laSalle

UNIVERSITAT RAMON LLULL

Escola Tècnica Superior d'Enginyeria La Salle

Treball Final de Màster

Màster Universitari en Enginyeria de Xarxes i Telecomunicació

Detecció de malaties segons els trets facials. Segona part: Intel·ligència artificial

Alumne
Guillem Fàbregas Margenats

Professor Ponent
Carles Vilella i Parra

ACTA DE L'EXAMEN DEL TREBALL FI DE CARRERA

Reunit el Tribunal qualificador en el dia de la data, l'alumne

D. Guillem Fàbregas Margenats

va exposar el seu Treball de Fi de Carrera, el qual va tractar sobre el tema següent:

Detecció de malaties segons els trets facials. Segona part: Intel·ligència artificial

Acabada l'exposició i contestades per part de l'alumne les objeccions formulades pels Srs. membres del tribunal, aquest valorà l'esmentat Treball amb la qualificació de

Barcelona,

VOCAL DEL TRIBUNAL VOCAL DEL TRIBUNAL

PRESIDENT DEL TRIBUNAL

ABSTRACT

En el mon actual, cada cop és més habitual l'ús de la tecnologia en processos que fins ara no l'utilitzaven, ja sigui com a substitut de les tècniques anteriors o com a ajut o complement per a aquestes.

El mon de la medicina no n'és una excepció, i les tècniques referents a imatges digitals han estat introduïdes en els darrers anys. Com exemple d'això ens trobem amb les imatges digitals produïdes per les ressonàncies magnètiques, les produïdes pels mètodes de medicina nuclear...

Tot i aquets avenços, encara ens trobem a l'inici de la utilització d'aquestes tècniques en el **diagnòstic assistit per computador**. La idea d'aquests softwares és ajudar als metges a diagnosticar malalties, en funció d' imatges que continguin informació sobre un pacient (ja siguin fotografies, radiografies, telemetries, ...).

Aquest ha estat el punt de partida del projecte, realitzar d'un software per al suport als metges en la diagnosi de malalties, a partir de les fotografies dels seus pacients.
La idea és entrar una fotografia d'un pacient al programa informàtic que s'ha realitzat, i que aquest indiqui a l'usuari certes malalties que pot patir.

No es pretén crear un mètode infal·lible per al diagnosis de malalties, sinó proporcionar als metges una eina per afitar el nombre de malalties a estudiar, de manera que el metge pugui realitzar posteriorment les probes necessàries i donar una diagnosis definitiva.

Aquest projecte s'ha dividit en dues parts, que són "**Detecció de malalties segons els trets facials - Primera part: Tractament d'imatge**" i "**Detecció de malalties segons els trets facials - Segona part: Intel·ligència Artificial**".

Aquesta és doncs la segona d'aquestes dues parts, i es centra en la part de tractament d'intel·ligència artificial que s'ha portat a terme en el projecte.

RESUM

La idea principal d'aquest projecte és la de crear un software per ajudar als metges en la predicció de certes malalties a partir de la fotografia d'un pacient. Aquest està realitzat per un encàrrec de l'hospital **Sant Joan de Déu**, i el seu objectiu és el de crear un prototip o esquelet d'aquest software.

En aquest programa, els metges podran introduir una fotografia d'un pacient, i aquest els indicarà les malalties que pot patir, afitant així les possibilitats existents. Amb aquesta informació el metge podrà posteriorment realitzar les proves que consideri necessàries al pacient, i realitzar una diagnosi final d'aquest.

En la realització d'aquest software ens trobem amb dues parts molt importants Per una banda tenim la part de tractament d'imatge, en la qual s'extreu la informació necessària de les fotografies estudiades. I per l'altre tenim la part d'intel·ligència artificial, en la qual es classifiquen i es prediuen aquestes dades.

L'explicació o memòria completa d'aquest software s'ha dividit en dues parts, que són: "**Detecció de malalties segons els trets facials - Primera part: Tractament d'imatge**" i "**Detecció de malalties segons els trets facials - Segona part: Intel·ligència Artificial**". Aquesta és doncs la segona part, en la qual es veuen els fonaments de la part de d'intel·ligència artificial del projecte.

En aquesta segona part d'intel·ligència artificial, inicialment es parla dels **objectius** que es desitgen assolir, així com el **plantejament** necessari que s'ha dut a terme per aconseguir-ho. Tot seguit, es veuen els fonaments teòrics de la part d'**intel·ligència artificial**, on tenim els conceptes del***Data Mining***, i algunes de les tècniques de classificació de dades més utilitzades, com són els **arbres de decisió** i el **SVM** (Màquines de Suport Vectorial) [1].

A continuació s'analitza el conjunt de llibreries d'intel·ligència artificial en JAVA **Weka**. En aquest apartat veiem una idea general de la seva utilitat, així com els fitxers que utilitza per a la classificació de dades.

Tot seguit es parla de les **funcions pròpies** utilitzades en el software, així com del **funcionament** d'aquest, tant a nivell individual d'aquesta part del projecte com a nivell general, on es veu el funcionament del programa complet.

Per acabar, es veuen els **resultats** que s'han extret de la realització del projecte, així com les **línies de futur** aconsellades per a aquest.

Índex

1 INTRODUCCIÓ

En els darrers anys, el desenvolupament de nou software i hardware en l'àmbit de l'anàlisi digital d'imatges ha experimentat un gran impuls, ja sigui en aplicacions de reconeixement facial, ocular, de seguretat, o en l'àmbit de la medicina, el que es tracta en aquest projecte.

Així doncs, un creixent nombre de tècniques referents a imatges digitals ha estat introduït en la pràctica mèdica en els últims anys. Molts radiòlegs i personal de laboratoris mèdics coneixen i manipulen imatges digitals com les produïdes per Tomografia Assistida de Computadora (TAC), ressonància magnètica o per mètodes de medicina nuclear. Degut al desenvolupament, capacitats i gran factibilitat d'execució dels programes informàtics, les imatges en medicina, que eren tradicionalment gravades sobre pel·lícula, ara es poden manipular de forma digital. D'aquesta manera, per exemple les imatges d'ultrasons i les de raigs X es poden emmagatzemar digitalment, reduint així considerablement els costos, i facilitant-ne tant l'accés com la classificació.

Tot i la clara implantació de les tècniques de tractament d'imatges en medicina, encara ens trobem a l'inici de la utilització d'aquests mètodes en sistemes de **diagnòstic assistit per ordinador**. És a dir, softwares que ens ajudin a diagnosticar malalties en els pacients.

Aquest ha estat precisament el punt de partida del projecte, realitzar un software per a **l'hospital Sant Joan de Déu**, que ens ajudi a diagnosticar malalties,partint de la fotografia frontal del pacient a tractar.

D'altre banda i dins de l'àmbit del tractament digital d'imatges, les tècniques d'extracció de les característiques més importants d'aquestes també han rebut un fort impuls en els darrers anys.

El que pretenen aquests mètodes és cercar els punts més importants o que ens aporten més informació en una fotografia, i un cop localitzats, extreure les seves característiques més destacades.

Existeixen varis mètodes de detecció i càlcul d'aquests punts molt efectius, com són el **SIFT (*Scale-Invariant Feature Transform)*[48],** el **SURF (*Speed Up Robust Features*) [49],** el **HOG(*Histogram of Oriented Gradient*) [50],** el **Harris Corner Detection**,... sobre els quals es segueix treballant actualment per aconseguir més i millors resultats.

Un dels punts claus d'aquest projecte ha estat doncs el tractament i la manipulació d'imatges, així com la detecció i extracció de les característiques més importants d'aquestes.

Un altre dels punts clau del projecte ha estat la part de ***Data Mining*** (**DM**), que consisteix en l'extracció d'informació no trivial resident de manera implícita en les dades. Les bases del DM es troben en la **Intel·ligència Artificial (IA)** i en l'**anàlisi estadístic**, i el seu procés consisteix en preparar i classificar tota la informació que es pugui extreure d'aquestes dades, creant així models d'entrenament que posteriorment seran utilitzats en problemes de predicció i segmentació.

Així doncs, el que s'ha desenvolupat en aquest projecte ha estat un software de predicció de malalties, que s'ha segmentat en les dues parts següents: "**Detecció de malalties segons els trets facials - Primera part Tractament d'imatge**" i "**Detecció de malalties segons els trets facials - Segona part Intel·ligència Artificial**".

Aquesta és la segona d'aquestes dues parts, i per tant parla sobre tots els processos d'intel·ligència artificial que s'han dut a terme en aquest treball.

2 OBJECTIUS

En aquest projecte s'ha realitzat un software de predicció de malalties en pacients, a partir de les fotos dels rostres d'aquests. Així doncs, s'ha iniciat la recerca en un sistema de **diagnòstic assistit per ordinador,** dels quals tenim molt poca informació avui en dia.

La idea principal és realitzar un software que ajudi al metge a detectar algunes malalties de les anomenades "rares", o de les quals el metge tingui molt poca experiència. Ara bé, cal dir que l'objectiu no és crear un detector infal·lible de malalties per estalviar-nos el paper del metge en la diagnosi, sinó crear un programa d'ajuda per a aquest. Així doncs, l'ideal seria entrar la foto d'un pacient al programa, i que aquests indiqués les malalties que aquest pot tenir (aquelles que té més possibilitats de patir). D'aquesta manera aconseguirem afitar el nombre de malalties a estudiar, de manera que serà molt més senzill per al metge aplicar les probes corresponents a les malalties en qüestió i realitzar un diagnosi final del pacient.

Ara bé, l'objectiu d'aquest projecte no és el de crear un software de diagnosis de malalties concretes,sinó crear la base tecnològica de la solució del problema. Així doncs, el que s'ha realitzat és un software d'extracció i predicció de característiques facials dels pacients, que posteriorment es podrà utilitzar per a fer proves amb les malalties concretes que es desitgi estudiar.

A més, s'ha de tenir en compte que molts metges no tenen un coneixement avançat sobre informàtica i tecnologia, i per tant, cal que el software creat sigui senzill i fàcil d'utilitzar. Per a fer-ho possible, s'ha creat una interfície gràfica per a Windows, molt intuïtiva, amb la qual qualsevol usuari amb coneixement bàsics sobre informàtica la pot fer servir sense problemes. A més, s'ha realitzat un manual d'usuari d'aquest software que acompanya el projecte.

Per a la realitzacióde l projecte s'ha treballat sobre dos processos bàsics, el **tractament digital de imatges**, per localitzar i extreure les característiques més destacades dels rostres dels pacients, i el ***Data Mining*** en la classificació i predicció d'aquestes característiques. Aquests projecte ha estat dividit en dues parts, cada una d'aquestes tractant un d'aquests dos processos bàsics, que es descriuen a continuació.

El primer d'aquests projectes tracta sobre l'anàlisi digital de fotografies en la detecció de les zones d'interès en la cara del pacient, i la posterior extracció de les característiques més destacades d'aquestes. L'objectiu principal d'aquesta part serà aconseguir un vector numèric que contingui les característiques més importants dels rostres dels pacients.

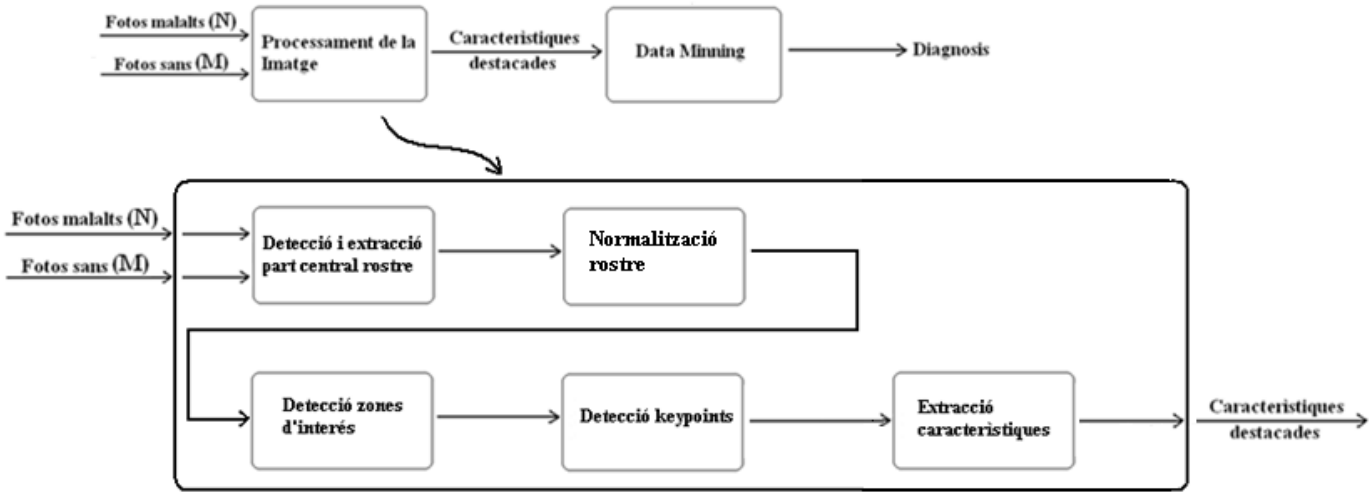
L'altre projecte (corresponent a aquesta memòria)estudia la mineria de dades, per aplicar sobre les característiques més importants dels rostres dels pacients, i per crear models de predicció basats en aquestes característiques. Així doncs, un cop obtinguts aquets models, es podran predir les malalties dels nous pacients a tractar. Aquest serà doncs l'objectiu principal d'aquesta part, classificar correctament els vectors numèrics descriptius de cada pacient, creant un model d'entrenament per a la predicció de nous cassos.

3 PLANTEJAMENT DEL PROBLEMA

En aquest apartat es veu el plantejament del problema que se'ns presenta. Com s'ha vist prèviament, el problema que se'ns proposa és el de facilitar el diagnosis de certes malalties en un pacients a partir de les fotos dels seus rostres. Per a fer-ho, tenim dos processos principals de treball, un d'**aprenentatge**(procés d'entrenament)i un de **diagnosi** (procés de testeig).

3.1 Aprenentatge

Veiem a continuació el diagrama funcional del procés d'**aprenentatge** a la figura [3.1], així com el seu bloc de **Processament de la Imatge**, descrit amb detall.



Il·lustració 3.1: Diagrama de blocs del procés d'aprenentatge.

Així doncs, veiem que en aquest procés s'entren N imatges de gent que pateixuna de les malalties a estudiar, i M imatges de gent que no la pateix. El que es farà és extreure les característiques més importants d'aquestes imatges, i crear un model de classificació d'aquestes dades. Tot seguit expliquem aquest procés amb més detall.

Com veiem en el diagrama de blocs de la part de **Processat de la Imatge** (procés explicat amb detall i amb exemples a l'apartat 7 d'aquesta memòria), el que primer que es realitza en una fotografia d'un pacient és una detecció i extracció de la part central del seu rostre. Un cop tenim aquesta zona per separat, el que fem és aplicar una normalització, ja que ens interessa que a partir d'aquest punt sempre treballem amb rostres que tinguin les mateixes dimensions i el mateix nombre de píxels. A partir d'ara només treballarem amb aquestes imatges normalitzades en tots els processos posteriors.

Arribats a aquest punt, el que fem és localitzar les 10 zones més destacades d'aquests rostres, es a dir, les zones que considerem que ens proporcionaran més informació sobre aquests. Aquestes 8 zones correspondran als llocs següents:

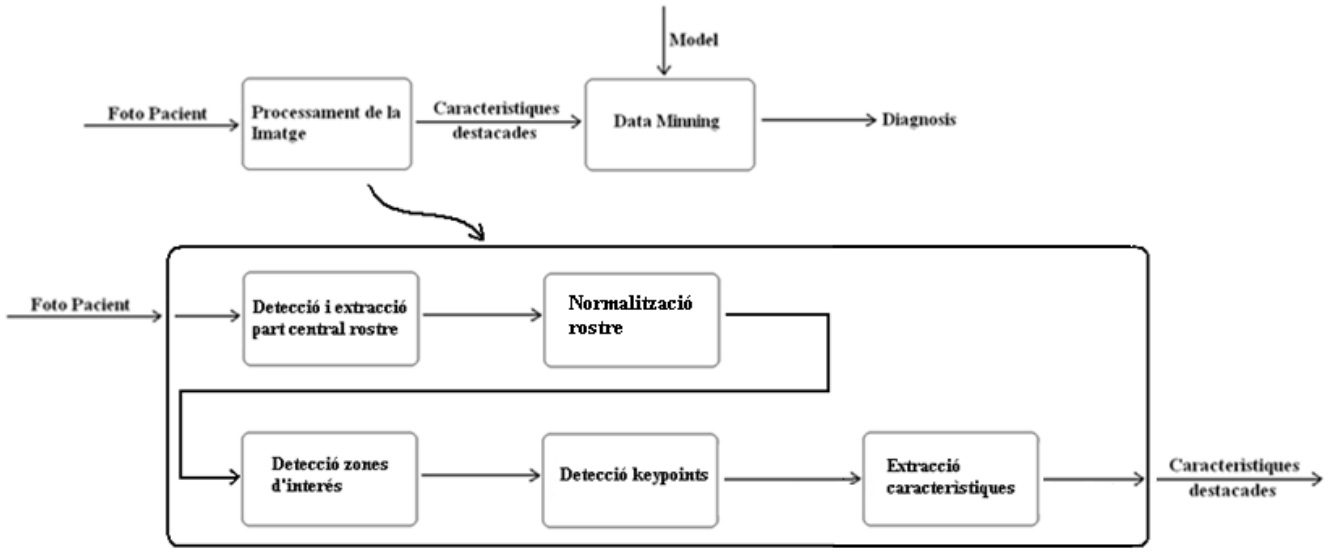
- Part esquerre de l'ull dret
- Part dreta de l'ull dret
- Part esquerre de l'ull esquerre
- Part dreta de l'ull esquerre
- Part esquerre boca
- Part dreta boca
- Part dreta nas
- Part esquerre nas
- Galta dreta
- Galta esquerre

Un cop disposem de cadascuna d'aquestes zones, el que es fa és extreure la informació més rellevant de cadascuna d'elles, en forma de vectors numèrics. Per a fer-ho disposem de vàries tècniques, com veurem a l'apartat 4.

Un cop es disposa d'aquestes dades descriptives de cada rostre, el que fa és classificar-les en el bloc de ***Data Mining***, obtenint així un **model** d'intel·ligència artificial, que ens servirà com a base per realitzar prediccions en el procés de diagnosi.

3.2 Diagnosi

Veiem a continuació el diagrama funcional del procés de **diagnosi** a la figura [3.2], així com el seu bloc de **Processament de la Imatge** explicat amb detall.



Il·lustració 3.2: Diagrama de blocs del procés de diagnosi.

En aquest cas, les dades d'entrada correspondran a una foto del pacient al qual es desitja diagnosticar alguna de les malalties estudiades, i el model d'aprenentatge creat anteriorment. El primer que es realitza és extreure les característiques més importants d'aquesta fotografia en el bloc

de **Processament de la Imatge**, seguint el mateix recorregut que s'ha vist anteriorment.

Un cop disposem de les dades més descriptives del rostre del pacient en format de vector numèric, el que fem en el bloc de ***Data Mining*** és utilitzar el model de classificació del que disposem amb aquestes dades, de manera que podem predir si el pacient dona o no símptomes de patir una de les malalties estudiades.

3.3 Conclusions

Observem que en els dos processos anteriors (aprenentatge i diagnosi) ens apareixen dos blocs principals, per una part el **Processat de la Imatge** i per l'altre la **Mineria de Dades**. Aquests dos blocs són similars en ambdós cassos, ja que s'utilitzaran les mateixes tècniques per extreure les característiques més importants de les imatges, i tècniques similars per a classificar-les. El treball o tasca realitzada en cadascun d'aquests dos grans blocs s'ha dividit en dos projectes, com s'ha comentat anteriorment.

En aquest projecte es veu la part d'intel·ligència artificial, la qual està centrada en dos punts importants. Per una banda tenimla creacióde l model d'entrenament a partir dels (N+M) vectors de característiques de les fotos de la gent sana i la malalta. I per l'altre, un cop ha estat creat el model d'entrenament esmentat, el que es fa és aplicar-lo, per predir si les característiques descriptives extretes de la foto d'un nou pacient donen o no símptomes de patir alguna de les malalties en qüestió. Com veurem posteriorment (apartat 4), existeixen diversos mètodes d'intel·ligència artificial per a classificar i predir les dades en qüestió.

Per tant, l'objectiu d'aquesta part del projecte serà crear un model d'aprenentatge d'intel·ligència artificial el màxim de sòlid possible, a partir dels N+M vectors dels que disposem. Aquest model ens permetrà realitzar prediccionspartint de un vector de dades d'un nou pacient.

4 INTEL·LIGÈNCIA ARTIFICIAL (FONAMENTS TEÒRICS)

4.1 DATA MINING

El *data mining* reuneix els avantatges de vàries àrees com la Estadística, la Intel·ligència Artificial, la Computació Gràfica, les Bases de Dades i el Processament Massiu, principalment utilitzant com a única matèria les bases de dades. Una definició tradicional és la següent: *El data mining és un procés no trivial d'identificació vàlida, innovadora, potencialment útil i entenedora de patrons comprensibles que es troben ocults en les dades* [51]. Des del punt de vista empresarial, i de la manera en que s'ha utilitzat en aquest projecte, podem definir-ho com: *La integració d'un conjunt d'àrees que tenen com a propòsit la identificació d'un coneixement, obtingut a partir de les bases de dades que aporten una experiència cap a la pressa de decisió*[52]. Així doncs, el que aconseguirem amb la mineria de dades és un model basat en una base de dades, que ens aportarà coneixements a l'hora de realitzar una nova predicció basada en aquestes dades.

La idea de la mineria de dades no és nova. Ja des dels anys setanta els estadístics utilitzaven termes com el *data fishing, data mining* o *data archaeology* amb la idea de trobar correlacions sense una hipòtesis prèvia en bases de dades amb soroll. A principis dels anys vuitanta, Rakesh Agrawal, Gio Wiederhold, Robert Blum i Gregory Piatetsky-Shapiro, entre d'altres, vam començar a consolidar els termes de *data mining* i coneixement de base de dades (KKD) [41]. A finals dels anys vuitanta només existien un parell d'empreses dedicades a aquesta tecnologia, actualment existeixen centenars d'empreses que s'hi dediquen.

La mineria de dades és una tecnologia composta per diverses etapes que integra vàries àrees i que no s'ha de confondre amb un gran software. Durant el desenvolupament d'un projecte d'aquest tipus s'utilitzen diverses aplicacions software en cada etapa que poden ser estadístiques, de visualització de dades o d'intel·ligència artificial, principalment. Actualment existeixen eines o aplicacions comercials de mineria de dades molt poderoses que ens faciliten el desenvolupament d'un projecte. Una d'aquestes eines és el software utilitzat en aquest projecte i del qual es parla extensament a l'apartat 5 d'aquesta memòria (Weka).

4.1.1 Fonaments de *Data Mining*

Les tècniques de mineria de dades són el resultat d'un llarg procés d'investigació i desenvolupament de productes. Aquesta evolució va començar quan les dades de negocis van ser emmagatzemades per primera vegada en ordinadors, i va continuar amb millores en l'accés de les dades, i més recentment amb tecnologies generades per permetre als usuaris navegar a traves d'aquestes en temps real.

La mineria de dades està suportada per tres tecnologies que ja són suficientment madures:

- Recol·lecció massiva de dades.
- Potents ordinadors amb multiprocessadors.
- Algorismes de *Data Mining*.

4.1.1.1 Principals característiques i objectius de la mineria de dades

- Explorar les dades que es troben en les profunditats de les bases de dades, algunes de les quals contenen informació emmagatzemada durant varis anys.

- En alguns casos, les dades es consoliden en un magatzem de dades i en mercats de dades. En altres, es mantenen en servidors de Internet i Intranet.

- L'entorn de la mineria de dades sol tenir una arquitectura client-servidor.

- Les eines de mineria de dades ajuden a extreure la part important de la informació emmagatzemada en registres.

- Les eines de mineria de dades es combinen fàcilment i poden analitzar-se i processar-se ràpidament.

- Degut a la gran quantitat de dades, algunes vegades resulta necessari utilitzar processament en paral·lel per la mineria de dades.

