Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques

Universitat de Barcelona

PLATAFORMA EN JAVA PER A L'ANÀLISI DE TEXTOS D'OPINIÓ EN LLENGUATGE NATURAL

ATOp

Bakary Singateh Queral

Director: Maria Salam Llorente
Realitzat a: Departament de Matemàtica Aplicada i Anàlisi. UB
Barcelona, 17 de gener de 2013

Índex

Índex de Figures

1.-Introducció

Actualment, en la societat de la informació en la que vivim es generen moltes dades a una velocitat vertiginosa. Un exemple molt clar d'aquest fet el trobem en la *web* on constantment es generen i transmeten quantitats molt grans d'informació. Part d'aquesta informació es pot filtrar per obtenir dades útils en determinats processos.

Molta de la informació que es genera avui dia en la web té un cert caràcter subjectiu, com a resultat del tarannà participatiu que comporta la web. Així articles d'opinió, textos d'opinió, comentaris personals són un clar exemple d'aquest fet en el qual qualsevol persona pot donar la seva opinió, valoració o punt de vista sobre un aspecte donat. Doncs, tota aquesta informació, accessible per tothom, té especial interès pel que fa a l'obtenció de dades sobre el seu autor o productor de manera no intrusiva, és a dir, sense que l'usuari hagi d'escriure explícitament informació personal sobre ell mateix.

Llavors cal processar d'alguna manera aquesta informació amb l'objectiu d'obtenir dades sobre els seus autors que puguin ser utilitzades en altres sistemes informàtics. Aquest procés no és gens fàcil. En el cas concret de textos d'opinió extrets de la web, d'una banda, necessitem trobar la manera de caracteritzar-los i, d'altra banda, trobar la manera de a partir d'aquestes característiques ser capaços d'aprendre per poder classificar tota aquesta informació. Això s'aconsegueix mitjançant l'ús de tècniques pel Processament del Llenguatge Natural (PLN) combinades amb l'aplicació de tècniques d'intel·ligència artificial (IA).

Aquest projecte es basa, doncs, en la creació d'una eina, que anomenarem ATOp, que integri aquestes tècniques, mencionades anteriorment, i faci aquest procés d'anàlisi dels textos de manera automàtica. Els textos amb els quals farem les proves d'anàlisi pertanyen al corpus *Hopinion* que conté més de 18.000 textos d'opinions escrites en castellà, provinents de la *web TripAdvisor*, principalment, sobre hotels. L'aplicació disposarà d'un entorn gràfic intuïtiu que ofereix a l'usuari totes les funcionalitats necessàries per la realització de les proves d'anàlisi. A més, tots els resultats obtinguts de les proves d'anàlisi es podran guardar al disc pel seu posterior estudi.

La implementació del projecte es farà en llenguatge de programació java i s'usarà la llibreria *weka* [1] per la incorporació de les funcions d'intel·ligència artificial (IA) .

1.-Introducció

1.1 Àmbit del projecte

Com s'ha dit anteriorment, en l'era digital en la que vivim cada cop és més fàcil que un usuari de la web estigui disposat a deixar informació degut a la gran quantitat de maneres de fer-ho. Així, un pot deixar comentaris a les xarxes socials, com *Facebook* o *Twitter*, donar una valoració sobre un tema mitjançant la simple selecció d'una determinada puntuació o, simplement, escrivint un text on plasmi la seva opinió sobre algun aspecte.

És important doncs, poder processar part d'aquestes dades, com són els textos d'opinió en llenguatge natural, amb la finalitat d'obtenir informació rellevant sobre els usuaris que pugui ser utilitzada en sistemes informàtics tals com els Recomanadors.

Per aconseguir-ho cal aplicar els processos i tècniques necessàries a aquests textos d'opinió en llenguatge natural per obtenir determinades propietats o característiques lingüístiques dels mateixos que puguin ser utilitzades en el seu estudi i anàlisi. Al mateix temps, cal fer ús dels programes i sistemes informàtics per tal de processar tota aquesta gran quantitat d'informació de manera àgil. Entra en joc, doncs, la ciència del Processament del Llenguatge Natural (PLN) que consisteix, precisament, en l'estudi i anàlisi d'aspectes lingüístics d'un text a través de programes informàtics. Segons l'enfocament pràctic que se li doni a aquesta ciència en front a l'enfocament teòric, del grau amb el qual s'espera comprensió i d'altres aspectes rep diversos noms com són: el PLN, processament de textos, tecnologia del llenguatge o lingüística computacional.

Un cop obtinguda aquesta representació dels textos d'opinió en llenguatge natural, tant des del punt de vista computacional com lingüístic, s'utilitzen tècniques d'intel·ligència artificial per realitzar l'aprenentatge automàtic a partir de la caracterització de tota aquesta informació. És per això, que el Processament del Llenguatge Natural, és considerada una disciplina de la Intel·ligència Artificial.

En resum, l'àmbit del projecte serà aquestes dues disciplines, el PLN i la IA, i, per tant, la combinació d'algunes tècniques d'aquestes dues, ens permetran la realització de les proves d'anàlisi dels textos d'opinió en llenguatge natural de l'eina ATOp.

1.-Introducció

1.2. Motivació

La motivació d'aquest projecte neix de la cerca, de vies no intrusives per l'obtenció de perfils d'usuari, necessaris en diversos sistemes informàtics actuals com són els Recomanadors (SR) que utilitzen el coneixement que tenen d'un usuari per personalitzar les recomanacions. Dades com l'edat, sexe o procedència estan presents en aquests perfils d'usuari. Això es pretén aconseguir mitjançant la interpretació de les accions i el comportament d'un usuari. És a dir, mitjançant l'anàlisi dels textos que produeix un usuari, més concretament, el seu comportament verbal, podem predir alguns d'aquests trets demogràfics dels autors. Tot i ser una manera indirecta de fer-ho, constitueix una via efectiva.

D'aquesta manera, l'estudi del comportament verbal d'un autor definit a través de mesures sobre la riquesa lèxica, extretes del seu text constituirà la base per les proves d'anàlisi que ens permetran predir determinats atributs demogràfics de l'autor, informació que després s'utilitzarà en l'elaboració de perfils d'usuari que serveixin en el disseny de sistemes com els Recomanadors (mencionats anteriorment).

1.3. Objectius generals

L'objectiu general del projecte és la creació i implementació de l'eina ATOp que permeti la realització de proves d'anàlisi sobre els textos d'opinió escrits en llenguatge natural, provinents de la web *TripAdvisor*, de manera automàtica.

1.4. Objectius específics

Com ja s'ha dit anteriorment, l'objectiu general de l'aplicació es la creació de la plataforma java, anomenada ATOp, que permeti l'anàlisi dels textos d'opinió escrits en llenguatge natural. Per tant, cadascun dels objectius específics que tot seguit es mencionaran constitueix un dels passos necessaris perquè l'objectiu general sigui realitzat satisfactòriament.

- Creació de l'estructura o model de dades on guardar la informació que s':utilitzarà en les proves d'anàlisi.
- Extracció i càlcul de les mètriques o mesures de la riquesa lèxica que representaran les característiques dels textos d'opinió en llenguatge natural (PLN).
- Realització de proves d'anàlisi on s'apliquin tècniques d'intel·ligència artificial per l'aprenentatge automàtic.
- Realització d'estadístiques, a partir dels resultats obtinguts en les proves d'anàlisi.

1.-Introducció

1.5. Organització de la memòria

L'organització en capítols de la memòria serà la següent:

- Capítol 2 -> Antecedents: es dóna una visió general de l'àmbit de la lingüística en el qual es situa aquest projecte, definint conceptes que ajuden a contextualitzar l'objecte .
- Capítol 3 -> Anàlisi: es fa una anàlisi detallada dels requeriments funcionals de l'aplicació, definint els diferents casos d'ús, el model de domini i els requeriments generals de l'eina ATOp.
- Capítol 4 -> Disseny: visió general de la implementació de l'aplicació, mitjançant l'ús de diagrames d'interacció, on es vegi la interacció entre les diferents classes, el diagrama de classes, així com una breu explicació dels principals algorismes.
- Capítol 5 -> Implementació i resultats: descripció d'alguns aspectes generals de la implementació, algunes eines utilitzades i realització de diverses proves per mostrar els resultats obtinguts.
- Capítol 6 -> Valoració econòmica: anàlisi del temps utilitzat per la realització del projecte.
- Capítol 7 -> Conclusió: valoració final del projecte, dels resultats obtinguts i les possibles línies continuació del mateix.

2.-Antecedents

En els darrers anys, els treballs d'investigació sobre el Processament del Llenguatge Natural ha crescut notablement, gràcies en certa manera a l'aplicació i ús de les noves tecnologies en els processos de cerca. Això ha donat lloc a l'aparició de diferents camps dins els PLN més específics cadascun dels quals aborda aspectes més concrets. A continuació es mencionaran i definiran conceptes clau per entendre el context en el qual es situa aquest projecte.

2.1. Còrpora lingüística

Com ja s'ha dit anteriorment, per la realització de les proves d'anàlisi del projecte utilitzem textos que pertanyen al corpus *Hopinion* que conté més de 18.000 textos d'opinions escrites en castellà, provinents de la *web TripAdvisor*, principalment, sobre hotels. Anem, doncs, a veure diferents definicions de corpus lingüístic i els avantatges del seu ús en el camp del PLN.

