UPTEC X 04 029
JAN 2004

ISSN 1401-2138

BJÖRN LINDH

Feature selection with support vector machines in functional genomics

Master's degree project

UPPSALA UNIVERSITET

Molecular Biotechnology Programme

Uppsala University School of Engineering

UPTEC X 04 029	**Date of issue 2004-01**

Author

Björn Lindh

Title (English)

Feature Selection with Support Vector Machines in Functional Genomics

Title (Swedish)

Egenskapsurval med supportvektormaskiner inom funktionell genomik

Abstract

There has been a rapid development of new measurement technologies to monitor the expression activity of the genome. The invention of micro-arrays permits simultaneous measurements of a large number of mRNA-molecules. This holds the promise to define diseases in molecular terms and could provide a basis for medical diagnoses. In this context it is crucial to develop computational analysing techniques that can classify and therefore differentiate between samples from normal and disease states. Support vector machines (SVM) is a learning system, which earlier have proved promising characteristics for this purpose. In this study we evaluated support vector machines for feature selection in gene expression data, and tried to give answer to the question why SVM seems to have other characteristics compared to less sophisticated classifiers. Also an alternative capacity evaluation method to "Leave One Out" (LOO) is discussed. The results showed how sparseness of data can affect the learning capacity and that either choice of margin softness or kernel seemed to affect the generalisation property of the system. This gives a hint of how to shape an SVM for classification of gene expression data.

Keywords

Lärande system, SVM, supportvektormaskin, cancerklassificering, microarray, genexpressionsdata

Supervisor: **Jesper Tegnér**

Scientific reviewer: **Bengt Persson**

Language: **svenska**	Security
ISSN 1401-2138	Classification
Supplementary bibliographical information	Pages **40**

Biology Education Centre Biomedical Center Husargatan 3 Uppsala

Box 592 S-75124 Uppsala Tel +46 (0)18 4710000 Fax +46 (0)18 555217

Egenskapsurval med supportvektormaskiner inom funktionell genomik

Björn Lindh

Populärvetenskaplig sammanfattning

Kan man träna en dator till att känna igen cancer? Det tror professor Jesper Tegnér som driver ett projekt vid Linköpings Universitet om just detta. För knappt 10 år sedan uppfann en rysk matematiker vid namn Vapnik ett lärande system som han kallade Supportvektormaskin (SVM). Det kan appliceras på cancerceller på följande sätt. Alla mRNA-koncentrationer i en cell med känd sekvens kan idag mätas i en så kallad mikroarrayanalys. Låt varje mRNA-koncentration vara en parameter med en egen axel i ett koordinatsystem. Mäts 5000 gener kommer således ett 5000-dimensionellt rum att skapas där varje cell får en viss koordinat beroende på sitt specifika genuttryck. Om en cell är en tumörcell av ett visst slag kommer koordinaten för denna cell hamna en bit bort i rummet i jämförelse med friska celler p.g.a. att några gener är utslagna i tumörcellen. Supportvektormaskinen räknar då ut det mest optimala hyperplanet som skiljer punktmängden av friska celler och punktmängden av tumörceller åt. Planet kan sedan användas som en urskiljningsregel vid test av nya celler. På så sätt kan en dator ”tränas” i att känna igen cancer.

Metoden har många fördelar gentemot den idag brukliga mikroskopmetoden. Allt sker automatiskt i en maskin, snabbt och enkelt, men framförallt erhålls direkt information om vilka gener som orsakar åkomman och en behandling kan sättas in just där. SVM har testats för andra system såsom postnummerigenkänning i brevsorteringsmaskiner med gott resultat, men kan nu alltså även användas för cancerklassificering. Min specifika uppgift har varit undersöka vissa inställningsparametrar för att ställa in maskinen så att den fungerar optimalt för detta syfte.

Examensarbete 20 p i Molekylär bioteknikprogrammet

Uppsala universitet Januari 2004

Innehåll

EGENSKAPSURVAL MED SUPPORTVEKTORMASKINER INOM FUNKTIONELL GENOMIK

BJÖRN LINDH

SAMMANFATTNING. Under den senaste tiden har en snabb utveckling skett av nya mätmetoder av genomets aktivtitet. Uppfinnandet av mikroarrayer tillåter simultana mätningar av ett stort antal mRNA-molekyler. Detta lovar gott inför att definiera sjukdomar i molekylära termer och kan utgöra en bas för medicinska diagnoser. Av stor vikt är att utveckla dataanalystekniker som kan klassificera och därför skilja mellan prov från normalt respektive sjukt tillstånd. Supportvektormaskiner (SVM) är ett lärande system som tidigare har visat goda egenskaper i detta syfte [18]. I denna studie vill vi utvärdera supportvektormaskiner för egenskapsurval i genuttrycksdata, samt besvara frågan om varför SVM verkar ha andra egenskaper än andra mindre sofistikerade klassificerare. Även en alternativ kapacitetsutvärderingsmetod till *lämna en utanför* (LOO) diskuteras. Resultaten visar hur glesheten av data kan påverka inlärningskapaciteten samt att varken val av mjukhet på marginalen eller val av kärna verkar ha betydelse för systemets generaliseringsegenskaper. Detta ger en fingervisning om hur en SVM bör formges för klassificering av genexpressionsdata.

1. Inledning

Biologin befinner sig för närvarande under stor utveckling. I tidskriften Science lista för de viktigaste upptäckterna år 2003 handlade sju av de tio första om bioteknik. Detta är en syn vi har vant oss vid det senaste deceniet. De revolutionerande upptäckterna handlar i stort sett uteslutande om insikter i hur livsprocesser går till på molekylär nivå. I och med en sådan förståelse börjar man också få insikt i hur dessa kan förändras. Detta kommer inte bara att inom en snar framtid innebära en mängd nya revolutionerande tekniska och medicinska tillämpningar, utan det kommer även att förändra synen på människan och på liv radikalt hos gemene man.

En revolutionerande följd av denna utveckling är att biologens uppgifter och krav på kunskaper förändras. Då biologin går ner på molekylär nivå kräver detta även kunskaper i fysik, matematik, datavetenskap och teknik. Inte bara utnyttjandet av biologiska organismer i tekniska processer kräver allt mer av dessa kunskaper. Även nya tekniker för att på ett effektivt sett kunna förstå biologiska fenomen kräver mer av sådana kunskaper. Experiment är ofta mycket dyra att genomföra, men med datamodeller över molekylära system och effektiva sökmetoder i världsomspännande databaser med biologisk information kan antalet experiment minimeras. I framtiden krävs ett allt större samarbete mellan biologi och teknik, den saken är klar. Det jag talar om är genombrottet för molekylär bioteknik.

1.1. **Bakgrund.** Biologin har nått ett stadium där analysprocesser för biologisk data har automatiserats och stora mängder information kan erhållas på kort tid.

Ett välkänt exempel är att DNA-sekvensen hos människa och en mängd andra organismer redan har blivit fullständigt kodade. Att analysera stora mängder biologisk information och systematisera denna har i biologikretsar kommit att kallas för -omik. Det benämns proteomik om det handlar om proteiner, genomik om det handlar om gener, metobolik om det handlar om metaboliter etc. Under de 4-5 senaste åren har det också skett en snabb utveckling av nya mätmetoder för att få en inblick i genomets aktivitet. Utveckling av biologiska chips och mikroarraysystem ger oss möjlighet att på ett effektivt sätt studera simultana mätningar på ett stort antal mRNA-molekyler. Detta är lovande inför möjligheten att definiera sjukdomar i molekylära termer och kan komma att utgöra en bas för medicinska diagnoser. I det sammanhanget är det av stor vikt att utveckla dataanalystekniker som kan klassificera och därmed skilja mellan prov från normalt respektive sjukt tillstånd. När funktionen klassificeras parallellt på en mängd gener och resultaten systematiseras benänms detta funktionell genomik. En viktig uppgift är att utveckla effektiva analystekniker för dessa stora datamängder, som kan användas för att skilja mellan exempelvis olika tumörer eller andra skilda genetiska tillstånd hos en cell.

Ett hitintills mycket populärt angreppssätt för att lösa sådana problem har varit att använda lärande system, det vill säga algoritmer som lär sig känna igen mönster i data av olika former med hjälp av träningsdata. De mest använda systemen inom biologin idag torde vara neurala nätverk, vilka även har sin idégrund inom biologin. I neurala nätverk har processer som finns i biologiska nervsystem försökt efterliknas. När några ryska matematiker (Vapnik 1992 [5], 1995 [26]) kombinerade dessa kunskaper med välkända matematiska redskap såsom optimering, statistiska inlärningsteori och kärnmaskiner föll bitarna förvånansvärt väl på plats och man skapade den så kallade *supportvektormaskinen, SVM:en.* Metoden läts mogna i några år och i slutet på 90-talet blev SVM det mest användbara lärande systemet för praktiska tillämpningar. Då mikroarraytekniken uppfanns testades olika lärande system för att analysera de stora datamängder som detta system genererar. SVM:n gav goda resultat [6], vilket inte är förvånande då supportvektormaskiner redan tidigare kunde användas lyckosamt i en rad olika praktiska tillämpningar, såsom handskifts-, siffer- och röstigenkänning. Det mest kända exemplet i Sverige är nog postverkets brevsorteringsmaskin [7], som använder en SVM för att känna igen postnummer på brev. Maskinen tränas med ett relativt stort antal data för att sedan kunna känna igen nya siffror.

Biologiska tillämpningar av SVM som har dykt upp under de senaste tre åren är [30]:

- Gensökning i DNA: En bit av en DNA-sträng är +1 om det är en del av en gen och -1 om den inte är det. Sekvensbaserad genklassificering: Ställ ja- och nejfrågor till en bit DNA.
- Sekundärstrukturspredikтion hos protein.
- 3D-strukturspredikтion hos proteiner.
- Proteinlokalisation i cellen [**?**].

Med dessa exempel vill jag understryka att metoden är mycket generell även om denna rapport främst beskriver klassificering av cancertyper med lärande system varav den mest användbara är SVM.

År 2000 publicerades den artikel som först beskrev hur man kan använda SVM:en för att beskriva mönster i de genetiska data som genereras då man analyserar celler med mikroarrayanalys [6]. Cancerklassificering har delvis varit komplicerat,

eftersom det historiskt sett främst förlitat sig på specifika biologiska insikter snarare än systematiska och statistiska metoder. I 30 år har metoden att klassificera cancer varit densamma, men med SVM introduceras en ny. Tidigare har klassificering gjorts med morfologisk närvaro av tumören, vilket givetvis har sina begränsningar. Några fördelar med SVM:en är att det går att spåra tumörer utan synliga spår, den går att skilja mellan likartade cancertyper samt att metoden automatiserar klassificeringsprocessen. Ännu är dock mikroarrayteknik relativt dyrt. En stor fördel är dock att tekniken inte bara kan användas för att klassificera cancerklasser, utan även att ge information om nya klasser och underklasser som inte går att urskilja med traditionella tekniker [9].

Genuttrycksexperiment producerar högdimensionell data genom att många gener mäts parallellt. Provtätheten är ofta låg på grund av stora experimentkonstnader. Ur dataanalyssynpunkt är för små datamängder inte tillfredställande och det blir inte bättre av att datan är mycket brusig beroende på olika typer av mätstörningar.

Vid en biologisk frågeställning kommer sällan alla dessa 10000-tals dimensioner, dvs gener, att vara relevanta. Det är därför av stor vikt att försöka hitta metoder som kan sålla bort onödigt information. Detta kallas egenskapsurval. Det förhåller sig emellertid så att olika metoder ger olika mängder av informativa gener, medan korrektheten hos klassificeringen med lärande system har varit relativt hög över lag [18].

En mycket intressant detalj vid tidigare undersökningar av SVM:en är att den beter sig annorlunda än andra metoder i den bemärkelsen att den i hög grad tar hänsyn till andra egenskaper än vad som vanligtvis görs med de andra metoderna [18]. Det finns således stor anledning att studera metoden närmare.

Idén med SVM:en som följande [7]. Först presenteras träningsdata för SVM:en i form av en träningsmängd $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}, \mathbf{x}_i \in \mathbb{R}^n, y_i \in Y = \{-1, 1\}$. SVM:en skapar med hjälp av träningsmängden en matematisk regel för hur nya exempel skall klassificeras mellan de två olika klasserna som definierar tecknet på y. Hur SVM:en formulerar klassificeringsregeln kan delas upp i två faser. I den första fasen beskrivs objekteten med hjälp av sina egenskaper som vektorer i ett mångdimensionellt egenskapsrum. En dimension för varje egenskap. Varje vektor projiceras in i ett nytt rum, eventuellt med annan, ofta högre, dimension än det ursprungliga. Det görs på ett sådant sätt att de olika klasserna kan separeras med ett hyperplan i $\mathbb{R}^n$. Uppgiften är att hitta en bra sådan beslutsregel, gärna den bästa. Med bra menas här goda generaliseringsegenskaper och en effektiv algoritm som kräver få beräkningar och även kan behandla stora datamängder.

Med beslutsregel menas att om ett nytt exempel introduceras tilldelas det y-värdet 1 eller -1 beroende på vilken sida av planet det befinner sig. På detta sätt väljer alltså algoritmen vilken klass ett nyintroducerat exempel tillhör.

I detta arbete har jag studerat ett specifikt problem. Man bör emellertid ha i åtanke att SVM:en, som tidigare påpekats, är en mycket generell metod, som kan användas i många sammanhang, både inom biologin och inom andra vetenskaper.