- La mineria de dades produeix cinc tipus d'informació:

    - Associacions.
    - Seqüencies.
    - Classificacions.
    - Agrupament.
    - Pronòstics.

La mineria de dades és un procés que inverteix la dinàmica del mètode científic en el següent sentit:

En el mètode científic, primer es formula la hipòtesis i després es dissenya l'experiment per col·leccionar les dades que confirmen o rebutgen la hipòtesis. Si això es realitza amb la formalitat adequada, s'obté un nou coneixement.

En la mineria de dades en canvi, es col·leccionen les dades i s'espera que de elles en surti una hipòtesis. És a dir, es busca que les dades indiquin per que són de la manera que són. Llavors, si es vàlida aquesta hipòtesis inspirada per les dades en les dades mateixes, els resultats seran numèricament significatius, però experimentalment invàlids. D'aquí s'extreu que la mineria de dades s'ha d'utilitzar per a obtenir hipòtesis sobre noves dades, i no sobre les mateixes dades utilitzades en la creació del model. Si ho féssim, els resultats podrien ser invàlids o falsos.

4.1.1.2 Abast de la mineria de dades

El nom de *Data Mining* deriva de les similituds d'entre buscar informació en grans bases de dades, (com per exemple utilitzem en aquest projecte, on les bases de dades corresponen a les característiques més destacades dels rostres dels pacients), i minar una muntanya per trobar una

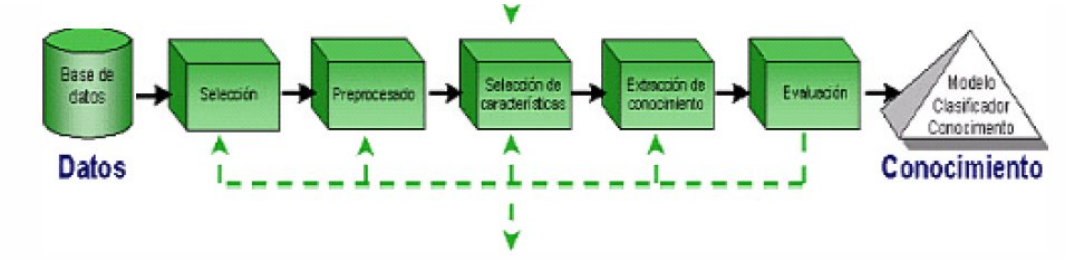
font de metalls preciosos. Ambdós processos requereixen examinar na immensa quantitat de material, o investigar intel·ligentment fins a trobar exactament on resideixen els valors. Donades bases de dades de suficient mida i qualitat, aquesta tecnologia pot proveir de les següents capacitats:

- Predicció automatitzada de tendències o comportament. S'automatitza el procés de trobar informació predictible en grans bases de dades. Preguntes que normalment requeririen un intens anàlisis manual, ara poden ser contestades directa i ràpidament des de les dades.
- Descobriment automatitzat de models prèviament desconeguts. Les eines de mineria de dades ataquen les bases de dades i identifiquen models prèviament amagats en un sol pas.

Les tècniques utilitzades poden ser implementades en nous sistemes a mesura que les plataformes de hardware i software existents s'actualitzin i nous productes siguin desenvolupats. Quan les eines de *Data Mining* són implementades en sistemes de processament paral·lel de alt rendiment, poden analitzar bases de dades massives en minuts. Processament més ràpid significa que els usuaris poden automàticament experimentar amb més models per entendre dades complexes. L'alta velocitat fa que sigui més pràctic pels usuaris analitzar immenses quantitats de dades. Si disposem de grans bases de dades, ens produiran millors prediccions.

4.1.2 Fases d'un projecte de *data mining*

Els passos a seguir per la realització d'un projecte de mineria de dades són sempre els mateixos, independentment de la tècnica específica d'extracció de coneixement utilitzada. En la figura [4.1] s'observen les Fases de un projecte de DM.



Il·lustració4.1: Fases del projecte de mineria de dades. Figura extreta de [40]

El procés de mineria de dades passa per les següents fases, que s'expliquen a continuació:

- Filtrat de dades.
- Selecció de Variables.
- Extracció de Coneixement.
- Interpretació i Avaluació.

4.1.2.1 Filtrat de dades

Normalment, el format de les dades contingudes en una font de dades (base de dades, Warehouse, ...) no és l'idoni per a la creació del model desitjat, i la majoria de cops no es possible ni tan sols utilitzar cap algorisme de mineria de dades sobre les dades originals.

Mitjançant el pre-processat, es filtren les dades (de manera que s'eliminen valors incorrectes, no vàlids, desconeguts,... segons les necessitats del sistema), s'obtenen mostres d'aquestes, o es redueixen el nombre de valors possibles (mitjançant arrodoniment, *clustering*, ...). Així obtenim un nombre de mostres útils per a crear el model d'entrenament desitjat.

4.1.2.2 Selecció de variables

Fins i tot després d'haver estat pre-processats, en la majoria de casos es disposa d'una quantitat alta de dades. La selecció de característiques redueix la mida de les dades escollint les variables més influents en el problema, casi sense sacrificar la qualitat del model de coneixement obtingut en el procés de mineria.

Els mètodes per la selecció de característiques són bàsicament dos:

- Mètodes basats en la elecció dels millors atributs del problema.

- Mètodes que busquen variables independents mitjançant texts de sensibilitat, algorismes de distància o heurístics.

4.1.2.3 Algorismes d'extracció de coneixements

Mitjançant una tècnica de mineria de dades, s'obté un model de coneixement, que representa els patrons de comportament observats en els valors de les variables del problema, o les relacions d'associació entre aquestes variables. També poden utilitzar-se vàries tècniques a la vegada per generar diferents models, encara que generalment cada tècnica obliga a un pre-processat diferent en les dades inicials.

4.1.2.4 Interpretació i avaluació

Un cop obtingut el model, s'ha de procedir a la seva validació, comprovant que les conclusions que després són vàlides i suficientment satisfactòries. En el cas d'haver obtingut varis models mitjançant l'ús de diferents tècniques, s'han de comprar els models per buscar aquell que s'ajusti millor al problema. Si amb cap dels models s'obté el resultat esperat, s'ha d'alterar alguns dels passos anteriors per generar nous models, intentant així de trobar-ne un que s'ajusti a les nostres pretensions.

4.1.3 Validació creuada

La validació creuada és una de les maneres més populars d'avaluar l'efectivitat dels models d'intel·ligència artificial creats.

Aquesta tècnica implica dividir la mostra en una sèrie de mostres més petites. A continuació, es generen els mètodes de IA, que no inclouen les dades de cada submostra. Per exemple, amb una validació creuada de deu vegades, les dades es divideixen en 10 submostres i després es generen 10 models de IA. El primer dels models es basa en tots els casos excepte els corresponents a la primera submostra. El segon es basa en tots els casos excepte els corresponents a la segona submostra, i així successivament. Per a cada model es calcula el risc de classificació errònia aplicant-lo a la submostra que es va utilitzar per a crear-lo. La estimació del risc mitjançant la validació creuada per a tot el model es calcula com el promig dels riscos de tots els arbres.

4.1.3.1 Llavor aleatòria

Quan s'utilitza la validació creuada, els casos o dades d'entrenament s'assignen de forma aleatòria a particions o nombres de submostres. La configuració de la llavor permet especificar el valor inicial que utilitza el generador del nombre aleatori per assignar els casos.

4.2 ARBRES DE DECISIÓ

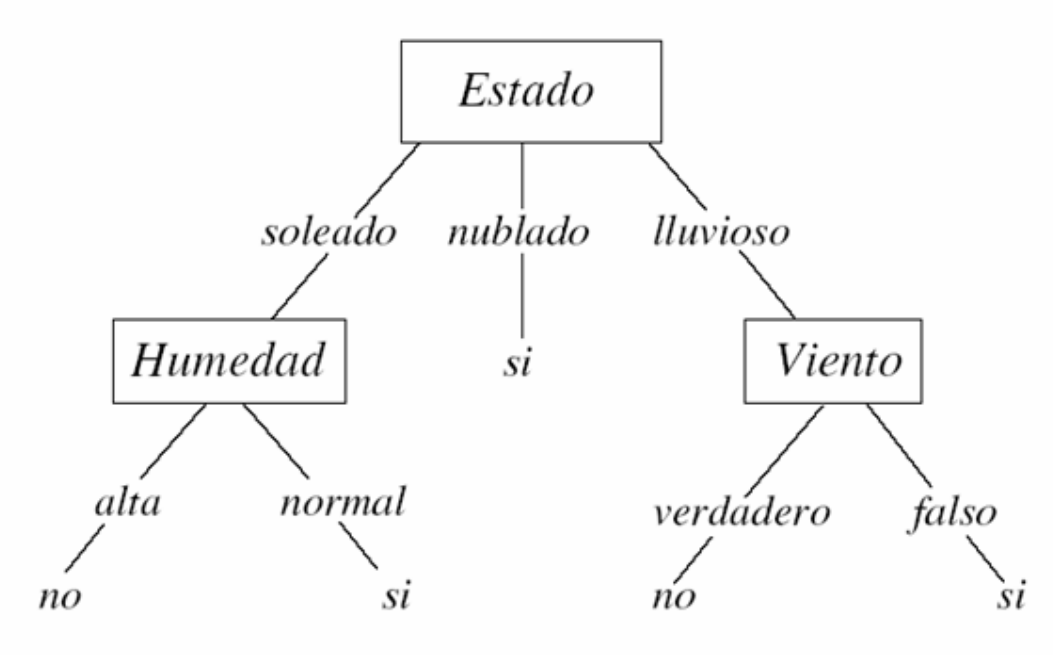
4.2.1 Introducció

L'ús d'arbres de decisió va tenir el seu origen en les ciències socials amb els treballs de Sonquist i Morgan (1964) i Morgan i Messenger (1979), realitzats en el Survey Research Center del Institut de Recerca Social de la Universitat de Michigan. El programa AIN (Automàtic Interaction Detection), de Sonquist, Baker i Morgar (1971), va ser un dels primers mètodes d'ajust de dades basat en arbres de classificació.

En estadística, Kass (1980) va introduir un algorisme recursiu de classificació no binari, anomenat CHAID (Chi-square automàtic interaction detection). Més tard, Breiman, Friedman, Olshen i Stone (1984), van introduir un nou algorisme per a la construcció d'arbres i el van aplicar a problemes de regressió i classificació. El mètode es conegut com CART (Classification and regression trees). Gairebé al mateix temps, el procés de decisió mitjançant arbres va ser començat a usar per la comunitat de "Machine Learning" (Michalski (1973), Quinlan (1983)) i la comunitat de "Patter Recognition" (Henrichon i Fu (1969)). "Machine Learning" és una sub-àrea d' Intel·ligència Artificial que està dins del camp de les ciències de computació, mentre que "Pattern Recognition" està dins de l'àrea d'Enginyeria Elèctrica.

4.2.2 Representació d'arbres de decisió

El terme de "arbres" ve donat per la representació d'aquests algorismes, com la que veiem a la figura [4.2]. L'arrel és el node superior, i en cada node es fa una partició fins a arribar a un node terminal o fulla. Cada node no-terminal conté una pregunta en la qual es basa la divisió del node. Cada node terminal conté el valor de la variable de resposta (arbres de regressió) o el nom de la classe a la qual pertany (arbres de classificació).



Il·lustració4.2: Exemple d'arbre de decisió. Figura extreta de [42].

L'aprenentatge dels arbres de decisió és un mètode molt simple, que ha estat àmpliament utilitzat i amb gran èxit en nombroses tasques d'aprenentatge inductiu. És un mètode d'aproximació de funcions robust a la presència de dades errònies, i es capaç d'aprendre expressions disjuntives. Existeixen molts algorismes d'arbres de decisió, entre els algorismes més utilitzats veiem els següents:

- CHAID: "*Chi-square automàtic interaction detection*", va ser introduït per Kass (1980), i és un derivat del THAID (A seqüencial search program for the analysis of nominal scale dependent variables) (Morgan i Messenger (1973)).

- C4.5: Introduït per Quinlan (1983) dins de la comunitat de "Machine Learning".

- NewId: Molt similar al C4.5.

Un arbre de decisió segmenta l'espai de variables predictores en un conjunt de híper-rectangles, i en cada un d'ells ajusta un model senzill, generalment una constant. És a dir $y=c$, on y és la variable de resposta.

La construcció d'un arbre de decisió es basa en quatre elements:

a) Un conjunt de preguntes binàries Q de la forma $\{ x \in A? \}$, on A és un subconjunt de l'espai de mostres.
b) El mètode usat per particionar els nodes.
c) La estratègia requerida per el creixement de l'arbre.

d) La assignació de cada node terminal a un valor de la variable de resposta (regressió) o a una classe (classificació).

Les diferències principals entre algorismes per construir arbres es troben en la estratègia per "podar-los", la regla per particionar els nodes i el tractament de valors perduts (missing values).

4.2.3 Problemes propis per l'aprenentatge d'arbres de decisió

Tot i que existeixen diversos mètodes per aprendre arbres de decisió, aquest tipus d'aprenentatge s'adapta en general a problemes amb les següents característiques:

- Les instàncies són representades per parells atribut-valor.
- La funció objectiu té valors de sortida discrets.
- Les dades d'entrenament poden contenir errors (en la classificació dels exemples de entrenament o en els valors dels seus atributs).
- Les dades d'entrenament poden tenir valors atributs desconeguts.

Els problemes pràctics que compleixen aquestes característiques són innombrables, com ho han estat les aplicacions d'arbres de decisió per a la seva resolució. En la majoria d'aquests casos, la tasca consisteix en classificar els exemples dins d'un conjunt discret de categories possibles, el que es denomina de manera col·loquial com *problema de classificació.*

4.2.4 Algorisme bàsic per l'aprenentatge d'arbres de decisió

El problema de trobar un arbre de decisió "consistent" amb els exemples d'entrenament pot semblar difícil, però en realitat hi ha una solució trivial. Un podria construir un arbre que tingués un pas a cada etapa per a cada exemple, on el pas testeja cada atribut i segueix el valor corresponent a l'exemple i la fulla té la classificació de l'exemple. Si el mateix exemple és presentat novament, l'arbre de decisió donarà la classificació correcta. Lamentablement, aquest arbre no tindrà molt a dir sobre cap exemple fora del conjunt d'entrenament, ja que només s'ha dedicat a "memoritzar" les observacions. Donat que no s'ha realitzat cap intent per extreure cap patró des dels exemples, no podem esperar que aquest arbre trivial pugui donar resultats en exemples que mai ha vist.

Una altre idea consistiria en seleccionar la hipòtesis més simple que s'ajusti a les dades, i trobar l'arbre de decisió més reduït que sigui consistent amb els exemples d'entrenament. Ja que aquesta tasca pot resultar impossible, una altre alternativa és utilitzar una heurística simple que faci una bona tasca a l'hora de trobar arbres "bastant" reduïts que siguin consistents amb les dades d'entrenament.

Per portar a la pràctica això, un algorisme d'aprenentatge d'arbres de decisió hauria d'intentar testejar l'atribut més important dels exemples en primer lloc. En aquest cas, per a terme "més important" ens referim a que l'atribut seleccionat fa la major contribució per la classificació d'un dels exemples. D'aquesta manera, si repetim el mateix raonament amb tots els atributs, esperem aconseguir la classificació correcte amb un nombre petit de tests, volent dir que tots els passos a l'arbre seran curts i que l'arbre en qüestió serà reduït.

4.2.5 L'elecció de l'atribut

Un dels aspectes centrals dels algorismes d'aprenentatge per a arbres de decisió serà doncs l'elecció de l'atribut correcte en cada node de l'arbre de les dades d'exemple per a crear el model d'entrenament desitjat. Així doncs, el que es desitja és seleccionar l'atribut "més útil" per a classificar els exemples. Suposem per el moment que la nostre intenció és aconseguir arbres de decisió el més reduïts possible, i que classifiquen adequadament els exemples. Així doncs, si tenim un algorisme recursiu el punt de parada principal del qual es produeix quan tots els exemples d'entrenament tenen el mateix valor per l'atribut objectiu. En aquest cas, podríem dir que el conjunt d'exemples d'entrenament és totalment "pur" o homogeni. El nostre objectiu és obtenir arbres el més reduïts possibles. Per tant, la idea serà la de detenir el procés recursiu el més aviat possible. Per això, hauríem d'arribar a particions del conjunt d'entrenament que siguin totalment pures, de la manera més ràpida possible. Si tinguéssim una mesura de la homogeneïtat de cada node, hauríem d'escollir l'atribut que produeix els nodes fills més purs.

Considerem com a exemple la taula de la figura [4.3]. En aquest cas, tenim un conjunt d'atributs A={Estat, Temperatura, Humitat, Vent, JugarTenis}, i la tasca d'aprenentatge consisteix en predir l'atribut objectiu A_0=JugarTenis, que pot prendre valors *si* o *no*, en funció dels altres atributs d'un dia arbitrari, donats pel conjunt $A' \equiv A - \{\text{JugarTenis}\}$. Els valors possibles pels atributs A' són els següents:

V(Estat)={assolellat, ennuvolat, plujós}
V(Temperatura)={calorós, temperat, fresc}
V(Humitat)={alta, normal}
V(Vent)={cert, fals}

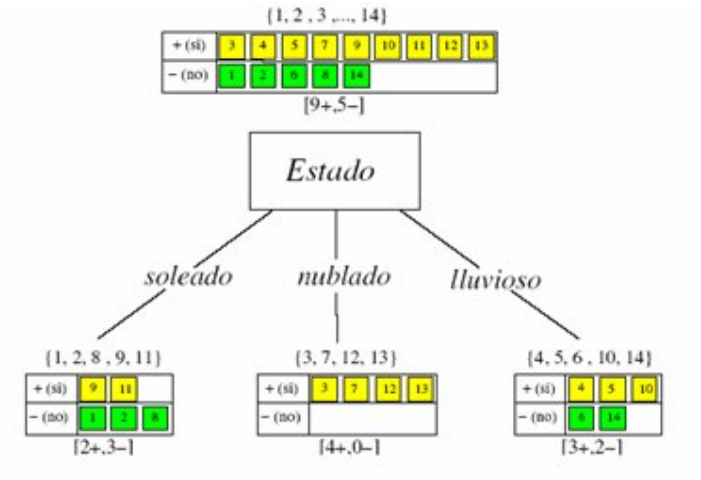
Exemple			**Atributs**		**Classe(A_0)**
	Estat	***Temperatura***	***Humitat***	***Vent***	***Jugar tenis***
e_1	assolellat	Calorós	alta	fals	no
e_2	assolellat	Calorós	alta	cert	no
e_3	ennuvolat	Calorós	alta	fals	si
e_4	plujós	Temperat	alta	fals	si
e_5	plujós	Fresc	normal	fals	si
e_6	plujós	Fresc	normal	cert	no
e_7	ennuvolat	Fresc	normal	cert	si
e_8	assolellat	Temperat	alta	fals	no
e_9	assolellat	Fresc	normal	fals	si
e_{10}	plujós	Temperat	normal	fals	si
e_{11}	assolellat	Temperat	normal	cert	si
e_{12}	ennuvolat	Temperat	alta	cert	si

e_{13}	ennuvolat	Calorós	normal	fals	si
e_{14}	plujós	Temperat	alta	cert	no

Il·lustració4.3: Taula exemple per a la creació d'un arbre de decisió.

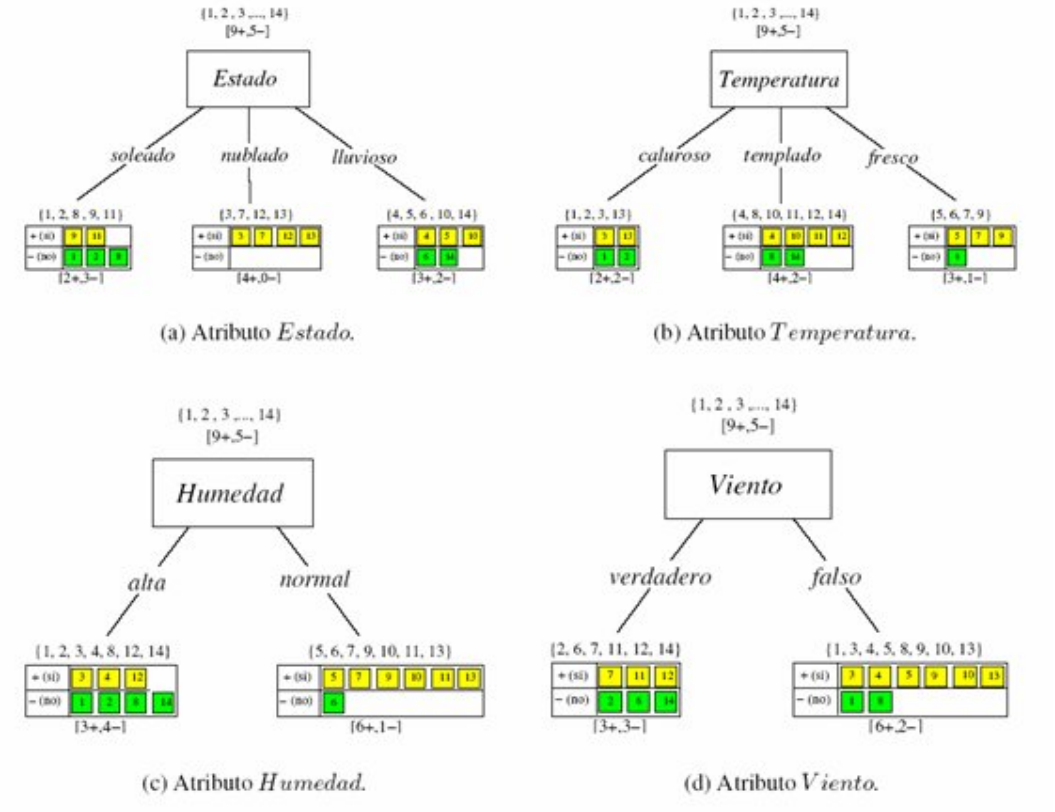
El conjunt d'entrenament consisteix en 14 exemples o mostres, 9 de les quals estan marcades com a positives (*JugarTenis*=si) i 5 com a negatives (*JugarTenis*=no). Per analitzar el grau de puresa en els conjunts resultants d'una partició, observem en primer lloc la figura [4.4], on s'aprecia quin seria l'efecte de segmentar el conjunt d'entrenament de la taula en funció de l'atribut *Estat.*

Podem veure a la taula que el conjunt d'entrenament inicial E = { e_1, e_2, ..., e_{14} }té 9 exemples positius i 5 exemples negatius (que denotem com [9+,5-]), i que si E és segmentat d'acord amb l'atribut *Estat*, obtindrem 3 subconjunts disjunts: $E_{assolellat}$, $E_{ennuvolat}$ i $E_{plujós}$, un per cada valor de l'atribut *Estat.* Cada un d'aquests subconjunts E_v, on $v \in V(Estat)$, tindrà aquells exemples d'entrenament del conjunt E, que tinguin el l'atribut *Estat* el valor v. Així per exemple, podem observar el la figura [4.4] que $E_{assolellat}$={ e_1, e_2, e_8, e_9, e_{11} }, i que en aquest subconjunt 2 exemples classifiquen *JugarTenis* com positiu (e_9 i e_{11}) i 3 exemples el classifiquen com a negatiu (e_1, e_2 i e_8,).



Il·lustració4.4: Segmentació del conjunt d'entrenament d'acord al atribut *Estat*. Figura extreta de [42].

Si es desitgés analitzar de manera intuïtiva quin és el grau d'impuresa que sorgeix de la segmentació del conjunt E, tindríem una situació com la que es mostra a la figura [4.5]. D'acord amb aquestes particions, caldria decidir quin seria el millor atribut per escollir com a arrel de l'arbre. Si l'objectiu és buscar aquells atributs que condueixen a particions més pures (homogènies), l'atribut *Estat* seria la millor opció. És la única alternativa per la qual un node fill és completament pur. L'atribut *Humitat* seria la segona millor opció, ja que produeix un node fill més gran que és casi completament pur.



(a) Atributo *Estado*. (b) Atributo *Temperatura*.

(c) Atributo *Humedad*. (d) Atributo *Viento*.