Convé plantejar què entenem avui dia per corpus lingüístic. Una definició *latu sensu* del concepte de corpus lingüístic la trobem en Payrató (1996:112) [2] qui opina que “un corpus és un magatzem organitzat de materials lingüístics, normalment textos”.

Alvar Ezquerra, Blanco Rodríguez i Pérez Lagos (1994:10) [3] complementen la definició anterior de corpus:

> Un corpus és un conjunt homogeni de documents lingüístics de qualsevol tipus (orals, escrits, literaris, col·loquials, etc.) que són presos com a model d'un estat o nivell de la llengua predeterminat, al qual representen o es pretén que representin. Aquest conjunt d'enunciats se sotmetran a un tractament informàtic els resultats del qual permetran el millor coneixement de les estructures lingüístiques de la llengua representada.

2.1.1. Anàlisi lingüístic basat en corpus

Les característiques d'un anàlisi lingüístic basat en còrpora han sigut presentades per Biber (1998:4) [4] en els següents termes:

- Una anàlisi d'aquest tipus és empíric, donat que analitza els patrons d'ús de textos reals.
- Utilitza una gran quantitat de textos, coneguts com corpus, com a base per la investigació.
- Fa ús dels ordinadors per analitzar els documents, utilitzant tècniques automàtiques i interactives.
- Depèn tant de mètodes quantitatius com qualitatius per realitzar les proves d'anàlisi.

Aquests autors plantegen alguns dels trets definitoris del nou enfocament donat als *corpus* lingüístics, com és, per exemple, el que actualment es realitza una explotació quantitativa (i estadística) de les dades proporcionades pel corpus estudiat. Referent a això, Biber [4] recorden que l'anàlisi basat en un corpus no ha de quedar-se en un simple recompte dels trets lingüístics, sinó que ha d'incloure estudis qualitatius i interpretacions funcionals dels patrons quantitatius que es presenten en el corpus.

2.-Antecedents

Segons McEnery i Wilson (1996) [5] les principals avantatges de l'ús de corpus lingüístics són:

a) Representativitat i quantificació. Si un corpus està estructurat per ser representatiu de la població, les troballes de la mostra poden ser generalitzades a una mostra àmplia de la població.

b) Facilitat d'accés. Un cop un corpus ha sigut recollit i codificat, és bastant senzill accedir a les seves dades.

c) Dades “enriquides”. Diversos corpus estan enriquits amb informació lingüística com la categorització gramatical, anotació sintàctica, i anotació morfològica la qual cosa facilita a l'estudiós l'explotació de les seves dades.

d) dades extretes de textos reals.

Tot i l'avenç en aplicacions informàtiques, encara resulta molt complicat portar a terme tasques com la desambiguació morfològica de manera automàtica.

2.1.2. Lingüística del corpus

Definit el concepte de corpus i el paper que juga en l'anàlisi lingüístic, passem a comentar què s'entén por lingüística del corpus. Aijmer i Altenberg (1991:1) [6] parteixen d'una visió àmplia quan diuen que “corpus lingüístic podria ser descrit com l'estudi del llenguatge sobre la base del text en el corpus”. L'objectiu de la lingüística del corpus, per tant, seria “l'estudi del llenguatge a través de l'establiment i desenvolupament de corpora lingüístics”. Seguint doncs, un dels comentaris de Leech (1992:108) [7] un dels principals focus d'atenció de la lingüística del corpus és l'èmfasi en el pla quantitatiu de la llengua sense oblidar el qualitatiu. Segons aquest autor, hi ha una forta connexió entre la lingüística del corpus i la lingüística quantitativa i això es dóna per dos raons: d'una banda, si tenim un corpus informatitzat de grans dimensions, una de les operacions més òbvies que podem fer amb ell és derivar freqüències; d'altra banda, si volem realitzar una descripció dels models quantitatius que subjauen en la llengua, el més normal és que per ells utilitzem un corpus lingüístic, donat que no podem confiar, en aquest cas, en la intuïció del lingüística.

2.1.3 Qüestions metodològiques

2.1.3.1 Anotació

L'anotació és la pràctica mitjançant la qual s'afegeix informació (lingüística o no lingüística) a un corpus de la llengua, amb l'objectiu que aquest pugui ser utilitzat per altres investigadors, així com implementat a partir de tècniques informàtiques. Existeixen dos tipus de corpus:

1. Els no anotats (és a dir, el text apareix sense cap tipus d'informació)
2. Els anotats (on el corpus proporciona informació lingüística i no lingüística).

2.-Antecedents

2.1.3.1 Anotació

Amb relació amb l'anotació dels corpus de la llengua, està comunament admesa per la comunitat investigadora la necessitat de treballar amb estàndards de marcació, ja que aquests es poden aplicar a corpus de diferents llengües i, a més a més, faciliten el processament informàtic de les dades. Com a conseqüència de la necessitat de crear llenguatges de marcació internacionalment acceptats, va sorgir SGML (*Standard Generalized Mark-up Language*) que segons alguns autors com Pérez Guerra 1999 [8] no ha de ser considerat com un llenguatge de marcació pròpiament dit, sinó, més bé, com un sistema de marcació de textos, o també com una normativa o conjunt de normes estàndards que permet a estudiosos codificar certs aspectes d'un text d'un mode universal.

2.1.3.2 Codificació

Seguint a Moure i Llisterri (1996:176) [9] utilitzem el terme de codificació per fer referència a un "conjunt de convencions amb les que s'associa cada paraula d'un corpus a la informació gramatical, morfològica i semàntica que pot ser rellevant pel seu anàlisi un cop s'ha extret del text". No és exagerat afirmar, per això, que un corpus serà més i millor utilitzat en la mesura amb que la seva codificació lingüística sigui més rigorosa i completa. Segons Berber (1999) [10] és necessari codificar els corpus per tres motius:

- Extracció posterior d'informació.
- Reutilització. Un mateix corpus pot ser utilitzat per diferents investigadors.
- Multifuncionalitat. Un mateix corpus pot ser utilitzat per diferents finalitats.

Les codificacions lingüístiques més utilitzades en lingüística del corpus, són la codificació morfològica, la sintàctica i la semàntica.

En l'àmbit d'aquest projecte només parlaré de la codificació morfològica del corpus que és la utilitzada com a pas previ en els textos d'opinió en llenguatge natural.

2.1.3.2.1. Anàlisi morfològic

Es denomina anotació morfològica a l'assignació d'una categoria morfològica a cadascuna de les paraules d'un corpus. En l'àmbit anglosaxó, aquest sol rebre el nom de *tagging* o de *part-of-speech annotation*, mentre que en els treballs hispànics podem trobar noms com la categorització o codificació morfològica.

McEnery i Wilson (1996:36) [5] resumeixen l'objectiu de l'anotació morfològica de la següent manera:

> El tipus més bàsic d'anotació lingüística del corpus és *part-of-speech tagging* (algunes vegades també conegut com a etiquetat gramatical o anotació morfosintàctic). L'objectiu de *part-of-speech tagging* és assignar a cada unitat lèxica (*token*) del text un codi indicant la seva categoria morfosintàctica (per exemple, nom comú singular, adjectiu comparatiu, participi passat, etc.).En la realització del projecte s'utilitza una llibreria anomenada Freeling [11] que conté un analitzador morfològic que ens permet fer l'anotació morfològica dels textos d'opinió en llenguatge natural.

2.-Antecedents

2.1.3.2.1. Anàlisi morfològic

D'altra banda, convé senyalar que un esquema d'anotació morfològica (*tag annotation*) consta dels següents apartats (Leech, 1993:276) [7]:

a) Un *tagset*, un grup d'etiquetes gramaticals.

b) Un conjunt de definicions per tals etiquetes.

c) Una sèrie de guies que descriguin l'aplicació d'aquestes etiquetes.

2.2. Corpus i Lingüística Computacional

En una panoràmica sobre els conceptes i mètodes de la lingüística del corpus, és imprescindible fer referència a la necessària relació que s'estableix entre la lingüística computacional i còrpora de la llengua, doncs els sistemes implementats pels estudis alineats en l'àmbit del processament del llenguatge natural sempre han d'estar basats en les dades provinents d'un corpus lingüístic. Entre les raons que justifiquen la constant utilització de còrpora per part de la lingüística computacional es contarien, entre d'altres:

- la necessitat de comprovar si els sistemes creats per aquesta disciplina estan adaptats a les dades reals de la llengua estudiada.
- El desenvolupament de models lingüístics que puguin donar compte no només de la llengua actual, sinó també del llenguatge utilitzat en el futur.

Segons diversos autors, els interessos de la lingüística computacional en relació amb les dades presents en un corpus lingüístic radica en els següents aspectes:

- Identificació de les diferents paraules d'un text, així com les combinacions i col·locacions de les mateixes.
- Investigació del comportament de les classes semàntiques i de la estructura lèxica del text,
- L'esbrinament dels esquemes i jerarquies que s'estableixen en el text.

En opinió de Bindi (1994:29) [12] **:** "còrpora del llenguatge parlat i escrit són una font essencial i primordial per qualsevol projecte del Processament del Llenguatge Natural, *NLP*, que estigui destinat a una aplicació real". Molts autors senyalen que l'anàlisi de còrpora és la principal font per obtenir l'evidència de com s'està usant realment el llenguatge.

