنی جداکننده مشخص شده‌اند.

سپس نوبت به تست بردار ماشین برای کاربرد دسته‌بندی می‌رسد. در این قسمت MATLAB از تابع svmclassify استفاده کرده و با ورود داده‌ی تست گروه متناظر با آنرا مشخص می‌نماید. بعنوان مثال اگر چند نقطه داخل دیسک مرکز و چندنقطه داخل دیسک محاطی انتخاب نماییم(که جز داده‌های ورودی نباشند)، خروجی‌های زیر را خواهیم داشت:

نقاط داخل دیسک مرکزی:

```
s1=[-0.9,0.4];
svmclassify(cl,s1)
   -1
s2=[0.3,0.9];
svmclassify(cl,s2)
   -1
s3=[0.8,-0.2];
svmclassify(cl,s3)
   -1
```

نقاط داخل دیسک محاطی:

```
sn1=[1.8,-1.2];
svmclassify(cl,sn1)
```

```
   1
sn2=[-2.2,-2.2];
svmclassify(cl,sn2)
   1
sn3=[-1.7,2.2];
svmclassify(cl,sn3)
   1
```

لازم به ذکر است ۱- و ۱+ اشاره به گروه بندی تعریف شده در قسمت آموزش دارند.

۶-۲- LIBSVM

بسته‌ی دیگری که برای کار با SVM ارائه شده استLIBSVM می‌باشد که به صورت مجموعه‌ای از توابع کتابخانه‌ای توسط Chih-Chung Chang and Chih-Jen Lin ارائه شده است. این مجموعه دارای امکانات فراوانی برای استفاده از SVM می‌باشد که برخی از آنها در زیر آمده‌اند:

- برای کاربردهای کلاسه‌بندی،رگرسیون و تخمین توزیع شده قابل استفاده می‌باشد.
- از دسته‌بندی چند کلاسه پشتیبانی می‌نماید.
- حمایت از تخمین احتمالاتی
- از توابع هسته‌ی گوناگون استفاده می‌نماید، ضمن آنکه ماتریس هسته در آنها از قبل محاسبه شده است.
- سورس آن به دو زبان C++ و Java موجود می‌باشد.
- در زبان‌ها و محیط‌های مختلف قابل استفاده می‌باشد.

(Python, R, MATLAB, Perl, Ruby, Weka, Common LISP, CLISP, Haskell, OCaml, LabVIEW, and PHP interfaces. C# .NET code and CUDA)

- از نسخه‌ی 2.8 به بعد از روش SMO استفاده می‌نماید.

همچنین چندین دیتاست مختلف توسط ارائه دهندگان LIBSVM طراحی و آماده‌ی استفاده می‌باشد.

آخرین نسخه‌ی این بسته تا زمان نگارش این مقاله، نسخه‌ی ۳.۱۸ می‌باشد که در آوریل ۲۰۱٤ منتشر شده است.

۶-۳- svm-struct-matlab

svm-struct-matlab عنوان بسته‌ی‌است که توسط Andrea Vedaldi برای استفاده‌ی ساده‌تر و کارآمدتر از SVM در MATLAB به صورت مجموعه‌ای از توابع پیاده‌سازی شده است که در MATLAB قابل فراخوانی و استفاده می‌باشد.

۷- جمع‌بندی و نتیجه‌گیری

در این مقاله مروری بر ماشین‌های بردار پشتیبان، حالت‌های مختلف دسته‌بندی داده‌ها، روش‌های حل مسئله‌ی بهینه‌سازی مرتبه‌ی دوم حاصل از SVM و کاربردهای مختلف SVM داشتیم. SVM در کاربردهای گوناگونی از شناسایی چهره تا تشخیص گفتار و ... را شامل می‌شود.

روش بردارهای پشتیبان یک روش بسیار کارا در دسته‌بندی داده‌ها می‌باشد از این رو یکی از ابزارهای مهم در یادگیری ماشین از نوع نظارت شده می‌باشد. این روش علاوه بر تمامی نقاط قوت خود محدودیت‌هایی را نیز دارا می‌باشد.