Il·lustració4.5: Segmentacions de *E* d'acord als diferents atributs. Figura extreta de [42].

Ja que cal portar aquestes idees intuïtives a la pràctica, caldrà una mesura més precisa sobre l'impuresa d'un conjunt d'exemples que pugui ser directament traduïda en un programa de computació. La teoria de la informació en ofereix una eina mitjançant la *entropia.*

La entropia pot ser considerada com la *quantitat d'informació* continguda en el resultat d'un experiment. Aquesta informació, dependrà sobre el coneixement previ que es tingui dels resultats d'aquest experiment. Quan menys coneixement previ es disposa, més informació s'obté (més s'aprèn).

Si un experiment pot tenir *m* resultats diferents v_1, v_2, ..., v_m que poden ocórrer amb probabilitats $P(v_1)$, $P(v_2)$, ... , $P(v_m)$, la quantitat d'informació *I* que s'obté al conèixer el resultat real de l'experiment és:

$$I(P(v_1),...,P(v_m)) \equiv \sum_{i=1}^{m} -P(v_i)\log_2 P(v_i)$$

Equació1

Per donar una idea d'aquest exemple, considerem l'experiment de tirar una moneda, la qual té com a possibles resultats *cara* i *creu*. Si coneixem anteriorment que la moneda va ser alterada per que sempre caigui en *cara,* la entropia (informació) *I* del resultat de l'experiment serà:

$$I(P(cara), P(ceca)) = I(1,0) = -1\log_2 1 - 0\log_2 0 = 0$$

Nota: Considerem

$$0 \log_2 0 = 0$$

Aquest resultat vol dir que, donat que ja sabem que la moneda caurà en *cara,* la informació que obtinguem al conèixer el resultat de l'experiment serà nul·la. Si en canvi utilitzem una moneda totalment balancejada, que produeix qualsevol dels dos resultats de manera equiprobable, tenim que:

$$I\left(\frac{1}{2},\frac{1}{2}\right) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$$

Equació 2

Com podem observar, l'entropia té el seu valor més baix (0) quan existeix una total certesa en el resultat de l'experiment, mentre que té el seu valor més alt en el cas de major incertesa en el resultat (casos equiprobables). Entre aquests dos valors extrems tindrem tota una sèrie de distribucions de probabilitat vàlides caracteritzades per tenir una entropia baixa quan existeixen events altament probables. Així per exemple, si la moneda és alterada per caure *cara en un 99%* dels casos, tindrem que:

$$I\left(\frac{99}{100},\frac{1}{100}\right) = 0,08 .$$

Si per aquest cas, amb dos resultats possibles, els valors de entropis es troben en l'interval [0,1], per el cas general amb *m*resultats possibles i *m*> 2 es compleix que $0 \leq I \leq \log_2 m$.

Per portar aquestes idees al problema de determinar el grau d'impuresa d'un conjunt d'entrenament *E*, observem que aquest conjunt pot ser considerat una mostra de l'atribut objectiu A_0. En aquest cas, les probabilitats d'ocurrència *P* (v_i) de cada valor $v_i \in V(A_0)$ podrien ser estimades com la proporció p_{vi} de exemples en *E* que tenen v_i com valor de A_0:

$$p_{v_i} = \frac{n_{v_i}}{\sum_{v_j \in V(A_\mathcal{O})} n_{v_j}}$$

Equació 3

On n_{vk} és elnombre d'exemples d'entrenament en *E* que tenen a v_k com valor deA_0.

D'aquesta manera, si anomenem el conjunt *E* com $[n_{v1},...,n_{vm}]$, o bé $[p_{v1},...,p_{vm}]$, l'entropia de *E* respecte la classificació A_0 pot ser calculada com:

$$I(E) = I([n_{v_1}, \ldots, n_{v_m}]) = I([p_{v_1}, \ldots, p_{v_m}]) = - \sum_{v_j \in V(A_\mathcal{O})} p_{v_j} \log_2 p_{v_j}$$

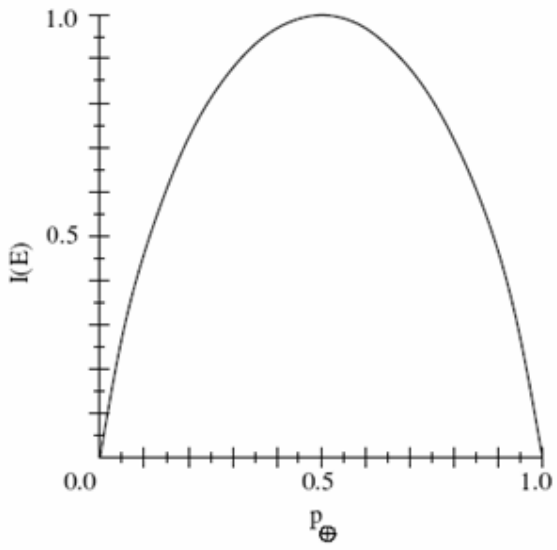
Equació 4

Per el cas particular d'un atribut objectiu booleà, tindrem una col·lecció *E,* amb exemples positius i negatius del concepte objectiu, i la entropia de *E* relativa a aquesta classificació booleana serà:

$$I(E) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

Equació 5

On $p_\oplus$ és la proporció d'exemples positius en *E,* i $p_\square$ és la proporció d'exemples negatius. La figura [4.6] Mostra la forma de la funció de entropia relativa a aquesta classificació booleana, amb $p_\oplus$ variant entre 0 i 1.



Il·lustració4.6: Funció de entropia relativa a una classificació booleana.

Seguint amb l'exemple anterior, si *E* és el conjunt de 14 exemples del concepte booleà *JugarTenis* de la figura [4.3], veiem que *E* conté 9 exemples positius i negatius, i que la entropia de *E* relativa a aquesta classificació booleana és:

$$I(E) = I([9+, 5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14)$$
$$= 0{,}940$$

Com ja havíem vist, la mínima entropia és 0, quan tots els membres pertanyen a la mateixa classe, i la màxima és 1 quan la col·lecció conté el mateix nombre de exemples positius i negatius. Per tant, l'entropia pot ser utilitzada com una mesura de la impuresa d'un conjunt de dades, prenent un valor 0 quan les dades són totalment pures (pertanyen a la mateixa classe) i el valor de màxima impuresa quan existeix la mateixa proporció d'exemples de cada una de les classes del atribut objectiu.

Observem que d'acord al concepte d'entropia, en el cas en que I(E) = 0 tenim màxima puresa. Tot i això, normalment els exemples no són purs (I(E) ≠ 0), i serà necessari analitzar la impuresa de les particions que resulten de considerar els valors possibles d'algun atribut A ∈ *Atribut.* A aquesta quantitat se l'anomena *informació residual* $I_{res}(E, A)$, i s'obté prenent com a base el concepte *d'entropia condicional mitjana* $H(Y|X)$[3], que es defineix com:

$$H(Y|X) \equiv \sum_{x_i} P(x_i) \cdot H(Y|x_i)$$
$$= \sum_{x_i} P(x_i)(-\sum_{y_i} P(y_i|x_i)\log_2 P(y_i|x_i))$$

Portant aquestes idees al context d'un conjunt d'entrenament *E* i un atribut arbitrari *A,* veiem que $I_{res}(E, A)$ pot ser definida com una estimació de $H(A_0|A)$, que s'obté sumant les entropies dels subconjunts resultants de segmentar *E* d'acord a *A*, ponderades per la probabilitat d'ocurrència dels valors de *A*:

$$I_{res}(E, A) \equiv \sum_{v \in V(A)} P(v) \cdot (-\sum_{c \in V(A_{\mathcal{O}})} P(c|v)\log_2 P(c|v))$$
$$= \sum_{v \in V(A)} P(v) \cdot I(E_v)$$
$$= \sum_{v \in V(A)} \frac{|E_v|}{|E|} \cdot I(E_v)$$

La informació residual respecte a un atribut *A,*és la informació que encara necessitem per classificar un exemple després de testejar l'atribut *A*. Per tant, el proper pas de l'algorisme d'aprenentatge de podria limitar-se a escollir aquell atribut *A* amb la menor informació residual.

Una altre alternativa consisteix en considerar el *guany d'informació* $G(E, A)$[4] que s'obté

al testejar l'atribut *A*, i que es calcula com la diferència entre el requeriment d'informació original i la informació requerida després de testejar l'atribut:

$$G(E,A) \equiv I(E) - I_{res}(E,A)$$
$$\equiv I(E) - \sum_{v \in V(A)} \frac{|E_v|}{|E|} \cdot I(E_v)$$

El guany d'informació és simplement la reducció esperada en la entropia causada al segmentar els exemples d'acord amb un atribut. Per tant, en el pas 4 de l'algorisme, l'atribut seleccionat hauria de ser aquell amb major guany d'informació. Per el cas de l'aprenentatge del concepte *JugarTenis*, si prenem per exemple el atribut *Vent* haurem de tenir:

$$V(Vent) = (fals, cert)$$
$$E = [9+, 5-]$$
$$E_{fals} \leftarrow [6+, 2-]$$
$$E_{cert} \leftarrow [3+, 3-]$$

I el guany de informació per aquest atribut pot ser calculat com:

$$G(E, Vent) = I(E) - \sum_{v \in (fals, cert)} \frac{|E_v|}{|E|} I(E_v)$$
$$= I(E) - (8/14) I(E_{fals}) - (6/14) I(E_{cert})$$
$$= 0,940 - (8/14)0,811 - (6/14)1,00$$
$$= 0,048$$

Si de la mateixa manera, calculem el guany d'informació per a tots els atributs de l'exemple, obtindrem els següents resultats:

$$G(E, Estat) = 0,246$$
$$G(E, Humitat) = 0,151$$
$$G(E, Vent) = 0,048$$
$$G(E, Temperatura) = 0,029$$

I per tant, l'algorisme de l'exemple seleccionarà l'atribut *Estat*, creant sota aquest node una branca per cada possible valor d'aquest atribut. L'arbre de decisió parcial resultant en aquest nou cas ja ha estat mostrat a la figura [4.4], junt als exemples d'entrenament que s'assignen a cada nou node de l'arbre i sobre els quals l'algorisme serà aplicat recursivament.

Observem que tots els exemples pels quals *Estat = ennuvolat* són exemples positius de *JugarTenis,* i per tant aquest node es transformarà en un node amb la classificació *JugarTenir = si*. Pel contrari, els descendents que corresponen a *Estat = assolellat* i *Estat = plujós* tenen encara entropia diferent de 0, i l'arbre haurà de ser elaborat sota aquets nodes. El procés de seleccionar un nou atribut i segmentar els exemples d'entrenament es repetit ara per cada node descendent no-terminal, utilitzant aquesta vegada només els exemples d'entrenament associats a aquest node. Els atributs que han estat incorporats més amunt a l'arbre són descartats, i per tant qualsevol atribut pot aparèixer una vegada en qualsevol pas de l'arbre. El procés continua per cada nou node fulla, fins que s'aconsegueix alguna deles següents 2 condicions:

1. Tot atribut ja ha estat inclòs en aquest pas a través de l'arbre.

2. Tots els exemples d'entrenament associats a aquest node tenen el mateix valor de atribut objectiu (entropia = 0).

L'arbre de decisió final aprés per l'algorisme, a partir dels 14 exemples d'entrenament de la variable *JugarTenis*, és el que s'ha mostrat a la figura [4.2].

4.2.6 Atributs numèrics

Fins el moment només hem considerat atributs nominals amb un conjunt discret de valors. Tot i això, en la majoria dels conjunts de dades reals apareixen atributs numèrics continus(com en el cas dels descriptors de característiques facials utilitzats en aquest projecte). Aquest tipus d'atributs poden ser fàcilment incorporats al esquema previ, mitjançant la definició dinàmica de nous atributs que segmenten els atributs continus en un conjunt discret d'intervals. La idea en aquest cas, és que un atribut numèric A pot ser segmentat de manera binària, creant dinàmicament un nou atribut booleà A_c , que és cert si $A < c$ i fals en el cas contrari. La única pregunta que queda per respondre és quin és el millor valor per el llindar (threshold) $c$en el nostre problema. Considerant que l'atribut A_c serà comparat amb altres atributs d'acord al seu guany d'informació, és evident que el valor de c haurà de ser aquell que entre tots els llindars possibles, maximitza el guany d'informació.

Un conjunt de llindars candidats, podria ser donat per els valors intermedis de l'atribut *A* en el que aquests canvis es produeixen. Aquesta elecció dels llindars candidats no és arbitraria. S'ha demostrat que el valor de *c* que maximitza el guany d'informació sempre estarà ubicat en algun d'aquests punts. Per tant, una aproximació per trobar el llindar òptim de *c* consistirà en, donats els possibles punts intermedis per el llindar, seleccionar aquell que produeix el major guany d'informació.

Per apreciar aquestes idees en un exemple, assumim que l'atribut *Temperatura* , a diferència de l'exemple de les seccions prèvies, no és un atribut nominal, sinó que és un atribut numèric (sencer). Considerem a més, que els valors de *Temperatura* que s'han observat en els exemples d'entrenament associats a un node particular són els mostrats a la taula de la figura [4.3], junt als valors corresponents per l'atribut *JugarTenis.*

Temperatura	40	48	60	72	80	90
JugarTenis	No	No	Si	Si	Si	no

En aquest exemple, existeixen dos llindars candidats per les dues línies verticals sobre la taula, que corresponen als valors de *Temperatura* on el valor *JugarTenis* canvia: (48+60)/2 i (80+90)/2. El guany de informació pot ser calculada per cada un dels atributs candidats, *Temperatura<54*i *Temperatura<85,* i el millor dels dos serà el seleccionat.

4.2.7 Exemples amb valors desconeguts dels atributs

En l'etapa prèvia a la aplicació d'un algorisme d'aprenentatge, es sol realitzar una mena de "neteja" de les dades d'entrenament per solucionar problemes freqüents, com per exemple la existència d'exemples d'entrenament dels que es desconeixen els valors d'alguns dels seus atributs. La solució en aquest cas, pot consistir en adoptar certs criteris molt dràstics, com quan s'eliminen els atributs afectats, o bé s'eliminen les mostres completes que tenen valors restants. Una altre alternativa és realitzar algun tipus de "reparació" de les dades, completant les mateixes de manera manual o fins i tot mitjançant tècniques automàtiques més sofisticades que inclouen la predicció de valors restants mitjançant un algorisme d'aprenentatge automàtic. Per altre banda, si el fet de que la falta del valor per un atribut és significatiu, també es pot considerar al valor restant com un altre valor possible de l'atribut.

Aquestes tècniques prèvies a l'aplicació de l'algorisme d'aprenentatge automàtic, es solen basar en criteris molt generals i en molts casos s'opta per deixar les dades d'entrenament incompletes i assumir que l'algorisme d'aprenentatge solucionarà aquest problema de manera robusta. Per el cas particular de l'aprenentatge d'arbres de decisió, el problema se'ns presenta quan existeix un exemple d'entrenament $e = \langle x, c(x) \rangle \in E$ que té un valor desconegut per l'atribut *A,* i necessitem estimar *x(A)* per poder calcular *G(E,A)* en un node arbitrari *n*.

Una estratègia per solucionar aquest problema consisteix en considerar que *x(A)* és el valor més comú de l'atribut *A* entre els exemples de *E* en el node *n,* que tenen la classificació *c(x).* En qualsevol d'aquests dos casos, l'exemple d'entrenament elaborat per assumir aquest valor de *x(A)* podrà ser usat directament per un algorisme d'aprenentatge d'arbres de decisió.

Un procediment més complex (utilitzat en l'algorisme C4.5) consisteix en segmentar (conceptualment) l'exemple *e* en tantes peces e_i, com valors $v_i \in$ V(A) existeixin. Cada peça e_i és ponderada amb una probabilitat p_i que pot ser estimada en base a la freqüència observada del valor v_i en els exemples d'entrenament en el node *n*. Posteriorment, el valor p_i pot ser utilitzat en tots els

càlculs per determinar el guany de l'atribut *A*, i cada fracció de la instància *e* és distribuïda per la branca corresponent al valor de l'atribut *A*.

4.3 SVM

4.3.1 Introducció

Les Màquines de Suport Vectorial (SVM) són un mètode de classificació i de reconeixement de patrons molt efectiu. En aquest apartat veurem els fonaments bàsics, tant teòrics com pràctics de les SVM, i com es pot suportar el seu potencial en tasques de classificació. La teoria de SVM va ser desenvolupada inicialment per V.Vapnik [1] a principis dels anys 80, i es centra en el que es coneix com Teoria de l'Aprenentatge Estadístic. L'objectiu de les SVMs és donar solució al problema fonamental que sorgeix en diferents camps, on s'estudia la relació entre tall i variància, el controls de la capacitat, el sobre ajust en les dades, etc.

Intuïtivament i a mode d'esquema, tenim que donat un grup de dades distribuïdes en dues classes, una SVM lineal busca un hiperplà, de tal manera que la major quantitat de mostres dela mateixa classe quedin al mateix costat, mentre que es maximitza la distància d'aquestes classes al hiperplà.

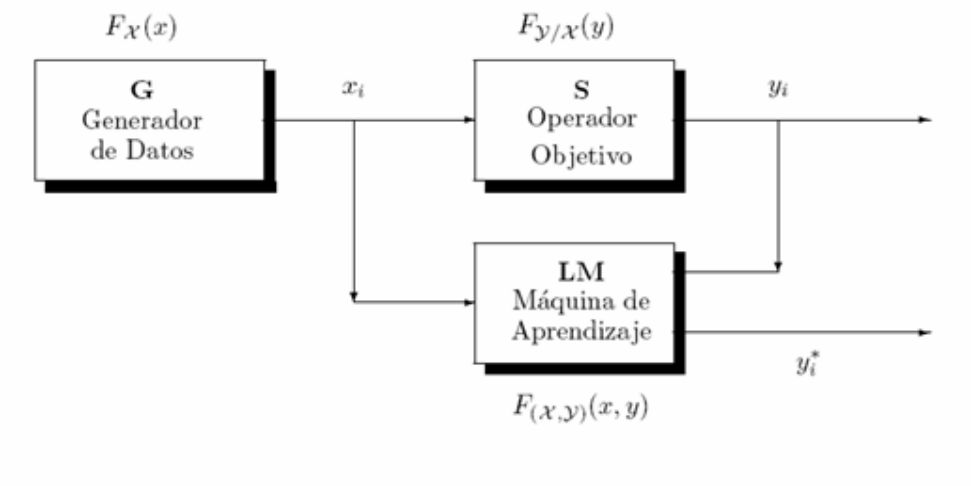
L'objectiu fonamental d'aquest tipus d'estudis és aprendre a partir de les dades conegudes, i cercar l'existència d'alguna dependència funcional entre un conjunt de vectors d'entrada o inputs:

$$\{x_i,\ i = 1, \cdots, n\} \subseteq \mathcal{X} \subseteq \mathbb{R}^d$$

I valors de sortida o outputs:

$$\{y_i,\ i = 1, \cdots, n\} \subseteq \mathcal{Y} \subseteq \mathbb{R}$$

El model representat a la figura [4.7] (denominat model d'aprenentatge a partir d'exemples) recull de manera clara l'objectiu que persegueix.



Il·lustració 4.7: Esquema de configuració d'una màquina d'aprenentatge a partir de exemples. Figura extreta de [34].

En aquest esquema, G representa un model generador de dades que ens proporciona els vectors $x_i \in X$, independents i idènticament distribuïts d'acord amb una funció de distribució $F_x(x)$, desconeguda però que suposem que no varia al llarg del procés d'aprenentatge. Cada vector x_i és la entrada de l'operador objectiu S, el qual el transforma en un valor y_i segons una funció de distribució condicional $Fx|y = x_i(y)$. Així, la màquina d'aprenentatge que anomenem recull el següent conjunt d'entrenament:

$$Z = \{(x_1, y_1), \cdots, (x_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y} = \mathcal{Z}$$

El qual és obtingut independentment i idènticament distribuït seguint la funció de distribució conjunta:

$$F_{(\mathcal{X},\mathcal{Y})}(x,y) = F_{\mathcal{X}}(x) \cdot F_{\mathcal{Y}/\mathcal{X}=x}(y)$$

A partir del conjunt d'entrenament *Z*, la màquina d'aprenentatge "construeix" una aproximació al operador desconegut, el qual proporciona un generador *G*, la millor aproximació (segons algun criteri) a les sortides proporcionades per el supervisor. Formalment, construir un operador vol dir que la màquina d'aprenentatge implementa un conjunt de funcions, de tal manera que durant un procés d'aprenentatge, escull d'aquest conjunt una funció apropiada seguint una determinada regla de decisió.

L'estimació d'aquesta dependència estocàstica basada en un conjunt de dades intenta aproximar la funció de distribució condicional $Fy|x\ (y)$, el qual en general porta a un problema realment complicat (veure [29] i [30]). Tot i això, el coneixement de la funció $Fy|x\ (y)$ no sempre és necessari, ja que molts cops només s'està interessat en només alguna de les seves característiques. Per exemple es pot desitjat estimar la funció d'esperança matemàtica condicional:

$$E[Y/X = x] \stackrel{def}{=} \int y\, dF_{\mathcal{Y}/x}(y)$$

Per això, l'objectiu del problema és la construcció d'una funció *f(x,y)* dins d'una determinada classe de funcions *F* escollida anteriorment, la qual ha de complir un determinat criteri

de la millor manera possible. Formalment, el problema es planteja de la següent manera:

Donat un subespai vectorial Z de $\square^{d+1}$ *on es té definida una mitja de probabilitat Fz(x), un conjunt* $F=\{f(z),z\in Z\}$ *de funcions reals i un funcional* $R: F \to \square$.

Buscar una funció $f^* \in F$ tal que:

$$R[f^*] = \min_{f \in \mathcal{F}} R[f]$$

Amb l'objectiu de ser el més general possible, seria bo escollir el funcional *R* de tal manera que es pogués plantejar amb ell, el major nombre de problemes possibles. Per això es defineix *R*[] com veiem:

Definició 1:*Donada una classe* $F=\{f(z),z\in Z\}$ *de funcions reals i una mitja de probabilitat Fz(z), es defineix el risc R com:*

$$R[f] = \int_Z c(z, f(z))\, dF_Z(z)$$

Equació 6

On c (.,.) es denomina funció de pèrdua (o de cost), i prendrà valors no negatius.

Veient la figura [4.7,] s'arriba a la conclusió de que els valors y_ii y_i* no han de coincidir necessàriament. Quan això sigui així, la màquina d'aprenentatge haurà comés algun error que s'ha de quantificar d'alguna manera, i per això es defineix aquesta funció de pèrdua.

Així, en aquest plantejament, donat un conjunt $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, el principal problema consisteix en formular un criteri constructiu per escollir una funció de *F* , ja que el funcional de la equació [6] per si mateix no serveix com a criteri de selecció, degut a que la funció *Fz(z)* inclosa en ell és desconeguda. Per elaborar aquest criteri s'ha de tenir en compte que el risc es defineix com l'esperança matemàtica d'una variable aleatòria respecte a una mesura de probabilitat, i per tant és lògic escollir com estimació la mitja mostral, i d'aquí s'extreu la següent definició:

Definició 1.2 *Donat un risc definit per la equació [6], un conjunt de funcions F i una mostra* $\{z_1, \ldots, z_n\}$. *El funcional risc empíric* (R_{emp}) *es defineix com:*

$$R_{emp}[f] = \frac{1}{n}\sum_{i=1}^{n} c(z_i, f(z_i)), \qquad f \in \mathcal{F}$$

La manera clàssica d'enfrontar-se a aquests problemes és la següent: si el valor mínim del risc s'aconsegueix amb una funció f_o i el mínim del risc empíric amb f_n per unamostra donada de mida *n,* llavors es considera que f_n és una aproximació a $f_o$en un determinat espai mètric. El principi que resol aquest problema es denomina principi de minimització del risc empíric. Aquest és el principi utilitzat en els desenvolupaments clàssics, per exemple quan es planteja a partir d'un conjunt de dades la regressió lineal mínim quadràtica.