1.2. **Problembeskrivning.** Grundsyftet med denna studie är att finna metoder för att ur stora datamängder göra bestämningar av genetiska nätverk. Med ett genetiskt nätverk menas en karta över en mängd gener av vilken det framgår hur genernas uttryck påverkar varandra. För att studera metoder som kan användas löses ett mindre problem: cancerklassificering. Det finns många tänkbara metoder för att lösa klassificeringsproblem. En av de mest användbara metoderna hittills

har varit det lärande systemet supportvektormaskiner (SVM). SVM:en visar sig ta vara på andra egenskaper än tidigare använda metoder, som andra lärande system och statistiska metoder. Det är relevant att ta reda på varför.

Det är också av stor betydelse att undersöka parameterinställningar till SVM:en i kombination med olika egenskaper av genuttrycksdata, vilka kontrollerar vilka egenskaper en SVM kan upptäcka.

1.3. **Syfte och omfattning.** Utifrån presentationen av det givna problemet ovan, kan syftet och omfattningen sammanfattas enligt följande:

Syftet med detta examensarbete är att:

(1) Teoretiskt förstå de matematiska processerna bakom supportvektormaskinen.
(2) Utvärdera supportvektormaskiner för egenskapsurval i genuttrycksdata, samt besvara frågan om varför SVM verkar ha helt andra egenskaper än andra mindre sofistikerade klassificerare.

Omfattningen på arbetet är:

(1) Grundligt förstå hur en SVM fungerar och kunna förklara detta teoretiskt samt kunna förstå dess programmeringskod.
(2) Översiktligt redogöra för vad som görs inom forskningsfältet för tillfället.
(3) Jämföra olika SVM, med olika parameterinställningar och kärnor och testa på mikroarraydata, dels artificiell och biologisk.

1.4. **Tidigare arbeten.** Vapnik och hans medarbetare lade grunden för teorin i hans två artiklar [5] och [10]. Därefter började metoden tillämpningas inom olika områden som [8], [26] och [19]. 1999 applicerades metoden för första gången på genexpressionsdata av Brown [6] och har sedan dess följts upp av ett antal artiklar exempelvis [2] och [29].

Mitt arbete kan ses som en uppföljning av Nilsson och Tegners tidigare arbete [18] med att utvärdera olika metoder för klassificering av genexpressionsdata. Olika metoder utvärderades och jämfördes. Det visas att SVM:en har bättre generaliseringsegenskaper än andra klassificeringsmetoder, som t-test, PCA, klustringstekniker, självorganiserande träd och variansfilter. SVM:en betedde sig också annorlunda jämfört med andra metoder. Då metoderna utvärderades med *lämna en utanför* (LOO) (se kapitel 5.1.5) felklassificerades andra punkter med SVM än de som felklassificerades av andra metoder. SVM:en utnyttjar således andra egenskaper i inlärningsprocessen än andra inlärningsmetoder. Därav följer intresset av en fördjupning i förståelsen av SVM:ens egenskaper.

Även andra resultat i litteraturen [10], [6] och [2] visar på att klassificering från mikroarraydata ofta ger tillfredsställande resultat jämfört med andra maskintest och kliniska diagnoser. Detta trots stora störningar i datan.

Ytterligare arbeten som berör detta arbete är bland andra [2] som undersöker mikroarraydata och visar att polynomiella kärnor ger samma resultatprestanda som lineära. Sålunda spelar inte kärnteori någon central roll vad gäller att hitta rätt parameterinställningar till SVM:en. Alferis visar också flertalet exempel på att risken generellt sett är stor för överanpassning i högdimensionela datamängder.

1.5. **Disposition.** Rapporten är strukturerad på följande sätt:

- Kapitel 2 beskriver de olika matematiska komponenterna i teorin för supportvektormaskiner: perceptronen, kärnmetoder, optimering och statistisk

inlärningsteori. I delkapitelet *SVM* beskrivs hur dessa komponenter knyts ihop och bildar en enhetlig teori. Därefter följer en kort beskrivning av implementering och vad som skiljer SVM:en från andra metoder, framför allt från det klassiska neurala nätverket.
- I kapitel 3 beskrivs hur data har genererats, både artificiell och biologisk. Det finns också beskrivet vilka program som har använts och hur simuleringsförsöken är uppbyggda.
- Kapitel 4 innehåller erhållna resultat från simuleringsförsöken.
- Kapitel 5 diskuterar resultat, slutsatser, felkällor samt förslag på fortsatt forskning.
- I kapitel 6 tackas alla som har bidriagit till arbetet.

2. Teori supportvektormaskiner

Supportvektormaskiner (SVM) är en familj av lärande algoritmer, vilka för tillfället anses som en av de mest effektiva för tillämpade problem. Det är en metod som är starkt på frammarsch. I många sammanhang har SVM bättre inlärningsegenskaper än exempelvis neurala nätverk, som är den mest kända algoritmfamiljen bland lärande system [4]. Syftet med SVM:er och andra inlärningsalgoritmer är att med hjälp av inlärningsdata känna igen mönster och på så sätt kunna avgöra egenskaper hos ny data som presenteras för algoritmen. Det har utvecklats många andra artificiella inlärningstekniker, som exempelvis neurala nätverk, Fischers diskriminant, klassificerings- och regressionsträd (CART), klustringstekniker och olika statistiska tekniker. SVM tycks emellertid för tillräckligt komplicerade system vara den mest användbara. SVM:er har också den fördelen att den är baserad på en mycket elegant och hållfast matematisk grund, vilket gör det möjligt att ha god kontroll över dess processer. SVM:en är ett resultat av teoretisk forskning, snarare än av prövning. Förutom detta faktum kan dess största fördelar sammanfattas på följande sätt [7]:

Goda generaliseringsegenskaper: Metoden klarar sig relativt bra från överinlärning, mycket bättre än exempelvis neurala nätverk. Det finns också metoder för att reglera och kontrollera detta fenomen.

Hög beräkningshastighet: Dual form gör att allt kan beräknas i ett steg. Dessutom finns det väl utvecklade algoritmer för de optimeringsproblem som uppkommer. Algoritmen är så konstruerad att den enbart använder information från en relevant delmängd av inlärningsdatan, så kallade supportvektorerna (förklaras nedan).

Robust: Antalet fria parametrar ökar inte med antalet dimensioner.

Anm: *Generaliseringsegenskaper* Med generaliseringsegenskaper menas förmågan hos en hypotes att korrekt klassificera data som inte är i träningsmängden.

Överinlärning är ett problem för inlärningsalgoritmer. Med begreppet menas att en algoritm ibland kan hitta en komplicerad regel som perfekt klassificerar objekten i träningsmängden. Regeln kan dock vara oanvändbar för att klassificera nya observationer om den är alltför relaterad till träningsmängden. Vi säger att en sådan regel inte generaliserar bra. Lärande system är alltid en avvägning mellan överinlärning och enkel approximation på beslutslinjen.

Hur går kontrollen av överinlärningsprocessen till? Svaret kommer från den statistiska inlärningsteorin [27]. Denna teori är ganska snårig och jag nöjer mig

med att presentera huvudresultaten. Förenklat skulle man kunna säga att Vapnik och medarbetare har med den statistika inlärningsteorin hittat en länk mellan förmågan hos en algoritm att lära sig en regel som är bra på att klassificera och samtidigt har bra generaliseringsegenskaper. Detta är en implikation i många steg från den statiska inlärnigsteorins huvudsats:

$$\mathbf{E}R(f) \leq R(f^*) + c\sqrt{\frac{VCdim}{N}}$$

där $\mathbf{E}$ är väntevärdesoperatorn, R risk (definieras $R(f) = P(f(X) \neq Y)$), $\hat{f}$ = en inlärningsregel med empirisk riskminimering på träningsdatan, f^* = en inlärningsregel generaliseringsmaximering, $VCdim$ = *ett mått på överanpassning* och N = *antalet observationer i träningsmängden*. Andemeningen med satsen är att risken hos en inlärningsalgoritm att klassificera fel är proportionell mot kvadratroten ur VC-dimensionen. En utförligare definition på VC-dimension hittas under kapitlet *kärnmaskiner*.

Satsen leder fram till ett minimeringsproblem av den så kallade *strukturella risken* för felklassificering, vilken minimeras genom att minimera marginalen, eller ekvivalent maximera viktvektorn för hyperplanet. Detta är ett optimeringsproblem med unik lösning, som visar sig ha många goda egenskaper. Dels är det ett kvadratiskt programmeringsproblem, vilket det finns effektiva lösningsmetoder till. Dels reduceras problemets lösning genom optimeringslärans KKT-villkor automatiskt ner till att bara bero på de vektorer som ligger närmast det avskiljande hyperplanet. Endast denna delmängd av träningsexempel ger således fullständig information om lösningen och har därför fått namnet *supportvektorer*.

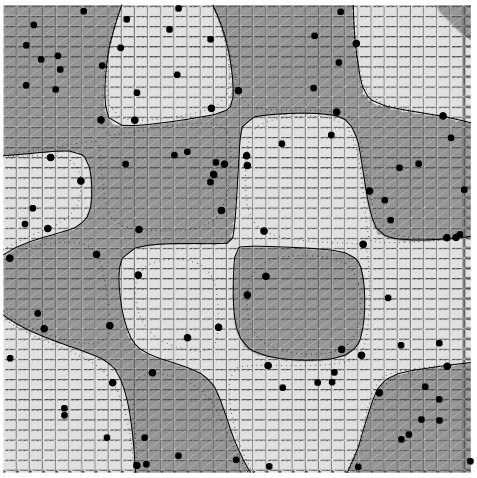
Långt ifrån alla inlärningsdatamängder är lineärt separabla, vilken ovan beskrivda perceptron kräver. Ett sätt att lösa detta problem är att introducera begreppet kärnor. Kärnmetoden ger unik optimallösning till samtliga ickelineära system med ändlig träningsmängd (förutsatt lämpligt val av kärna) genom att projicera problement in i ett rum, som ofta (men inte alltid) har högre dimension än indatarummet. Det fina är att en så kallad *dual representation* fortfarande gör det möjligt att definiera projektion endast implicit genom en inre produkt, vilket implicerar att antalet parametrar då inte kommer att öka med antalet dimensioner. Dimensionberoendet utrönes utförligare i kapitlet om statistisk inlärningsteori. Det finns dock två problem som uppkommer vid användningen av kärnor:

(i) Hur erhålls ett separerande hyperplan som generaliserar bra även i ett högdimensionellt egenskapsrum?

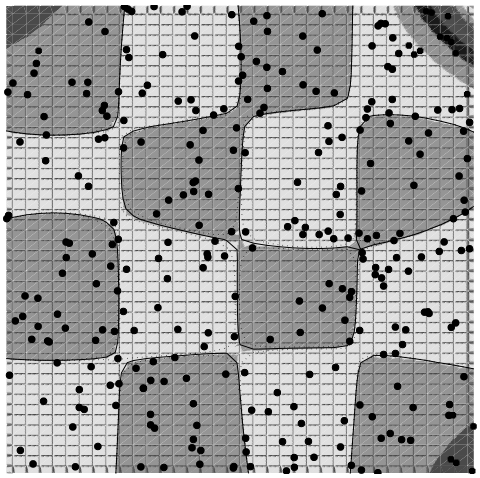
(ii) Hur behandlar man högdimensionella rum rent beräkningstekniskt.

SVM:en löser problem (i) genom att konstruera en γ-marginal och mjuka marginaler. Definition finns i kapitlet *Supportvektormaskiner*. En stor γ-marginal ger nämligen liten VCdim. Problem (ii) löses genom inre produkter och kärnteori [27]. Mercers sats implicerar att om bara kärnor väljs på ett lämpligt sätt så går det att kontollera VCdim även för högdimensionella eller till och med ∞-dimensionella egenskapsrum (Hilbertrum). En sådan kärna är radiella basfunktioner, vilket följande exempel visar.

Exempel Schackbräde: Exemplet visa hur ett schackbrädemönster kan återskapas genom att använda en SVM med radiell basfunktionkärna. 100 respektive 400 punkter slumpas ut på ett 4 x 4 rutor stort schackbrädemönster. Om punkten hamnar på en vit ruta ger vi dess tillhörande y-värde värdet 1,



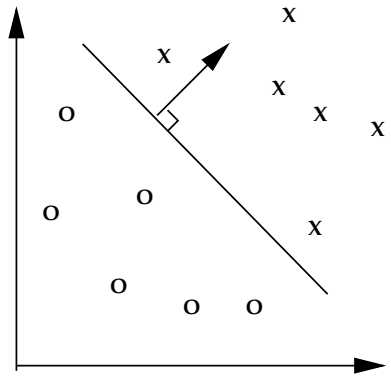
FIGUR 1. 100 punkter slumpas ut på ett schackbräde. SVM:en får information om punkterna hamnade på vit eller svart ruta och försöker därefter återskapa schackmönstret. I detta fall har radiella basfunktioner använts som kärna.



FIGUR 2. Samma test som ovan fast för 400 träningspunkter

annars -1. SVM:en låts sedan försöka återskapa rutmönstret med hjälp av träningsdatan. I det här fallet används exponentiella radiella basfunktioner som kärna, vilket är en projektion in i ett Hilbertrum där träningsexemplerna kan separeras lineart med ett hyperplan (se figur 1 och figur 2).