بزرگ‌ترین محدودیت روش بردارهای پشتیبان در انتخاب Kernel می‌باشد. زمانی که Kernel ثابت باشد، طبقه بندی SVM تنها دارای یک پارامتر قابل تغییر توسط کاربر می‌باشد (پارامتر جریمه‌ی خطا). انتخاب بهترین Kernel برای یک مسئله‌ی خاص از مسائل مهم و مطرح می‌باشد.[5]

دومین محدودیت در سرعت و اندازه در آموزش و تست می‌باشد. آموزش پایگاه داده‌های بسیار بزرگ، مسئله‌ای غیرقابل حل به روش عددی می‌باشد .راه‌حل‌هایی برای این قبیل از مسائل ارائه گردیده که به برخی از آنها در مقاله اشاره گردید.

داده‌های گسسته مشکل دیگری را سبب می‌شوند، هرچند با مقیاس‌بندی‌های مجدد می‌توان به نتایج خوبی رسید.[5] یکی دیگر از مسائل، مشکل در طراحی تفکیک‌کننده برای SVM چند کلاسه می‌باشد که در اینجا نیز راه‌حل‌هایی ابداع گردیده است.

این روش به نسبت سایر روش‌های کلاسه‌بندی داده‌ها مانند Neural Network در صورتی که انتخاب‌ها مناسب انجام شوند(انتخاب حل‌کننده‌ی درجه‌ی دوم، انتخاب Kernel و ...)از دقت بسیار خوب و بالایی برخوردار می‌باشد و در کارهایی نظیر شناسایی الگو و شناسایی دست‌خط کارائی بالایی را به نمایش گذارده است.

پیوست ۱

کد مربوط به تولید شکل ۱۰ در MATLAB :

```
r = sqrt(rand(100,1)); % radius
t = 2*pi*rand(100,1); % angle
data1 = [r.*cos(t), r.*sin(t)]; % points

r2 = sqrt(3*rand(100,1)+1); % radius
t2 = 2*pi*rand(100,1); % angle
data2 = [r2.*cos(t2), r2.*sin(t2)]; % points

plot(data1(:,1),data1(:,2),'r.')
hold on
plot(data2(:,1),data2(:,2),'b.')
ezpolar(@(x)1);
ezpolar(@(x)2);
axis equal
hold off
```

پیوست ۲:

روش ضرائب لاگرانژ و شرایط KKT

روش ضرائب لاگرانژ

هنگامیکه با قیود به شکل معادله (تساوی) سر و کار داریم، این روش بهینه‌سازی بسیار کارا خواهد بود. به‌علاوه، هنگامیکه مقادیر ثابت قیود تغییر می‌کند، اطلاعات با ارزشی از مقدار بهینه تابع هدف در اختیار می‌گذارد.

ابتدا، تابع هدف F را در نظر بگیرید که بر زیرمجموعه‌ای چون X از فضای اقلیدسی تعریف شده است (پس عناصر X می توانند بردار باشند)؛ که این زیرمجموعه، توسط قیود به شکل g(x)=b تعریف می‌شود. تابع لاگرانژ در این حالت عبارت خواهد بود از

$$L(x,y)=F(x)+y.(b-g(x))$$

(۷۳)

شرایط لازم را می‌توان، از طریق یافتن نقاط بحرانی تابع لاگرانژ (ماکزیمم‌سازی بدون قید)، به دست آورد.

مثال: فرض کنید مساله‌ی بهینه‌سازی زیر را داریم:

$$\max\{u(x_1,x_2):px_1+px_2=m\}$$

(۷٤)

لاگرانژین این مسئله به شکل زیر خواهد بود:

$$L(x_1,x_2,\lambda) \coloneqq u(x_1,x_2)+\lambda(m-p_1x_1-p_2x_2)$$

(۷۵)

شرایط کاروش-کان-تاکر(KKT)