La pregunta que sorgeix és si es pot assegurar que el risc $R[f_n]$ està aprop del $mín_{f \in F} R[f]$. La resposta és que en general això no és veritat (veure [31]), el qual porta a la conclusió que la classe de funcions *F* no pot ser arbitrària, necessàriament s'han d'imposar algunes condicions de regularitat als seus elements, amb la via d'un funcional Q[f]. Així, en l'elaboració del problema s'ha de buscar una adequada relació entre la precisió aconseguida amb un particular conjunt d'entrenament, mesurat a través de $R_{emp}[f]$, i la capacitat de màquina mesurada per Q[f]. Això porta a considerar el problema de minimitzar un risc regularitzat, on aquest es defineix per algun $\lambda > 0$ de la manera següent:

$$R_{reg}[f] = R_{emp}[f] + \lambda Q[f]$$

Com es pot veure a la figura [4.7], aquest hiperplà minimitza el risc de classificacions errònies en el grup pres per a realitzar en procés de validació.

4.3.2 Classificació amb Vectors de Suport

Per un grup d'entrenament de mida *N* compost de parells atribut-etiqueta $(x_i, y_i)_{1 \le i \le N}$, sent $x_i \in \mathbb{R}^n$ i $y_i \in [-1,1]$, es desitja obtenir una equació per un hiperplà que divideixi aquest grup d'entrenament, de manera que aquells punts amb la mateix etiqueta quedin al mateix costat del hiperplà. Això vol dir trobar un *w* i una *b* tal que:

$$y_i(w' x_i + b) > 0, \; i = 1, ..., N$$

Equació 7

Si existeix un hiperplà que satisfaci les condicions anteriors, es diu que les dades són *linealment separables.* En aquest cas, *w* i *b* es poden escalar així:

$$\min_{1 \le i \le N} \; y_i(w' x_i + b) \ge 1$$

De tal manera que el punt més proper al hiperplà tingui una distància *1/||w||* . Així doncs, l'equació [7] es pot escriure com:

$$y_i(w' x_i + b) \ge 1$$

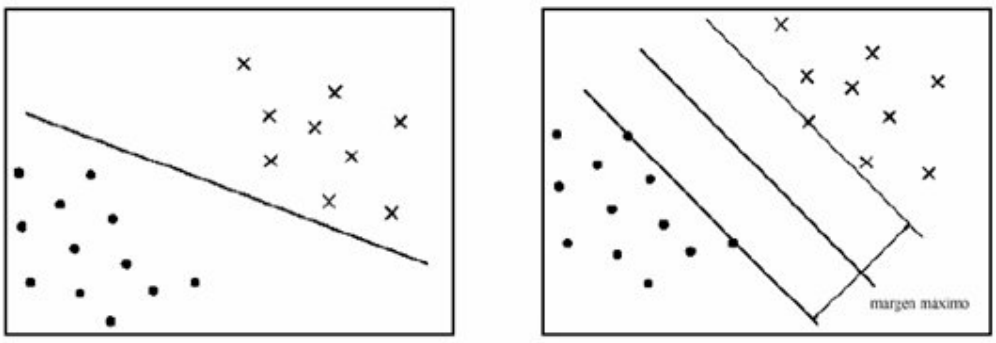
Equació 8

Així, entre tots els possibles hiperplans, aquell al qual la seva distància al punt més proper és màxima es denomina el "hiperplà òptim de separació"(OSH). És a dir, que s'escull l'hiperplà que proporcioni una major separació entre els grups de treball, ja que d'aquesta manera es permet distingir de forma més clara les regions on cauen els punts amb diferents etiquetes. Mentre la distància al hiperplà òptim sigui *1/||w||* , trobar el OSH equival a resoldre el següent problema:

$$\min_{w \in \mathbb{R}^d} \frac{1}{2} \|w\|^2$$
$$s.a. \quad y_i \left(x_i \cdot w + b\right) - 1 \geq 0, \ i = 1, \cdots, n$$

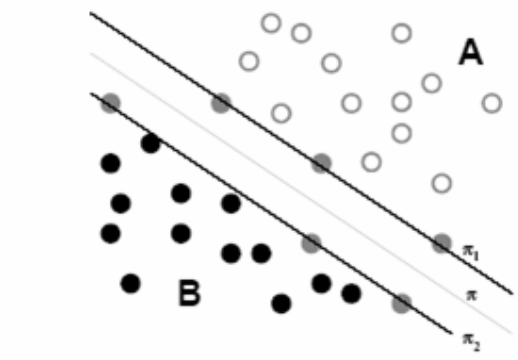
Equació 9

La quantitat *2/||w||* és anomenada "marge", i l'hiperplà que maximitza aquest marge, OSH. El marge pot ser vist com una mesura de la dificultat del problema, així, com més petit sigui el marge més difícil és el problema; o vist d'una altre manera, si el marge és gran s'espera una millor capacitat de generalització(veure figura [4.8]).



Il·lustració 4.8: Hiperplans que separen correctament les dades. El OSH de la dreta té un marge major, i per tant s'espera una millor generalització. Figura extreta de [35].

La solució per el cas de dimensió dos es pot interpretar gràficament a partir de la figura [4.9]. A la vista d'aquesta figura, és fàcil adonar-se d'una característica important de les SVM's, i és que si s'afegeix o s'elimina qualsevol nombre de vectors que compleixin la desigualtat estricta (equació [8]), la solució del problema d'optimització no es veu afectada. Tot i això, n'hi ha prou amb afegir un vector que es trobi entre els dos hiperplans, perquè la solució canviï totalment.



Il·lustració4.9: Hiperplans paral·lels i vectors de suport en R^2. Figura extreta de [35].

Per resoldre el problema d'optimització amb restriccions (equació [9]), s'utilitzen els multiplicadors de Lagrange. Així la funció objectiu és:

$$L(w,b,\alpha) = \frac{1}{2}w'w - \sum_{i=1}^{N} \alpha_i[y_i(w'x_i + b) - 1]$$

Equació 10

Llavors ens queda com un problema de programació quadràtica on la funció objectiu és convexa, i els vectors que satisfan les restriccions formen un conjunt convex. Això vol dir que es pot resoldre el següent problema dual associat al problema principal: maximitzar la funció $L(w,b,\alpha_i)$ respecte a les variables duals $\alpha_i$subjecte a les restriccions imposades per que els gradients de L respecte a w i b siguin nuls, i subjecte també al conjunt de restriccions $C_2 = \{\alpha_i \geq 0, i = 1, \ldots, n\}$. La solució d'aquest problema s'expressa de la manera següent:

$$\frac{\partial L(w,b,\alpha)}{b} = \sum_{i=1}^{N} y_i\alpha_i = 0$$
$$\frac{\partial L(w,b,\alpha)}{w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0$$

Equació 11

Substituint la equació [11] a [10], s'aconsegueix la funció objectiu dual:

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i\, \alpha_j\, y_i\, y_j\, x_i\, x_j$$

Equació 12

Els vectors del conjunt d'entrenament que proporcionen un multiplicador a $\alpha_i > 0$ són anomenats **vectors de suport,** i clarament, aquests vectors es troben en un dels hiperplans π_1 o π_2. Per a aquest tipus de model d'aprenentatge, els vectors de suport són els elements crítics, per que ells són els que proporcionen l'aproximació del problema, ja que si tots els elements restants del conjunt d'entrenament són eliminats (o són canviats per altres que no es troben entre els dos hiperplans), i es repeteix el problema d'optimització, es troben els mateixos hiperplans separadors.

Aquesta característica dels models de vectors de suport pot ser utilitzada en molt problemes on es desitja destacar la importància de determinades entrades. També si es treballa amb una gran quantitat d'entrades, és útil treballar amb els vectors de suport, ja que aquests formen un esquema de compressió que permet reconstruir la solució al problema, és a dir, si considerem exclusivament els vectors de suport i descartem la resta de vectors d'entrenament tindríem un problema d'optimització amb menys restriccions que proporciona la mateixa informació.

Les condicions del problema d'optimització ens porta a que es compleixi la següent equació(veure [32]):

$$\alpha_i \cdot (y_i \cdot (x_i \cdot w + b) - 1) = 0$$

Anomenada condició (complementària) de Karush-Kuhn-Tucker (KKT). Aquestes restriccions indiquen que el producte de les restriccions del problema primal ($y_i \cdot (x_i \cdot w + b) - 1 \geq 0$) i les restriccions del problema dual ($\alpha_i \geq 0$) s'anul·len en tots els vectors d'entrenament. D'aquesta manera es segueix que les condicions KKT (veure [33]):

$$\begin{aligned}
w_j - \sum_{i=1}^{n} \alpha_i \, y_i \, x_{ij} &= 0 \qquad j = 1, \cdots, d \\
\frac{\partial}{\partial b} L_P = -\sum_{i=1}^{n} \alpha_i \, y_i &= 0 \\
y_i(x_i \cdot w + b) - 1 &\geq 0 \qquad \forall i = 1, \cdots, n \\
\alpha_i &\geq 0 \qquad \forall i = 1, \cdots, n \\
\alpha_i(y_i(x_i \cdot w + b) - 1) &= 0 \qquad \forall i = 1, \cdots, n.
\end{aligned}$$

Dels desenvolupaments inicials no es segueix una manera explicita de determinar el valor de *b,* tot i això, la condició KKT complementària ens permet determinar-lo. Per això, n'hi ha prou amb escollir un $\alpha_i > 0$ i aïllar *b,*obtenint :

$$b_0 = y_i - w_0' x_i$$

Encara que s'ha determinat *b*, és més adequat realitzar els càlculs amb tots els $\alpha_i > 0$ i escollir amb a valor de *b* un valor mitjà dels resultats obtinguts, amb l'objectiu d'arrodonir els errors intrínsecs associats a tot mètode de càlcul numèric:

$$b = \frac{1}{\#\{\alpha_i > 0\}} \sum_{\alpha_i > 0} (y_i - x_i \cdot w)$$

El problema de classificar un nou punt *x,* es resolt examinant el signe de $w_0' x + b_0$. Ara, considerant la expansió de w_0:

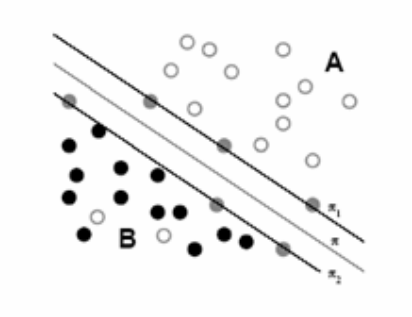
$$w_0 = \sum_{i=1}^{N} \alpha_i^0 y_i x_i$$

La funció de decisió f(x) per l'hiperplà pot ser escrita com:

$$f(x) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i^0 y_i x_i' x + b\right)$$

4.3.3 Cas linealment no separable

A la pràctica, treballar amb conjunts de dades separables no és habitual. En aquest cas, es troben vectors d'una classe dins de la regió corresponent als vectors de una altre classe i per tant mai es podran separar aquestes classes amb hiperplans. (veure figura [4.10]). En aquestes situacions es dirà que el conjunt és **no separable**.



Il·lustració4.10: Exemple de hiperplans separadors per el cas de dades no separables.

Davant aquests casos, el problema d'optimització no troba una solució possible. Tot i això, no és difícil ampliar les idees generals del cas separable al cas no separable introduint una variable ξ en les restriccions i plantejar un nou conjunt de restriccions:

$$\begin{aligned} x_i \cdot w + b &\geq +1 - \xi_i \quad \text{para } y_i = +1 \\ x_i \cdot w + b &\leq -1 + \xi_i \quad \text{para } y_i = -1 \\ \xi_i &\geq 0 \quad \forall i = 1, \cdots, n \end{aligned}$$

El propòsit de les variables ξ_i és permetre punts erròniament classificats, és a dir, que una entrada no sigui ubicada a la classe correcte, els quals corresponen a $\xi_i > 1$, i per tant, $\sum_i \xi$ és una cota superior del nombre d'errors d'entrenament. Ja que en el cas de no separable, necessàriament s'han de cometre errors, sembla natural assignar a la funció objectiu un cost extra que en certa manera penalitzi els errors (funció de pèrdua). Per tot això, plantegem el problema:

$$\min_{w,b} \frac{1}{2} w'w + C \sum_{i=1}^{N} \xi_i$$
$$s.a.\ y_i(w'x_i + b) \geq 1 + \xi_i \text{ y } \xi \geq 0,\ \forall i$$

Equació 13

El primer terme és minimitzat per controlar la capacitat d'aprenentatge del mateix mode que en el cas separable; el segon terme permet mantenir sota control el nombre de classificacions errònies. El paràmetre *C* és escollit per l'usuari de manera que un valor gran és equivalent a assignar una alta penalització als errors. En analogia amb el cas separable, la utilització de multiplicadors de Lagrange deriva en el següent problema d'optimització:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i^k$$

Si es considera un valor de *C* gran, vol dir que s'està assignant un pes molt alt als errors enfront de || w ||², i pel contrari, si *C* és petita assigna un pes major a || w ||². Aquesta interpretació resulta més intuïtiva si s'interpreta que || w ||² és un factor de suavització de la solució buscada. Per altre banda, si *k* és gran, el que fem és donar molt més pes als errors quan més grans siguin aquests. S'arriba per tant a plantejar un problema de programació convexa per qualsevol valor de *k.* Si *k* = 2 o *k* = 1, es té un problema de programació convexa quadràtic. Considerem aquí que *k* = 1, ja que en aquest cas es té el avantatge de que cap valor de ξ_i, ni cap dels seus corresponents multiplicadors de Lagrange, apareixen en el problema dual. Per tant el problema d'optimització que es planteja és:

$$\min_{w \in \mathbb{R}^d} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$
$$s.a. \begin{cases} y_i\ (x_i \cdot w + b) - 1 + \xi_i \geq 0,\ \forall i \\ \xi_i \geq 0,\ \forall i \end{cases}$$

Equació 14

Utilitzant la tècnica dels multiplicadors de Lagrange, s'arriba a la funció objectiu dual:

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i\, \alpha_j\, y_i\, y_j\, x_i\, x_j$$

La qual s'ha de maximitzar respecteα_i subjecte a:

$$0 \leq \alpha_i \leq C,\ \ i = 1, \cdots, n \qquad \sum_{i=1}^{n} \alpha_i\, y_i = 0$$

La solució final de la qual ve donada per:

$$w = \sum_{i=1}^{N_{SV}} \alpha_i \, y_i \, s_i$$

On N_{SV} denota el nombre de vectors de suport i s_i els vectors de suport del conjunt $\{x_1, ..., x_n\}$. Clarament $N_{SV} \leq n$, i una de les característiques més importants d'aquests models és que escollint adequadament els paràmetres, és possible aconseguir que N_{SV} sigui molt inferior a n, amb el que s'aconsegueix una representació "curta" de la solució en funció dels vectors d'entrada sense perdre capacitat de generalització.

Es pot observar que la única diferència entre aquesta solució respecte en el cas separable és que els multiplicadors de Lagrange α_i estan acotats superiorment per la constant C.

Les condicions de KKT associades a aquest problema són les següents:

$$\begin{aligned} \frac{\partial}{\partial w_j} L_P = w_j - \sum_{i=1}^{n} \alpha_i \, y_i \, x_{ij} &= 0 \\ \frac{\partial}{\partial b} L_P = -\sum_{i=1}^{n} \alpha_i \, y_i &= 0 \\ \frac{\partial}{\partial \xi_i} L_P = C - \alpha_i - \mu_i &= 0 \end{aligned}$$

Equació 15

$$\begin{aligned} y_i \, (x_i \cdot w + b) - 1 + \xi_i &\geq 0 \\ \xi_i, \, \alpha_i, \, \mu_i &\geq 0 \\ \alpha_i (y_i \, (x_i \cdot w + b) - 1 + \xi_i) &= 0 \end{aligned}$$

Equació 16

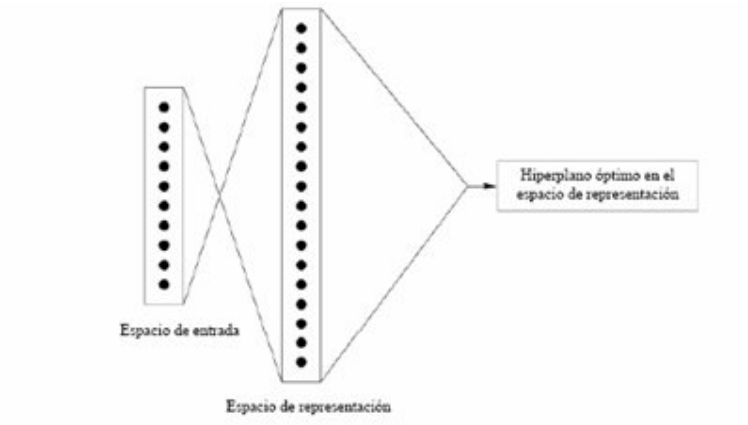
$$\mu_i \, \xi_i = 0$$

Equació 17

Com s'ha comentat en el cas separable, es poden utilitzar les condicions complementàries de KKT (les igualtats [16] i [17]), per determinar el valor de b. Es pot notar que la equació [15] combinada amb la [17] mostra que, si $\xi_i = 0$, llavors $\alpha_i < C$. Així es pot simplificar b prenent vectors d'entrenament tals com $0 < \alpha_i < C$ i utilitzar l'equació [16] amb $\xi_i = 0$ (és més indicat promitjar aquest valor entre tots les vectors d'entrenament amb $\xi_i = 0$).

4.3.4 Màquines de suport no Lineals

El principi de SVM no lineal consisteix en mapejar l'espai d'entrada a un espai de representació de dimensió més alta a través d'una funció no lineal escollida a priori [4], veure figura [4.11].



Il·lustració4.11: La SVM mapeja l'espai d'entrada en un altre de representació de dimensió més alta i després construeix un OSH sobre aquest últim. Figura extreta de [34].

Tot i això, apareix un problema computacional, ja que la dimensió de l'espai de representació pot ser molt alta i la dificultat radica en com construir un hiperplà de separació en aquest espai. La resposta al problema parteix de que per a construir aquest hiperplà, el mapeig $z = \phi(x)$ no necessita ser explícit, de manera que substituint x per $\phi(x)$ en la equació [12] s'aconsegueix:

$$\text{máx} \sum_{i}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j \phi(x_i)' \phi(x_j)$$

$$s.a. \sum_{i=1}^{N} y_i \alpha_i = 0 \text{ y } \alpha_i \geq 0, \ \forall i$$

D'aquestes equacions, l'algorisme d'entrenament només depèn de les dades a través dels productes de punts en l'espai de representació, això són funcions de la forma $z = \phi(x_i)'\phi(x_j)$. Sigui donada una funció de kernel simètrica *K* tal que $K(x_i, x_j) = \phi(x_i)'\phi(x_j)$, de manera que l'algorisme d'entrenament depengui només de *K* i el mapeig ϕ no sigui utilitzat explícitament.

Donat $\phi: \mathbb{R}^d \to H$, el kernel de *K* és $K(x_i, x_j) = \phi(x_i)'\phi(x_j)$, però de manera inversa, donat un kernel *K* s'han d'establir les condicions per a que el mapeig existeixi. Aquestes condicions són assegurades per les condicions de Mercer:

Teorema: *Sigui K(x,y) una funció simètrica contínua en L²(C), després, existeix un mapeig* ϕ *i una expansió, tal que:*

$$K(x,y) = \sum_{i=1}^{\infty} \phi(x)_i' \phi(y)_i$$

Equació 18

Si i només si, per algun $g \in L^2(C)$*, tal que:*

$$\int_{C\times C} K(x,y)g(x)g(y)dxdy \geq 0$$

Equació 19

Cal destacar que per casos específics, pot ser difícil mostrar quan les condicions de Mercer es compleixen, mentre que la equació [18] ha de mantenir-se per algun $g \in L^2(C)$. Tot i això, és fàcil provar que la condició es compleix per el kernel polinomial vist a [2]:

$$K(x,y) = (x'y)^p$$

Els primers kernels investigats per el reconeixement de patrons van ser els següents:

- Polinomial: $K(x,y) = \left(x'y + c\right)^d$ per c > 0
- Funció de base radial (RBF): $K(x,y) = \exp\left(-\lambda \left|\left|x - y\right|\right|^2\right)$ *per* $\lambda > 0$
- Sigmoide: $\tanh\left(k\ x'y\ + v\right)$

Del primer en resulta en un classificador amb funció de decisió polinomial, del segon un classificador amb funció de base radial i de l'últim un tipus particular de xarxa sigmoïdal de dos capes. Per el cas de RBF, el nombre de centres (nombre de SV), els centres (SV), els pesos (α_i) i el desplaçament (*b*) són generats automàticament per la SVM en l'etapa d'entrenament i donen excel·lents resultats en comparació a la xarxa RBF clàssica [5]. De la mateixa manera, per el cas del perceptró multicapa (MLP), la arquitectura (nombre de nodes ocults) es determina per l'entrenament de la SVM.

4.3.5 Algorisme de Entrenament

Considerant la formula general per la SVM, és a dir, no lineal i no separable:

$$\text{máx} \sum_{i}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$
$$s.a. \sum_{i=1}^{N} y_i \alpha_i = 0 \text{ y } 0 \leq \alpha_i \leq C, \ \forall i$$

Equació 20

El mètode de descomposició és tingut en compte considerant la densitat de la matriu kernel $K\left(x_i, x_j\right)$ de la equació [18]. Bona part del treball al voltant d'aquest mètode es pot trobar a [6], [7], [8] i [9].

4.3.5.1 Mètode de Descomposició

Partint de l'equació [20], es pot realitzar la següent representació vectorial:

$$\min_{\alpha} \frac{1}{2} \alpha' Q \alpha - e' \alpha$$
$$s.a. \quad y' \alpha = 0 \text{ y } 0 \leq \alpha_i \leq C, \ \forall i$$

Equació 21

On $Q_{ij} = y_i y_j K(x_i, x_j)$ y $e = 1, \ \forall i.$

Algorisme 1:

- *Donat un nombre q < N, com la mida del conjunt de treball, trobem a α^1 la solució inicial, i tenim k =1*

- *Si α^k és la solució òptima de la equació [21] i per tant aquesta s'acaba, una altre manera de buscar un conjunt $B \subset \{1,...,N\}$ amb mida q. Es defineixen $L \equiv \{1,...,N\} \setminus B$, α_B^k i α_L^k com subvectors de α^k corresponents a B i a L respectivament.*

- *Es resol el següent problema respecte de α_B:*

$$\min_{\alpha_B} \frac{1}{2}\alpha_B' Q_{BB}\alpha_B - (e_B + Q_{BL}\alpha_L^k)'\alpha_B$$
$$s.a.\ \ y_B'\alpha_B = -y_L'\alpha_L^k \text{ y } 0 \le (\alpha_B)_i \le C,\ \forall i$$

Equació 22

On $\begin{bmatrix} Q_{BB} & Q_{BL} \\ Q_{LB} & Q_{LL} \end{bmatrix}$ *és una permutació de la matriu Q.*

- *Es deixa* α_B^{k+1} *com a solució òptima de (2.18) i* $\alpha_L^{k+1} \equiv \alpha_L^k$. *Es fa que k=k+1 i es torna al pas 2.*

La idea bàsica de l'algorisme de descomposició és que en cada iteració, els índex {1, ..., N} del conjunt d'entrenament, siguin separats en dos més petits, que seran *B* i *L*, on *B* és la feina. El vector α_L és fixat de manera que l'objectiu sigui:

$$\frac{1}{2}\alpha_B^{'} Q_{BB}\alpha_B - \left(e_B - Q_{BL}\alpha_L\right)^{'}\alpha_B + \frac{1}{2}\alpha_L^{'} Q_{LL}\alpha_L - e_B\alpha_L$$

Tot seguit, es resol un subproblema respecte de α_B, *B* és actualitzat en cada iteració (es pot observar que per simplificar la notació s'utilitza *B* en comptes de B^k), i el decreixement estricte de la funció objectiu es sosté.