I följande kapitel fördjupas de olika delarna mer i detalj. Först presenteras de olika matematiska områdena: perceptronen, kärnteorin, optimeringteorin och statistiska inlärningsteorin var för sig. Därefter knyts delarna ihop under delkapitlet *Supportvektormaskiner*. På slutet diskuteras även kort implementerings-tekniker samt



FIGUR 3. Med hjälp av träningsdata skapas ett beslutsplan med en normal $\mathbf{w}$. Detta illustreras här i $\mathbb{R}^2$.

en jämförelse i teori och prestanda med andra metoder. Det mesta materialet i detta kapitel är hämtat ur [7] och [27].

Slutligen vill jag också nämna att SVM:en även kan användas, förutom till binär klassificering, till ickelineär (och lineär) regression [7].

2.1. **Perceptronen.** Den lineära diskriminanten, eller som den också kallas: perceptronen, beskrevs första gången av Frank Rosenblatt 1956 [25]. Den är grundidén för all artificiell inlärning och har sedan dess introduktion enbart förfinats och generaliserats till mer avancerade problem, men har fortfarande samma grundstomme. Det enklaste fallet är en binär klassificerare som lär sig skilja mellan två klasser. Detta går att generalisera mycket enkelt till separation mellan flera klasser. Jag nöjer mig däremot med att presentera teorin för separation mellan två klasser.

För att kunna skapa en beslutsregel behövs en träningsmängd:

Def: *Träningsmängd* För $X \subseteq \mathbb{R}^n$, ett indatarum och $Y = \{-1, 1\}$, utdatarum definieras en träningsmängd: $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\} \subseteq (X \times Y)^l$, där $\mathbf{x}_i$ är exempel, y_i tillhörande ettiketter och l antalet exempel.

Utifrån denna mängd skapas en beslutsregel i form av ett hyperplan som delar in rummet i två halvrum. I 3 nedan visas hur ett sådant hyperplan definierat av träningsdatan skulle se ut i två dimensioner. Ett där den ena klassen hör hemma och ett för den andra klassen. Hyperplanet kan beräknas med många olika metoder, vilka oftast bygger på statistiska metoder eller optimeringsmetoder. I fallet med SVM:er uppstår ett kvadratiskt programmeringsproblem, som diskuteras under kapitlet *optimeringslära*.

Klassificeringen i perceptronen går till på följande sätt. Låt $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ vara en klassificeringsfunktion på så sätt att $\mathbf{x} = (x_1, \ldots, x_n)'$ antingen har en egenskap som definieras av f, annars är den negativ. Vi säger då att $\mathbf{x}$ är positiv. Detta gäller om $f(x) \geq 0$, annars är den negativ. Anta att klassen av sådana funktioner $\mathcal{F} = \{f | f \text{ lineär}\}$ av lineära funktioner på formen $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$, där $(\mathbf{w}, b) \in \mathbb{R}^n \times \mathbb{R}$, $\mathbf{x} \in X$ är parametrarna som kontrollerar beslutsregeln. Dessa ges av $sgn(f(\mathbf{x}))$, $(sgn(0) = 1)$. Det gäller alltså att finna en så bra beslutsregel som möjligt. Med bra menas här en regel som kan klassificera träningsdatan korrekt och även kunna generalisera regeln till nyintroducerade exempel. Det är bara möjligt

om datan är lineärt separabel. Om vi antar att så är fallet, kan vi låta beslutsregelns parametrar i $f(\mathbf{x})$ helt bestämmas av inlärningsdatan. I det lineära fallet bildar f ett hyperplan. Hyperplanet kan således ses som ett affinet delrum av R^n i $n-1$ dimensioner, som otvetydigt definierar två distinkta klasser. $\mathbf{w}$ är normalvektorn till planet. b kallas för *viktning* eller *bias* om man vill använda ett utländskt ord.

Utgående från ovanstående teori kan perceptronsalgoritmen beskrivas som följande:

(1) Utgå från $\mathbf{w}_o, b_o = 0$
(2) Om marginalen ≤ 0 för ett träningsexempel, dvs exemplet är felklassificerat, flytta $\mathbf{w}$ och b ett steg i riktning så att marginalen ökas.
(3) Fortsätt tills exemplet är på rätt sida om planet.
(4) Upprepa de tidigare stegen för alla träningsexempel tills alla ligger på rätt sida om hyperplanet.

Denna enklaste form av perceptronen kräver att träningsdatan är lineärt separabel. Det behöver den naturligtvis inte vara. Jag kommer nu att diskutera ett sätt att lösa detta problem. Men först några definitioner:

Def: Marginalen för ett exempel $(\mathbf{x}_i, y_i)$ med avseende på hyperplanet $(\mathbf{w}, b)$ ges av:

$$\gamma_i = y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$$

Anm: Det är dock brukligt att använda sig av en normerad marginal för att få goda egenskaper hos det optimeringsproblem som uppkommer (se kap. opimeringslära). Planet på normerad form blir då:

$$(\frac{1}{\|\mathbf{w}\|}\mathbf{w}, \frac{1}{\|\mathbf{w}\|}b)$$

Den geometriska marginalen mäter det Euklidiska avståndet mellan punkterna och det separerande hyperplantet i X.

Marginalen γ_s till S definieras som den maximala marginalen över alla hyperplan, vilket benämns maximala marginalhyperplanet. Storleken av dess marginal är alltid positiv för lineärt separabla träningsmängder. Det kan nu bevisas (se [7]) att ett hyperplan kommer att hittas inom ett ändligt antal iterationssteg med perceptronalgoritmen.

Def: Fixera $\gamma > 0$, definiera *slapp marginalvariabel* till exemplet $(\mathbf{x}_i, y_i)$ med avseende på hyperplanet $(\mathbf{w}, b)$ och målmarginalen som

$$\xi((\mathbf{x}_i, y_i), (\mathbf{w}, b), \gamma) = \xi_i =$$

$$= max(0, \gamma - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b))$$

Om ett exempel $(\mathbf{x}_i, y_i)$ är på fel sida om γ blir således ξ_i positivt och dess storlek ger ett mått på hur mycket på fel sida om den korrekta marginalen den befinner sig. Detta tillvägagångssätt har många fördelar. Vi behöver inte kräva lineär separabilitet hos data och vi får ett mått på exempel som med stor sannolikhet innehåller stora mätfel. Metoden är alltså särskilt gynnsam för ickelineärseparabel data med stora störningar.

Vi ska observera några saker angående planets ekvation. Det inses lätt att $\mathbf{w}$ kan beskrivas som en lineärkombination av träningspunkter

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

Notera att α_i är proportionell mot antalet missklassificeringar för respektive i. Det vill säga att svårklassificerade punkter erhålls stora α, av vilket det följer att det därför går enkelt att detektera punkter med till exempel stora mätfel. I kapitlet *optimeringsmetoder* nedan beskrivs hur man beräknar dessa.

Fördelen med att kunna skriva $\mathbf{w}$ som en lineärkombination gör att vi kan beskriva beslutsregeln direkt utifrån en träningmängd S med en så kallad dual beslutsregel. Detta är en av de vinnande egenskaperna hos SVM:en. Den duala beslutsregeln ser ut på följande sätt:

$$\begin{aligned} h(\mathbf{x}) &= sgn(\langle \mathbf{w}, \mathbf{x} \rangle + b) = \\ &= sgn(\langle \sum_{j=1}^{l} \alpha_j y_j \mathbf{x}_i, \mathbf{x} \rangle + b) = \\ &= sgn(\sum_{j=1}^{l} \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x} \rangle + b) \end{aligned}$$

Det innebär att endast den inre produkten mellan exemplen behöver beräknas. Icke lineärseparabla kan då lösas genom att ändra rummets inre produkt. Jag kommer att diskutera detta djupare i kapitlet nedan om *kärnmetoder*. Formen för indata är oftast enbart samtliga inre produktkombinationer samlade i en matris:

$$\mathrm{G} = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle)_{i,j=1}^{l}$$

G kallas för en Grammatris.

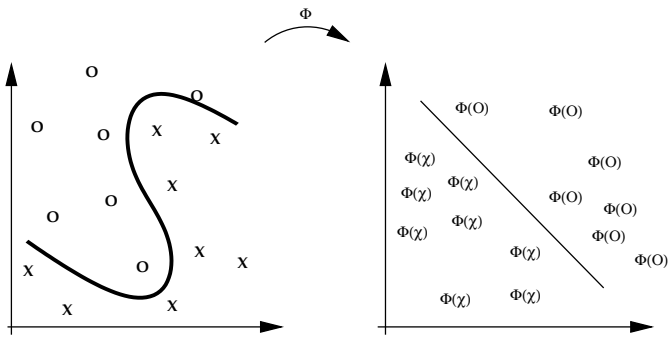
Perceptronen är alltså en algoritm som hittar ett separerande hyperplan. I kapitlet *optimeringslära* kommer jag att diskutera hur detta kan göras på bästa sätt. Där presenteras också mera effektiva algoritmer.

Anm: Även neurala nätverk bygger på perceptronen. Se figur 7 i kapitel 2.6.

En av förutsättningarna för algoritmen är som nämnts att datan som presenteras krävs vara lineärt separabel. För data med stora störningar går detta att lösa med slappa variabler. Om algoritmen skall lära sig känna igen mönster som uppenbart inte är lineärt separabla måste dock en annan teknik användas nämligen att introducera kärnfunktioner.

2.2. **Kärnmetoder.** Supportvektormaskiner är medlemmar i klassen av en större klass av inlärningsalgoritmer, vilken brukar benämnas kärnmetoder. Idéen med kärnmetoder är att avbilda indata indirekt ickelinjärt in i ett teoretiskt rum, ett så kallat *egenskapsrum*, E, genom att byta ut den inre produkten i indatarummet mot en kärnfunktion $K(\mathbf{x}, \mathbf{z})$. E väljs ofta till ett högdimensionellt rum på så sätt att träningsdatans klasser blir lineärt separabla. Detta är enligt [34] alltid möjligt för ändliga träningsmängder S.

Kärnmetoder kan studeras helt för sig själv och har utvecklats sedan 60-talet, långt innan den första SVM:en såg dagens ljus. Teorin smälter dock förvånansvärt fint in och är en högst naturlig komponent i teorin för supportvektormaskiner.



FIGUR 4. Många system kräver en ickelinär beslutsregel. Rummet kan då transformeras in i ett rum, ofta av högre dimension, där beslutsregeln är lineär.

Vi börjar med att definiera en avbildning för projektionen av datan in i egenskapsrummet. Låt oss kalla den för en målfunktion:

$$\phi : X \to E = \{\phi(\mathbf{x}) | x \in X\}$$

$$\mathbf{x} = (x_1, \ldots, x_n) \mapsto \phi(\mathbf{x}) = (\phi_1(x), \ldots, \phi_d(x)),$$

Kvantiteterna som introduceras i E kallas egenskaper medan orginalkvantiteterna kallas attribut.

Uppgiften är att välja ϕ på sådant sätt att E blir lineärt separabelt (se 4). Om en sådan avbildning har lyckats hittas kan sedan ett separerande hyperplan finnas i detta rum med perceptronalgoritmen. Den explicita egenskapavbildningen behöver dock inte kännas till. I stället byts skalärprodukten, det vill säga Grammatrisen G, ut till en kärnfunktion K. Det som krävs av en funktion för att den skall kallas kärnfunktion är:

Def: *Kärnfunktion* kallas en funktion K, sådan att $\forall \mathbf{x}, \mathbf{z} \in X$ gäller att:

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$$

där $\phi : X \to F$

Definitionen kommer ursprungligen från integrationsteorin och används bland annat vid skalärproduktbyten i Hilbertrum.

Hur kommer då beslutsfunktionen att se ut? På primal form får vi följande funktion:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \mathbf{w}_i \phi_i(\mathbf{x}) + b$$

Om denna skrivs på dual form erhålls i stället följande beslutsfunktion. Jämför denna med beslutsfunktionen som beskrevs i perceptronkapitlet:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i \langle \phi_i(\mathbf{x}), \phi(\mathbf{x}) \rangle + b$$

Vi behöver således bara beräkna alla $\langle \phi_i(\mathbf{x}), \phi(\mathbf{x}) \rangle = K(\mathbf{x}_i, \mathbf{x})$. Om skalärprodukten nu byts ut mot en kärnfunktion kan alltså beslutsfunktionen skrivas på en form som innehåller K:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Kärnmetodklassen definierar således implicit klassen av möjliga mönster genom att introducera ett begrepp för likhet mellan data.