En relació amb la diferent utilitat que pot tenir un corpus lingüístic pels estudis de lingüística computacional Ostler (1992:2) [13] distingeix dos possibles usos:

- Com a font de dades.
- Com una manera de provar el funcionament de sistemes informàtics.

En el primer cas, el corpus presenta interès per si mateix, mentre que en el segon, simplement de manera instrumental, com un "input" per al procés de perfeccionament d'un sistema electrònic.

2.-Antecedents

2.3. Lingüística Computacional

La lingüística del corpus es troba lligada tant a les disciplines encarregades de l'anàlisi del discurs, com a la Lingüística Computacional. En aquest apartat ens aproparem als trets definitoris de la LC per ubicar els estudis referents a la lingüística del corpus.

Des d'un punt de vista globalitzador i general, Payrató (1998:108) [14] senyala que "la lingüística computacional estudia les aplicacions i aportacions de la informàtica a l'anàlisi del llenguatge". Aquest autor, per tant, defensa una perspectiva àmplia de la disciplina, que s'urgiria de la correlació d'informàtica i llenguatge; a més a més, apunta que la corrent que s'encarrega d'aquests aspectes també es denomina lingüística informàtica o processament del llenguatge natural.

Tot i això, aquesta opinió globalitzadora no és majoritària. Així, Grishman (1991:15) [15], "la lingüística computacional és l'estudi dels sistemes de computació utilitzats per la comprensió i la generació de les llengües naturals". Segons aquest autor, per tant, no forma part de la lingüística computacional qualsevol aplicació informàtica per l'estudi del llenguatge, sinó només aquelles que poden ser utilitzades per a uns objectius molt concrets.

Allen (1987) [16] defineix el Processament del Llenguatge Natural d'una manera similar: "L'objectiu d'aquesta investigació és crear models computacionals del llenguatge suficientment detallats que permetin escriure programes informàtics que realitzin les diferents tasques on intervé el llenguatge natural."

La definició més actual d'aquest segon grup correspondria a Moreno Sandoval (1998:16) [17] qui senyala el següent:

- Per oposició a altres disciplines, LC tracta de la construcció de sistemes informàtics que processen estructura lingüística i l'objectiu del qual sigui la simulació parcial de la capacitat lingüística dels parlants d'una llengua, independentment del seu caràcter comercial o d'investigació bàsica.

Hi ha, per tant, una sèrie d'autors que no inclouen la lingüística del corpus dins de la lingüística computacional, donat que pensen, com Gazdar i Mellish (1989:16) [19] que "la parcel·la del coneixement que tracta de la investigació lingüística i literària amb mitjans informàtics no es considera part de la lingüística computacional".

Una de les visions més clarificadores del problema correspon a Gómez Guinovart (1999:7-8) [20] , qui opina que una delimitació del camp d'estudi de la lingüística computacional implica parlar de tres línies principals d'investigació. La primera línia i la més important seria aquella en la que apareix un grup d'estudis basats en l'aplicació dels ordinadors a la investigació lingüística, que podria rebre el nom de lingüística informàtica o d'informàtica aplicada a la lingüística. Gómez Guinovart [20] senyala que algunes de les àrees d'estudi que s'inclouen en aquesta línia d'investigació són la lingüística del corpus, la lingüística estadística, la estilometria, la informàtica aplicada a la sociolingüística i la lexicografia assistida per ordinador.

2.-Antecedents

2.3. Lingüística Computacional

Per últim, existeix una orientació de la lingüística computacional més tecnològica i informàtica que les anteriors, que es centra en el disseny i l'elaboració de sistemes informàtics capaços de treballar amb enunciats orals i escrits provinents de llengües naturals. Els productes als quals pot donar lloc aquesta orientació són: eines d'ajuda a la escriptura i a la traducció, aplicacions de la tecnologia de la parla, sistemes de gestió documental, aplicacions didàctiques per l'ensenyament de llengües i sistemes de diàleg. L'autor senyala que depenent de l'aspecte en el qual es posi més èmfasi, aquest camp de treball rep les denominacions de enginyeria lingüística, tecnologies de la llengua, processament del llenguatge natural o indústries de la llengua.

Des d'aquest darrer punt de vista, la creació de l'eina ATOp per l'anàlisi de textos en llenguatge natural a partir de les mesures de la riquesa lèxica (comportament verbal) si es podria incloure dins la lingüística computacional, donat que pertany als àmbits de la lingüística del corpus i de la estadística lingüística, pròpies de l'orientació de la disciplina que Gómez Guinovart [20] denomina lingüística informàtica.

Les definicions presentades fins al moment han plantejat dues de les qüestions principals a les quals s'ha d'enfrontar qualsevol autor que desitgi aclarir l'objecte d'estudi i la definició de la lingüística computacional: per una banda, s'ha d'aclarir si partim d'una visió àmplia de la disciplina (com a intersecció d'informàtica i lingüística) o estricta (com una corrent que s'encarrega del disseny d'eines computacionals). D'altra banda, i en segon lloc, és necessari diferenciar la lingüística computacional d'altres disciplines similars.

El primer dels problemes està encara per resoldre, però Moure i Llisteri (1996:209) [9] aporten un poc de claredat quan senyalen que:

Com qualsevol altra àrea de coneixement, la Lingüística Computacional pot definir-se mitjançant tres paràmetres:

- L'objecte que estudia
- La metodologia amb la que aborda aquest estudi
- La finalitat que persegueix.

D'entrada, aquesta disciplina, s'ocupa d'elaborar teories i procediments per aconseguir el tractament automàtic de les llengües. Per aquest motiu, des del punt de vista metodològic, posseeixi un caràcter híbrid, a mig camí entre la informàtica i la lingüística. El seu objectiu, per fi, es xifraria en obtenir productes tecnològics relacionats amb les indústries de la llengua.

Del caràcter híbrid de la metodologia d'aquest corrent han parlat també Meya i Huber (1986:12-13) [21], els quals opinen que els mètodes de la lingüística computacional provenen d'altres disciplines: procedents de l'acústica, dins del processament de la parla, figuren els següents mètodes: l'anàlisi de la freqüència, l'anàlisi espectral, l'anàlisi de *Fourier*, processos de transformació anàleg-digital, etc. Entre els mètodes matemàtics, desenvolupen un paper rellevant els següents: estadística, sistemes algebraics, sistemes formals, i teoria d'autòmats, etc. De la informàtica s'han adoptat sistemes algorítmics i tècniques de cerca. D'altra banda, per al tractament semàntic de la llengua s'ha acudit, o bé, als mètodes de la lògica, com per exemple, el càlcul de predicats i el càlcul de la *lambda*, o bé, les matemàtiques, amb la teoria de grafs.

2.-Antecedents

2.3. Lingüística Computacional

En quant a la disciplina conceptual de la lingüística computacional respecte d'altres disciplines similars, convé indicar, en primer lloc, que actualment es parla de processament del llenguatge natural com un terme sinònim de la lingüística computacional, encara que convé senyalar que és una denominació utilitzada, sobretot, pels partidaris de la concepció estricta de la disciplina.

Amb respecte a la diferència entre lingüística computacional i altres disciplines en ocasions considerades com a sinònims, com la Enginyeria Lingüística o les indústries de la llengua, Moreno Sandoval (1998:14-15) [18] creu que no es convenient la identificació entre aquests corrents d'investigació, i utilitza els següents arguments:

- Per Enginyeria Lingüística s'entén tota aquella aplicació potencialment comercial que impliqui l'ús de les noves tecnologies i llengües. En aquest sentit, s'inclou l'edició electrònica (diccionaris, llibres, periòdics,etc.), els productes multimèdia, etc. Per descomptat, també té cabuda tot sistema PLN comercial (traducció automàtica, corrector gramatical, reconeixedor de la parla...). No obstant això, no podem incloure una part important de la recerca en la lingüística computacional: aquella que no té com a primer objectiu la comercialització d'un producte, com és el cas de nombrosos projectes en universitats i centres de recerca públics i privats.

És a dir, el que separa l'Enginyeria Lingüística de la LC és el caràcter marcadament comercial de la primera.

Com es desprèn d'aquestes definicions, l'existència de les indústries de la llengua, enteses com una activitat d'índole fonamentalment comercial, requereix del desenvolupament de l'enginyeria lingüística per disposar de les eines i tècniques a partir de les quals es creen productes per realitzar diverses funcions relacionades amb la utilització del llenguatge.

Amb respecte a la diferència entre Lingüística Computacional i Lingüística Informàtica, l'opinió de Moreno Sandoval (1998:14-15) [18] es pot considerar representativa de la majoria d'investigadors. Aquest autor creu que:

- es podria afirmar que la LC és una part integrant de la Lingüística Informàtica si entenem aquesta última com la disciplina que comprèn tot ús d'ordinadors amb relació al llenguatge i les llengües. Aquí, s'inclourien no només els sistemes que simulen el llenguatge humà, sinó tot tipus de programa i eina informàtica que ajudi a l'estudi de les llengües i de la Lingüística.