4.3.5.2 Selecció del Conjunt de Treball i Criteri de Parada

Una de les parts important de l'algorisme de descomposició és la selecció del grup de treball *B*. La condició de Karush-Kuhn Tucker (KKT) a la equació [21], mostra que existeix un escalar i dos vectors no negatius λ i μ, tals que:

$$Q\alpha + e + by = \lambda - \mu$$
$$\lambda_i\alpha_i = 0,\ \mu_i(C-\alpha)_i = 0$$
$$\lambda_i \ge 0,\ \mu_i \ge 0,\ \forall i$$

Equació 23

Es pot observar que si s'escriuen les condicions de KKT per el primari i el dual, resulten ser les mateixes, i el multiplicador de Lagrange de la restricció lineal $y'\alpha = 0$ coincideix amb el valor de desplaçament *b* en la funció de decisió. Després, la equació [23] pot re-escriure's com:

$$
\begin{aligned}
Q\alpha + e + by &\geq 0, \text{ si } \alpha = 0 \\
&= 0, \text{ si } 0 < \alpha < C \\
&\leq 0, \text{ si } \alpha = C
\end{aligned}
$$

Ara, utilitzant $y = + - 1, \ \forall i$ i assumint que $C > 0$, s'obté que:

$$
\begin{aligned}
y = 1, \ \alpha_t < C \ &\Rightarrow \ (Q\alpha + e)_t + b \geq 0 \Rightarrow b \geq -(Q\alpha + e)_t = -\nabla f(\alpha)_t \\
y = -1, \ \alpha_t > 0 \ &\Rightarrow \ (Q\alpha + e)_t - b \leq 0 \Rightarrow b \geq (Q\alpha + e)_t = \nabla f(\alpha)_t \\
y = -1, \ \alpha_t < C \ &\Rightarrow \ (Q\alpha + e)_t - b \geq 0 \Rightarrow b \leq (Q\alpha + e)_t = \nabla f(\alpha)_t \\
y = 1, \ \alpha_t > 0 \ &\Rightarrow \ (Q\alpha + e)_t + b \leq 0 \Rightarrow b \leq -(Q\alpha + e)_t = -\nabla f(\alpha)_t
\end{aligned}
$$

On $f(\alpha) = \frac{1}{2}\alpha' Q\alpha + e'\alpha$ i $\nabla f(\alpha)$ és el gradient de *f(α)* en α i considerant:

$$
\begin{aligned}
i &\equiv \text{argmax}(\{-\nabla f(\alpha)_t | y_t = 1, \ \alpha_t < C\}, \{\nabla f(\alpha)_t | y_t = -1, \ \alpha_t > 0\}) \\
j &\equiv \text{argmin}(\{\nabla f(\alpha)_t | y_t = -1, \ \alpha_t < C\}, \{-\nabla f(\alpha)_t | y_t = 1, \ \alpha_t > 0\})
\end{aligned}
$$

Equació 24

De manera que $B = \{i \ , \ j\}$ pot utilitzar-se com un grup de treball per el subproblema en la equació [22] del mètode de descomposició, on i i j són els dos elements que més violen les condicions de KKT. La idea d'utilitzar dos elements com a grup de treball és pressa a partir de l'algorisme d'optimització seqüencial mínima (SMO) [10]. El principal avantatge d'això, és que la solució analítica de la equació [21] pot ser obtinguda sense la necessitat de un programa de optimització comercial. Es pot veure que l'equació [24] és un cas especial del mètode SVM^{light} en [11]. Per ser més exactes, en SVM^{light}, si α és la solució actual del problema, aquest és resolt a continuació:

$$
\begin{aligned}
&\text{mín}_d \nabla f(\alpha)' d \\
&y' d = 0, \ -1 \leq d \leq 1, \\
&d_t \geq 0, \text{ si } \alpha_t = 0, \ d_t \leq 0, \text{ si } \alpha_t = 0 \\
&|\{d_t \ d_t \neq 0\}| = q
\end{aligned}
$$

Equació 25

Es pot observar que $\left|\{d_t d_t \neq 0\}\right|$ és el conjunt de components de d que no són cero. La restricció en la equació [25] implica que la component descendent involucra només q variables. Tot seguit, les components de α amb d_t diferents de cero són incloses en el grup de treball B utilitzat per construir el subproblema a la equació [22]. En efecte, d únicament s'utilitza per identificar B i no

per trobar la direcció de cerca.

Pot ser vist clarament que si $q = 2$, la solució de la equació [25] és:

$$i = \text{argmin}\{\nabla f(\alpha)_t d_t | y_t d_t = 1;\ d_t \geq 0,\ \text{si } \alpha_t = 0;\ d_t \leq 0, \text{si } \alpha_t = C\}$$
$$j = \text{argmin}(\nabla f(\alpha)_t d_t | y_t d_t = -1;\ d_t \geq 0,\ \text{si } \alpha_t = 0;\ d_t \leq 0, \text{si } \alpha_t = C\}$$

Que és igual a la equació [24]i correspon a la segona modificació del algorisme SOM en [12].

Ara, podem definir:

$$g_i \equiv \begin{cases} -\nabla f(\alpha)_i \text{ si } y_i = 1,\ \alpha_i < C \\ \nabla f(\alpha)_i \text{ si } y_i = -1,\ \alpha_i > 0 \end{cases}$$

Equació 26

$$g_j \equiv \begin{cases} -\nabla f(\alpha)_j \text{ si } y_j = -1,\ \alpha_j < C \\ \nabla f(\alpha)_j \text{ si } y_j = 1,\ \alpha_j > 0 \end{cases}$$

Equació 27

De l'equació [24] s'obté que:

$$g_i \leq -g_j$$

Equació 28

El qual implica que α és una solució òptima de la equació [20], de manera que el criteri de parada pot ser escrit o implementat de la següent manera com:

$$g_i \leq -g_j + \epsilon$$

Equació 29

On ϵés una constant positiva de valor reduït.

4.3.5.3 Convergència del Mètode de Descomposició

La convergència dels mètodes de descomposició va ser inicialment estudiada en [13]. Tot i això, en aquesta secció només es tenen en compte els resultats de convergència pel mètode específic de descomposició a partir de [14].

Teorema 1: *Donat qualsevol* $\epsilon > 0$*, després d'un nombre finit d'iteracions la equació [29] serà satisfeta.*

El teorema anterior estableix la anomenada propietat de l'acabament finit, de manera que es té la seguretat de que després d'un nombre finit de passos l'algorisme finalitzarà.

Teorema 2: *Si* $\{\alpha^k\}$ *és la seqüencia generada per l'algorisme de descomposició en l'apartat 4.3.5.1, el límit de qualsevol de les seves subseqüències és la solució òptima de la equació [21].*

El teorema 1 No implica el teorema 2,si es consideren g_i i g_j a l'equació [29] com a funcions de αque no són continues. Per tant no es pot afirmar que qualsevol punt convergent satisfà les condicions de KKT.

El teorema 2 va ser inicialment demostrat com un cas especial dels resultats generals de [15], on es realitzen algunes suposicions. Partint de la demostració de [16], les suposicions són eliminades, i per tant el teorema és completament vàlid.

Considerant la convergència local, degut a que l'algorisme utilitzat és un cas especial de un discussió de [17], s'obté el següent teorema:

Teorema 3: *Si Q és definida positiva i el dual del problema de optimització és degenerat, existeix un c < 1, tal que si k és prou gran:*

$$f(\alpha^{k+1}) - f(\alpha^*) \leq c(f(\alpha^k) - f(\alpha^*))$$

On α és la solució òptima de [21].*

Amb això, aquest mètode de descomposició és linealment convergent. Els resultats mostrats en aquest apartat són vàlids per kernels que poden ser considerats com el producte entre dos vectors de característiques, el qual vol dir que Q és semidefinida positiva.

4.3.5.4 Solució analítica

Amb la selecció del grup de treball a l'apartat 4.3.5.2, la equació [22] es converteix en un problema amb dues variables:

$$\min_{\alpha_i,\alpha_j} \frac{1}{2}[\alpha_i \alpha_j] \begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ji} & Q_{jj} \end{bmatrix} \begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + (Q_{i,L}\alpha_L - 1)\alpha_i + (Q_{j,L}\alpha_L - 1)\alpha_j$$

$$s.a.\ y_i\alpha_i + y_j\alpha_j = 0 \equiv -y'_L\alpha_L^k$$

$$0 \leq \alpha_i,\ \alpha_j \leq C$$

Equació 30

En [8], es substitueix α_i per $y_i\left(-y'_L\alpha_L - y_j\alpha_j\right)$ en la funció objectiu de la equació (2.18) i es resol la minimització sense restriccions respecte a α_i, obtenint la següent solució:

$$\alpha_j^{new} \equiv \begin{cases} \alpha_j + \frac{-G_i - G_j}{Q_{ii}+Q_{jj}+2Q_{ij}} \text{ si } y_i \neq y_j \\ \alpha_j + \frac{G_i + G_j}{Q_{ii}+Q_{jj}-2Q_{ij}} \text{ si } y_i = y_j \end{cases}$$

Equació 31

$$\text{On } G_i \equiv \nabla f \alpha_i \text{ y } G_j \equiv \nabla f(\alpha)_j$$

Si aquest últim valor està fora de la possible regió per α_i, el valor en la equació [31] és arrodonit i assignat a α_j^{new}. Per exemple, si $y_i = y_j$ i $C \leq \alpha_i + \alpha_j \leq 2C$, α_j^{new} s'ha de satisfer:

$$L \equiv \alpha_i + \alpha_j - C \leq \alpha_j^{new} \leq C \equiv H$$

De manera que el màxim valor per α_j^{new} i α_i^{new} és C. Per tant, tenim que:

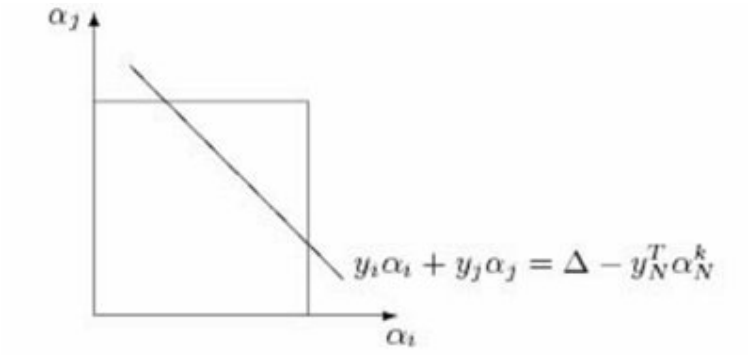
$$\alpha_j + \frac{G_i + G_j}{Q_{ii} + Q_{jj} - 2Q_{ij}} \leq L$$

Llavors, $\alpha_i^{new} = L$ i tenim que:

$$\alpha_i^{new} = \alpha_i + \alpha_j - \alpha_j^{new} = C$$

Equació 32

Veiem aquest procés a la figura [4.12], en la que s'optimitza una funció quadràtica sobre un segment de recta. El segment de recta és la intersecció entre la restricció lineal $y_i \alpha_i + y_j \alpha_j$ i les restriccions acotades $0 \leq \alpha_i$ i $\alpha_j \leq C$.



Il·lustració4.12: Solució analítica d'un problema d'optimització de dues variables.

Tot i això, la igualtat a l'equació [32] podria no mantenir-se si la operació de punt flotant

causes que $\alpha_i + \alpha_j - \alpha_j^{new} = \alpha_i + \alpha_j - (\alpha_i + \alpha_j - C)$, el qual és diferent de *C*. En la majoria dels casos, una petita tolerància ε_α és definida, de manera que tot $\alpha_i \geq C - \varepsilon_\alpha$ és una cota superior i $\alpha_i \leq \varepsilon_\alpha = 0$. Això últim és necessari, ja que algunes dades podrien ser considerades erròniament com a vectors de suport. Cal dir que el càlcul del valor de desplaçament també necessita correcció per aquells valors lliures de α_i ($0 \leq \alpha_i \leq C$).

A [18], és pot veure que si tots els α_i obtenen els seus valors mitjançant assignacions directes, no cal utilitzar un valor de tolerància. Per ser més precisos, en una operació de punt flotant, si $\alpha_i \leftarrow C$ és assignat, una futura comparació entre α_i i *C retornarà CERT* sempre que continguin la mateixa representació interna.

Un altre problema que ens apareix és que el denominador de la equació [31] pot ser cero. Quan això succeeix:

$$Q_{ij} = \pm(Q_{ii} + Q_{ij})/2$$

I per tant tenim que:

$$Q_{ii}Q_{jj} - Q_{ij}^2 = Q_{ii}Q_{jj} - (Q_{ii} + Q_{jj})^2/4 = -(Q_{ii} - Q_{ij})^2/a \leq 0$$

Ara, considerant que Q_{BB} és definida positiva, el denominador cero en la equació [31] no és un resultat possible. D'aquí extraiem que aquesta problema només pot succeir quan *Q* sigui singular de 2 x 2. A continuació es discuteixen dues situacions en les que aquesta matriu pot ser singular:

- La funció *Ø* no mapeja les dades en vectors independents en l'espai d'alta dimensionalitat fent que *Q* sigui només semidefinida positiva. Per exemple utilitzant un Kernel lineal o polinomial de baix ordre.

- Alguns kernels tenen una interesant propietat per la qual $Ø(x_i)$ són independents sempre que $x_i \neq x_j$. Un exemple d'això és el Kernel RBF (veure [19]), degut a que en moltes situacions pràctiques alguns x_i són els mateixos, el que implica que les files (o columnes) de *Q* són exactament iguals, i amb això apareix la possibilitat de que Q_{BB} sigui singular.

De qualsevol manera, encara que el denominador de la equació [31] sigui cero, no hi ha problemes numèrics des de que a l'equació [20] es pot veure que:

$$g_i + g_j \geq \epsilon$$

I en el procés d'iteració:

$$g_i + g_j = \pm(-G_i - G_j), \text{ si } y_i \neq y_j, \ (y)$$
$$g_i + g_j = \pm(G_i - G_j), \text{ si } y_i = y_j$$

Si la matriu de Kernel no es semidefinida positiva, $Q_{ii} + Q_{jj} \pm Q_{ij}$ pot no ser positiva. Llavors, la equació [31] pot no produir una actualització de mode que el valor objectiu sigui disminuït. A més, l'algorisme pot mantenir-se en un sol punt en un cicle infinit. Aquest problema s'estudia en detall a [20], que proposa la següent modificació:

$$\alpha_j^{new} \equiv \begin{cases} \alpha_j + \frac{-G_i - G_j}{\text{màx}(Q_{ii}+Q_{jj}+2Q_{ij},0)} \text{ si } y_i \neq y_j \\ \alpha_j + \frac{G_i + G_j}{\text{màx}(Q_{ii}+Q_{jj}-2Q_{ij},0)} \text{ si } y_i = y_j \end{cases}$$

Així es garanteix el decreixement estricte de la funció objectiu.

4.3.5.5 Càlcul de b

Després de trobar la solució α al problema de optimització, la variable b ha de ser calculada per ser utilitzada en la funció de decisió. Les condicions de KKT de la equació [21]han estat mostrades a la equació [24]. Ara, per el cas $y = 1$, si existeixen α_i que compleixin $0 \leq \alpha_i \leq C$, llavors fem $r_1 = \nabla f(\alpha)_i$. Per evitar errors numèrics, es promitgen com:

$$r_1 = \frac{\sum_{0 \leq \alpha_i \leq C, y_i = 1} \nabla f(\alpha)_i}{\sum_{0 \leq \alpha_i \leq C, y_i = 1} 1}$$

Per altre banda, si no existeix α_i, $r_1$ha de satisfer:

$$\max_{\alpha_i = C, y_i = 1} \nabla f(\alpha)_i \leq r_1 \leq \min_{\alpha_i = 0, y_i = 1} \nabla f(\alpha)_i$$

On $r_1$pren el punt mig del rang. Per y_i= -1, r_2 es calcula de manera similar i obtenim la expressió següent:

$$-b = \frac{r_1 - r_2}{2}$$

Veiem que les condicions de KKT poden ser escrites com:

$$\max_{\alpha_i > 0, y_i = \pm 1} \nabla f(\alpha)_i \leq \min_{\alpha_i < C, y_i = \pm 1} \nabla f(\alpha)_i$$

De manera que el següent criteri de parada pot ser utilitzat pràcticament, i per tant l'algorisme de descomposició per a ell en la iteració α satisfà:

$$\begin{aligned}\text{máx}(&- \min_{\alpha_i<C, y_i=1} \nabla f(\alpha)_i + \max_{\alpha_i>0, y_i=1} \nabla f(\alpha)_i, \\ &- \min_{\alpha_i<C, y_i=-1} \nabla f(\alpha)_i + \max_{\alpha_i>0, y_i=-1} \nabla f(\alpha)_i) < \epsilon\end{aligned}$$

On ε > 0 és una constant escollida com a tolerància de parada.

4.3.5.6 Contracció

Considerant que en molts dels problemes pràctics, el nombre de vectors de suport lliures ($0 \le \alpha_i \le C$) és petit, la tècnica de contracció redueix la mida del problema de treball sense considerar algunes variables acotades. En un punt proper al final del procés iteratiu, el mètode de descomposició identifica un possible conjunt *A* de manera que tots els vectors de suport lliures queden continguts en ell. Per això, el següent teorema mostra que en les iteracions finals de la descomposició proposta en la secció 4.3.5.2 només les variables corresponents a un conjunt petit tenen possibilitat de moure's, com veiem a [21].

Teorema 4: *Si* $\lim_{k\to\infty} \alpha_k = \bar{\alpha}$ *per el teorema 2, llavors* $\bar{\alpha}$ *és una solució òptima. Fins i tot quan k és prou gran, només els elements:*

$$\begin{aligned}\{t | -y_t \nabla f(\bar{\alpha})_t &= \text{máx}(\max_{\bar{\alpha}_i<C, y_i=1} -\nabla f(\bar{\alpha})_i, \max_{\bar{\alpha}_i>0, y_i=-1} \nabla f(\bar{\alpha})_i) \\ &= \text{mín}(\min_{\bar{\alpha}_i<C, y_i=-1} \nabla f(\bar{\alpha})_i, \min_{\bar{\alpha}_i>0, y_i=1} -\nabla f(\bar{\alpha})_i)\end{aligned}$$

Encara poden ser modificats.

Per tant, es sol pensar que si la variable α_i és igual a *C* per algunes iteracions, al final de la solució, aquesta es manté com a cota superior. D'aquí que en comptes de resoldre tot el problema de la equació (2.17), es treballi amb una de menor mida:

$$\begin{aligned}&\min_{\alpha_A} \frac{1}{2}\alpha_A' Q_{AA}\alpha_A - (e_A + Q_{AL}\alpha_L^k)'\alpha_A \\ &s.a.\ y_A'\alpha_A = -y_L'\alpha_L^k \text{ y } 0 \le (\alpha_A)_i \le C,\ \forall i\end{aligned}$$

Equació 33

On $L = \{1,...,N\} \backslash A$.

Tot i això, aquesta demostració pot fallar si la solució de la equació [33] no és una part corresponent a la de la equació [21]. Quan això succeeix, el problema complert es torna a optimitzar des d'un punt on α_B és una solució òptima de la equació [33] i α_L són variables acotades identificades abans del procés de contracció. Es pot veure que mentre que s'està solucionant el problema de la

contracció només es coneix el gradient $Q_{AA}\alpha_A + Q_{AL}\alpha_L + e_A$ de la equació [33]. Considerant això últim, quan s'optimitza de nou el problema de la equació [21], s'ha de reconstruir completament el gradient de $f(\alpha)$, el qual és bastant costós en termes computacionals. Per evitar-ho, en comptes d'iniciar el procés de contracció al final del procés iteratiu, s'inicia des del principi com veiem a continuació:

- Després de cada *mín*(*N*,1000) iteracions s'intenten contreure algunes variables. Així, durant el procés iteratiu:

$$\begin{aligned} &\text{mín}(\{\nabla f(\alpha_k)_t | y_t = -1, \alpha_t < C\}, \{-\nabla f(\alpha_k)_t | y_t = 1, \alpha_t > 0\}) = -g_{ii} \\ &< \text{máx}(\{-\nabla f(\alpha_k)_t | y_t = 1, \alpha_t < C\}, \{\nabla f(\alpha_k)_t | y_t = -1, \alpha_t > 0\}) = g_{jj} \end{aligned}$$

L'equació [28] encara no es satisfà. Llavors, es suposa que si $g_i \leq -g_{ii}$ de l'equació [26] i α_t estan dintre del marge, és molt possible que α_t no torni a canviar, i per tant es desactiva la variable. De la mateixa manera per $-g_j \geq g_{jj}$ de la equació [27] amb α_t dins del marge. D'aquesta manera, el conjunt *A* de variables actives és dinàmicament reduït cada {*L,* 1000} iteracions.

- Està clar que l'estratègia de contracció mencionada anteriorment és molt agressiva considerant que el mètode de descomposició té una convergència lenta i una gran quantitat de les iteracions són consumides per arribar al dígit final de precisió requerit, no és desitja que es perdin iteracions innecessàriament degut a una contracció errònia. Amb això, quan el mètode descomposició arriba primer la tolerància $g_i \leq -g_j + 10_\varepsilon$, el gradient complet és reconstruït. Després, basats en la informació correcte, s'utilitzen les equacions [26] i [27] per desactivar algunes variables i continuar amb el mètode de descomposició.

Com que la mida del conjunt *A* és dinàmicament reduït, per dissimular el cost computacional del gradient $\nabla f(\alpha)$, durant les iteracions es manté sempre:

$$\bar{G}_i = C \sum_{\alpha_j = c} q_{ij}, \ \forall i$$

Així, per el gradient $\nabla f(\alpha)_i$ amb $i \ni A$ s'obté:

$$\nabla f(\alpha)_i = \sum_{i=1} Q_{ij}\alpha_j = \bar{G}_i + \sum_{0<\alpha_j<C} Q_{ij}\alpha_j, \ \forall i$$

4.3.5.7 Caching

Una altre tècnica per reduir el cost computacional és el *caching*. Si tenim en compte que Q és molt pesada i no pot ser guardada a la memòria de l'ordinador, els elements Q_{ij}són calculats només quan és necessari. Després, utilitzant la idea de emmagatzemament de caché es poden guardar els element de Q_{ij} recentment utilitzats [11], fent que el cost computacional de les iteracions posteriors sigui menor.

El teorema 4 Suporta l'ús de caching degut a que en les iteracions finals, només algunes columnes de la matriu Q segueixen sent necessàries, de manera que si el cache conté aquestes columnes, es poden evitar la majoria de les avaluacions de Kernel per aquesta etapa.

Per a la implementació pràctica, s'utilitza una estratègia simple consistent en guardar dinàmicament només les columnes més recentment utilitzades de la matriu Q_{AA}de la equació [33].

4.3.5.8 Complexitat Computacional

La discussió en la secció 4.3.5.3 és sobre la convergència global asimptòtica del mètode de descomposició. En adició, la convergència lineal (teorema 3) és una propietat de la taxa global de convergència. Tot seguit discutim la complexitat computacional del mètode.

La major quantitat d'operacions resideixen en el càlcul de $Q_{BL}\alpha_L^k + e_B$ i la actualització de $\nabla f\left(\alpha^k\right)$a $\nabla f\left(\alpha^{k+1}\right)$. Es pot observar que $\nabla f\left(\alpha\right)$ és utilitzada tant en la selecció del grup de treball com en la condició de la parada, de manera que pot considerar-se tot junt com:

$$Q_{BL}\alpha_L^k + e_B = \nabla f(\alpha^k) - Q_{BB}\alpha_b^k$$

Equació 34

$$\nabla f(\alpha^{k+1}) = \nabla f(\alpha^k) + Q_{:,B}(\alpha_b^{k+1} - \alpha_B^k)$$

Equació 35

On $Q_{:,B}$ és la submatriu de Q amb índex en B. Això és, en la k-ésima iteració amb $\nabla f\left(\alpha^k\right)$ conegut i la part dreta de l'equació [34] com a constructor del subproblema. Després de que el subproblema és resolt, l'equació [35] és utilitzada per obtenir el proper $\nabla f\left(\alpha^{k+1}\right)$. Com que B conté només 2 elements i resoldre el subproblema és fàcil, el cost substancial apareix en el càlcul de $Q_{:,B}\left(\alpha_b^{k+1} - \alpha_B^k\right)$. La operació en si pren $O(n)$, tot i això, si $Q_{:,B}$no està disponible en el caché i cada operació de Kernel costa $O(n)$, cada columna de $Q_{:,B}$necessita $O(nN)$. De manera que la complexitat és igual a *iteracions x O(n)* o *iteracions x O(2n)* segons el cas, tenint en compte que si s'utilitza contracció, N disminueix gradualment. Malauradament, no es sap gaire sobre la complexitat del nombre d'iteracions. Tot i això, [22] va obtenir alguns resultats interesants, però només per els mètodes de descomposició descrits en [13].