Nu till några karaktäristiska egenskaper hos kärnor. Följande egenskaper gäller för kärnor:

Cauchy-Schwarz:

$$K(\mathbf{x}, \mathbf{z})^2 = \langle \phi(\mathbf{x}) \phi(\mathbf{z}) \rangle^2 \leq \|\phi(\mathbf{x})\|^2 \|\phi(\mathbf{z})\|^2 =$$

$$= \langle \phi(\mathbf{x}) \phi(\mathbf{x}) \rangle \langle \phi(\mathbf{z}) \phi(\mathbf{z}) \rangle = K(\mathbf{x}, \mathbf{x}) K(\mathbf{z}, \mathbf{z})$$

Symmetriska:

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \phi(\mathbf{z}) \rangle = \langle \phi(\mathbf{z}) \phi(\mathbf{x}) \rangle = K(\mathbf{z}, \mathbf{x})$$

Det finns också många metoder att skapa nya kärnor utifrån gamla. Här följer några räkneregler som gäller för kärnor. Bevis finns i [7]:

Proposition: *Räkneregler för kärnor* Låt K_1 och K_2 vara kärnor i $X \times X, X \subseteq \mathbb{R}^n, a \in +, f(\cdot)$ en reellvärd funktion på X,

$$\phi : X \to \mathbb{R}^n$$

med K_3, en kärna över $\mathbb{R}^n \times \mathbb{R}^n$, och $\mathbf{B}$ en symmetrisk positivt semidefinit $n \times n$-matris. Då gäller att följande funktioner är kärnor:

(1) $K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z}) + K_2(\mathbf{x}, \mathbf{z})$,
(2) $K(\mathbf{x}, \mathbf{z}) = aK_1(\mathbf{x}, \mathbf{z})$,
(3) $K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z}) K_2(\mathbf{x}, \mathbf{z})$,
(4) $K(\mathbf{x}, \mathbf{z}) = f(\mathbf{x}) f(\mathbf{z})$,
(5) $K(\mathbf{x}, \mathbf{z}) = K_3(\phi(\mathbf{x}), \phi(\mathbf{z}))$,
(6) $K(\mathbf{x}, \mathbf{z}) = \mathbf{x}' \mathbf{B} \mathbf{x}$.

För att kärnkonceptet skall framgå ännu tydligare presenteras nedan några av de mest använda exemplena i tillämpade problem.

Ex: *Generell polynomkärna*

Kärnan är på formen $K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + C)^n$. Konceptet framgår med ett specialfall i $\mathbb{R}^2$ (här $C = 0$ och $n = 2$):

$$\langle \mathbf{x}, \mathbf{y} \rangle^2 = \langle (x_1, x_2), (y_1, y_2) \rangle^2 = x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 y_1 x_2 y_2 =$$

$$= \langle (x_1^2, x_2^2, \sqrt{2} x_1 x_2), (y_1^2, y_2^2, \sqrt{2} y_1 y_2) \rangle$$

Detta är också en skalärprodukt i ett lineärt rum med tre dimensioner, med axlarna x_1^2, x_2^2 och $\sqrt{2} x_1 x_2$. I detta rum är betydligt fler träningsmängder separabla. Kärnan plockar på så sätt ut vissa egenskaper ur data och gör den i någon mening mer separabel. Detta utan att beräkningsordningen på problemet stiger. Det går också mycket enkelt att visa att om bara lämplig kärna väljs är alla träningsmängder separabla (utom specialfallet då: $\mathbf{x}_i = \mathbf{x}_j$ om $y_i \neq y_j, i \neq j$).

Ex: *Radiella kärnfunktioner*
Kärnan är på formen $K(\mathbf{x}, \mathbf{y}) = K(\|\mathbf{x} - \mathbf{y}\|)$. Till skillnad från det klassiska angreppssättet med radiella basfunktioner kommer vid användningen av SVM antalet funktioner vara antalet supportvektorer och lineär-kombinationskoefficienterna vara vikterna i SVM:en, vilket gör att dessa parametrar blir optimala i stället för att de ska bestämmas heuristiskt. Se fig 1 ovan Schack

Ex: *Tvålagers nevralt nätverk*
Kärnan är på formen $K(\mathbf{x}, \mathbf{y}) = tanh(\kappa \mathbf{x}\mathbf{x}' + \theta)$, där κ och θ är parametrar som kallas *styrka* respektive *tröskel*. Beslutsfunktionen är då ett tvålagers neuralt nätverk.

Valet av kärna blir en balansgång mellan goda separationsegenskaper och överinlärning. En ökning av dimensionsantalet ger ofta det senare problemet och tvärtom. Det finns dock metoder inom statistisk inlärningsteori som gör graden av över-inlärningen kontrollerbar om bara kärnfunktionen väljs på rätt sätt. Det går då att till och med välja ∞-dimensionella rum och utnyttja teorin från funktionalanalys för fördelaktiga egenskaper hos Hilbertrum. Det som behövs är ett mått på generaliseringsegenskaperna. Även inom detta område är det Vapnik och medarbetare [28], som har utrett detaljerna. De definierar ett begrepp som de kallar Vapnik-Chovalvski-dimension (VCdim), vilket visar sig vara proportionellt mot överinlärning i en viss mening. Med detta verktyg kan överinlärningsprocessen kontrolleras. Detta är dock mycket tekniskt och jag kommer inte att gå in på detaljerna utan hänvisar till [27]. En förenklad sammanfattning av huvudresultaten följer dock:

Till en lineärt separabel träningsmängd finns det alltid flera olika separerande hyperplan. Det behövs ett principiellt sätt att välja det mest optimala i någon mening. Många fungerande metoder har utvecklats som exempelvis Bayes, MDL, Statistisk inlärningsteori med mera. Det jag menar med en bra metod i detta sammanhang är en metod med goda generaliserande egenskaper, det vill säga en metod, som begränsar risken för överanpassning. Med *VCdim* menas den största delmängd av X, som kan delas av en beslutsregel. I praktiken för de exempel som här behandlas gäller det att $VCdim = dim(X) + 1$. Felmarginalen blir enligt VCteori:

$$\varepsilon = \widetilde{\mathcal{O}}(\frac{VCdim}{N}) = \widetilde{\mathcal{O}}(\frac{(\frac{R}{\gamma})}{N})$$

Här betyder $\widetilde{\mathcal{O}}$ beräkningsordningen, $VCdim$ VC-dimensionen (ovan definierad),N antal träningsexempel och R den empiriska risken (ovan definierad).

Detta innebär att vi minimerar risken för överanpassning genom att välja det maximala marginalhyperplanet i egenskapsrummet. Det vill säga att SVM:en kontrollerar sin kapacitet genom att öka marginalen inte genom att öka antalet frihetsgrader. Ett fördelaktigt val av kärna är således om den marginalen är så stor som möjligt. Två sorters marginaler finns:

(i)

$$func = \min_i y_i f(x_i)$$

(ii)

$$geom = \min_i \frac{y_i f(x_i)}{\|f\|}$$

Om vi fixerar den funktionella marginalen till 1, blir den den geometriska marginalen $\frac{1}{\|\mathbf{w}\|}$. Det är alltså på detta sätt marginalen kan maximeras genom att minimera normen på viktvektorn. Följande minimeringsproblem erhålles:

$$\text{minimera } \|\mathbf{w}\|^2 = \langle \mathbf{w}, \mathbf{w} \rangle$$

$$\text{då } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + \mathbf{b}) \geq 1$$

Inom optimeringsläran brukar ett sådant problem kallas för kvadratiskt prograMeringsproblem (QP). Hur ett sådant löses diskuteras i nästa delkapitel.

Det finns en i lärande-system-kretsar välkända tumregeln kallad dimensionalitetsförbannelsen, som i en tolkning säger att ju högre dimension ett rum har desto större risk finns det för överanpassning. Det har på ovan beskrivna sätt kringgåtts, vilket kan ses som ett motbevis.

2.3. **Optimeringsteori.** Optimeringslära är en gren av den tillämpade matematiken som omfattar användningen av matematiska modeller och metoder för att finna ett bästa handlingalternativ i olika beslutssituationer. Teorin utvecklade sporadiskt hand i hand med den tidiga matematiska analysens. Först under andra värdskriget började optimering ses som ett eget delområde inom matematiken, då under namnet *Research on (military) operations*, OR. I samband med utvecklandet av datorn har den blivit en allt mer nödvändig komponent i optimeringteorins lösningsmetoder och idag ses optimeringsteori snarare som en datavetenskap än matematik, om än med matematisk grundstomme. Följande teori är hämtad främst ur [15] och [7].

Ett optimeringsproblem beskrivs oftast i form av en *målfunktion*, vilken beskriver hur målvariabeln maximeras eller minimeras. Vi begränsar tillåtenheten hos lösningar med så kallade bivillkor, vilka ger restriktioner för hur lösningar får se ut. Man brukar klassificera optimeringsproblem efter formen på målfunktionen tillsammans med bivillkoren. Av orsaker som förklaras nedan kommer det vid träning av en SVM alltid att uppstå ett optimeringsproblem med en kvadratisk målfunktion, emedan samtliga bivillkor kommer att vara lineära. Sådana problem brukar benämnas kvadratiska optimeringsproblem (QP). Kvadratiska problem är relativt vanliga (om inte i närheten så vanliga som lineära problem (LP), där även målfunktionen är lineär) och det finns väl utvecklade algoritmer för att lösa dessa problem effektivt. Optimeringsläran ger oss tillgång till nödvändiga och tillräckliga villkor för att en given funktion skall vara en lösning till sådana problem. Även dualitetsteori kommer att komma till användning.

Ett generellt optimeringsproblem kan skrivas på följande form:

Def: *(Optimeringsproblem, primär form)* Givet funktionerna $f, g_i, i = 1, \ldots, k$ och $h_j, j = 1, \ldots, m$, definierade på området $\Omega \in \mathbb{R}^n$,

$$\text{minimera } f(\mathbf{w}), \mathbf{w} \in \boldsymbol{\Omega},$$

$$\text{då } g_i(\mathbf{w}) \leq 0, i = 1, \ldots, k,$$

$$h_j(\mathbf{w}) = 0, j = 1, \ldots, m,$$

där f kallas *objektfunktion* och g_i, h_j kallas *olikhets-* respektive *likhetsbivillkor.*

Det område där objektsfunktionen definieras kallas tillåtet område och kan betecknas:

$$A = \{\mathbf{w} \in \Omega : \mathbf{g}(\mathbf{w}) \leq \mathbf{0}, \mathbf{h}(\mathbf{w}) = \mathbf{0}\}$$

En optimallösning till ett optimeringsproblem kallas en punkt $\mathbf{w}^* \in \mathbb{R}$ sådan att det inte finns någon annan punkt $\mathbf{w} \in \mathbb{R}$ för vilken det gäller att $f(\mathbf{w}) < f(\mathbf{w}^*)$. En sådan punkt kallas också ett *globalt minimum*. Observera att ett maximeringsproblem alltid kan beskrivas som ett minimeringsproblem genom att negera målfunktionen. En punkt $\mathbf{w}^* \in \Omega$ kallas ett *lokalt minimum* till $f(\mathbf{w})$ om $\exists \varepsilon > 0$ sådant att följande utsaga är sann: $\forall \mathbf{w} \in \Omega, f(\mathbf{w}) \geq f(\mathbf{w}^*)$ och $\|\mathbf{w} - \mathbf{w}^*\| < \varepsilon$.

Man kan då beskriva likhetsvillkor som två lika olikhetsvillkor fast med olika riktningar på olikheterna. Det går också att beskriva olikhetsvillkor med likhetsvillkor genom användningen av så kallade *slappa variabler*:

Def: *slappa variabler* betecknas ξ och transformerar olikhetsbivillkor till likhetsvillkor på följande sätt:

$$g_i(\mathbf{w}) \leq 0 \Leftrightarrow g_i(\mathbf{w}) + \xi_i = 0, \text{ där } \xi_i \leq 0$$

Jämförs detta med begreppet slappa variabler som introducerades i samband med perceptronen inses snart att det är samma begrepp.

Def: En reellvärd funktion f kallas *konvex* för $\mathbf{w} \in \mathbb{R}^n$ om, $\forall \mathbf{w}, \mathbf{u} \in \mathbb{R}^n$ och $\forall \theta \in (0, 1)$, gäller det att

$$f(\theta\mathbf{w} + (1 - \theta)\mathbf{u}) \leq \theta f(\mathbf{w}) + (1 - \theta) f(\mathbf{u})$$

Om ett optimeringsproblem uppfyller att Ω, objektsfunktionen samt att samtliga bivillkor är konvexa sägs optimeringsproblemet vara ett *konvext* optimeringsproblem.

Ett enkelt test för att se om en funktion är konvex är att undersöka om dess Hessian är positivt semi-definit. Konvexitet är en egenskap som ger många fördelar i optimeringsteorin. Det stora fördelen med att arbeta med konvexa problem är att varje lokalt minimum är också globalt, vilket är en mycket fördelaktig situation. QP är alltid konvexa.

Def: *Aktiva bivillkor* Ett bivillkor $g_i(\mathbf{x}) \leq b_i$ sägs vara aktivt för en lösning $\mathbf{x}_0$ om $g_i(\mathbf{x}_0) = b_i$. Annars sägs villkoret vara inaktivt (se 5).

Vad kan vi använda för optimalitetsvillkor? De klassiska Karush-Kuhn-Tuckervillkoren (KKT) ger oss nödvändiga optimalitetskriterier. Dessa karaktäriserar optimallösningen till problemet och hjälper oss att formulera lösningsmetoder för att hitta denna.