2.-Antecedents

2.4. Lingüística Computacional i Lingüística Matemàtica

Alguns autors, com Meya i Huber (1986:8-9**)** [21] s'han ocupat de la definició de lingüística computacional en relació amb altres disciplines properes a la Lingüística Matemàtica. Els autors citats pensen que l'objectiu de la lingüística matemàtica és la fonamentació metòdica de la lingüística i la formació teòrica estricta. La lingüística recorre en aquest cas a les matemàtiques, ciència d'estructures abstractes i de processos formals, per adoptar els seus mètodes. Per altra part, la lingüística algebraica s'apropia de mètodes matemàtics per la fixació de teories abstractes, tals com la teoria de conjunts, de grafs, de funcions recursives i d'autòmats. El seu objectiu fonamental és la recerca i definició de gramàtiques formals. La lingüística estadística, al contrari que la lingüística algebraica, estaria orientada a l'estudi quantitatiu controlat dels fets lingüístics. La meta final de la lingüística estadística seria obtenir resultats quantitatius sobre determinades propietats de la llengua, les quals han sigut eventualment pressuposades o definides en teories lingüístiques. El seu objectiu és, doncs, investigar les regularitats de la distribució dels elements d'un text. En quant a la lingüística quantitativa, lligada metodològicament amb la lingüística de corpus, els autors pensen que "la lingüística quantitativa és la part de la lingüística matemàtica que intenta determinar les lleis que subjauen a l'organització estadística del llenguatge a través de l'anàlisi del comportament de les unitats de text." (Meya i Huber, 1986:47 [21]).

2.5. Lingüística Teòrica i la Lingüística Computacional

Per últim, és interessant distingir entre lingüística teòrica i la lingüística computacional. Moreno Sandoval (1998:30) [17] ens diu el següent: la lingüística teòrica i la computacional tenen bastants diferències en quant als seus enfocaments, mètodes i objectius. Per una banda, la lingüística teòrica es centra en:

- Analitzar la competència dels parlants.
- Utilitza la introspecció com a principal font per obtenir les seves dades.
- Sol arribar a les seves conclusions mitjançant mètodes deductius.
- Els seus principals objectius són aconseguir una teoria gramatical, simple, elegant, restringida i que donin raó dels universals lingüístics.

D'altra banda, la lingüística computacional:

- Està interessada en l'ús lingüístic.
- Utilitza dades procedents de situacions comunicatives reals
- Els seus mètodes poden ser tant deductius com inductius.
- El seu objectiu final és obtenir un sistema que funcioni, és a dir, que processi estructura lingüística de una forma computacionalment eficient.

2.-Antecedents

2.6. Lingüística Computacional i aplicacions

Després de totes aquestes opinions, es pot comprovar que la definició de la lingüística computacional, així com la delimitació del seu objecte d'estudi mètodes i separació de disciplines afins és encara una tasca difícil d'aclarir; per això, molts autors han preferit apropar-se al camp de la lingüística computacional a través de l'estudi de les seves aplicacions.

Segons Moreno (1998:27-29) [17] les aplicacions de la lingüística computacional es poden agrupar en els següents blocs:

- Sistemes que tracten d'emular la capacitat humana de processar llengües naturals. Dins d'aquest grup, les aplicacions més importants són la traducció automàtica, la recuperació i extracció d'informació i les interfícies home-màquina.
- Sistemes que ajuden en les tasques lingüístiques. Aquest segon grup està format per eines que poden ser utilitzades pels lingüistes per facilitar-los certes tasques complexes. Algunes aplicacions d'aquest tipus són les eines d'anàlisi textual, les eines per l'ús de corpus i les bases de dades lexicogràfiques.
- Programes d'ajuda a l'escriptura i la composició textual. Les aplicacions compreses en aquest grup han sigut àmpliament desenvolupades i qualsevol usuari habitual d'un processador de textos està familiaritzat amb elles, com per exemple, els correctors ortogràfics i els correctors sintàctics i d'estil.
- Ensenyament assistit per ordinador.

3. Anàlisi

En aquesta secció s'explica, entre altres, els requeriments funcionals de l'aplicació, els principals casos d'ús que es donen, es defineix el model de domini i es menciona el requeriments generals de software i hardware de l'aplicació així com les tecnologies utilitzades.

3.1 Model Vista Controlador

L'eina ATOp per l'anàlisi de textos utilitza el patró MVC de Java, ja que disposa d'una interfície gràfica d'usuari que ens permet dur a terme totes les funcionalitats necessàries adaptant-se a aquest patró.

Model Vista Controlador

D'acord amb aquest patró d'arquitectura de software les dades de l'aplicació es divideixen en tres capes que separen les dades de l'aplicació, la interfície d'usuari i la lògica de negoci en tres components diferents:

1. Model: és l'encarregat de portar tota la lògica de negoci de l'aplicació Java.
2. Vista: aquest presenta el model en un format adequat per interactuar, la interfície d'usuari.
3. Controlador: respon a esdeveniments que usualment són les accions dels usuaris i delega la responsabilitat de realitzar certes accions al model.

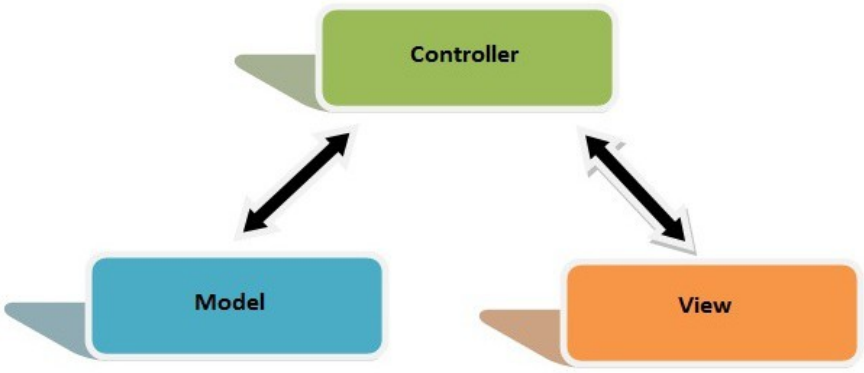

Figura 1: Patró MVC

Tot i que, es poden trobar diferents implementacions de ***MVC***, el flux habitual que segueix el seu funcionament sol ser el mateix:

- L'usuari interactua amb la interfície gràfica d'usuari, realitzant les accions d'alguna manera (p. e. prem un botó).
- El controlador rep (per part d'alguns dels objectes de la interfície gràfica) la notificació de l'acció realitzada per l'usuari a través d'un gestor esdeveniments.
- El controlador s'encarrega de transmetre la petició de l'usuari al model el qual realitza les accions corresponents i un cop ha acabat envia la resposta al controlador.

3. Anàlisi

3.1 Model Vista Controlador

- Si la petició feta per l'usuari i la posterior acció del model implica algun canvi de vista, les modificacions són visualitzades en la interfície gràfica perquè l'usuari les pugui veure.

D'aquesta manera el model mai es comunica directament amb la vista, sinó que ho fa a través del controlador. De la mateixa manera, la vista mai interactua de forma directa amb el model, per tant, el controlador és l'encarregat de recollir les peticions realitzades per l'usuari a través de la vista i delega la responsabilitat al model de dur a terme determinades accions. El resultat d'aquestes accions, si implica canvis en la vista, seran retornats pel model al controlador, que s'encarregarà que arribin a la vista perquè siguin mostrats a l'usuari de l'aplicació.

3.2 Diagrama de casos d'ús

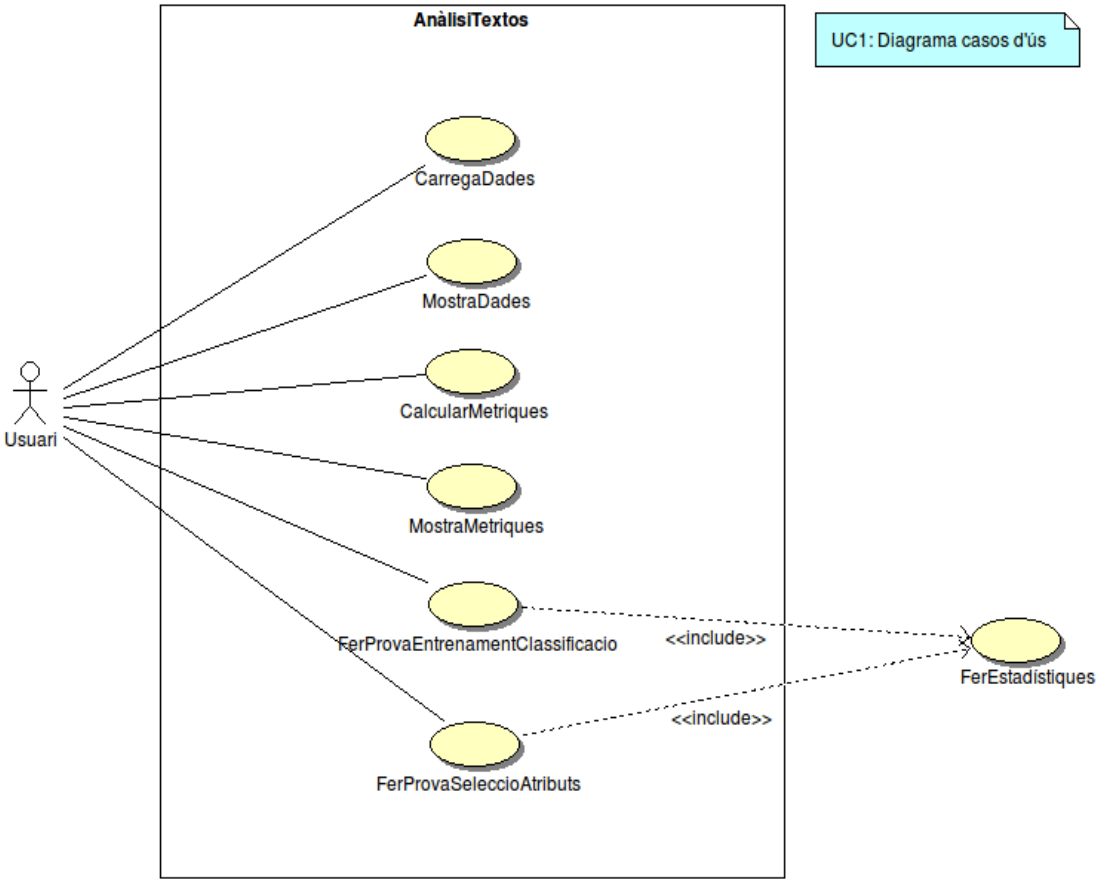

Figura 2 : Diagrama casos d'ús.