4.3.6 Màquines de Suport Multiclasse

Degut a la naturalesa dicotòmica de SVM, va sorgir la necessitat d'implementar nous mètodes que poguessin resoldre problemes multiclasse. Amb aquest objectiu, s'han proposat diferents aproximacions. Per una banda, com aproximació directa, a [46] es proposa una modificació de la funció d'optimització que té en compte totes les classes:

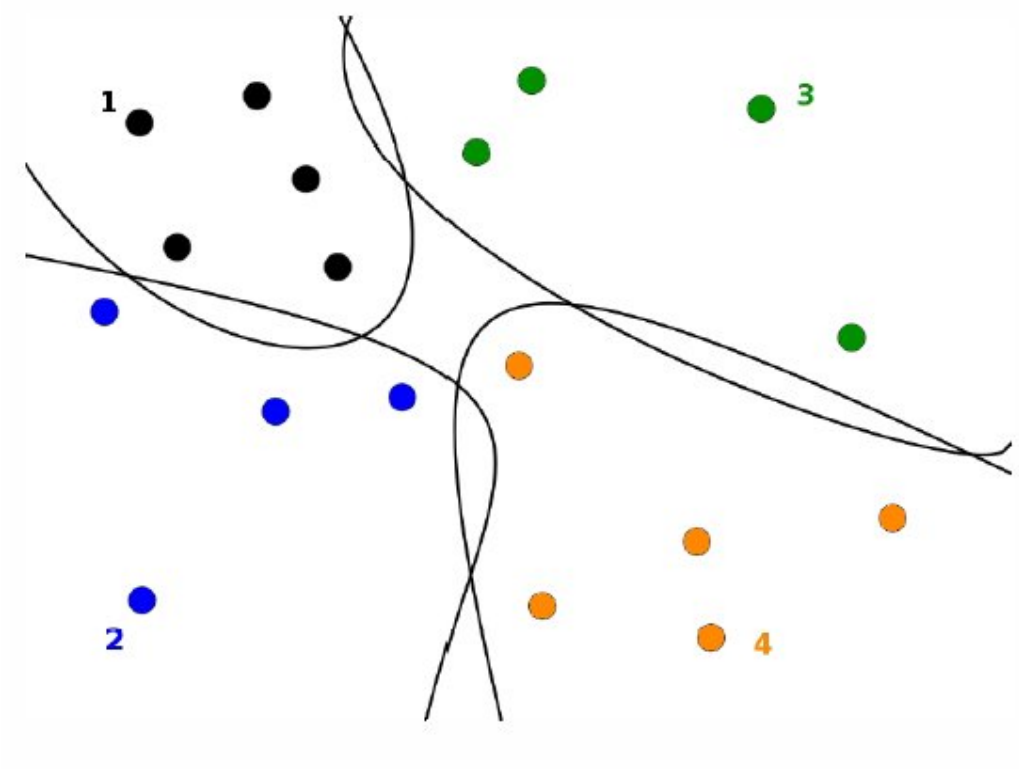
$$\text{mín}\ \frac{1}{2}\sum_{m=1}^{n}||w_m||^2 + C\sum_{i=1}^{l}\sum_{m\neq y_i}\xi_i^m$$

Subjecte a: $w_{y_i}\cdot x_i + b_{y_i} \geq w_m \cdot x_i + b_m + 2 - \xi_i^m, \xi_i^m \geq 0$

Per altre banda, diverses tècniques per a la aproximació a SVM multiclasse de *k* classes s'han basat en la combinació de classificadors binaris [47]:

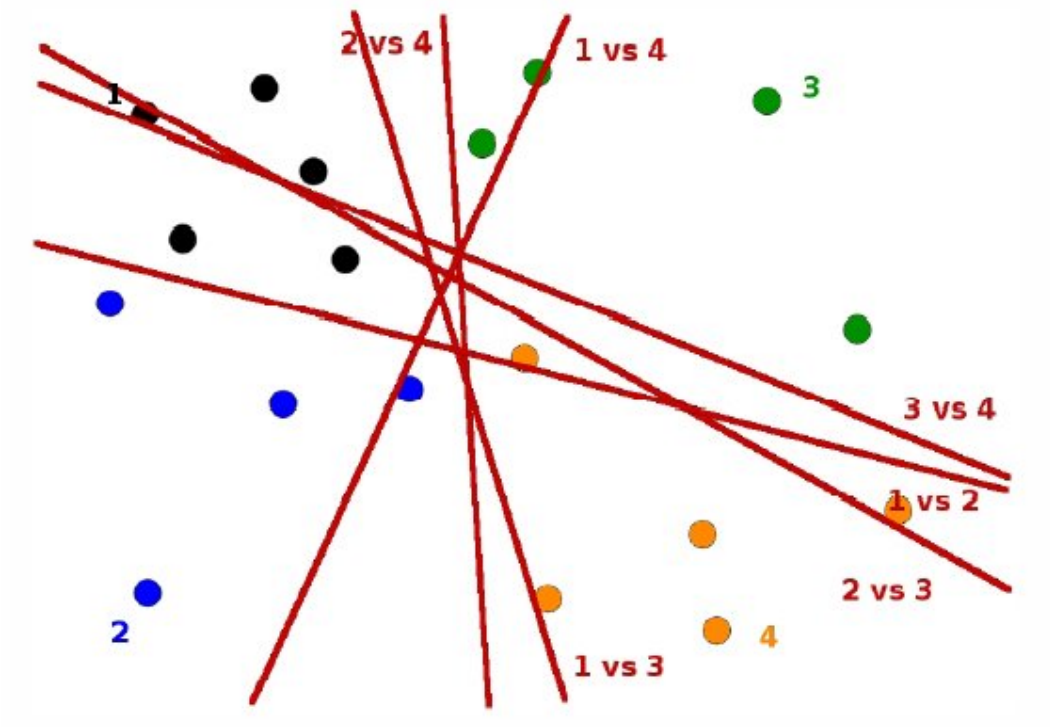
- **One-against-all**: Construeix *k* classificadors que defineixen altres hiperplans que separen la classe *i* del *k-1* restants. Per exemple, per un problema de 4 classes, es creen classificadors *1* vs *2-3-4*, *2* vs *1-3-4*, *3* vs *1-2-4* i *4* vs *1-2-3*. Al rebre noves dades, aquestes són sotmeses als *k* classificadors, escollint com a resultat aquells que maximitzi el marge:

$$\hat{C}_i = \arg\ \underset{i=1,\ldots,k}{\text{máx}}\ (w_i x + b_i)$$



Il·lustració 4.13: Classificació one-against-all

- **One-against-one**: Construeix $\frac{k(k-1)}{2}$ classificadors, un per cada parell de classes possibles, enfrontant així a totes les classes una a una. Per exemple, un problema amb 4 classes generaria els següents classificadors: *1* vs *2*, *1* vs *3*, *1* vs *4*, *2* vs *3*, *2* vs *4* i *3* vs *4*. Un cop realitzat això, es sotmet a cada una de les dades una col·lecció de test a tots aquests classificadors, on s'afegeix un vot a la classe guanyadora en cada cas. Finalment, aquella que més vots obté serà la classe proposta per el sistema.



Il·lustració 4.14: Classificació one-against-one

Existeixen altres aproximacions derivades d'aquestes, entre les que destaca la anomenada *Directly Acyclic Graph SVM (DAGSVM)* [27], que es basa en la aproximació *one-against-one*, però modifica el procés de decisió a la fase de classificació. En ella es crea un algorisme en forma d'arbre de $\frac{k(k-1)}{2}$ nodes, on k nodes són finals. Recorrent tots els nodes d'aquest arbre es decideix la classe a la que pertany la instància.

4.4 COMPARATIVA ENTRE SVM I ARBRES DE DECISIÓ

4.4.1 Estudi realitzat per John William

En aquest apartat veiem les conclusions més interesants d'un estudi realitzat per John William i Jaime López [43], on es va realitzar una comparació entre diferents models de classificació de patrons. Els models implementats van ser el classificador Bayesià, les xarxes neuronals, les màquines de suport de vectors (SVM) i elsarbres de decisió. S'analitzen doncs a continuació els resultats comparatius més destacats entre SVM i els arbres de decisió, les dues tècniques estudiades en aquest projecte.

Si s'analitzen directament els resultats obtinguts per aquest estudi, es pot comprovar que l'error és més elevat en els arbres de decisió, i per contra l'efectivitat és més elevada en SVM.

A més, en aquest estudi s'arriba a la conclusió de que els arbres de decisió no són viables per

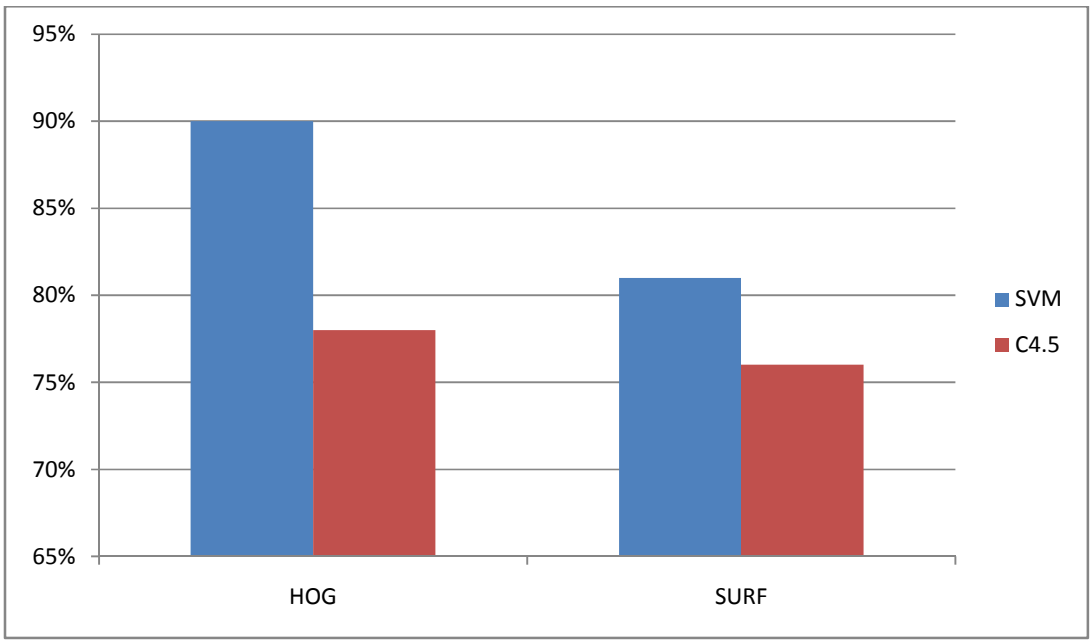
obtenir un bon resultat amb les dades tractades, degut a la seva alta sensibilitat davant conjunts d'entrenament sorollosos, i per tant donen una predicció pobre.

4.4.2 Experiències pròpies

En aquest apartat veiem una comparativa entre els resultats obtinguts en la utilització de l'algorisme amb la tècniques SVM o amb un arbre de decisió del tipus C4.5 (veure apartat 4.2).

Veiem els resultats de l'efectivitat d'aquests mètodes en el nostre algorisme a la figura [4.16]. L'anàlisi ha estat realitzat amb un total de 120 fotografies, i mostra el tant per cent d'encert de cada un d'aquestes tècniques utilitzant els mètodes de detecció i extracció de keypoints HOG i SURF.

Es pot observar a la figura [4.16] que els resultats obtinguts en el cas de SVM són clarament favorables als obtinguts amb C4.5.



Il·lustració 4.16: Comparació de l'efectivitat de SVM i C4.5 en el nostre cas.

Podem observar també que SVM aporta alguns avantatges sobre les altres tècniques de classificació de patrons, tal com s'explica a [45]:

- No es necessita una selecció o reducció de termes. En cas de que una classe es distribueixi en àrees separades de l'espai vectorial, serà la redimensió mitjançant la funció de kernel la que s'encarregui de solucionar-ho.

- No cal realitzar un esforç d'ajust de paràmetres en el cas de problemes linealment separables, ja que disposa del seu propi mètode per a això.

Una altre dels avantatges que ofereix SVM és que la seva transformació a aprenentatge semisupervisat es converteix, generalment, en un comportament transductiu, el que possibilita el màxim refinament de la definició del classificador.

Finalment, i per tots aquest motius, s'ha escollit la tècnica SVM com a mètode de classificació de patrons per davant dels arbres de decisió, ja que apart dels avantatges dels quals disposa, ens ha donat millor resultat amb el tipus de dades tractades en aquest projecte. Per tant, queda justificat l'ús de SVM en aquest projecte.

5 WEKA

5.1 Introducció

La Weka (*Gallirallus australis*) és un au endèmica de Nova Zelanda. Aquesta au dona nom a una extensa col·lecció d'algorismes de Màquines de coneixement o Intel·ligència Artificial desenvolupades per la universitat de Waikato (Nova Zelanda). Aquests algorismes són útils per ser aplicats sobre les bases de dades mitjançant les interfícies que ofereix, o per afegir-los en qualsevol aplicació Java en forma de codi.

Weka són un conjunt de llibreries JAVA per la extracció de coneixements des de bases de dades. És un software que ha estat desenvolupat sota llicencia GLP, cosa que ha impulsat que sigui una de les llibreries més utilitzades en aquesta àrea en els últims anys.

Les darreres versions d'aquesta llibreria inclouen les següents característiques:

- Possibilitat de diverses fonts de dades.
- Interfície visual basada en processos/fluxes de dades (rutes).
- Diferents eines de mineria de dades: regles de associació (a priori, Tertius, ...), agrupació/segmentació/conglomerat, classificació (xarxes neuronals, regles i arbres de decisió, aprenentatge Bayesiona) i regressió (Regressió lineal, SVM, ...).
- Manipulació de dades (pick & mix, mostreig, combinació i separació).
- Combinació de models (*Bagging, Boosting*, ...).
- Visualització anterior (dades en múltiples gràfiques) i posterior (arbres, corbes ROC, corbes de cost,...).
- Entorn d'experiments, amb possibilitat de realitzar proves estadístiques.

Alguns dels motius que han portat a la utilització d'aquest software per davant d'altres algorismes d'intel·ligència artificial són els següents:

- Està disponible sota la llicencia pública general de GNU.
- És molt portable, ja que està completament implementat en Java i pot córrer en casi qualsevol plataforma.
- Conté una extensa col·lecció de tècniques per pre-processament de dades i modelat, que ens permet realitzar les operacions desitjades.

5.2 Fitchers de dades

Nativament, Weka treballa amb un format anomenat *arff,* acrònim de *Attribute-Relation File Format.* Per tant, les dades d'entrada del nostre algorisme han d'estaren un fitxer d'aquestes característiques.Aquest format està compost per unaestructura clarament diferenciada en tres parts:

1. **Capçalera:** Es defineix el nom de la relació. El seu format és el següent:

   @relation<nom-de-la-relació>

   On <nom de la relació>és de tipus String (com String ofert per Java). Si aquest nom conté algun espai serà necessari expressar-lo entre cometes.

2. **Declaracions d'atributs:** En aquesta secció es declaren els atributs que compondran el nostre arxiu junt amb el seu tipus. La sintaxi és la següent:

   @attribute <nom-de-l'atribut><tipus>

   On<nom-de-l'atribut>és de tipus String tenint les mateixes restriccions que en el cas anterior. Weka accepta diversos tipus, que són:

   a) **NUMERIC:** Expressa nombres reals (Degut a que Weka és un programa Anglosaxó, la separació de la part decimal i entera dels nombres reals es realitza mitjançant un punt en comptes de una coma).

   b) **INTEGER:**Expressa nombres enters.

   c) **DATE:**Expressa dates, per a fer-ho, aquest tipus ha d'anar precedit d'una etiqueta de format entre cometes. La etiqueta de format està composta per caràcters separadors (Guions i/o espais) i unitats de temps com les següents:

      - **dd** Dia.
      - **MM** Mes.
      - **yyyy** Any.
      - **HH** Hores.
      - **mm** Minuts.
      - **ss** Segons.

   d) **STRING:**Expressa cadenes de text, amb les restriccions del tipus String comentades anteriorment.

   e) **ATRIBUT:**L'identificador d'aquest tipus consisteix en expressar entre claus i separats per comes els possibles valors (caràcters o cadenes de caràcters) que pot

prendre l'atribut. Per exemple, si tenim un atribut que indica que el temps podria definir-se com:

@attribute temps {assolellat, plujos, ennuvolat}

3. **Secció de dades.** Declarem les dades que composen la relació separant entre comes els atributs i les relacions amb salts de línia.

```
@data
4, 3.2
```

Encara que aquest és el mode "complet", es possible definir les dades d'una manera abreviada (*sparse data*). Si tenim una mostra en la que hi ha moles dades nul·les o de valor zero, podem expressar les dades sense tenir en compte els elements nuls, arrodonint cada una de les files entre claus i situant davant de cada una de les dades el nombre de l'atribut.

Es pot veure això al'exemple següent:

```
@data
{1 4, 3 3}
```

En aquest cas s'ha prescindit dels atributs 0 i 2 (com a mínim), i s'ha assignat al atribut 1 el valor 4 i al atribut 3 el valor 3.

En el cas de que alguna de les dades sigui desconeguda s'expressarà amb un símbol de tancar interrogació ("?").

Es possible afegir comentaris amb el símbol "%", com passa a C++, que indicarà que des d'aquest símbol fins el final de la línia es tot un comentari auxiliar. Els comentaris poden situar-se en qualsevol lloc del fitxer.

5.3 Paràmetres a configurar

Observem a continuació les línies principals d'un dels programes d'entrenament creats en el nostre software. Aquest programa concret és el del mètode SURF:

"java weka.classifiers.functions.SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0" -t "Descriptors/SURF-Descriptors.arff" -x 10 -d Descriptors/SURFmodSVM.out"

Podem observar com el programa crida a una funció en SMO (també anomenat SVM), que crearà un model d'aprenentatge segons les característiques dels pacients.Per a crear-lo s'ha utilitzat una funció polinomial de Kernel, ja que ens ha donat més bon resultat que la funció radial.

Observem també que tenim com a dades d'entrada l'arxiu de característiques "**Descriptors/SURF-Descriptors.arff**", i que dipositem les dades de sortida (en aquest cas el

model d'aprenentatge) a l'arxiu "**Descriptors/SURFmodSVM.out**".

Veiem a continuació una breu descripció dels paràmetres que s'han configurat en l'arxiu anterior:

-C (double): Constant de complexitat (*default* 1).

-N (0,1,2): 0 = normalització, 1 = estandarització, 2 = cap dels dos anteriors (*default*0).

-E (double): Exponent del kernel polinomial (*default* 1).

-K: Model de Kernel utilitzat en la creació del model.

-L (double): Llindar de tolerància (*default* 1.0e-3).

-P (double): Conjunt d'epsilons per a l'arrodoniment (*default* 1.0e-12).

-V (double): Nombre de cops usat en validació-creuada per a generar els models logístics.(*default* -1, *use training data*).

-W (double): Nombre aleatori de llavors per a la validació creuada.

-d: Indica arxiu de sortida.

-t: Indica arxiu de dades d'entrada.

I dins de la configuració del polinomide Kernel:

-C (double): Mida de la memòria cache.

-E (double): Exponent a utilitzar.

Tot seguit podem veure les línies de codi més rellevants utilitzades en la predicció de resultats. En aquest cas també veiem el cas concret del mètode SURF:

"java weka.classifiers.functions.SMO -T Descriptors/SURF-Descriptors-Pacient.arff -l Descriptors/SURFmodSVM.out -p 0 >SURFsortidaSVM.txt"

Com en el cas anterior, veiem una breu descripció dels paràmetres utilitzats per a configurar l'algorisme:

-T: Indica arxiu de dades d'entrada.

-l: Indica arxiu amb el model d'aprenentatge a utilitzar.

-p: Indica arxiu de dades de sortida.

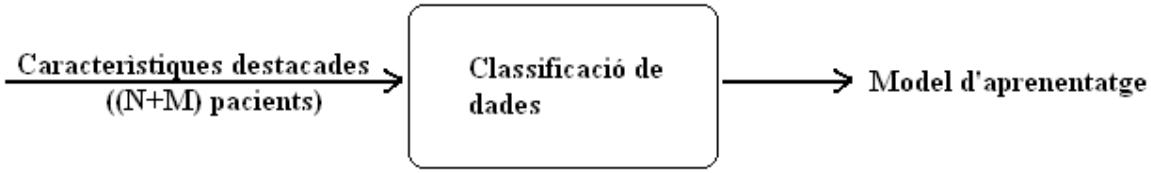
6 FUNCIONAMENT DE LA PART D'INTEL·LIGÈNCIA ARTIFICIAL

Com es pot veure a l'apartat 3, la part de *Data Mining* del problema té dos parts molt diferenciades,com són els processos d'**aprenentatge** i de **diagnosi**. L'objectiu d'aquest apartat és explicar la resolució d'aquests processos a nivell conceptual, explicant amb detall cada un dels passos que es realitzen.

6.1 Aprenentatge

L'objectiu principal d'aquest procés és el de crear un model d'aprenentatge amb les dades dels N pacients que pateixen una malaltia, juntament amb les dels M que no la pateixen. Aquest serà el model que s'utilitzarà en el procés de diagnosi (veure apartat 6.2) per a determinar si un nou pacient dona o no símptomes de patir la malaltia estudiada.

Veiem aquest procés d'aprenentatge a la figura [6.1].



Il·lustració 6.1: Procés d'aprenentatge

Com s'observa, partim de les característiques descriptives més importants dels rostres de gent malalta i de gent sana que disposem en un arxiu en format *.arff*. Es pot veure la metodologia per a l'obtenció d'aquest fitxer de característiques al projecte:**"Detecció de malalties segons els trets facials - Primera part: Tractament d'imatge".**

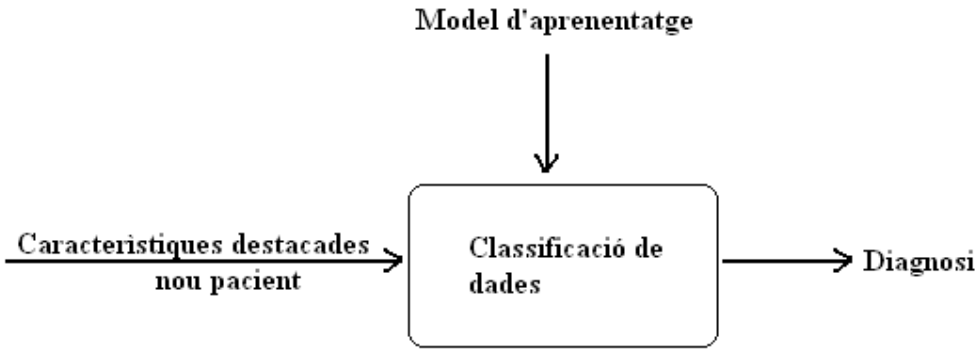
Amb aquest arxiu de característiques, el que es fa és crear un model de classificació basat en SVM (veure apartat 4.3). Aquest model correspon a una classificació de les dades de les que disposem, de manera que aquestes quedin separades en grups, segons si les dades corresponen a les d'una persona que pateixi una de les malalties tractades o a un pacient sa.

Per a crear aquests models s'utilitzen les llibreries de Data Minning de Weka (veure apartat 5). Amb aquestes llibreries, el que s'ha fet és realitzar uns executables en Java, que són cridat des del software principal en C++, i el que fan és entrar l'arxiu en format *.arff* amb totes les dades al programa Weka, que executa una classificació d'aquestes. En finalitzar el procés, Weka crea un arxiu en format *.out* amb el model de característiques desitjat.

6.2 Diagnosi

En aquest procés, es parteix d'un sol vector de característiques descriptives del rostre d'un nou pacient, al qual s'aplica una predicció per determinar si dona o no senyals de patir la malaltia estudiada.

Veiem el procés de diagnosi a la figura [6.2].