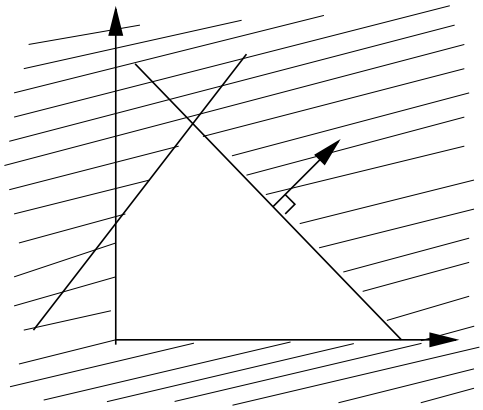
Antag att vi har ett ickelinjärt problem, till exempel ett kvadratiskt problem på formen

$$\text{minimera } f(\mathbf{w}), \mathbf{w} \in \mathbf{\Omega},$$

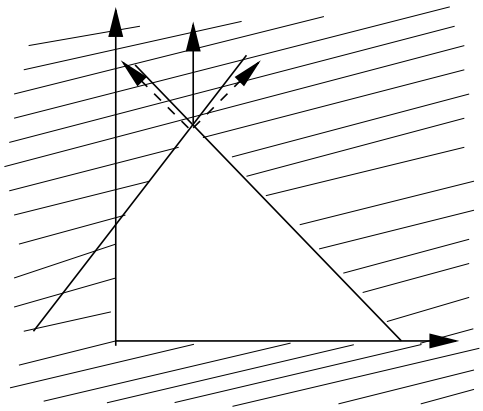
$$\text{då } g_i(\mathbf{w}) \leq 0, i = 1, \ldots, k,$$

Det tillåtna området X är illustrerat som skärningen mellan mängderna som definieras av bivillkoren. Gradienten till alla bivillkor $\bigtriangledown g_i$ pekar alltid ut från detta område. Detta innebär att vi kan använda oss av följande begrepp.

Def: *Kon* En kon definieras av mängden:



FIGUR 5. Aktiva bivillkor. För en lösning som finns vid roten av den utritade normalen i bilden är endast detta bivillkor aktivt. Observera att marginalen alltid är riktad utåt från det tillåtna området, vilket alltid är konvext.



FIGUR 6. I detta fall ligger gradienten till målfunktionen inuti konen som spänns upp av normalerna till de aktiva bivillkoren. KKT-villkoret är således uppfyllt och vi har en optimallösning.

$$\{C = \mathbf{y} | \mathbf{y} = \sum_{i=1}^{s} \alpha_i \mathbf{h}_i, \alpha_i \geq 0, i = 1, \ldots, s\}$$

Om gradienten till målfunktionen (vi förutsätter att $f \in \mathcal{C}^1$) inte ligger i konen av aktiva bivillkor för en viss randpunkt $\mathbf{x}$ kan punkten inte vara en optimallösning (inses geometriskt, se 6).

KKT-villkoren säger då att:

(1)

$$\nabla f(\mathbf{x}) = \sum_{i=1}^{m} v_i \nabla g_i(\mathbf{x})$$

$$v_i \geq 0, i = 1, \ldots, m$$

(2)
$$g_i(\mathbf{x}) \leq b_i, i = 1, \ldots, m$$

(3)
$$v_i(b_i - g_i(\mathbf{x})) = 0, i = 1, \ldots, m$$

(1) kallas dual tillåtenhet och säger just att för att $\mathbf{x}$ ska kunna vara en optimallösning måste gradienten ligga i konen av aktiva bivillkor. Aktiviteten hos bivillkoren kommer in i tredje bivillkoret (3), komplementaritetsvillkoret, som säger att antingen är $\mathbf{x}$ ett villkor aktivt ($g_i = b_i$) eller så är ger bivillkorsgradienten inget bidrag till lineärkombinationen i konen ($v_i = 0$). Detta visar sig vara mycket viktigt då vi på detta sätt kommer att kunna reducera stora separationsproblem till problem som endast använder farliga exempel, s.k. supportvektorer, dvs punkter som ligger närmast beslutsplanet. Se kapitlet om supportvektorer.

Villkor (2) kallas den primala tillåtenheten och är helt enkelt de ursprungliga bivillkoren som givetvis måste vara uppfyllda.

Vi kommer att använda en metod från 1797 utvecklad av Lagrange [1]. Det är egentligen en generalisering av Fermats resultat från 1629 [1]. Ytterligare generalisering av resultatet gjordes av Karush, Kuhn-Tucker 1951 [11].

Def: *Lagrangefunktion* Givet ett optimeringsproblem med en given objektsfunktion $f(\mathbf{w})$ och likhetsbivillkor $h_i(\mathbf{w}) = 0, i = 1, \ldots, m$ definieras *Lagrangefunktionen* till optimeringsproblemet som

$$L(\mathbf{w}, \beta) = f(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w})$$

där koefficienterna $\beta_i \in \mathbb{R}$ kallas *Lagrangemultiplar*.

Sats: *Lagrange* Ett nödvändigt villkor for en normal punkt $\mathbf{w}$ att vara ett minimum till $f(\mathbf{w})$ då $h_i(\mathbf{w}) = 0$, $i = 1, \ldots, m$, med $f, h_i \in \mathcal{C}^1$, är

$$\frac{\partial L(\mathbf{w}^*, \beta^*)}{\partial \mathbf{w}} = 0$$

$$\frac{\partial L(\mathbf{w}^*, \beta^*)}{\partial \beta} = 0$$

för vissa värden β^*. Ovan ställda krav är också tillräckligt förutsatt att $L(\mathbf{w}^*, \beta^*)$ är en konvex funktion på $\mathbf{w}$.

Det primala problemet kan transformeras till ett dualt genom att sätta alla derivatorna med avseende på de primala variablerna till Lagrangianen till 0, och sedan substituera de uppkomna relationerna in i Lagrangianen. På så sätt kan beroendet av samtliga primala variabler elimineras. Detta motsvarar att explicit beräkna Lagrangedualitetskriteriet:

$$\theta(\alpha, \beta) = \inf_{\mathbf{w} \in \Omega} L(\mathbf{w}, \alpha, \beta)$$

Den resulterande funktionen innehåller bara duala ariabler och ska maximeras, vilket ger ett enklare problem än det tidigare. Vi kommer att se explicit hur detta tillämpas på det optimeringsproblem som uppkommer i nästa delkapitel.

2.4. **SVM.** Det är dags att knyta ihop teorierna från de föregående kapitlena, dvs att knyta ihop perceptronen, kärnteorin från funktionalanalysen, optimeringsläran, statistiska inlärningteorin tillsammans med implementationstekniker från dataveten-skapen till en enhetlig teori. Det var precis det Vapik gjorde när han för första gången 1992 fick de olika delarna att samverka [5]. Målet med SVM:en är alltså att effektivt hitta ett bra separerande hyperplan mellan de olika klasserna av punkter i egenskapsrummet. Med bra menas, som tidigare nämnts, ett hyperplan med goda generaliseringsegenskaper och en effektiv algoritm som kräver få beräkningar och som kan behandla även stora datamängder.

Det finns ett antal vägar att gå för att hitta optimala hyperplan. Några sätt är att maximera antalet supportvektorer, marginalen eller marginalfördelningen. Ett annat sätt som är ekvivalent med att maximera marginalen är att minimera normen (2-normen) på viktvektorn $\mathbf{w}$, vilket förklaras som: Funktionen som associeras till hyperplanet $(\mathbf{w}, b)$ ändras inte om vi skalar om parameriseringen till $(\lambda\mathbf{w}, \lambda b), \lambda \in \mathbb{R}^+$, däremot ändras normen på normalen och marginalen. Vi kan således välja skalning från början innan vi påbörjar optimeringen. Som marginal använder vi den geometriska marginalen och vi fixerar marginalen så att:

$$f(\mathbf{x}^+) = \langle \mathbf{w}, \mathbf{x}^+ \rangle + b = +1$$

$$f(\mathbf{x}^-) = \langle \mathbf{w}, \mathbf{x}^- \rangle + b = -1$$

där $\mathbf{x}^+/\mathbf{x}^-$ är en positiv respektive negativ punkt.

Anm: Denna ekvation kan användas för att beräkna viktningen (biasen), b, genom att bara bryta ut b i någon av ekvationerna ovan.

Vi kan nu beräkna marginalen γ genom att betrakta avståndet mellan de två konvexa mängderna som de olika klasserna definierar:

$$f(\mathbf{x}^+) - f(\mathbf{x}^-) = \langle \mathbf{w}, (\mathbf{x}^+ - \mathbf{x}^-) \rangle = 2$$

Om vi fixerar den funktionella marginalen till 1, blir den geometriska marginalen precis halva avståndet mellan dessa klasser, det vill säga $\frac{1}{\|\mathbf{w}\|}$ enligt:

$$\langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}^+ - \mathbf{x}^-) \rangle = \frac{2}{\|\mathbf{w}\|}$$

Marginalen beror alltså uteslutande på normalvektorn och den maximeras om normen av normalvektorn minimeras.

$$\text{maximera}\gamma \Leftrightarrow \text{minimera}\|\mathbf{w}\|_2 \Leftrightarrow \text{minimera}\frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle$$

Vi kan således ställa upp följande sats:

Sats: För en givet lineärt separabel träningsmängd $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\} \subseteq (X \times Y)^l$ är hyperplanet $(\mathbf{w}, b)$ som löser optimeringsproblemet:

$$\text{minimera}_{\mathbf{w},\mathbf{b}} \ \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle, \mathbf{w} \in \Omega,$$

$$\text{då } \gamma_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, i = 1, \ldots, l,$$

det maximala marginalhyperplanet med geometrisk marginal $\gamma = \frac{1}{\|\mathbf{w}\|}$

Om detta problem skrivs om på dual form genom att samma substituering som beskrivs i kapitlet för optimering ovan erhålls följande QP (för detaljer se [7]):

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

$$\alpha_i \geq 0$$

$$\sum_i \alpha_i y_i = 0$$

vilket har följande lösning:

$$\Rightarrow \gamma = \frac{1}{\|\mathbf{w}\|} = (\sum_{i \in sv} \alpha_i^*)^{-\frac{1}{2}}$$

$$\Leftrightarrow \mathbf{w} = \sum_{i \in sv} \alpha_i^*$$

Vi har nu utgående från perceptronen kommit fram till lösningen på ett optimeringsproblem. Denna innehåller en skalärprodukt, som kan bytas ut mot en kärna K. Därmed har samtliga ovan beskrivna teorier har knutits ihop. På detta sätt kan alltså en beslutsregel f beräknas utifrån en träningsmängd S.

2.5. **Implementering av SVM.** Det finns olika varianter för implementering av en SVM. Oftast används följande omskrivning av minimeringsproblemet som nämns i kapitlet ovan:

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

$$\alpha_i \geq 0$$

$$\sum_i \alpha_i y_i = 0$$

Minimeras W uppkommer ännu ett QP, vilket det finns många väl utvecklade algoritmer för att lösa [7]. Den mest använda benämns *Sekvensiell minimaloppti-mering* (SMO), vilken uppdaterar två vikter w simultant i varje steg. Optimeringenssteget sker sedan med en så kallad brantaste lutningen-metod, som i varje steg uppfyller de lineära bivillkoren [23].

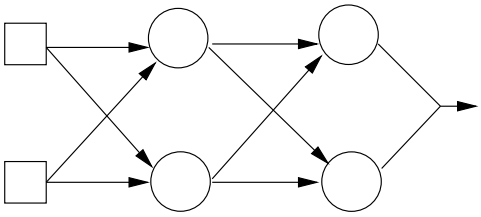
Om fördelningar antas vara överlappande är det vanligt att så kallade mjuka marginaler införs. Här kan ν anta värdena 1 eller 2 vilket medför något olika egenskaper:

$$\text{minimera } < \mathbf{w}, \mathbf{w} > + C \sum_{i=1}^{l} \xi_i^\nu$$

$$\text{då } y_i(< \mathbf{w}, \mathbf{x}_i > + b) \leq 1 - \xi_i$$

$$\xi_i \geq 0,\ i = 1, \ldots, n$$

Även detta är ett QP och kan lösas med SMO-algoritmen.



FIGUR 7. SVM som ett tvålagersnätverk.

2.6. **Jämförelse med andra metoder.** SVM:en skiljer sig från andra metoder dels i prestanda vad gäller klassificering av genexpressionsdata. Det finns också rent teoretiska skillnader i hur SVM:en är uppbyggd i jämförelse med andra system och i synnerhet i jämförelse med neurala nätverk, vilka jag finner intressanta att ta upp.

För genexpressionsdata har en del utvärderingar av SVM i jämförelse med andra metoder gjorts. Den mest utförliga torde vara [34] som skriver att SVM fungerar minst 25 % bättre än det bästa neurala nätverket (Pedersen och Nielsen [22]) i fråga om generaliseringsegenskaper.

I ett tidigare examensarbete för Jesper Tegner har Roland Nilsson, som tidigare nämnts jämfört olika inlärningsmetoder [18]. Här jämförs olika binära klassificering mot varandra. Det visas att SVM:en har bättre generaliseringsegenskaper än andra klassificeringsmetoder, som t-test, PCA, klustringstekniker, självorganiserande träd och variansfilter. SVM:en betedde sig också annorlunda jämfört med andra metoder.

Skillnaden mellan neurala nätverk och supportvektormaskiner är att neurala nätverk håller konfidensintervallet fixt medan den empiriska risken minimeras. Supportvektormaskinen minimerar i stället konfidensintervallet medan den empiriska risken hålls fix [20]. SVM blir härigenom betydligt enklare att kontrollera, även om neurala nätverk ofta ger goda resultat vid tillämpningar. Neurala nätverk fungerar som ett framåtmatat nät av perceptroner där signalerna mellan noderna modifieras, ofta med hjälp av sigmoida funktioner. I SVM:en används endast en perceptron. För att komma åt olineäriteter ändras i stället skalärprodukten för det linjära rummet som träningsdatan definierar. SVM:er och neurala nätverk har stora teoretiska likheter. Det går att skissa upp en SVM som ett tvålagersnätverk enligt figur 7.