3. Anàlisi

3.3 Casos d'ús

UC1: L'usuari carrega informació de la base de dades *hopinion*.

Actor principal: usuari

Precondicions: l'usuari ha engegat l'aplicació.

Flux bàsic

1.-L'usuari prem el botó per fer la càrrega de les dades sobre els textos d'opinió que requerirem per l'anàlisi.

2.-El sistema mostra una pantalleta on l'usuari ha d'escollir el tipus de *driver* que necessita per fer la connexió amb el Sistema Gestor de Bases de Dades (de moment, es pot escollir entre *MySQL* i *Postgresql*), introduir el nom de la base de dades (*hopinion*), l'adreça IP de l'ordinador on està localitzada, la contrasenya i el nom de l'usuari autoritzat per l'accés.

3.-Tot seguit, l'usuari prem el botó *Carrega dades* amb els valors dels camps demanats emplenats.

4.-El sistema inicia el procés es connecta amb la base de dades amb els valors que l'usuari ha introduït, fa les consultes i descarrega la informació requerida.

5.-La finestra es tanca i es mostra un missatge al panel *Estat* de la pantalla principal informant que les dades han sigut carregades correctament i, a més, al panel Dades, també de la pantalla principal, es mostra el número de mostres carregades, donant la possibilitat de la informació emmagatzemada.

Flux alternatiu

3.1.- L'usuari prem el botó *Carrega dades* però el nom de la base de dades no és el correcte.

1-El sistema mostrarà una missatge d'error avisant d'aquesta circumstància i que, per tant, no s'ha pogut realitzar la connexió correctament.
2-L'usuari accepta el missatge d'error.
3-La finestreta d'error es tanca i torna a la pantalleta *Carrega de dades*.

3.2.- L'usuari prem el botó *Carrega dades,* però l'adreça IP no és la correcta.

1-El sistema mostrarà una missatge d'error avisant d'aquesta circumstància i que, per tant, no s'ha pogut realitzar la connexió correctament.
2-L'usuari accepta el missatge d'error.
3-La finestreta d'error es tanca i torna a la pantalleta *Carrega de dades*.

3.3.- L'usuari prem el botó *Carrega dades,* però la contrasenya no és la correcta.

1-El sistema mostrarà una missatge d'error avisant d'aquesta circumstància i que, per tant, no s'ha pogut realitzar la connexió correctament.
2-L'usuari accepta el missatge d'error.
3-La finestreta d'error es tanca i torna a la pantalleta *Carrega de dades*.

3. Anàlisi

Flux alternatiu

3.4.- L'usuari prem el botó *Carrega dades*, però l'usuari introduït no està autoritzat.

1-El sistema mostrarà una missatge d'error avisant d'aquesta circumstància i que, per tant, no s'ha pogut realitzar la connexió correctament.
2-L'usuari accepta el missatge d'error.
3-La finestreta d'error es tanca i torna a la pantalla *Carrega de dades*.

UC2: L'usuari mostra les dades carregades de la base de dades *hopinion* Actor principal: l'usuari.

Precondicions: l'usuari ha engegat l'aplicació i ha fet la càrrega de les dades.

Flux bàsic

1.-L'usuari prem el botó *Mostra Dades* que es troba a la finestra principal.

2.-El sistema mostra una finestra on apareix una taula amb totes les dades carregades amb el nom de les corresponents columnes. L'usuari pot avançar entre les diferents files prement els botons Anterior i Següent. També té la possibilitat de modificar el valor d'algun dels camps dels diferents registres carregats.

3.-L'usuari modifica el valor d'alguns dels camps d'un registre i prem el botó *Modifica* per canviar el seu valor.

4.-El sistema fa els canvis efectuats per l'usuari en les dades i mostra un missatge informant que els canvis han sigut correctament realitzats.

5.-L'usuari accepta el missatge i torna a la finestra on es mostren les dades.

6.-L'usuari surt de la finestra *Visualització de les dades carregades* prement el botó *Sortir.*

7.-El sistema tanca la finestra.

UC3:Calcular les mètriques.

Actor principal: l'usuari.

Precondicions: l'usuari ha engegat l'aplicació i ha carregat les dades.

Flux bàsic

1.-L'usuari prem el botó *Calcular les mètriques* o l'opció equivalent del menú *Accions*.

2.-El sistema mostra la finestra Calcular Mètriques on es troben totes les opcions possibles per dur a terme el càlcul de les mesures de la riquesa lèxica. L'usuari ha d'introduir o seleccionar el directori on es troben el arxius d'extensió *xml* que s'utilitzaran per extreure els valors necessaris pel càlcul de les mètriques, escollir les mètriques que desitja calcular, seleccionar l'atribut de classe que desitja predir, també, haurà de marcar si desitja incorporar alguns dels atributs que són utilitzats com atributs de classe juntament amb les mètriques perquè siguin considerats com a part del coneixement.

3. Anàlisi

UC3:Calcular les mètriques.

3.-L'usuari escriu el directori on es troben els arxius *xml* per dur a terme les proves.

4.-L'usuari selecciona les mètriques que vol calcular.

5.-L'usuari selecciona si vol incorporar alguns dels atributs de classe com a atributs per l'entrenament junt amb les mètriques marcant la casella corresponent.

6.-L'usuari selecciona de la llista l'atribut de classe que vol predir.

7.-L'usuari prem el botó Calcular mètriques per iniciar el procés.

9.-El sistema calcula les mètriques a partir de les opcions introduïdes per l'usuari i quan ha acabat tanca la finestra *Calcular Mètriques*. Mostra un missatge confirmant que les mètriques han sigut calculades en el panel *Estat* de la finestra principal de l'aplicació.

Fluxos alternatius

7.1.-L'usuari prem el botó *Calcular les Mètriques*, però no ha realitzat la càrrega de les dades de referència sobre els arxius *xml* (formats a partir dels textos d'opinió en llenguatge natural).

1-El sistema mostrarà un missatge d'avís on advertirà a l'usuari que, primer, ha de realitzar la càrrega de dades.
2-L'usuari prem la tecla *Acceptar*.
3-La finestreta d'avís i la pantalla *Calcular mètriques* es tanquen.

7.2.-L'usuari prem el botó *Calcular les Mètriques*, però no ha introduït cap directori on trobar els arxius *xml*.

1-El sistema mostrarà un missatge d'avís on advertirà a l'usuari que ha de seleccionar o introduir la ruta a un directori on estiguin els arxius.
2-L'usuari prem la tecla *Acceptar*.
3-La finestreta d'avís es tanca i torna a la pantalla *Calcular mètriques*.

7.3.-L'usuari prem el botó *Calcular mètriques* per fer el càlcul de les mètriques, però no ha seleccionat un mínim de quatre mètriques per les proves.

1-El sistema mostrarà un missatge d'avís on advertirà a l'usuari que ha de seleccionar un mínim de quatre mètriques per poder fer les proves.
2-L'usuari prem la tecla *Acceptar*.
3-La finestreta d'avís es tanca i torna a la pantalla *Calcular mètriques*.

3. Anàlisi

UC4: L'usuari mostra les mètriques calculades

Actor principal: l'usuari.

Precondicions: l'usuari ha engegat l'aplicació, ha carregat les dades i calculat les mètriques.

Flux bàsic

1.-L'usuari prem el botó *Mostra Mètriques* que es troba a la finestra principal de l'aplicació.

2.-El sistema mostra la finestra *Visualització de les mètriques calculades* on es mostra una taula amb tots els valors de les mètriques calculades amb els noms corresponents de les columnes. L'usuari té la possibilitat de canviar algun dels valors de les mètriques calculades directament modificant el camp de la taula.

3.-L'usuari modifica algun dels valors.

4.-L'usuari prem el botó *Acceptar*.

5.-El sistema realitza el canvis fets per l'usuari en les dades de les mètriques i tanca la finestra *Visualització de les mètriques calculades*.

Flux alternatiu

4.1- L'usuari realitza canvis en els valors, però en comptes de prémer el botó *Acceptar*, prem el botó *Cancel·lar*.

1.-El sistema tanca la finestra sense realitzar cap canvi en els valors de mètriques.