در ریاضیات، شرایط کاروش-کان-تاکر، (Karush-Kuhn-Tucker)و یا کان-تاکر(Kuhn-Tucker) ، شرایط لازم برای جواب بهینه در برنامه‌ریزی غیرخطی می‌باشد، مشروط بر آنکه برخی شرایط Regularity برقرار شود. کان-تاکر در واقع تعمیمی است بر روش ضرایب لاگرانژ که در آن قیود نامساوی هم در نظر گرفته شده است .مسئله بهینه‌سازی غیرخطی زیر را در نظر بگیرید:

$$\text{Minimize } f(x)$$

$$\text{subject to: } g_i(x)\leq 0 \text{ , } h_l(x)\leq 0$$

(۷٦)

تابع هدف$f(.)$ تابعی است که قصد حداقل سازی آن را داریم؛ $g_i(.)$ $(i=1,...,m)$ توابعی هستند که قیود نامساوی را تشکیل می‌دهند و $h_l(.)$ $(l=1,...,m)$ توابعی‌اند که قیود تساوی را شکل می‌دهند، mو l به ترتیب تعداد قیود نامساوی و تساوی هستند. شرایط کان-تاکر، به افتخار هارولد و. کان (Harold W. Kuhn) و نیز آلبرت و. تاکر (Albert W. Tucker) نام‌گذاری شده است که اول بار، این شرایط را منتشر کرده‌اند. پژوهشگران بعدی، شرایط لازم این مسئله را یافتند .ویلیام کاروش (William Karush) در پایان‌نامه کارشناسی‌ارشد خود، این شرایط را تشریح کرده است.

- شرایط لازم

فرض که تابع هدف $f:R^n\rightarrow R$ باشد و توابع قیود نامساوی به شکل $g_i:R^n\rightarrow R$ و توابع قیود مساوی به شکل $g_i:R^n\rightarrow R$ باشند. اگر x^* ، مینیمم موضعی باشد که برخی شرایط Regularity را برآورده کند، آنگاه ثابتی چون $u_i(i=1,...,m)$ و $\lambda_j(j=1,...,m)$ وجود خواهد داشت بطوریکه:

$$\nabla f(x^*)+\sum_{i}^{m}\mu_i\nabla g_i(x^*)+\sum_{j=1}^{l}\lambda_j\nabla h_j(x^*)=0$$

(۷۷)

$$g_i(x^*)\leq 0, for\ all\ i=1,\dots,m$$

(۷۸)

$$h_j(x^*) = 0, for\ all\ j = 1, \dots, l$$

(۷۹)

$$\mu_i \geq 0, for\ all\ i = 1, \dots, m$$

(۸۰)

$$\mu_i g_i(x^*) = 0, for\ all\ i = 1, \dots, m$$

(۸۱)

- شرایط کافی

اگر مجموعه جواب‌های ممکن، مجموعه‌ای ناتهی و فشرده و ضمناً محدب باشد، و تابع هدف بر این مجموعه، پیوسته و مقعر باشد؛ در این صورت، ماکزیمم موضعی، ماکزیمم مطلق است. بعلاوه، اگر تابع هدف اکیدا مقعر باشد، در این صورت جواب یکتاست، یعنی یک ماکزیمم مطلق یکتا وجود دارد. پس یک راه اطمینان حاصل کردن از اینکه، جواب‌های حاصل از شرایط مرتبه اول کان-تاکر، ماکزیمم هستند، بررسی شرایط فوق‌الذکر است.

هرگاه تابع هدف f و قیود نامساوی g_i بطور پیوسته مشتق‌پذیر و محدب باشند و قیود تساوی h_i توابع آفین باشند؛ شرایط لازم، کافی نیز خواهند بود.

مارتین ۱۹۸۵(Martin 1985) ، دسته‌ای از توابع را مشخص کرد که در آن‌ها شرایط کان-تاکر، وجود بهینه سرتاسری را تضمین می‌کند، که توابع invex خوانده می‌شوند. اگر قیود تساوی، توابع آفین باشند، قیود نامساوی و تابع هدف بطور پیوسته مشتق‌پذیر و invex باشند، شرایط کان-تاکر برای بهینه سرتاسری، کافی نیز خواهد بود.

مراجع

[1] K.M.Lin,C.J.Lin, " A Study on Reduced Support Vector Machines," IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 14, NO. 6,pp.1449-1459,NOVEMBER 2003.