I SVM:en uppkommer också ett annorlunda optimeringsproblem än i neurala nätverk. Det har fördelen att vara ett begränsat bivillkorsproblem, vars målfunktion är kvadratiskt. Till sådana problem, vilka benämns kvadratiska programmeringsproblem finns det många väl kända lösningsmetoder, som alla har den stora fördelen att de endast utnyttjar den indata som ligger närmast beslutsregeln. All annan data sållas automatiskt bort. Det gör att algoritmer som bygger på denna teknik är mycket minnes- och beräkningseffektiva. De ovan beskrivna egenskaperna är de mest väsentliga skillnaderna, men det finns även andra, som inte tas upp i detta arbete.

3. MIKROARRAYSYSTEM

3.1. **Microarrayteknik.** Mikroarrayer är en teknik som utför simultana mätningar av tusentals RNA-transkript genom att använda oligonukleotidprober immobiliserade på en glasskiva, ofta med mycket liten storlek (därav namnet mikroarrayer) [18] och [13]. Tekniken detekterar den relativa förekomsten av olika mRNA-molekyler

som motsvarar transkriberade gener. Den stora fördelen med tekniken är att tusentals gener kan detekteras parallellt. Om detta görs för flera olika cellpopulationer, är det möjligt att göra unika jämförelser.

Mikroarrayer är en ordnad mängd små mätpunkter som innehåller 10-20 mg DNA. Dessa mätpunkter motsvaras av tusentals punkter på en liten glasskiva eller ett plastmembran där kända DNA-sekvenser fästs till specifika positioner. DNA-sekvenserna har antingen genererats via PCR av långa DNA-fragment eller syntetiserats av korta oligonukleotider direkt på glaset.

Till respektive mätpunkt kan sedan cDNA hybridiseras med mRNA, som extraherats från cellprover. Detta görs genom att tillsätta komplementära oligodeoxythymidinmolekyler (oligo(dT)) vilka är bundna till en fast yta i form av en kromatografisk kolonn eller en samling magnetiska kulor. RNA är instabilt och bryts snabbt ner. Därför transkriberas de snabbt tillbaka till mer stabilt cDNA med enzymet reverserat transkriptas. Reaktionen startar från poly(A)-svansen och fortlöper längs med hela molekylen.

För detektion märks mikroarray-DNA:et in med fluorescerande färger. Fluoroforer med olika ljusabsorbansspektra används för varje prov. De märkta cDNA-proverna benämns *prober* och används som sonder för den ordnade samlingen av punkter som arrayen är uppbyggd av. Färgerna kan detekteras direkt med ögat då de belyses med laserljus av specifika våglängder. För kvalitativ detektion används emellertid en spektrofotometer som genererar data i form av mättal för varje brunn och för respektive cell.

Den teknik som används i dagsläget brukar klassificeras beroende på längden av den immobilicerade proben. Tillverkningsprocesserna mellan olika mikroarraysystem kan också skilja en del. Den mest använda tekniken hittills har varit *Affymetrix GeneChip* [14], vilka har korta prober (25 nukleotider).

Den data som tekniken genererar utgör ofta ett stort antal mätvärden. Först bearbetas mätvärdena med olika statistiska metoder för att korrigera eventuella metodstörningar. Detta är dock svårt och stora störningar och variationer i kvalitet, är något som måste tas hänsyn till. Därefter kan olika klassificeringsmetoder appliceras som exempelvis SVM:er.

4. Material och metoder

4.1. **Data.**

4.1.1. *Syntetisk data.* Den syntetiska data som har använts består av två multidimensionella normalfördelningar. Väntevärdet i varje dimension är detsamma för båda fördelningarna, med undantag för en dimension, för viken väntevärdet skiljer med ett visst reglerbart avstånd. På så sätt kan den teoretiskt korrekta beslutsregeln exakt bestämmas, nämligen som ett hyperplan med normal som är en ortonormal enhetvektor med nollskild komponent i den väntevärdesskiljande dimensionen. (Se mer under kapitlet *mjukvaruimplementering.*)

Biologiska nätverk har dock inte normalfördelad struktur [18]. En exakt beskrivning av dess struktur är svår att fastställa i termer av vedertagna statistiska fördelningar. Ett alternativ är då att skapa syntetiska nätverk med egenskaper som biologiska nätverk kan förväntas ha. För att kunna utvärdera SVM:en även för stora träningsmängder, vilka det i dagsläget inte finns tillgång till, är syntetiska nätverk ett bra alternativ.

4.1.2. *Biologisk data från mikroarrayer.* Den biologiska datamängd, som jag har arbetat med beskrivs i arbetet [3]. Denna datamängd består av *Colon Adenocarcinoma Specimens* tillsammans med normal colon-vävnad, tagen från samma patient. Data erhölls med Affymetrix Hum6000-arrayer, för 7469 gener i 72 prov.

4.2. **Mjukvaruimplementering.** Samtliga försök implementerades i MatLab 6.1.0.450 (R12.1) från The MathWorks, Inc [16], för Windows 2000. MatLabkod finns tillgänglig mot förfrågan. Jag har använt en SVM-verktygslåda som skapades av Steve Gunn [srg@ecs.soton.ac.uk] .

4.3. **Simuleringar.** De simuleringar som har utförts är:

4.3.1. *Gleshetstest.* Programmet slumpar ut data i ett flerdimensionellt lineärt rum enligt en given fördelning. Två multinormalfördelade datamängder har använts, vilka har samma varians och väntevärde i samtliga dimensioner med undantaget för en dimension, för vilket väntevärdet skiljer sig åt med ett varierbart avstånd. På så sätt kan man förutsäga det teoretiskt korrekta hyperplanet, vilket kommer att ha en normalvektor, som normaliserad är exakt enhetvektorn i den separerande dimensionen.

Den ovan beskrivna delen av programmet körs flera gånger med glesare datamängder från samma fördelning. Som mått på hur bra SVM:en klarar att hitta en bra klassificeringsregel med en glesare fördelning, det vill säga en mindre träningsmängd, plottas den separerande komponenten i normalvektorn till planet för varje iteration i programmet. I idealfallet torde den första normalvektorskomponenten vara 1 enligt ovan beskrivna teori. Ju närmare 1 den är desto bättre klarar SVM:en att hitta den korrekta klassificeringsregeln. Som jämförelse görs även ett lämna-en-utanför-test (LOO, se nästa delkapitel) för att jämföra dessa metoder och för att kunna jämföra mot andras resultat.

Meningen med testet är att undersöka hur korrektheten hos SVM:en påverkas när antalet data är litet, vilket ofta är fallet med mikroarrayanalyser. Detta eftersom mikroarrayteknik ännu är mycket dyrt att genomföra. Ett antal parametrar kan varieras i testet såsom antalet dimensioner på det lineära rummet, avståndet mellan fördelningar, utgångsantalet datapunkter, olika kärnor och olika fördelningar med olika parametrar.

I fallet då klinisk data undersöks kan givetvis inte hyperplanet förutsägas, vilket är grunden för att normalkomponentanalysen skall kunna genomföras. Testet kan dock ge en indikation på hur bra SVM:en klarar utglesning av data. Detta görs genom att applicera en SVM på hela datamängden och anta att hyperplanet, som då genereras är det korrekta. Normalvektorn till detta hyperplan normaliseras, varefter en ny ON-bas skapas med denna som enhetvektor. Detta görs med en Gram-Schmidtprocess. Fördelningen glesas sedan ut genom att slumpvis ta bort träningsexempel. Normalvektorn i den nya basen kan sedan plottas mot antalet träningsexempel. Även om det ursprungliga hyperplanet troligen ej utgör den teoretiskt korrekta, ger testet ändå ett mått på hur förmågan att hitta en korrekt beslutsregel påverkas då träningsmängden minskas. Den sistnämnda delen av programmet behöver dock utvecklas ytterligare för att fungera.

4.3.2. *Marginalberoende.* Programmet fungerar i grunden på samma sätt som Gleshetstest. I stället för att variera träningsmängden varieras konstanten C i optimeringsproblemet:

$$\text{minimera } < \mathbf{w}, \mathbf{w} > + C \sum_{i=1}^{l} \xi_i^{\nu}$$
$$\text{da } y_i(< \mathbf{w}, \mathbf{x}_i > + b) \leq 1 - \xi_i$$
$$\xi_i \geq 0, \ i = 1, \ldots, n$$

Konstanten C reglerar hur mjuk marginal som önskas. Denna bör ej väljas för liten, då de tillåtna värdena på α begränsas för mycket för att kunna erhålla en tillräckligt generaliserande beslutsregel. Men ej heller för stor då lösningen kan påverkas mycket av värden med stora mätfel [7]. Detta är en avvägning som måste behandlas för varje specifikt problem eftersom det beror på vilken typ av indata som behandlas.

Programmet körs för ett antal C-värden, som kan väljas. LOO och normalvektorskomponenten i separerande dimension plottas sedan mot olika C-värden.

4.3.3. *Kärntest.* Två lika stora punktmängder slumpas ut likformigt på de vita respektive svarta rutorna på ett 4 x 4 rutor stort schackbräde. En supportvektormaskin försöker sedan återskapa rutmönstret med olika typer av kärnor och olika antal träningspunkter. Se figur .

4.4. **Lämna-en-utanför.** Lämna-en-utanför är ett klassiskt test för att undersöka generaliserbarheten hos ett lärande system. Idéen är att plocka ut ett exempel i taget, applicera det lärande systemet, för att sedan testa om det borttagna exemplet klassificeras korrekt. Andelen korrektklassificerade av samtliga exempel ges som ett mått på hur bra generaliceringsegenskaper systemet med den givna datan har. De bästa SVM-testerna på mikroarraydata klarar LOO runt 94%, men reslutatet beror givetvis på storleken och kvalitén på inlärningsdatan.

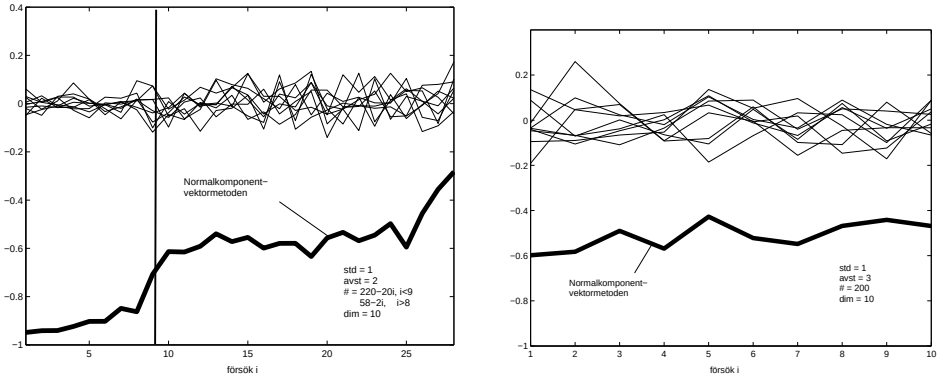
Det finns emellertid framför allt en stor nackdel med testet gentemot normalvektorstestet och det är att SVM:en måste köras en gång för varje träningsexempel, vilket kräver det lång simuleringstid. Denna nackdel har ej normalvektorstestet. Dock är LOO mer generellt än normalvektorstestet då det går att köra utan att känna till den korrekta separeringregeln. Om ingen a priorikunskap finns tillgänglig som i fallet med biologisk data kan beslutsregeln gissas med hjälp av andra test, eller antas vara den beslutsregel som erhålls då samtliga träningsdata används. När antalet träningsdata sedan minskas ger förändringen på första normalkomponenten ändå ett mått på hur kapaciteten försämras, även om jämförelsen är med den teoretiskt korrekta beslutsregeln.

5. Resultat

5.1. **Gleshetstest.** Figurerna 8-15 beskriver hur väl en SVM klassificerar en träningsmängd med 20, 200 eller 400 punkter, vilken efterhand glesas ut. I varje steg minskas antalet träningspunkter. Den feta kurvan visar resultatet från LOO i respetive steg, den halvtunna grafen första komponenten i normalvektorn till beslutsplanet och de tunna graferna övriga komponenter till normalvektorn. Olika parametrar i olika försök har varierats vilket beskrivs under respektive figur.

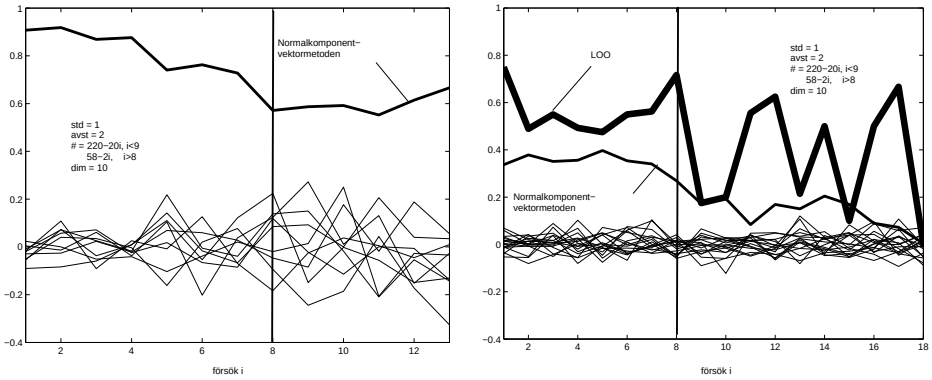
I samtliga försök har lineär kärna, mjuk marginal med lineär störningsterm och $C = \infty$ använts.