UC3: Fer la Classificació

Actor principal: l'usuari.

Precondicions: l'usuari ha engegat l'aplicació, ha carregat les dades i ha calculat les mètriques.

Flux bàsic

1.-L'usuari prem el botó de Classificació

2.-El sistema mostra la finestra *Classificació* on l'usuari ha d'introduir les funcions de classificació amb les que vol fer la prova, el número de plecs (*folds*) del *cross-validation*, el conjunt de dades que s'ha d'utilitzar per l'entrenament que poden ser les mètriques calculades o unes dades alternatives guardades en un fitxer i l'arxiu on vol guardar els resultats de les proves.

3.-L'usuari selecciona totes les funcions amb les que vol fer la prova.

5.-L'usuari escull si vol fer el *cross-validation* amb 10 o 5 plecs.

6.-L'usuari selecciona si vol utilitzar les mètriques calculades com a conjunt de dades per les proves de classificació o si vol seleccionar un arxiu que contingui les dades a utilitzar. En aquest darrer cas, haurà d'indicar la ruta o camí a l'arxiu.

3. Anàlisi

UC3: Fer la Classificació

Flux bàsic

7.-L'usuari indica el lloc i el nom de l'arxiu on vol guardar les dades resultants de la prova.

8.-L'usuari prem el botó *Comença*.

9.-El sistema inicia la prova de classificació i, un cop ha acabat, tanca la finestra *Classificació* i mostra els resultats estadístics en el quadre amb títol *Resultats anàlisi* que es troba a la finestra principal.

Fluxos alternatius

8.1-L'usuari prem el botó *Comença*, però no ha carregat, prèviament, les dades de referència dels textos.

1-El sistema mostra un avís, avisant que s'han de carregar les dades abans de poder realitzar la prova de classificació.
2-L'usuari prem el botó Acceptar del missatge d'avís
3-El sistema tanca el missatge d'avís i torna a la finestra de Classificació.

8.2-L'usuari prem el botó *Comença*, però no ha escollit cap funció de classificació.

1-El sistema mostra un avís, avisant que ha d'escollir alguna funció de classificació.
2-L'usuari prem el botó Acceptar del missatge d'avís.
3-El sistema tanca el missatge d'avís i torna a la finestra de Classificació.

8.3-L'usuari prem el botó *Comença*, però no ha determinat el conjunt de dades d'entrenament (aquest cas, només si ha seleccionat l'opció *Seleccionar unes dades diferents*, a les mètriques, sense haver introduït el camí a l'arxiu amb les dades).

1-El sistema mostra un avís, avisant que ha d'introduir la ruta a l'arxiu on es troben les dades per poder fer la prova.
2-L'usuari prem el botó Acceptar del missatge d'avís.
3-El sistema tanca el missatge d'avís i torna a la finestra de Classificació.

8.3-L'usuari prem el botó *Comença*, però no ha introduït la ruta o camí a l'arxiu on vol guardar els resultats del *cross-validation*.

1-El sistema mostra un avís, avisant que ha d'introduir la ruta a l'arxiu on es vol guardar els resultats de la prova.
2-L'usuari prem el botó Acceptar del missatge d'avís.
3-El sistema tanca el missatge d'avís i torna a la finestra de Classificació.

3. Anàlisi

UC4: Calcular funcions de selecció

Actor principal: l'usuari.

Precondicions: l'usuari ha engegat l'aplicació, ha carregat les dades de referència dels textos i ha calculat les mètriques.

Flux bàsic

1.-L'usuari prem el botó *Selecció d'atributs*, o bé, selecciona l'opció *Selecció d'atributs* del menú *Accions.*

2.-El sistema mostra la finestra *Selecció d'atributs* amb les opcions per realitzar la prova. Ha d'introduir el *dataset* per la prova, seleccionar de la llista la funció de classificació, els selectors que vol fer anar junt amb els seus avaluadors. També ha d'escriure o seleccionar el destí on han d'anar els resultats del *cross-validation*.

3.-L'usuari selecciona d'on vol extreure el dataset que pot ser de les mètriques calculades, que només serà possible seleccionar si les ha calculat prèviament, o, alternativament, d'un fitxer que contingui les dades.

4.-Selecciona la funció de classificació.

5.-Selecciona tots els selectors que vol fer anar junt amb els avaluadors en cas que vulgui.

6.-L'usuari selecciona el destí dels resultats, és a dir, l'arxiu on han d'anar.

7.-L'usuari prem el botó *Comença*.

8.-El sistema inicia la prova de selecció d'atributs i, un cop ha acabat, tanca la finestra *Selecció d'atributs* i mostra els resultats estadístics en el quadre amb títol *Resultats anàlisi* que es troba a la finestra principal.

Fluxos alternatius

7.1-L'usuari prem el botó *Comença*, però no ha carregat, prèviament, les dades de referència dels textos.

1-El sistema mostra un avís, avisant que s'han de carregar les dades abans de poder realitzar la prova de selecció d'atributs.
2-L'usuari prem el botó *Acceptar* del missatge d'avís
3-El sistema tanca el missatge d'avís i torna a la finestra Selecció d'atributs.

7.2-L'usuari prem el botó *Comença*, però no ha determinat el conjunt de dades d'entrenament (aquest cas, només si ha seleccionat l'opció *Seleccionar unes dades diferents*, a les mètriques, sense haver introduït el camí a l'arxiu amb les dades).

1-El sistema mostra un avís, avisant que ha d'introduir la ruta a l'arxiu on es troben les dades per poder fer la prova.
2-L'usuari prem el botó *Acceptar* del missatge d'avís.
3-El sistema tanca el missatge d'avís i torna a la finestra *Selecció d'atributs*.

3. Anàlisi

UC4: Calcular funcions de selecció

Fluxos alternatius

7.3-L'usuari prem el botó *Comença*, però no ha introduït la ruta o camí a l'arxiu on vol guardar els resultats del *cross-validation*.

1-El sistema mostra un avís, avisant que ha d'introduir la ruta a l'arxiu on es vol guardar els resultats de la prova.
2-L'usuari prem el botó *Acceptar* del missatge d'avís.
3-El sistema tanca el missatge d'avís i torna a la finestra *Selecció d'atributs*.

3. Anàlisi

3.4 Model de domini

Model de domini de l'eina ATOp.

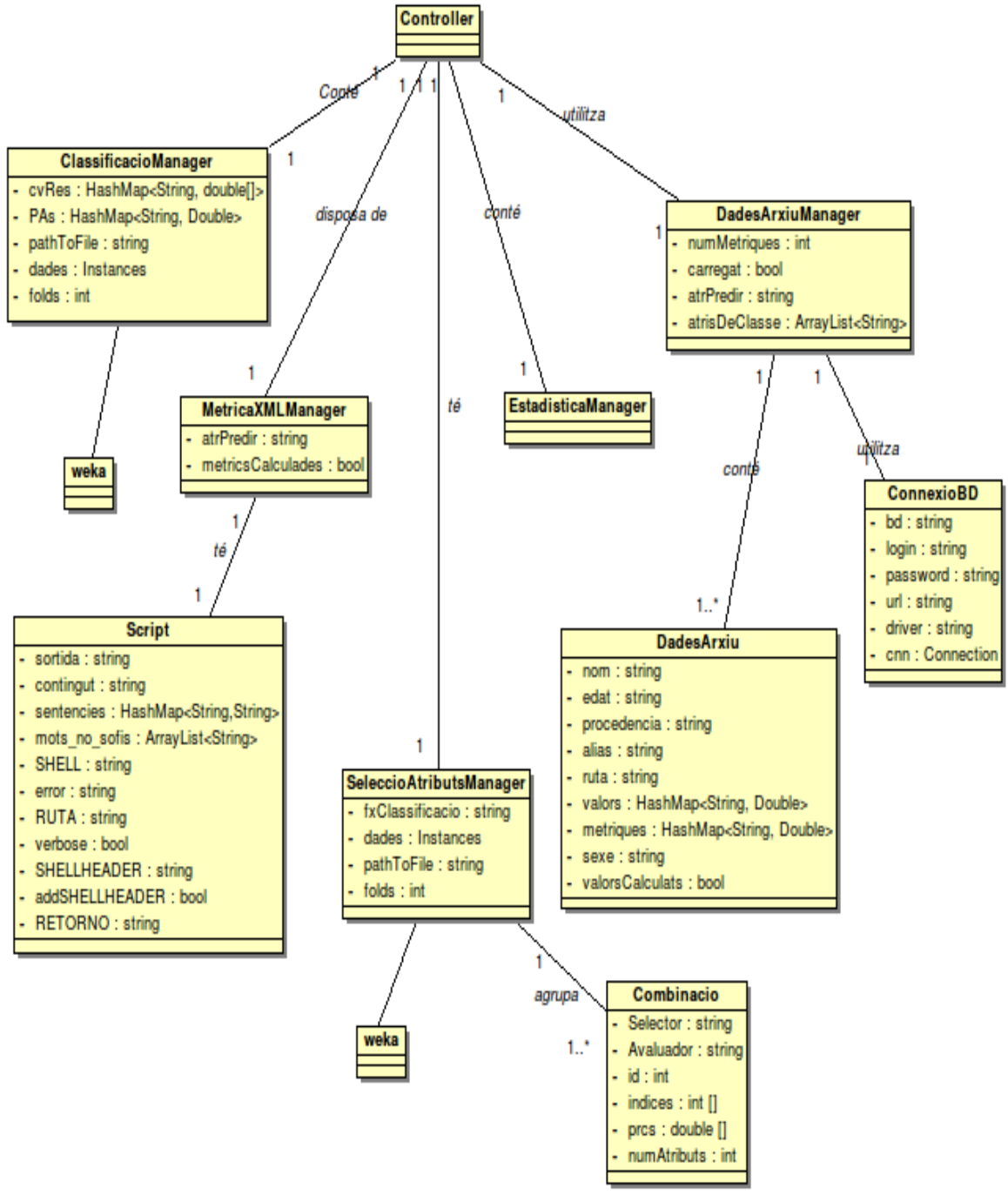

Figura 3 : Model de domini de l'eina ATOp

3. Anàlisi

3.5 Definició del model de domini

Per donar la modularitat requerida al sistema l'aplicació ATOp està dividida en diferents *packages* cadascun dels quals conté la classe o les classes que satisfan unes funcionalitats determinades. Així, les diferents classes agrupades per *packages* són:

-CarregaDades: conté les classes necessàries per estructurar tota la informació sobre els textos d'opinió i els diferents resultats.