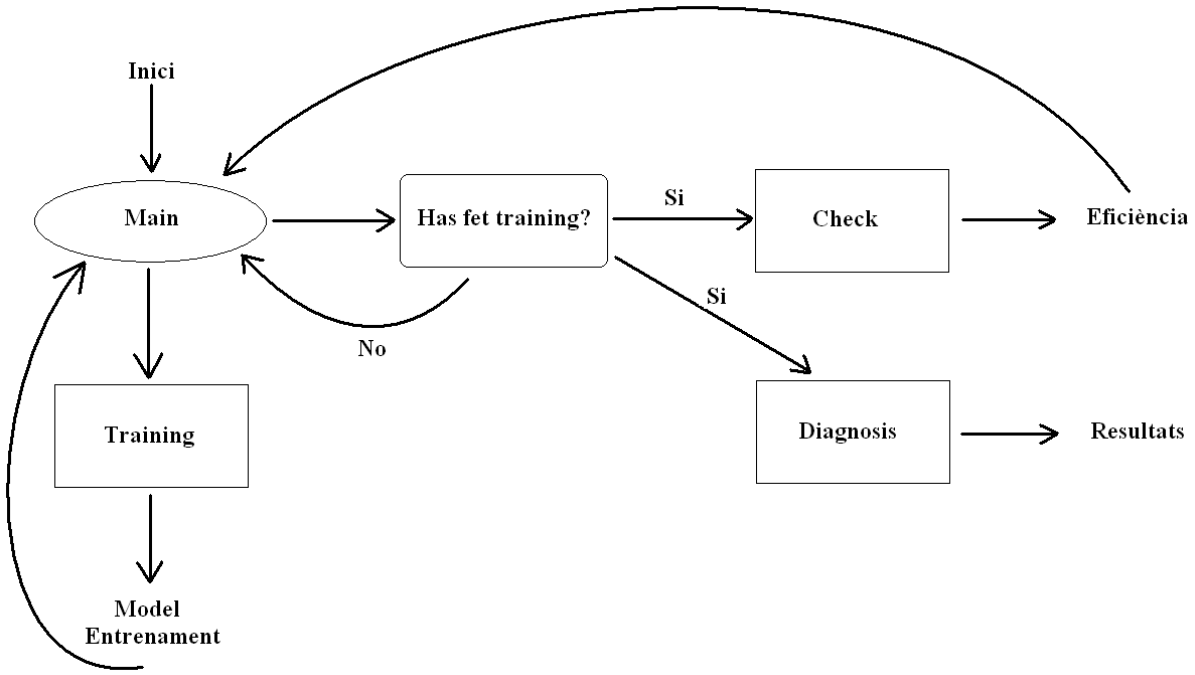
Il·lustració 6.2: Procés de diagnosi.

Observem que en aquest procés partim de les característiques calculades en la imatge del pacient a diagnosticar, també en un arxiu en format *.arff*, i del model d'entrenament calculat anteriorment. Amb aquest model, el que es fa és utilitzar també la llibreria en Java Weka (veure apartat 5) per comprovar si les dades del pacient es poden classificar a amb les dades de gent malalta o si pel contrari s'han de classificar amb les dades de gent sana.

Si les dades del pacient es classifiquen amb les dades de la gent malalta, considerem que el pacient dona senyals de que pot patir aquella malaltia, i per tant s'informarà a l'usuari d'aquesta possibilitat. Si per el contrari les dades es classifiquen amb les dades de la gent sana, considerem que l'usuari no dona senyals de patir la malaltia en qüestió.

Per a fer aquesta predicció, de la mateixa manera que en el procés d'entrenament, el que s'ha fet és crear un sèrie d'executables en Java, que cridats des del programa principal en C++, seleccionen com a dades d'entrada el fitxer *.arff* i el model (*.out*) creat en el procés d'entrenament. Aquests executables en Java configuren i executen les llibreries Weka, que prediuen si el pacient en qüestió pot patir o no la malaltia tractada, objectiu inicial del projecte. Per acabar, els resultats d'aquesta predicció són mostrats per pantalla a l'usuari.

7 FUNCIONAMENT DEL SOFTWARE A NIVELL D'USUARI



Il·lustració 7.1: Diagrama de flux del programa.

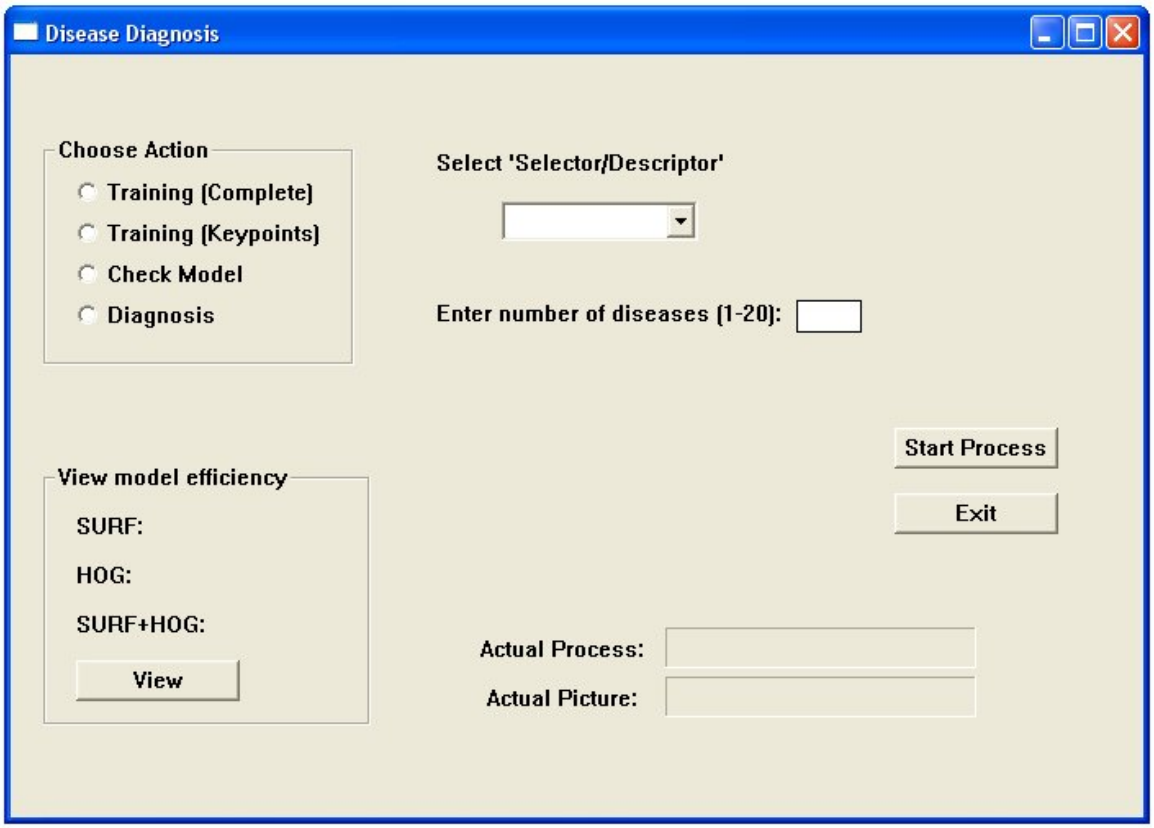
Com es veu a l'apartat 3 i a la figura [9.1], el programa està realitzat en C++, i es divideix en nombrosos blocs, cadascun dels quals consta de una o vàries funcions. En aquest apartat es procedirà a una descripció interna dels blocs més importants que constitueixen el programa. No es mostrarà el codi pertanyent a cada funció, el que es pretén és donar una visió detallada de com s'ha estructurat el programa de detecció de malalties. Tampoc pretén ser una guia d'usuari per al funcionament d'aquest programa, ja que per això ja es disposa d'una guia apart. Així doncs, el que s'intenta en aquest apartat és analitzar el funcionament "tècnic" de l'algoritme en totes les seves accions per al diagnosis de malalties.

7.1 Main (programa principal)

Aquest és el bloc més complex i important del que es disposa. La funció MAIN és el programa de detecció de malalties en sí, i com a programa principal, realitza totes les tasques i crides a altres funcions necessàries per el correcte funcionament del sistema. També interactua amb els diferents processos dels que consta, inicialitzant paràmetres al inici i alliberant-los al finalitzar l'execució del programa.

El programa està construït sobre la família de funcions "Win32" per a la creació d'una interfície gràfica o GUI, és a dir, que està construït sobre les "WinApi" de Windows. Aquesta interfície gràfica ha estat creada per facilitar l'ús d'aquest programa per les persones amb pocs coneixements informàtics.

Així doncs, el primer que realitza el programa principal és inicialitzar totes les variables i crear la pantalla principal de l'aplicació, que té un aspecte com el que es pot veure a la figura [9.2].



Il·lustració 7.2: Menú principal del software.

Fet això el programa entra en un bucle esperant una acció per part de l'usuari.

Com s'ha vist a l'apartat 4, el programa consta de dues parts o accions molt diferenciades. En primer lloc, tenim la part d'entrenament o *training* del sistema, i en segon lloc la part de chequeix del model i diagnosis de malalties pròpiament dita. El programa principal cridarà aquestes funcions segons les esculli l'usuari.

El nostre programa és basa en un procés de *Data Minning* per a la classificació i predicció de nous casos. Per tant, el primer que s'ha de fer abans de predir noves malalties, és crear un model d'aprenentatge amb dades de gent que pateixi o no les malalties a estudiar. A l'hora de crear aquest model es podran escollir 3 mètodes: SURF, HOG i una combinació d'aquests dos. Així doncs, el primer que es farà és crear el model amb un d'aquests mètodes esmentats. A continuació ja es podran predir noves malalties amb aquest model, tot i que el més recomanable serà comprovar la seva eficàcia o eficiència prèviament, estant així segurs del bon funcionament del model creat.

Tot seguit es procedeix a detallar els blocs esmentats, que realitzaran les opcions disponibles.

7.2 Training

Com s'ha comentat, el primer que es necessita és un bon model d'aprenentatge del sistema. Per tant, el primer que es farà és escollir un cert nombre de fotos de gent que pateixi o no les malalties a estudiar, i es col·locaran a la carpeta corresponent. L'objectiu serà trobar els punts més descriptius dels rostres dels pacients, i calcular una sèrie de descriptors de característiques al voltant d'aquests punts, per a poder crear el model d'entrenament anteriorment esmentat.

El primer que realitza aquest bloc d'entrenament és analitzar totes aquestes fotos una per una, cercant el rostre del pacient a cadascuna d'elles. Un cop localitzats els rostres de cada fotografia, es demana a l'usuari que realitzi una inspecció visual dels resultats per a comprovar que són satisfactoris, i un cop obté la seva aprovació es continua amb el procés.

Tot seguit, i un cop s'han normalitzat aquests rostres, el que es fa és trobar les seves zones més importants (Com són els ulls, el nas, la boca i les galtes), i partint d'aquestes zones, es procedeix a extreure les seves característiques més destacades. Aquestes característiques es col·loquen en forma de vector numèric en un arxiu en format *arff*.

En aquest procés podem escollir el tipus de mètode que s'utilitzarà, ja sigui SURF, HOG o una combinació d'aquests dos. Aquests mètodes són explicats amb detall a l'apartat 4.

Realitzat aquest procés de cerca de punts clau i extracció de característiques, es procedeix a executar un algorisme en Java de classificació basat en el software d'intel·ligència artificial "Weka", que crea el model d'aprenentatge desitjat en format *.out*.

7.3 Chek

Un cop realitzat el model d'aprenentatge, el programa ja es pot utilitzar per predir malalties en nous pacients, però el més recomanable abans de realitzar-ho és fer un chequeig del model creat, comprovant així si aquest ens dona uns resultats satisfactoris.

Per a realitzar aquest procés de chequeig, s'han d'escollir un cert nombre de fotos de gent que pateixi les malalties estudiades i de gent sana, però cal que sigui gent diferent a la utilitzada per crear el model d'entrenament, ja que sinó els resultats d'aquest procés no serien reals ni fiables.

Així doncs, el que fa aquest bloc és agafar cadascuna de les fotos esmentades, i en cerca els punts més descriptiusdel seus rostres. Un cop es disposa d'aquests *keypoints*, en calculen els descriptors de característiques, de la mateixa manera que en el procés d'entrenament, i s'aplica un algorisme en Java de classificació d'informació basat en el software d'intel·ligència artificial "Weka", que prediu si les característiques del pacient s'assemblen més a les de la gent que pateix la malaltia o a les de la gent que no la pateix.

Com que el programa sap el resultat d'aquesta predicció abans de realitzar-la (coneixem amb anterioritat si les fotos usades per testejar el model pertanyen a pacients sans o malalts), el software analitza el % d'encert del seu model, i el mostra per pantalla a l'usuari. Amb això s'aconsegueix comprovar l'efectivitat de cadascun dels models creats en l'algorisme.

7.4 Diagnosis

Un cop es disposa del model d'entrenament, ja es pot procedir a realitzar un diagnosis de la malaltia estudiada ens els nous pacients. Cal tenir en compte que per realitzar-ho, s'ha de portar a terme amb una de les tècniques que s'han usat per crear els models comentats, ja siguin SURF, HOG o una combinació dels dos. És a dir, que si per exemple s'ha creat un model d'aprenentatge només per mètode de HOG, també podrem realitzar un diagnosis de malalties seguint aquest model, però no podrem predir malalties amb SURF.

Així doncs, el que fa aquest bloc es un procés semblant al del chequeig del model, però amb la diferència de que ara només treballem sobre una sola fotografia, i a més ara no sabem el resultat de la predicció abans de realitzar-la. Així doncs, el que es realitza és agafar la fotografia i localitzar-hi el rostre del pacient. Un cop localitzat, es busquen les zones més importants d'aquest, i se'n calculen els descriptors de característiques, que s'emmagatzemen com a vector numèric en un arxiu en format *.arff*.

Tot seguit, es procedeix a aplicar un algoritme de classificació de dades basat en el model d'aprenentatge creat. Per a fer-ho, s'executa un algorisme en Java, que realitza una crida al software d'intel·ligència artificial "Weka". Feta aquesta crida, se'ns mostra per pantalla si les característiques de l'usuari s'assemblen més a les dels pacients que pateixen la malaltia o a les dels pacients sans. És a dir, s'informa a l'usuari si el pacient dona indicis de patir o no una de les malalties estudiades.

8 RESULTATS

En aquest apartat es mostren els resultats obtinguts en l'execució de l'algorisme creat, per cada una de les tècniques de detecció i extracció de característiquesde les que es disposa. A més, es comprova l'eficàcia del software creat. Per a fer-ho, en primer lloc s'explica el procés de validació de cadascuna d'aquestes tres tècniques, i posteriorment, es veuen els resultats que ens proporcionen, i es compara el cost de cadascuna d'elles, a partir dels seus temps de processat.

Abans d'analitzar els resultats, ens plantegem de quina manera s'avaluen normalment aquest tipus d'algorismes.

Al tractar-se d'un software de diagnòstic assistit per ordinador, el més important o el que ens mostrarà si hem obtingut uns bons resultats, és precisament el fet que aquest sigui capaç d'encertar si un pacient pot patir o no una malaltia. Per tant, per avaluar l'efectivitat dels models creats, el que s'ha de fer és partir d'una base de dades de gent que pateix aquesta malaltia, i aplicar un procés de chequeig en que el software provarà d'encertar si el pacient està malalt o no. El software creat en aquest projecte disposa d'un procés de chequeig intern (com podem veure a l'apartat 7), on es comprova la fiabilitat dels models creats. Així, el tant per cent d'encert que obtingui el programa ens mostrarà l'efectivitat d'aquest per al seu ús en el diagnosis de la malaltia estudiada.

Ara bé, per a la creació dels models d'aprenentatge desitjat, ens trobem amb el problema de reunir un bon nombre de fotografies de gent que pateixi una determinada malaltia.

En la societat en que vivim, un dels valors més importants és la privacitat i dret a la intimitat de les persones. Degut a la irrupció d'aquest tipus de tècniques de diagnosis basades en imatges dels pacients, cada cop és més freqüent que els hospitals demanin autorització per realitzar fotografies de gent que pateixi una certa malaltia, per poder-les utilitzar en softwares d'aquest tipus. Així doncs, aquest prototip d'algorisme de diagnosis ha estat creat pensant en un breu futur, on es disposin de completes bases de dades de gent que pateixi tot tipus de malalties per estudiar.

En el moment de la finalització d'aquest projecte tot just s'estaven començant a recol·lectar fotografies de gent malalta a l'hospital**Sant Joan de Déu**. Tot i aquests avanços, ha estat impossible aconseguir un bon nombre de fotografies de gent que pateixi una malaltia concreta per a testejar el programa. A més, l'objectiu d'aquest projecte és el de resoldre un problema "tècnic", per l'extracció i la predicció de característiques facials, no una recerca mèdica sobre quines malalties es poden detectar gràcies a aquestes característiques. Com es veu a l'apartat 10, un bon punt de partida a partir de la feina feta en aquest projecte és precisament valorar quines malalties es poden identificar a través dels trets facials de les persones.

Així doncs, el que s'ha fet per a poder testejar el software és partir d'un gran nombre de fotos amb característiques ben diferenciades entre elles, com són fotos de **homes** i de **dones**. Amb aquestes fotografies s'han creat els models d'entrenament, i s'ha comprovat l'efectivitat del software en cada cas.

Procedim doncs a veure la creació dels models d'aprenentatge utilitzats, així com dels resultats aconseguits.

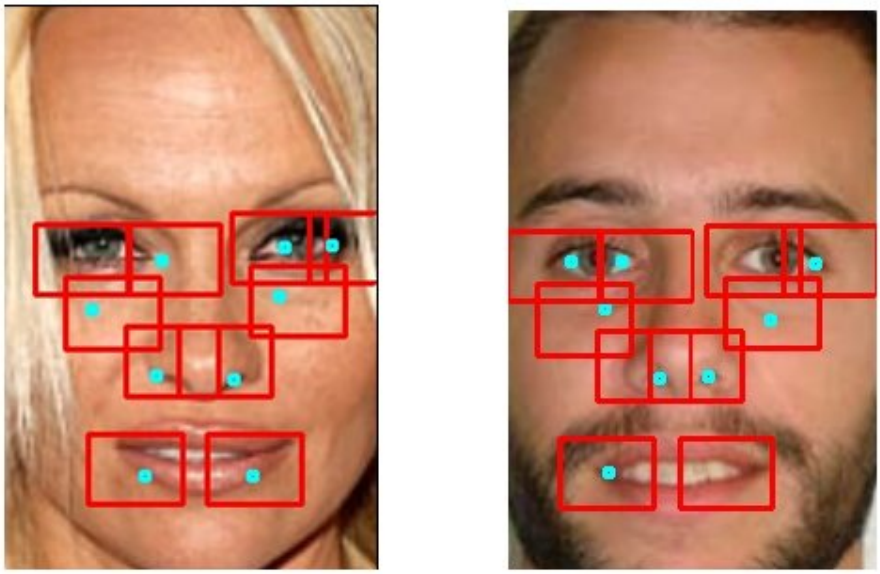
Per a la creació del model d'entrenament del programa s'han utilitzat 225 fotografies de homes i 225 de dones, col·locant-los respectivament en les carpetes de **"Sick Faces"** i **"Healthy Faces"** de les que disposem.

Un cop fet això, s'ha procedit a crear els models d'entrenament seguint les tres tècniques de les que disposa el programa, és a dir, SURF, HOG i una combinació d'aquests dues.

El primer que realitza el programa és una cerca de la part central dels rostres de les fotografies, que guarda normalitzats a les carpetes **"Sick Faces Output"** i **"Healthy Faces Output".**

Un cop tenim localitzats els rostres de les fotografies, s'ha procedit a cercar els punts més característics en cadascuna d'elles. Sobre aquests punts calcularem posteriorment els valors numèrics descriptius de cada imatge, i per tant és necessari que es corresponguin amb les zones més importants de cada un dels rostres. A més, cal que sempre es detectin els mateixos punts, ja que és important tenir els valors descriptius dels mateixos punts de les cares per poder comparar-los entre ells.

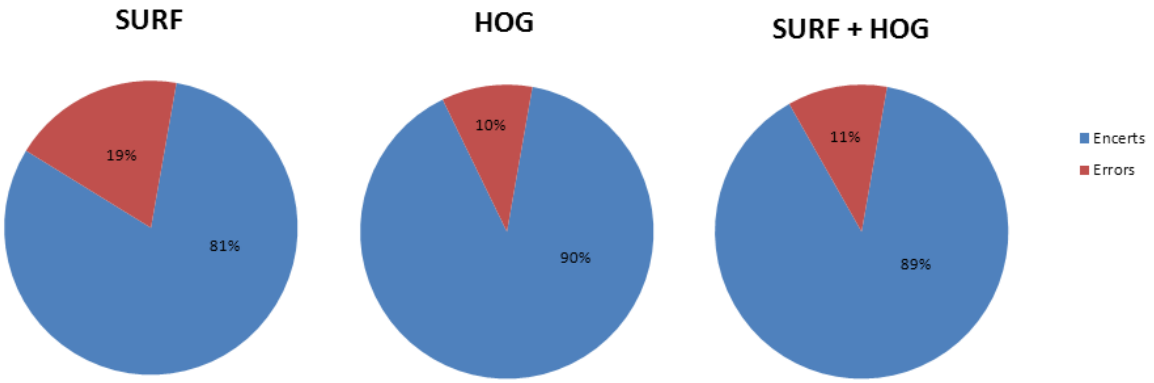
Així doncs, a la figura [8.1] podem veure una mostra les zones de treball trobades en aquestes imatges. Els punts marcats en blau corresponen als *keypoints* trobats amb el mètode SURF, i els requadres vermell a les zones de treball del el mètode HOG.



Il·lustració 8.1: Exemple de les zones d'interès trobades als rostres dels pacients.

Un cop trobats els punts més descriptius de les fotografies, s'ha procedit a extreure les seves característiques més importants, i posteriorment s'ha creat el model d'aprenentatge desitjat. Aquests processos es realitzen amb les funcions de les que disposa el software, explicades amb detall a l'apartat 7.

Un cop obtinguts els tres models d'entrenament, s'ha procedit al seu chequeig. Com també es veu a l'apartat 7, el software disposa d'una opció de testeig automàtic per a realitzar-ho. Els resultats d'efectivitat que s'han obtingut són els que es mostren a la figura [8.2].



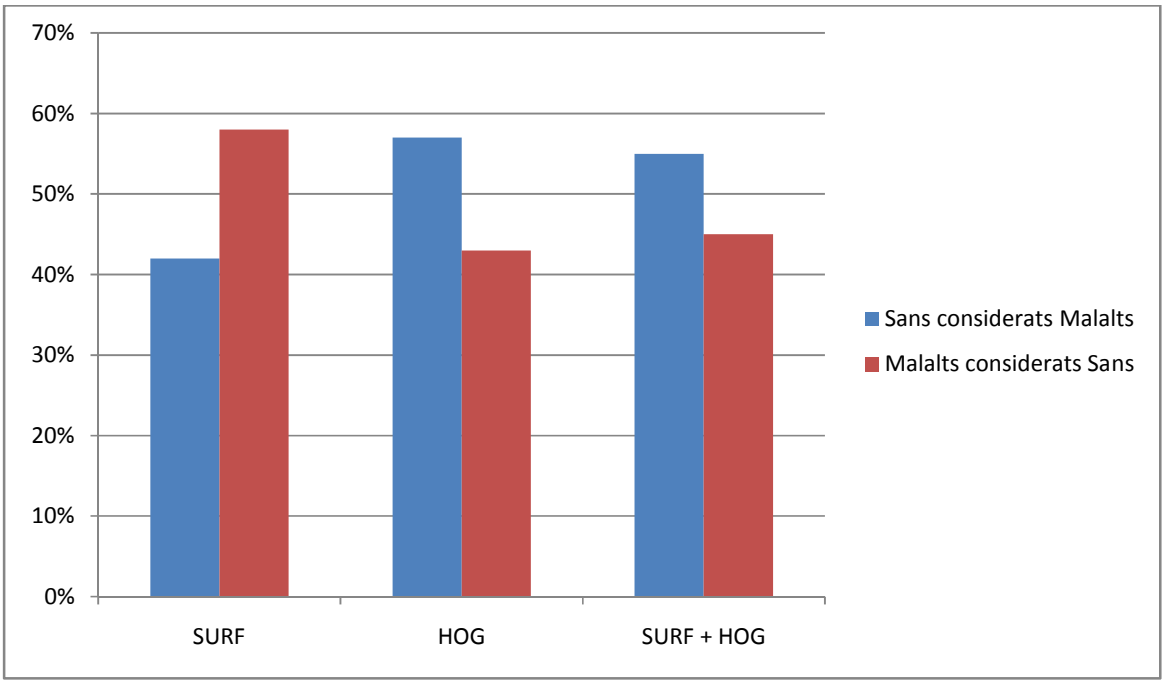
Il·lustració 8.2: Efectivitat obtinguda amb cada mètode.