Klassificeringen blir sämre ju mindre träningsmängd som används enligt vad som vore rimligt att anta (se figur 8). Var den kritiska gränsen går för vad som



FIGUR 8. Gleshetstest. Mätning med normalvektorkomponentanalys av SVM:ens kapacitet att finna den korrekta klassificeringsregeln då antalet träningsdata glesas ut. Den feta linjen bör vara 1 eller -1 förr teoretiskt korrekt klassificering.

FIGUR 9. Gleshetstest. Samma försök fast med större avstånd mellan väntevärdena på träningsfördelningarna

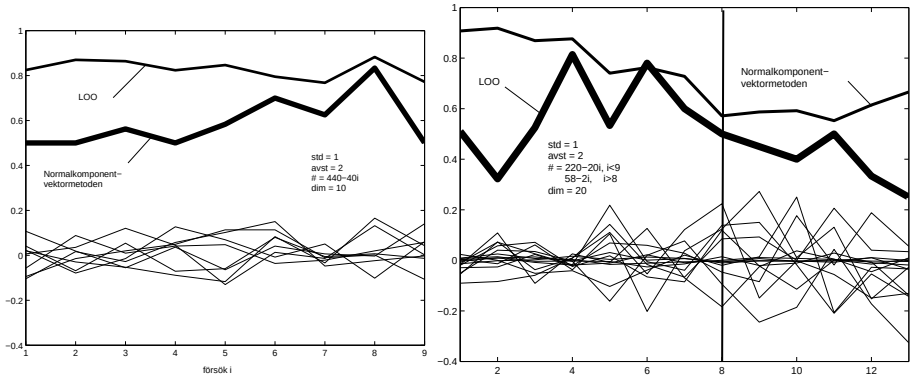


FIGUR 10. Gleshetstest med fokus på 40 träningsdatapunkter och lägre

FIGUR 11. Gleshetstest. Samma test som till vänster, fast i 100 dimensioner och med avstånd 1 mellan väntevärdena på träningsfördelningarna

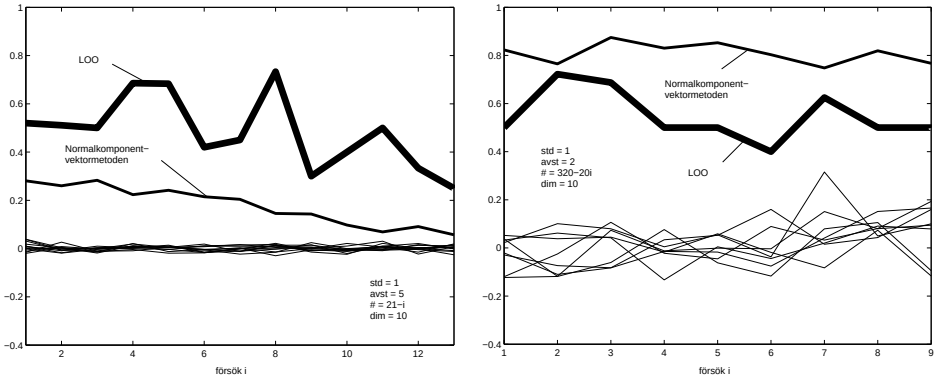
är acceptabelt beror på valet av olika parametrar som exempelvis dimension samt avståndet mellan fördelningarna. En intressant observation är att om dimensionen ökas blir båda testmetoderna betydligt mer brusig än om i stället avståndet mellan fördelningarna skulle ökas. Detta syns framför allt i figur 11.

5.2. **Marginalberoende.** I programmet Marginalberoende testas SVM:s klassificeringsförmåga för olika värden på konstanten C, se figurerna 16-20. Enligt [7] bör C ha ett maximum för ett visst värde på $0 < C < \infty$. Klassificeringsförmågan mäts både med LOO och med den ovan beskrivna normalkomponentmetoden.



FIGUR 12. Gleshetstest. Med minskning av träningsdatapunkter från 400 och neråt.

FIGUR 13. Gleshetstest med 20 dimensioner.

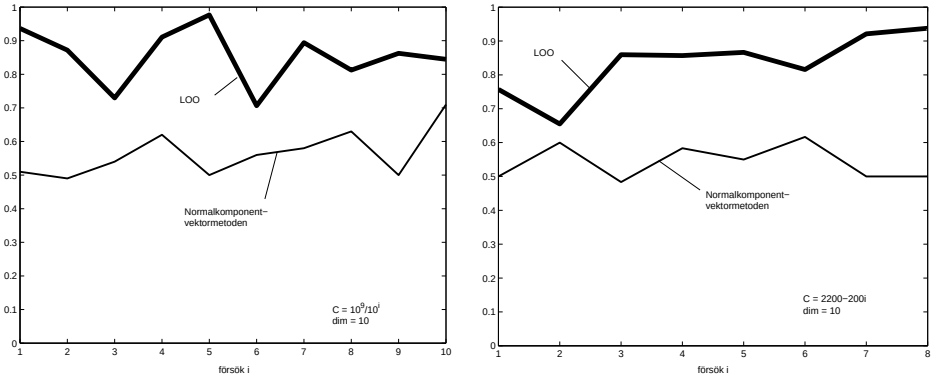


FIGUR 14. Gleshetstest med avstånd 5 mellan väntevärdena på träningsfördelningarna. Fokus ligger på mycket små träningsmängder.

FIGUR 15. Gleshetstest. Med minskning av träningsdatapunkter från 300 och neråt.

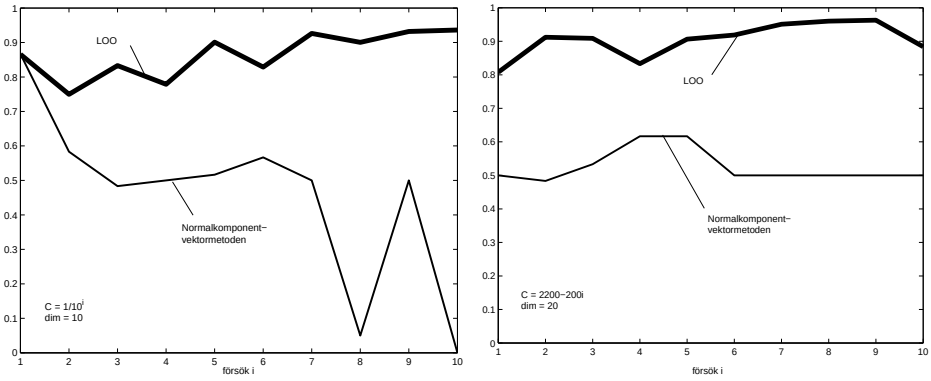
I vissa av figurerna kan trender anas, som för den feta grafen (LOO) i figur 17, men motsägs ofta av andra figurer, 16 eller 18 eller av normalvektorkomponenttestet i figur 17. Ingenting kan därför sägas om C:s inverkan på SVM:ens förmåga. De båda olika testmetoderna verkar inte heller korrelera. Den slumpmässiga faktorn verkar dock vara mycket stor i förhållande till eventuella trender.

5.3. **Kärntest.** Se figurerna 1 och 2 och gå tillbaks till exemplet *Schackbräde* i kapitlet kärnteori. Exemplet går ut på att återskapa ett schackrutemönstret med en SVM. Resultatet var betydligt bättre med radiell basfunktionskärna än polynomkärnor av grad 1,2,3 eller 10. En träningsmängd med 400 punkter klarar klassificeringen bättre, men inte avsevärt bättre än en träningsmängd med 100 punkter.



FIGUR 16. Mjukhetstest av marginalen. Mjukhetsparametern C varieras i de följande figurerna på olika sätt varpå SVM:ens kapacitet testas med LOO respektive normalvektorkomponentanalys. I denna figur gäller att för försök i antas värdet på mjukhetskonstanten $C = \frac{10^{10}}{10^i}$

FIGUR 17. Mjukhetstest av marginalen. För försök i antas värdet på mjukhetskonstanten $C = 2200 - 200i$

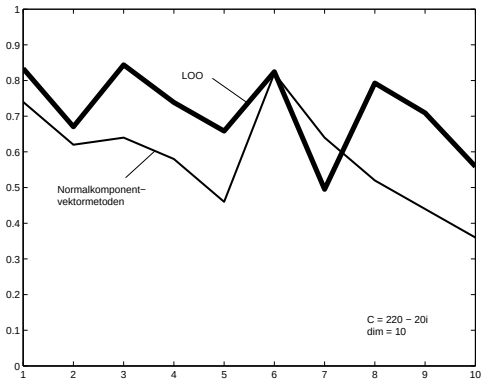


FIGUR 18. Mjukhetstest av marginalen. För försök i antas värdet på mjukhetskonstanten $C = \frac{1}{10^i}$

FIGUR 19. Mjukhetstest av marginalen. För försök i antas värdet på mjukhetskonstanten $C = 2200 - 200i$, dim=20

6. DISKUSSION

Gleshetstest: Olika parametrar varierades i programmet. Om avståndet mellan fördelningarna ökades fick systemet som väntat bättre generaliseringsegenskaper. En betydligt mer intressant observation är att om antalet dimensioner på det lineära rummet ökades hände samma sak, men variationen i resultatkapaciteten blev också betydligt större. Detta beror troligtvis på att variationen i varje dimension fortfarnde är konstant. Vid extremvärden



FIGUR 20. Mjukhetstest av marginalen. För försök i antas värdet på mjukhetskonstanten $C = 220 - 20i$

för träningsmängden i någon av dimensionerna förändras troligvis klassificeringsegenskaperna drastiskt. Att notera är att LOO är betydligt mer känsligt för denna typ av variationer än vad normalvektorkomponentmetoden är (se figur kvantifiera). Detta är en intressant observation då genexpressionsdata ofta är extremt högdimensionell.

Den kritiska fråga, som testet gör ett försök att ge svar på, är hur få datapunkter en SVM kan använda sig av och ändå klara av att hitta en tillräckligt bra klassificeringsregel. Figurerna 8-15 visar att det beror av de olika parametrarna. Eftersom testerna inte ännu har gjorts på biologisk data är det meningslöst att spekulera i ett specifikt antal som skulle vara en nedre acceptabel gräns. Att finna en sådan är dock en naturlig fortsättning på det som har gjorts i detta arbete. Det borde gå ganska snabbt att utföra då program för att testa detta redan finnes.

En nackdel med testet är att det endast utfördes för artificiella mycket tillrättalagda fördelningar. Vissa trender i hur prestanda beroende på vissa parametrar kan urskiljas, men det vore även i det här fallet meningslöst att diskutera specifika parametervärden då dessa ändå är annorlunda för artificiell data. Vi kan emellertid konstatera att optimala parametrar finns, men kan inte beräknas med befintliga metoder.

Marginalberoende: Tidigare resultat visar på ett optimum för en viss mjukhet på marginalen [27]. Ingenting i mina resultat tyder på att detta skulle vara sant. Trots variationer av C över flera tiopotenser kan inga trender skådas i resultaten. Dock är slumpfaktorn (signal-störningskvoten) i mina försök mycket stor, vilket skulle kunna ge en förklaring genom att eventuella trender dränks i stora variationer hos prestandan.

Kärntest: Resultatet är en verifiering av tidigare resultat [7]. Att polynomkärnor av grad 2 eller 3 inte visade lika goda resultat som den radiella basfunktionskärnan är dock förvånande.

Allt tyder på att valet av kärna inte har någon betydelse vad gäller mikroarraydata. Detta beror antagligen på att träningsdatan är väl separerad, vilket i sin tur beror på att antalet dimensioner är betydligt högre än antalet träningsdata.

Generellt kan sägas att det som är speciellt klassificering av genexpressiondata är att datan är högdimensionell och gles. En annat drag är att den är separabel. Detta är med stor sannolikhet delvis en följd av de två förstnämnda dragen. Vi kan se att en klar försämring av klassificerarens kapacitet sker då data glesas ut. Frågan kvarstår dock var acceptansgränsen går för biologisk data.

6.1. **Felkällor.** Några felkällor som kan nämnas är:

- Samtliga artificiella fördelningar som använts under simuleringar är normalfördelade. Genexpressionsdata är inte normalfördelad eftersom de påverkas av de genetiska nätverk [18]. Det har ej gjorts någon utvärdering på hur grov denna förenkling är.
- Testresultaten var ofta mycket brusiga och signal-störningskvoten liten. För-söken skulle kunna förbättras och göras mer pålitliga genom att utvärdera statistik av flera körningar.
- Klassificeringsproblems natur: Ibland går det inte att hitta en perfekt beslutsregel. Om en klassificerare skall skilja mellan en hund och ett bord och man presenterar ett exempel som har egenskaperna brun och fyra ben har man helt enkelt valt fel egenskaper för att separera klasserna. Att veta vilka egenskaper som särskiljer klasserna från varandra är inte alltid lätt att veta. I fallet tumörklassificering vet vi från början mycket lite om de gener vi vill separera med avseende på. Intuitionen säger oss dock att det borde gå att hitta bra beslutsregler här eftersom vi vet att vissa gener har påverkats olika hos de olika klasserna.