- **DadesArxiu:** emmagatzema la informació de referència sobre cadascun dels textos d'opinió, així com els valors necessaris pel càlcul de les mètriques i el resultat de les mètriques calculades.
- **DadesArxiuManager:** s'encarrega de guardar les referències a totes les instàncies de DadesArxiu creades, proporcionant l'accés a les mateixes, així com realitzar algunes tasques com la càrrega de dades sobre els objectes DadesArxiu a fitxers pel seu processament.

-Connexió: conté la classe encarregada de realitzar la connexió amb la base de dades.

- **ConnexióBD:** es la classe encarregada de realitzar la connexió amb la base de dades, proporcionar la connexió, carregar la informació i tancar la connexió un cop finalitzada la seva missió.

-OperacionsXml: conté la classe Script que és l'encarregada d'extreure uns valors dels arxius xml mitjançant l'execució de comandes shell proporcionades per l'eina de línia de comandes XmlStarlet [22], des del codi java.

- **Script:** porta a terme les tasques necessàries per executar les *comandes shell* que permetran l'extracció dels valors dels arxius xml sobre els textos d'opinió en llenguatge natural i que són necessaris pel càlcul de les mètriques.

-Mètrica: conté la classe que s'encarrega del càlcul de les mètriques.

- **MetricaXMLManager:** és la classe encarregada de realitzar el càlcul de les mètriques, valent-se de la classe *Script* per obtenir els valors amb els que fer els càlculs, i guardar els resultats dins de cada instància de l'objecte *DadesArxiu,* pel seu posterior processament.

-Mineria: conté les classes que s'encarreguen de tot el procés d'aprenentatge automàtic i mineria de dades, utilitzant la llibreria weka [1].

- **ClassificacióManager:** és la classe encarregada de realitzar les proves d'entrenament automàtic, a partir de les mètriques dels diferents textos (fent *cross-validation*).
- **SelecciónAtributsManager:** és la classe encarregada de fer la selecció d'atributs a partir de les mètriques i determinar quines mètriques tenen més pes a l'hora de predir les característiques.
- **Combinacio:** és la “classe suport” que guarda la combinació de selector+avaluador feta per l'usuari, així com tots els resultats que aquesta combinació generi.

3. Anàlisi

3.5 Definició del model de domini

-Estadística: conté la classe que s'encarrega de calcular estadístiques sobre els resultats obtinguts en la fase d'aprenentatge automàtic.

- **EstadísticaManager:** a partir dels resultats de les proves, tant de classificació com de selecció d'atributs realitza tasques estadístiques per mostrar-les en l'aplicació.

-analisiTextos: conté la nostra classe *controller* que serà l'encarregada de mostrar l'entorn gràfic a l'usuari permetent-li fer accions i recollint les seves peticions per ser processades.

- **AnalisiTextos:** és la nostra classe interfície d'usuari que permetrà a l'usuari dur a terme les funcionalitats de l'aplicació a partir d'un entorn gràfic.

3.6 Requeriments generals: *software* i *hardware*

- El projecte ha estat realitzat amb l'entorn de programació *NetBeans* i totes les proves s'han realitzat amb el SO Ubuntu 12.04
- S'ha d'instal·lar l'eina XmlStarlet [22] per executar comandes *shell* per obtenir els valors necessaris pel càlcul de les mètriques a partir dels arxius *xml*.
- Es requereix 1 GB mínim de memòria RAM donat que s'han de guardar tots els valors que es requereixen per les proves d'anàlisi.
- Es requereix d'un processador d'almenys 1 Ghz, donat que s'ha de processar moltes dades tant pel càlcul de les mètriques com la realització de les proves de mineria de dades.

4. Disseny

4.1 Diagrames d'interaccions

A continuació es mostren els diagrames de seqüència pels events de sistema que corresponen a cada cas d'ús.

DS1. L'usuari realitza la càrrega de les dades

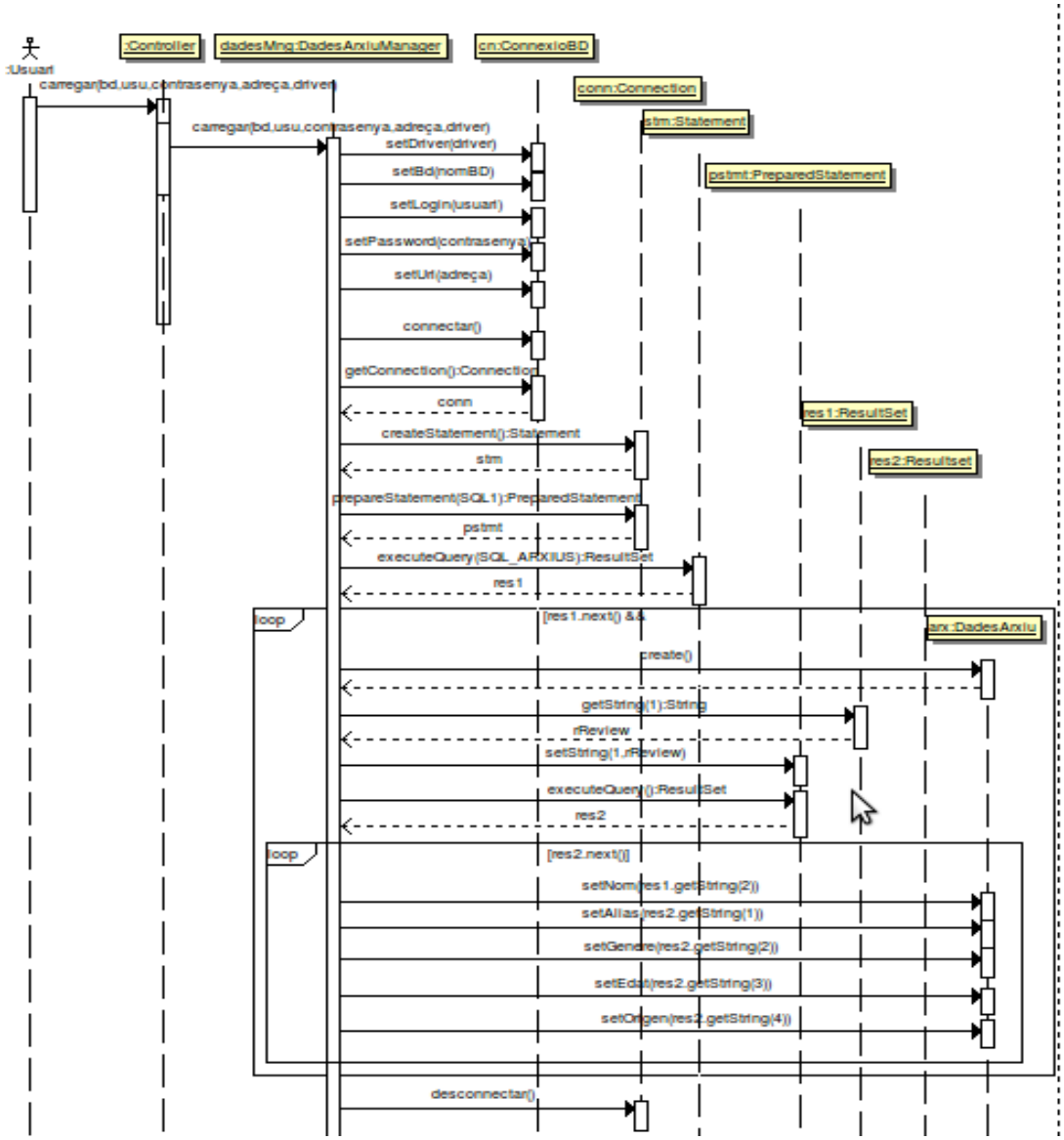

Figura 4: DS1 L'usuari realitza la càrrega de les dades

Explicació: l'usuari selecciona el *driver* del SGBD, introdueix el nom de la base de dades(hopinion en aquest cas), l'usuari autoritzat, la contrasenya, i l'adreça IP de la màquina on es troba la informació. Després prem el botó *Carregar* i amb les dades introduïdes es crea la connexió a partir de la qual s'executaran les consultes *sql* amb les quals descarregarem les dades que necessitem sobre els autors dels textos d'opinió i anirem creant els objectes DadesArxiu a cada iteració on guardarem aquestes dades per cada text.