Podem observar que els millors resultats són els obtinguts amb la tècnica HOG, un 9% per sobre de l'efectivitat obtinguda amb SURF. Ens trobem més informació sobre aquestes tècniques a la primera part d'aquest projecte: "**Detecció de malalties segons els trets facials - Primera part: Tractament d'imatge**". Això ja es podia intuir en aquella memòria. El motiu de que el mètode HOG sigui més robust que SURF, és degut a les diferències entre les característiques descriptives de cada rostre calculades per ambdós mètodes. Per una banda, la tècnica SURF busca els punts més descriptius en cada rostre. I encara que aquesta cerca està acotada, és comú que els punts trobats entre diferents rostres no siguin exactament els mateixos, sinó que es cometin lleugeres variacions entre ells. Aquestes variacions poden ser fins i tot menyspreables a simple vista, però a nivell de càlcul d'extracció de valors descriptius d'un imatge si que poden ser perceptibles.

La tècnica HOG en canvi, el que fa és calcular i extreure un conjunt de característiques d'una certa àrea d'aquestes imatges, no de punts concrets. Amb això s'aconsegueix més robustesa en aquest mètode, ja que els valors calculats en cada imatge són exactament els mateixos, i per tant la comparació entre aquestes és molt més efectiva.

Pel que fa a la combinació entre els dos mètodes anteriors (HOG + SURF), veiem que és tant efectiva com la tècnica HOG, pel qual també es considera que pot donar bons resultats per al diagnosis de certes malalties.

A la figura [8.3] veiem la estadística concreta del tipus d'errors comesos en cadascun d'aquests tres mètodes.

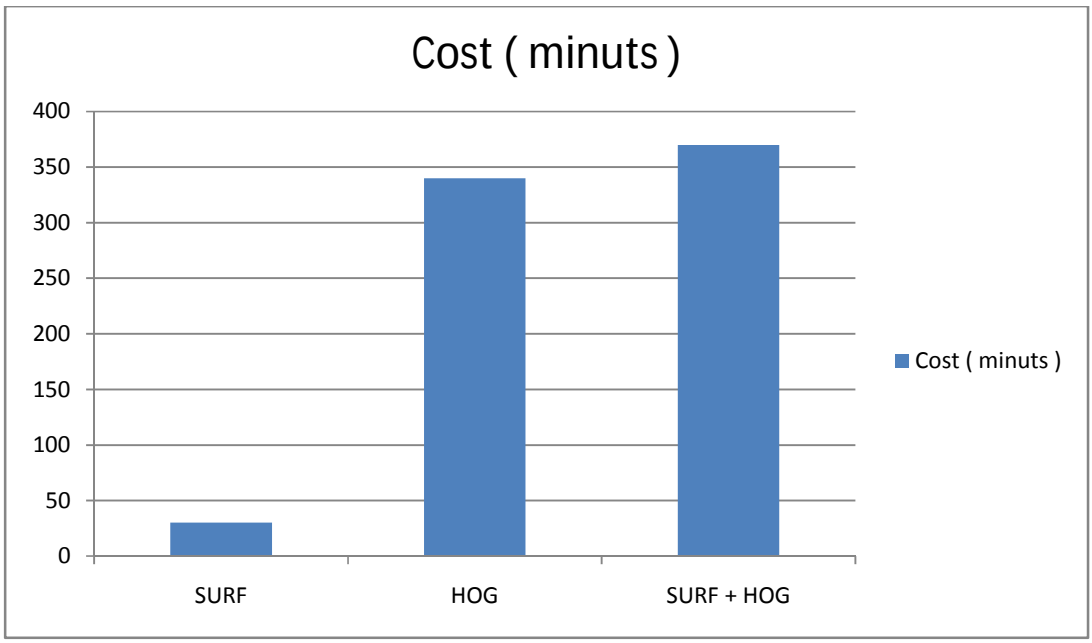


Il·lustració 8.3: Estadística tipus de errors

En aquest punt cal destacar que és molt més greu considerar una persona malalta com a sana que al revés, ja que es pot donar el cas de que el metge no tingui en compte el seu anàlisi en una malaltia que si que pot patir. En canvi al diagnosticar un pacient sa com a malalt, el màxim que pot passar és que el metge apliqui les probes per a aquella malaltia concreta, i arribi a la conclusió de que el pacient no pateix la malaltia.

Així doncs, només s'haurien de considerar com a erronis els casos en que la gent malalta és detecta com a sana, i per tant l'efectivitat del programa ens augmentaria considerablement.

Un altre assumpte a tractar és el cost de cadascuna d'aquestes tècniques. Podem observar a la figura [8.4] el cost d'execució aproximat de cada una d'elles.



Il·lustració 8.4: Gràfica del cost de creació d'un model en el programa, fet amb 225 fotografies de qualitat mitjana.

Es pot observar que la tècnica SURF té un cost considerablement inferior al de la tècnica HOG, ja que el nombre de càlculs que ha de realitzar és menor (només actuem sobre un keypoint a cada zona). Ara bé, cal dir que aquest cost és només per a la creació dels models d'entrenament, on s'han d'analitzar un elevat nombre de fotografies (en aquest cas 225). En el cas en el que ja disposem del model d'entrenament creat, el cost d'execució de l'algorisme per a realitzar el diagnosis d'un nou pacient és de pocs segons. Com s'ha vist anteriorment, en l'anàlisi d'una malaltia determinada n'hi haurà prou amb realitzar el model d'entrenament una vegada, i a partir d'aquí es podran realitzar el nombre de diagnosis que es desitgin. Per tant, tot i que el cost d'execució d'aquests mètodes sigui elevat, val la pena realitzar-los amb un destacat nombre de fotografies, ja que només s'haurà de portar a terme una sola vegada per cada tipus de malaltia estudiada.

Així doncs, es valoren molt positivament els resultats obtinguts, ja que es considera el tant per cent d'encert un molt bon percentatge. A més, cal dir que moltes de les fotografies usades en la creació del model no tenien les mateixes condicions d'il·luminació i de qualitat que d'altres. Això és degut a la gran dificultat d'obtenir un gran nombre de fotografies amb les mateixes condicions per als nostres propòsits. Per tant, es considera que amb les fotografies realitzades per a la utilització d'aquests software en els hospitals, que tindran les mateixes condicions externes, s'aconseguirà fins i tot pujar l'efectivitat del programa.

9 CONCLUSIONS

En aquest apartat es veuen les conclusions que podem extreure de la realització d'aquest projecte. Per tant, es comprova si s'ha aconseguit complir els objectius establerts, i de quina manera s'han assolit.

Com es pot observar a l'apartat 2 d'aquesta memòria, l'objectiu principal d'aquest projecte és el de crear un software base per al diagnosis assistit de malalties. Ara bé, no es pretenia efectuar un estudi mèdic sobre els tipus de malalties que es poden diagnosticar amb els trets facial, sinó solucionar la part "tècnica" del problema. Un cop creat aquest software base, els metges o científics de l'hospital Sant Joan de Déu podran començar a realitzar proves amb malalties reals dels seus pacients.

A més, es desitjava que el software creat fos senzill d'utilitzar, i que qualsevol persona amb uns coneixement mínims sobre informàtica fos capaç de fer-lo servir.

La solució al problema consta de dues parts molt diferenciades, com són **el tractament digital de les imatges a tractar**, i els **algorismes de classificació de dades** utilitzats per predir nous resultats. Cada una d'aquestes parts s'ha efectuat en un projecte diferent, i en aquest cas s'ha tractat el segon d'ells.

Per abordar aquesta part del problema, en primer lloc, ha calgut efectuar un ampli estudi sobre les diferents tècniques de classificació de característiques existents, comprovant quines donen més bons resultats i per tant quines són les òptimes per utilitzar en el programa. Per a fer-ho, s'han estudiat les dues tècniques de *Data Mining* més populars, com són els **arbres de decisió** i el **SVM**. Com es pot veure a l'apartat 4.4, s'ha comprovat que SVM dona millors resultats en el nostre algorisme, i per tant és la solució que s'ha acabat adoptant.

Així doncs, en aquest projecte s'ha vist com a partir dels vectors de característiques dels trets facials dels pacients, (dels quals podem veure la seva obtenció al projecte "**Detecció de malalties segons els trets facials - Primera part Tractament d'imatge**"), es classificaven aquestes dades en tants grups com malalties estudiades, a més del grup corresponent a la gent sana. Un cop la classificació d'aquestes dades s'ha efectuat, el nostre software és capaç de predir si un nou pacient dona símptomes o no de patir una d'aquestes malalties segons els seus trets facial.

Si observem la solució global aconseguida, el que s'ha creat és un software d'extracció de trets facials i predicció de malalties a partir d'aquests. Com es veu a l'apartat 8, la efectivitat obtinguda per el programa és satisfactòria, ja que s'ha aconseguit un tant per cent molt alt en el seu diagnosi. A més, arribem a la conclusió de que utilitzant fotografies amb condicions molt més favorables, l'efectivitat aconseguida serà fins i tot superior a la analitzada en aquesta memòria.

S'ha observat també que la tècnica **HOG** dona millors resultats que **SURF**, però també suposa un cost d'execució bastant més elevat. Per tant, caldrà tenir en compte el temps del que disposem per a efectuar l'anàlisi desitjat, i triar quina de les dues tècniques convé utilitzar en cada moment.

10 LINIES DE FUTUR

Com s'ha vist en altres apartats d'aquest projecte, l'objectiu principal d'aquest ha estat el de crear la part tecnològica d'un software de predicció de malalties. Així doncs, el que s'ha desenvolupat és un algorisme que funciona sobre una interfície gràfica senzilla d'utilitzar, i que diagnostica si un pacient dona o no símptomes de patir unes determinades malalties en base als seus trets facials més característics.

Ara bé, en aquest projecte només s'ha abordat la part tecnològica d'aquesta solució, creant un software capaç d'extreure i classificar les característiques més destacades d'un rostre. El següent pas a realitzar seria realitzar un estudi sobre quin tipus de malalties es poden predir amb el mateix. De fet aquest projecte ha estat creat amb aquest objectiu, i per tant és lògic que aquesta sigui la línia de futur a seguir més evident.

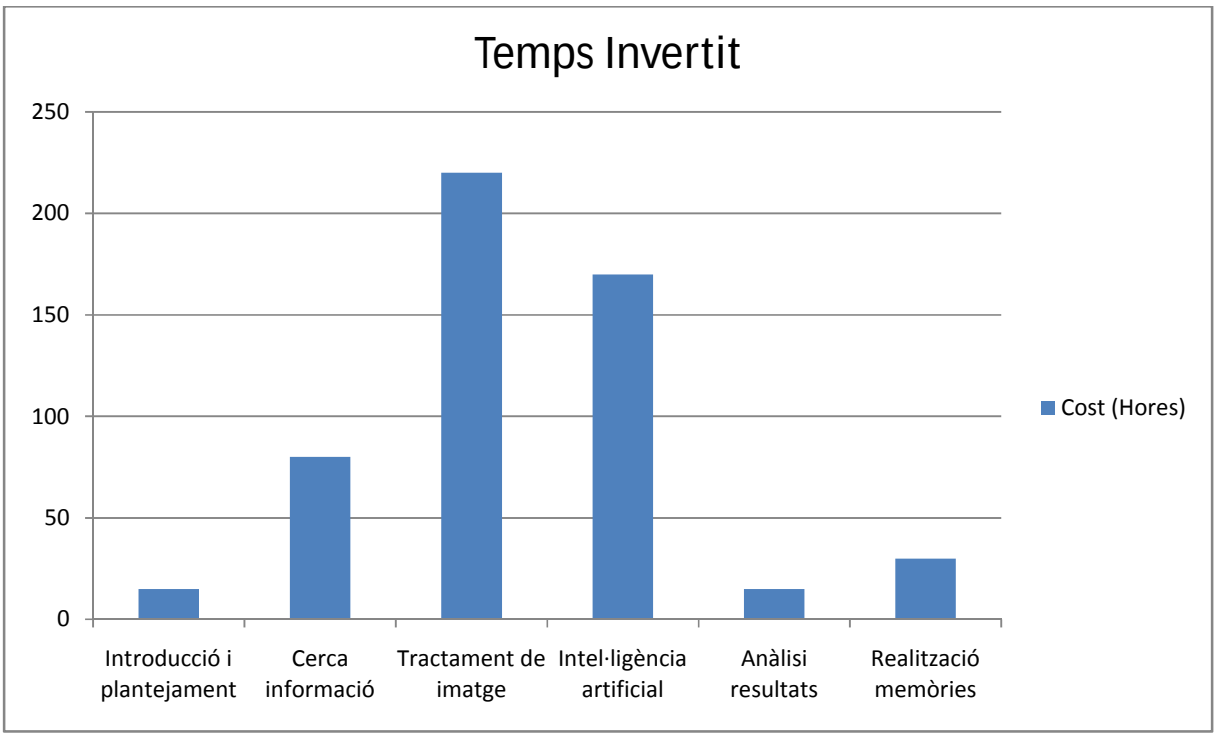
El que s'hauria de realitzar doncs és començar a fer proves de creació de models amb fotografies persones amb una determinada malaltia, i comprovar l'efectivitat del model creat per a la predicció d'aquesta. Cal saber que encara que la classificació i extracció del software es basa en els trets facials dels pacients, aquets poden ser imperceptibles a l'ull humà, i per tant no es pot saber a simple vista quin tipus de malalties podríem arribar a predir amb aquesta eina.

Una altre línia de futur interessant per al projecte seria modificar el software de manera que es pogués assignar un pes diferent a cada fotografia. Amb això es podria prioritzar si desitgem eliminar falses alarmes o malalties no desitjades. A més, també ajudaria a reduir el nombre de fotografies de malalts necessàries per a realitzar un bon entrenament en el sistema. Aquest punt pot ser molt important, ja que avui en dia resulta molt difícil trobar un bon nombre fotografies en bones condicions de gent que pateixi una malaltia concreta.

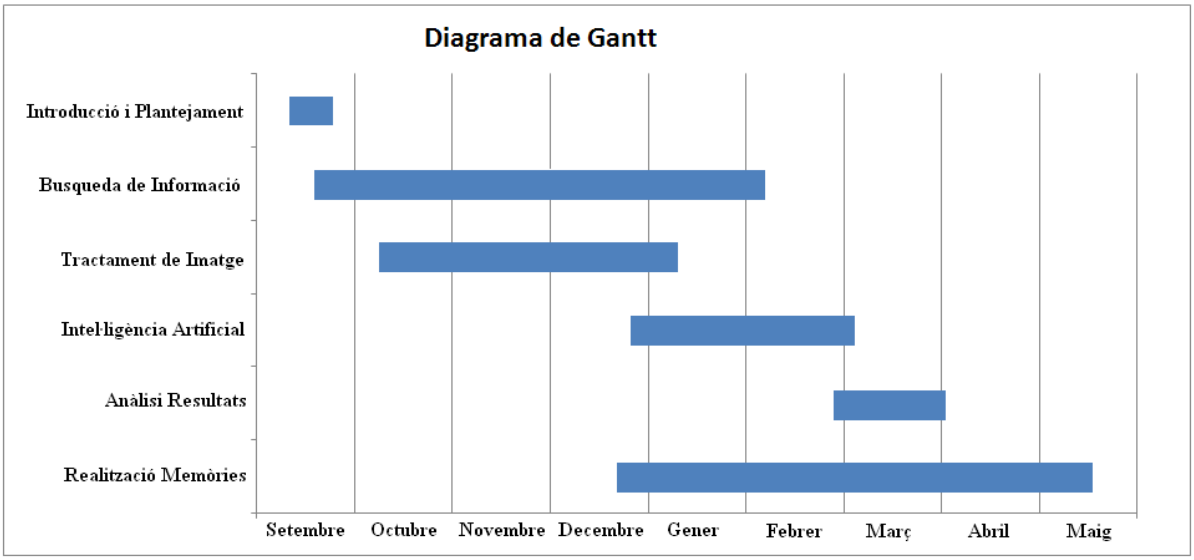
Per acabar, també seria interessa interessant adequar el programa per a que pogués diagnosticar més d'una malaltia a la vegada. De moment s'utilitza la tècnica SVM (veure apartat 4.3), que permet la classificació de les dades en el procés d'aprenentatge en varis grups diferenciats. Quan s'efectua una diagnosi, l'algorisme efectua una classificació de les noves dades en el grup que correspongui, i així s'efectua la predicció desitjada. Però només classifica les dades en el grup on aquestes encaixen millor, i per això només permet predir la malaltia que l'usuari doni més símptomes de patir, però sempre només una malaltia. Seria important doncs adaptar l'algorisme per a poder diagnosticar totes les malalties que l'usuari donés símptomes de patir, no només la més destacada.

11 TEMPS INVERTIT EN EL PROJECTE

En aquest apartat es veu el temps invertit en el projecte, tant a nivell d'hores de feina (figura [11.1]) com a nivell d'organització d'aquestes hores (figura [11.2]).



Il·lustració 11.1: Diagrama d'hores invertides en el projecte



Il·lustració 11.2: Diagrama de Gantt del projecte

12 BIBLIOGRAFIA

[1] V. Vapnik. The Nature of Statistical Learning Theory. Springer, N.Y., 1995. ISBN 0-387-94559-8.

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

[3] C. Cortes and V. Vapnik. Support vector network. Machine Learning, 20(1).

[4] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifer. In Pat Langley, editor, In Proc. 5th ACM Workshop on Computational Learning Theory, pages 144–152, Pittsburgh, PA, 1992.

[5] B. Schölkopf, K. Sung, C. Burges, F. Girosi, P.˜Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. IEEE Transactions onSignal Processing, A.I. Memo (1599), 45(11):2758–2765, 1996.

[6] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In IEEE, editor, In Proceedings of CVPR˙S97, pages 130–136, New York, NY, 1997.

[7] T. Joachims. Making Large-Scale Support Vector Machine Learning Practical, in Advances in Kernel
Methods - SupportVector Learning. MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3.

[8] John C. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization, in Advances in Kernel Methods - Support Vector Learning. MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3.

[9] C. Saunders, M. O. Stitson, J. Weston, L. Bottou, B. Schlolkopf, and A. Smola. Support vector machine reference manual. Technical report, Royal Holloway, University of London, Egham, UK, 1998.

[10] John C. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization,in Advances in Kernel Methods - Support Vector Learning. MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3.

[11] T. Joachims. Making Large-Scale Support Vector Machine Learning Practical, in Advances in Kernel Methods - SupportVector Learning. MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3.

[12] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to platt's smo algorithm for svm classifier design. Technical report, Control Division, Dept. of Mechanical Engineering, National University of Singapore, 1999.

[13] C.-C. Chang, C.-W. Hsu, and C.-J. Lin. The analysis of decomposition methods for support vector machines. IEEE Transactions on Neural Networks, 11(4):1003–1008, 2000.

[14] S. S. Keerthi and E. G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. Machine Learning, 46(1-3):351–360, 2002.

[15] C.-J. Lin. On the convergence of the decomposition method for support vector machines. IEEE Transactions on Neural Networks, 12(6):1288–1298, 2001c.

[16] C.-J. Lin. Asymptotic convergence of an smo algorithm without any assumptions. IEEE Transactions on Neural Networks, 13(1):248–250, 2001a.

[17] C.-J. Lin. Linear convergence of a decomposition method for support vector machines. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, 2001b.

[18] C.-W. Hsu and C.-J. Lin. A simple decomposition method for support vector machines. Machine Learning, 46(1):291–314, 2002b.

[19] C. A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. Constructive Approximation, 2(1):11–22, 1986.

[20] H.-T. Lin and C.-J. Lin. A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, 2003.

[21] C.-J. Lin. A formal analysis of stopping criteria of decomposition methods for support vector machines. IEEE Transactions on Neural Networks, 13(5):1045–1052, 2002.

[22] D. Hush and C. Scovel. Polynomial-time decomposition algorithms for support vector machines. Machine Learning, 51(1):51–71, 2003.

[23] S. Knerr, L. Personnaz, and G. Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network, in Neurocomputing: Algorithms, Architectures and Applications. Springer, N.Y., 1990. ISBN 3-540-53278-1.

[24] J. Friedman. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, 1996.

[25] U. KreSSel. Pairwise Classification and Support Vector Machines, in Advances in Kernel Methods and Support Vector Learning. MIT Press, Cambridge, MA, 1999. ISBN 0-262-19416-3.

[26] J. Weston and C. Watkins. Multi-class support vector machines. Technical report, Royal Holloway, University of London, Egham, UK, 1998.

[27] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. Advances in Neural Information Processing Systems, 2000.

[28] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks, 2002.

[29] L. González. Análisis discriminante utilizando máquinas núcleos de vectores soporte. Funci_on núcleo similitud. Tesis doctoral, Dpto. Economía Aplicada I. Universidad de Sevilla, Junio 2002.

[30] V.N. Vapnik. Statistical Learning Theory. John Wiley & Sons, Inc, 1998.

[31] L. González. Análisis discriminante utilizando máquinas núcleos de vectores soporte. Función

núcleo similitud. Tesis doctoral, Dpto. Economía Aplicada I. Universidad de Sevilla, Junio 2002.

[32] L. González. Teoría del aprendizaje estadístico de la regresión. Máquinas de regresión de vector base. Trabajo interno del departamento de economía aplicada i, Facultad de Ciencias Económicas y Empresariales, Universidad de Sevilla, Diciembre 2000.

[33] R. Fletcher. Practical methods of optimization. John Wiley and Sons, Inc, 2 edition, 1987.

[34] L.González Abril. Modelos de Classificación basados en Màquinas de Vectores de Soporte. Universidad de Sevilla.

[35] Ricardo Henao. Selección de hiperparámetros en máquinas de soporte vectorial. Universidad de Colombia. Maig 2004

[36] María José Ramírez Quintana José Hernández Orallo. Extracción Automática de Conocimiento en Bases de Datos e Ingeniería del Software. España, 2003.

[37] Diego García. Manual de Weka

[38] Pàgina web de la aplicació Weka: "http.//www.cs.waikato.ac.nz/ml/weka"

[39] Manual de Weka: http://www.scribd.com/doc/41476871/apuntesAD-informacion-adicional

[40] Sofia J. Vallejos. Minería de datos. Universidad Nacional del nordeste, facultad de Ciencias exactas, naturales y agrimensura. 2006

[41] María Jose Ramírez Quintana, José Hernandez Oralla. Extracción Automática de conocimiento de bases de datos e Ingenieria del software. España, 2003.

[42] Departament d'Informàtica. Universidad Nacional de San Luis (UNSL). San Luis, Argentina. 2006.

[43] Jaime López Carvajal / John William Branch Bedoya. Universidad Nacional de Colombia. Comparación de modelos de classificación automàtica de patrones, 2005.

[44] Arkaits Zubiaga Mendialdua. Aproximaciones a SVM semisupervisado multiclasse para clasificación de páginas web. ETS. De Ingeniería Informática – UNED. 2008

[45] Joachims, Thorsten. Text categorization with support vector machines: learning with many relevant features. 10th european conference on machine learning. Verlag, Heidelberg, 1998.

[46] Weston, J., & Watkins. Multi-class support vector machines. Department of Computer Science, Royal Holloway,University of London. 1998

[47] Hsu, C., & Lin, C. 2001. A comparison of methods for multi-class support vector machines.

[48] David G. Lowe. Object recognition from local Scale-Invariant features. *Computer science department. University of British Columbia.* 1999

[49] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. European

Conference on Computer Vision, 2006.

[50] Navneet Dalal , Bill Triggs, “Histograms of Oriented Gradients for Human Detection,” in Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, 2005.

[51]Usama M. Fayyad i altres. “Advances in Knowledge Discovery and Data Mining.” 1996

[52] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan. “ACM Transactions on Internet Technology”, 2001.