[2] I.W. Tsang, J.T. Kwok, P.M.Cheung, "Core Vector Machines:Fast SVM Training on Very Large Data Sets," Journal of Machine Learning Research 6 (2005) 363–392,2005.

[3] Y.J.Lee, S.Y.Huang, "Reduced Support Vector Machines:
A Statistical Theory, " IEEE TRANSACTIONS ON NEURAL NETWORKS, 2006.
[4] J.Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," 1998.

[5] C.J.C.Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery* 2.2, 121-167, 1998.

[6] C.M.Bishop, M.E. Tipping, "Variational relevance vector machines," *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2000.

[7]J. Quinonero-Candela, L.K.Hansen, "Time series prediction based on the Relevance Vector Machine with adaptive kernels," *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on* , vol.1, no., pp.I-985,I-988, 13-17 May 2002.

[8] H.Byun, S.W.Lee, "Applications of support vector machines for pattern recognition: A survey," *Pattern recognition with support vector machines*. Springer Berlin Heidelberg, pp. 213-236, 2002.

[9] C.Campbell, "Kernel methods: a survey of current techniques," Elsevier, *Neurocomputing* 48.1, pp. 63-84, 2002.

[10] R.E.Fan, C.Pai-Hsuen, C.J.Lin, "Working set selection using second order information for training support vector machines," *The Journal of Machine Learning Research* 6, pp.1889-1918, 2005.

[11] M.Tipping, "Sparse Bayesian learning and the relevance vector machine," *The journal of machine learning research* 1, pp.211-244,2001.

[12] "Hilbert space," http://en.wikipedia.org/wiki/Hilbert_space, [Online Access:Jol. 11, 2014]

[13] " Sequential minimal optimization," http://en.wikipedia.org/wiki/Sequential_minimal_optimization, [Online Access: Jol. 11, 2014]

[14] E. Osuna, R. Freund, and F. Girosi, “Training support vector machines: An application to face detection,” in Proc. CVPR ’97, 1997.

[15] The Analysis of Decomposition Methods for Support Vector Machines

[16] T.Joachims, “Making large-scale SVM learning practical,” in Advances in Kernel Methods—Support Vector Learning, B. Schölkopf, C. J. C.Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1998.

[17] L.Kaufman, “Solving the quadratic programming problem arising in support vector classification,” in Advances in Kernel Methods—Support Vector Learning, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999.

[18] O.L.Mangasarian, D.R.Musicant, “Successive overrelaxation for support vector machines,” IEEE Trans. Neural Networks, vol. 10, pp. 1032–1037, Sept. 1999.

[19] J.C.Platt, “Fast training of support vector machines using sequential minimal optimization,” in Advances in Kernel Methods—Support Vector Learning, B. Schölkopf, Ed. Cambridge, MA: MIT Press, 1998.

[20] C.Saunders, M.O.Stitson, J.Weston, L.Bottou, B.Schölkopf, and A.Smola, “Support vector machine reference manual,” Royal Holloway Coll., Univ. London, Egham, U.K., Tech. Rep. CSD-TR-98-03, 1998.

[21] N.Takahashi, M.Kuranoshita, Y.Kawazoe, J.Guo, J.I.Takeuchi, "A New Working Set Selection for Decomposition-Type SVM Learning Algorithms."

[22] Lin, Chih-Jen. "On the convergence of the decomposition method for support vector machines," Neural Networks, IEEE Transactions on 12.6,p.p.1288-1298, 2001.

[23] Takahashi, Norikazu, N.Tetsuo, "Global convergence of decomposition learning methods for support vector machines," Neural Networks, IEEE Transactions on 17.6 ,p.p.1362-1369, 2006.

[24] Joachims, Thorsten, "Introduction to support vector machines," 2003.

[25] Vasant Honavar, “Sequential Minimal Optimization for svm,” Lowa State University.

[26] R.R.Padron, “A Roadmap to SVM Sequential Minimal Optimization for Classification, ” School of Electrical Engineering and Computer Science University of Central Florida Orlando.

[27] S.Keerthi, Sathiya,"Improvements to Platt's SMO algorithm for SVM classifier design," Neural Computation 13.3, p.p. 637-649, 2001.

View publication stats