6.2. **Slutsatser.** En sammanfattning av de slutsatser som kan dras av de experiment som har utförts är:

- Vid gleshetstest varierades olika parametrar. Följande slutsatser kan dras om de olika parametrarna avstånd, dimension, och antal punkter.
  · Om avståndet mellan fördelningar ökar ökas generaliserbarheten hos systemet.
  · Ökas antalet dimensioner i det lineära rummet ökas generaliserbarheten hos systemet, men till skillnad från om avståndet ökas blir även störningskänsligheten större. Det betyder att det torde vara fördelaktigt att reducera bort opåverkade dimensioner då det enligt ovanstående skulle leda till mindre brus.
- Storleken på C tycks inte spela någon roll. Trots att det i literaturen [7] hävdas att ett optimum av klassifieringskompetens hos systemet bör finnas tyder inga resultat på att ett sådant finns. Det bör nämnas att störningarna på data i dess försök var stora, troligtvis även i jämförelse med eventuella trender.
- Kärnor, lineära verkar vara tillfredsställande [2]. Allt tyder på att valet av kärna inte har någon betydelse vad gäller mikroarraydata. Detta beror antagligen på att träningsdatan är väl separerad, vilket i sin tur beror på att antalet dimensioner är betydligt högre än antalet träningsdata.
- LOO korrelerar med normalvektorsmetoden. Dock inte så väl som min intuition förutsåg. Även här är stora störningar ett stort problem i synnerhet hos LOO.

6.3. **Framtida forskning.** Målet är att finna ett lärande sytem som optimalt klassificerar genexpressionsdata. Tidigare arbeten har visat att SVM:en är ett lärande

system som väl uppfyller det målet. En SVM kan dock varieras mycket i när det gäller parameterinställningar, val av kärnor etc. I detta arbete har vissa parameterar testats främst för artificiella fördelningar. En naturlig fortsättning vore att grundligt undersöka dessa parametrar även för klinisk data. I detalj skulle alltså följande kunna göras:

(1) Finna kriterium för val av mjukhet på marginalen. Det borde gå att finna beroende av olika variabler till strafftermen C. En empirisk formel, vilken direkt skulle kunna användas i syfte att välja ut optimal SVM till en given träningsdatamängd. Helt enkelt en receptlösning för hur SVM:ens parametrar bör ställas in i en given situation.
(2) Jämföra LOO med normalvektorkomponentmetoden. Den sistnämnda metoden har den stora fördelen att SVM:en endast behöver köras en gång för att få ett mått på generaliseringsegenskaperna hos det lärande systemet. Med LOO krävs en körning per träningsexempel, vilket alltså innebär att simuleringstiden blir träningsdatans storlek gånger längre än med normalkomponentmetoden. Min erfarenhet är att simuleringstiden är ett stort problem. Om detta påstående är sant även med framtidens datorer och andra metoder och algoritmer låter jag vara osagt. Det vore hur som helst intressant att grundligt jämföra metoderna. Erfarenheten från mina försök är att metoderna inte korrelerar perfekt. Detta kan bero på stora störningar. I synnerhet är LOO mycket känsligt för störningar.
(3) Implementera normalvektorkomponentmetoden även för klinisk data. Nackdelen med normalvektorkomponentmetoden är att den kräver ett korrekt hyperplan. Detta går att komma runt med ovan beskrivna Gram-Schmidt-metod (se kap mjukvaruimplementation).
(4) Utföra gleshetstest för klinisk data. Utföra marginaltest för klinisk data. Detta kräver mycket långa simuleringstider, men är i teorin enkelt att utföra. Programmen finns redan för mina data.
(5) Då biologisk data från mikroarrayförsök idag är svår att få tag på i stora mängder är ett naturligt angreppssätt att skapa artificiella genetiska nätverk och generera data därifrån. Utifrån denna kan sedan parametrar och kärnor till SVM:en optimeras och utvärderas. Det måste naturligtvis göras klart vilka egenskper det artificiella nätverket måste inneha för att kunna ersätta biologisk data i detta syfte.
(6) Kombinera med andra metoder. Kanske går det att finna ännu bättre inlärningsegenskaper om man kombinerar olika lärande system. Det som talar för detta är att olika system tycks utnyttja olika egenskaper hos datan. Om mer information utnyttjades borde bättre beslutsregler kunna finnas.
(7) Undersöka vilka möjligheter som finns till att lägga in apriorikunskap i en SVM. Går det att utttrycka sådan kunskap som någon typ av bivillkor i det matematiska problemet som uppkommer? Kanske går det att låsa eller väga vissa drag tyngre hos inlärningsdatan redan på förhand?

En förutsättning för att kunna hitta ett optimum av parametrar är naturligtvis att ha tillgång till tillfredställande mått på hur bra en SVM klarar att klassificera en viss datamängd. I detta arbete har föreslagits en alternativ metod till den allmänt vedertagna LOO. En utvärdering är av stor vikt om normalkomponentmetoden skall kunna ha någon praktisk betydelse i framtida forskning.

Slutligen skulle jag också vilja peka på den största fördelen med att utnyttja datatolkning av genexpressionsdata vid tumörklassificering. Det är att en bild direkt kan erhållas av vilka gener som har påverkats vid uppkomsten av tumören. Detta ger ledtrådar till att reda ut de genetiska nätverk som finns i cellen. Om dessa utrönas erhålls stor information om inte bara om uppkomsten av sjukdomen på ett genetiskt plan, utan även en större förståelse om de molekylära livsprocesser som äger rum i cellen, vilket kan leda till ytterligare kunskap samt tekniska och medicinska framsteg.

7. Tack till

Jesper Tegnér: Min handledare för goda diskussioner och bra hjälp genom hela arbetet.

Hanna Härdin och Johan Hattne: Mina opponenter, som hjälpt mig med slutkorrigeringen.

Guiyuan Lei: För stor hjälp med diverse problem genom arbetet. Men också för att ha hållt stämningen uppe på arbetsrummet.

Anders Bresell, Christofer Hallén och Roland Nilsson: Främst för hjälpen med alla datorproblem.

Bengt Hiller och Håkan Abrahamsson: För att i min uppväxt ha skapat mina stora intressen för matematik respektive biologi.

Mathias Henniksson: För att ha hjälpt mig med optimeringsavsnittet.

Referenser

1. R. A. Adams *Calculus of Several Variables* 3 ed., Addison-Wesley, Brittish Columbia, 1996.
2. C.F. Aliferis, I. Tsamardinos, P. Massion, A. Statnikov, D. Hardin, *Why classification models using array gene expression data perform so well: a preliminary investigation of explanatory factors*, International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS), 2003.
3. U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by loigonucleotides arrays*, Proc. Natl. Acad. Sci. USA **96** (1999), nr 12, 6745-6750, juni.
4. C.M. Bishop, *Neural networks for pattern recognition*, Oxford University Press, New York, 1995.
5. B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algoithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, sidorna 144-152. ACM Press, 1992.
6. M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, Jr. and D. Haussler, *Knowledge-based analysis of microarray gene expression data by using support vector machines*, Proceedings of the National Academy of Sciences, **97** (2000), nr 1, 262–267.
7. N. Cristianini, J Shawe-Taylor *Introduction to Support Vector Machines and other kernal-based learning methods*, Cambridge University Press, Cambridge, Storbritanien, 2000.
8. S. Dumais, J. Platt, D. Hecerman, M. Sahami. Inductive learning algorithms and representations for text categorization. I *7th International Conference on Information and Knowledge Managemant,*, 1998.
9. T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield and E.S. Lander, *Molecular classification of cancer: class discovery and class prediction by gene expression monitoring*, Science, **286** (1999), no. 15, 531–536.
10. I. Guyon, N. Matic´, V. Vapnik. Discovering informative patterns and data cleaning. In U.M. Fayad, G. Piatetsky-Shapiro, P. Smythand, R. Uthurusamy, editors, *Advances in Knowladge Discovery and Data Mining*, sidorna 181-203. MIT Press, 1996.

11. h. Kuhn and A. Tucker. Nonlinear programming. I *Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, sidorna 481-492. University of California Press, 1951.
12. B. Lewin, *Genes VII*, Oxford University Press, New York, 2000.
13. B. Lewin *Genomes*, Oxford University Press, 2000.
14. R. J. Lipschutz, S.P.A. Fodor, T.R. Gingeras, D.J. Lockhart. *High density synthetic oligonucleotide arrays*, Nature genetics **21**, (1999), 20-24, Supplement.
15. J. Lundgren, M. Rönnqvist och P. Värbrand, *Linjär och ickelinjär optimering*, Författarna och Studentlitteratur, Lund, 2001.
16. MATLAB. *User's Guide*. The MathWorks, Inc., 1992.
17. J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans- Roy. Soc. London*, A 209:415-446, 1909.
18. R. Nilsson, *Feature selection methods for classification and network discovery from gene expression data*, Examensarbete, Linköping University, 2003.
19. M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio. Pederstrian detection using wavelet templates. I *Proceedings Computoer Vision and Pattern Recognition*, sidorna 193-199, 1997.
20. A. Osareh, M. Mirmehdi, B Thomas, R. Markham, *Comparitive Exudate Classification using Support Vector Mashines and Neural Networks* 2002, http://www.cs.bris.ac.uk/%7Eosareh/index.html (31 Dec. 2003)
21. P. Pavlidis, J. Weston, J. Cai, and W.S. Noble, *Learning gene functional classifications from multiple data types*, Journal of Computational Biology, **9** (2002), no. 2, 401–411.
22. A. G. Pedersen, H. Nielsen, *Neural Network Prediction of Translation Initiation Sites in Eukaryotes: Perspectives for EST and Genome analysis* Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology, 1997, sidorna 226-233.
23. J. C. Platt. Sequentioal minimal optimization: *A fast algorithm for training support vector machines.* Teknisk rapport MSR-TR-98-14, Microsoft Research, 1998.
24. S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.H. yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J.P. Mesirov, T. Poggio, W. Gerald, M. Loda, E.S. Lander and T.R. Golub, *Multiclass cancer diagnosis using tumor gene expression signatures*, Proceedings of the National Academy of Sciences, **98** (2001), nr 26, 15149–15154.
25. F. Rosenblatt, *Principles of neurodynamics*, Spartam Books, (1962).
26. B. Scölkopf, C. Burges, V. Vapnik. *Extracting support data for a given task.* I U. M. Fayyad and R. Uthurusamy, editors, *First International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1995.
27. V.N. Vapnik, *The nature of statistical learning theory*, 2 upplagan, Springer-Verlg, New York, 2000.
28. V. Vapnik, A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities *Theory of Probabilityand its Applications* , 16(2):264:264-280, 1971.
29. K. Veropoulos, N. Cristianini and C. Campbell, *The application of support vector machines to medical decision support: a case study*, ACAI99, Workshop on Support vector machines theory and applications", Crete, Greece, 14:e juli , 1999.
30. K. Veropoulos, C. Campbell and N. Cristianini, *Controlling the sensitivity of support vector machines*, Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI99), Stockholm, Sverige, 1999.
31. J.-P. Vert, *Introduction to support vector machines and applications to computational biology*, DRAFT, 17:e juli, 2001.
32. J.-P. Vert, *Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings*, Proceedings of the Pacific Symposium on Biocomputing, 2002, sidorna 649–660.
33. J.-P. Vert and M. Kanehisa, *Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA*, Advances in Neural Information Processing Systems 15, MIT Press, 2003.
34. A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. lengauer and K.-R. Müller, *Enginnering support vector machine kernels that recognize translation initiation sites*, Bioinformatics, **16** (2000), nr 9, 799-807.

Errata

Sida 2: ”deceniet” ska vara ”decenniet”.

Sida 3: ”benänms” ska vara ”benämns”, ”handskifts-” ska vara ”handskrifts-”.

Sida 5: ”tillämpningas” ska vara ”tillämpas”, ”Alferis” ska vara ”Aliferis”.

Sida 6: ”Kapitel 5 diskuterar” ska vara ”Kapitel 5 och 6 diskuterar”, ”i kapitel 6 tackas” ska vara ”i kapitel 7 tackas”, ”bidriagit” ska vara ”bidragit”.

Sida 7: ”statistika” ska vara ”statistiska”, ”inlärnigsteorin” ska vara ”inlärningsteorin”.

Sida 9: ”ettiketter” ska vara ”etiketter”.

Sida 10: ”opimeringslära” ska vara ”optimeringslära”, ”hyperplantet” ska vara ”hyperplanet”.

Sida 15: ”värdskriget” ska vara ”världskriget”.

Sida 19: ”Vapik” ska vara ”Vapnik”.

Sida 20: ”optimeringenssteget” ska vara ”optimeringssteget”.

Sida 21: ”immobilicerade” ska vara ”immobiliserade”.

Sida 22: ”immobilicerade” ska vara ”immobiliserade”, ”enhetvektor” ska vara ”enhetsvektor”.

Sida 27: ”fortfarnde” ska vara ”fortfarande”.

Sida 29: ”literaturen” ska vara ”litteraturen”, ”sytem” ska vara ”system”.

Referenser:
7. ”Storbritanien” ska vara ”Storbritannien”.
8. ”Knowledge Managemant” ska vara ” Knowledge Management”.
10. ”Knowladge” ska vara ”Knowledge”.
11. ”h. Kuhn” ska vara H. Kuhn”.
19. ”Computoer” ska vara ”Computer”
23. ”Sequentioal” ska vara ”sequential”.
27. ”Springer-Verlg”ska vara Springer-Verlag”.
34. ”lengauer” ska vara ”Lengauer”,”Enginnering” ska vara ”Engineering”.