4. Disseny

DS2. L'usuari realitza el càlcul de les mètriques

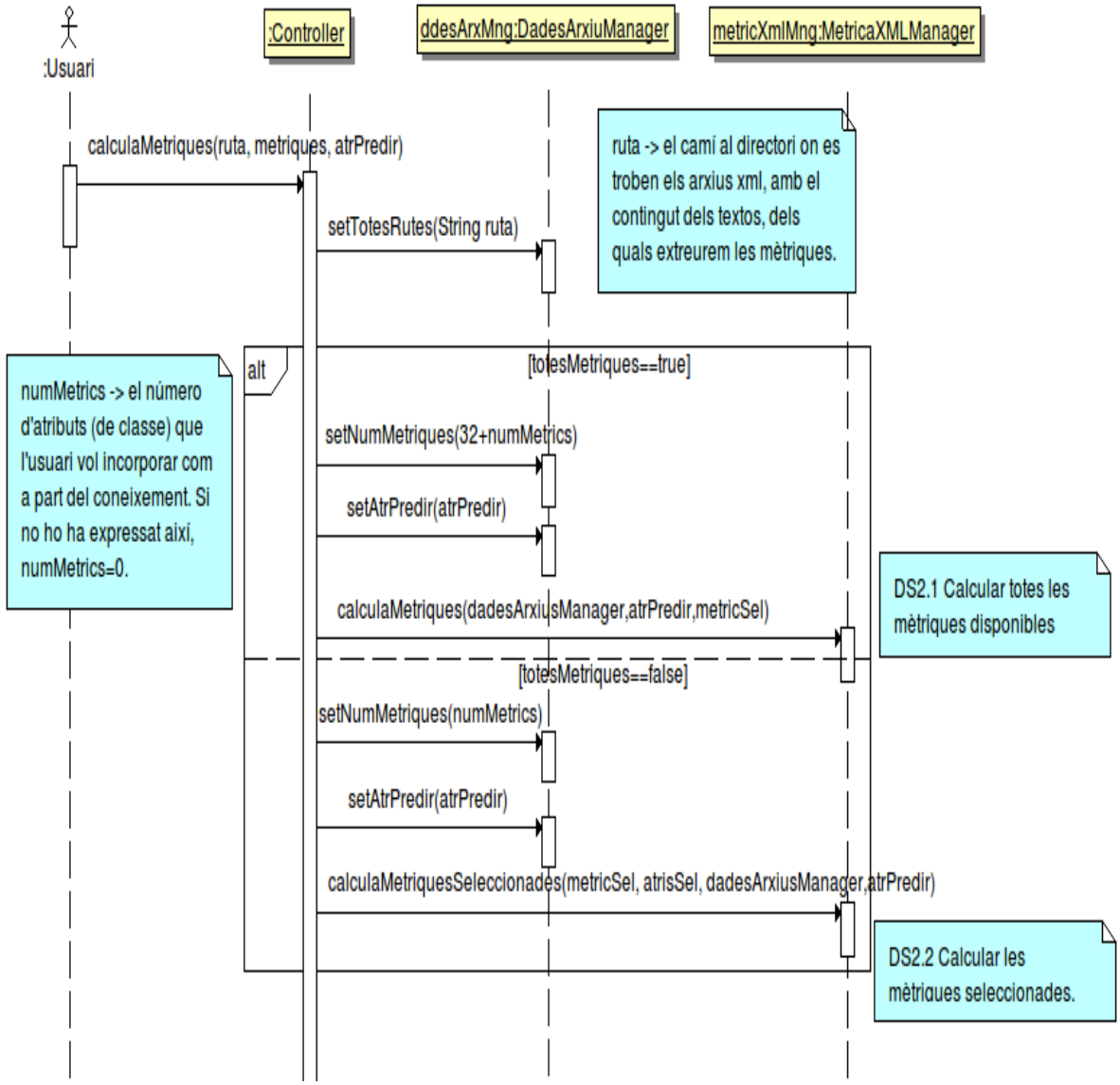

Figura 5: DS2. L'usuari realitza el càlcul de les mètriques

Explicació: L'usuari fa el càlcul de les mètriques. Primer, indica el camí al directori on es troben els arxius *xml* i selecciona les mètriques que vol calcular, així com l'atribut de classe. Tot seguit, el sistema estableix el camí en cada objecte *ArxiuDades* (setRuta()) a través de la classe *DadesArxiuManager* on es troba la llista dels objectes *ArxiuDades* cadascun dels quals guarda la informació referent a un arxiu *xml*. També emmagatzema el número de mètriques escollides i el nom de l'atribut de classe dins la classe DadesArxiuManager. Després crida a la funció de la classe *MetricaXMLManager* per calcular totes les mètriques, o només les seleccionades per l'usuari, passant-li com a arguments, les mètriques seleccionades, la instància de *DadesArxiuManager* i l'atribut de classe a predir.

4. Disseny

DS2.1 Calcular totes les mètriques disponibles

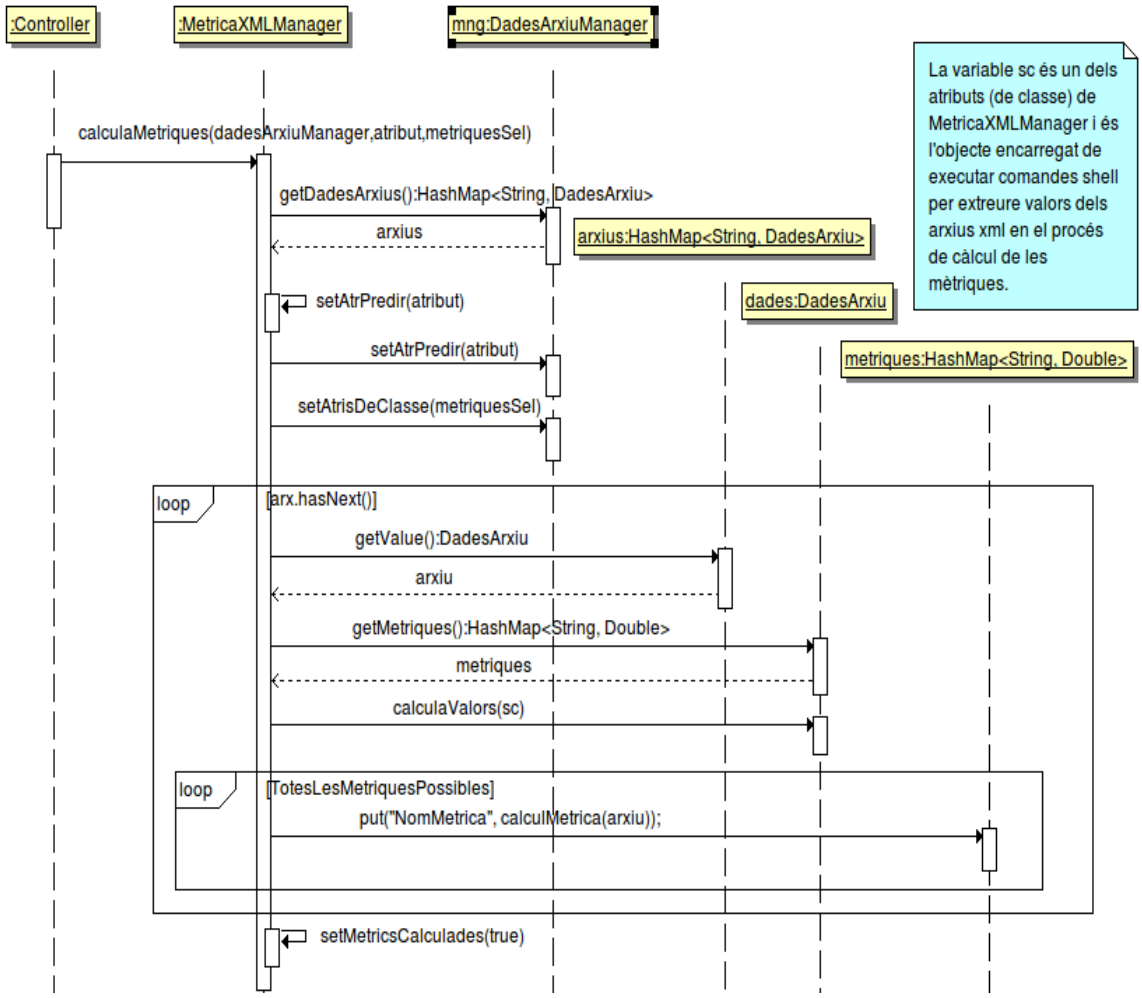

Figura 6: DS2.1 Calcular totes les mètriques disponibles

Explicació: es crida al métode *calculaMetriques*() de la classe *MetricaXMLManager* per calcular totes les mètriques. Primer, dins la funció obtenim totes les instàncies de la classe *DadesArxiu* amb *getDadesArxius()* de la classe *DadesArxiuManager*. Guarda l'atribut a predir i després per cada instància de la classe DadesArxiu calcula els valors que es necessiten per calcular les mètriques cridant la funció *calculaValors(sc)* passant-li com a argument la instància *sc* de la classe *Script* que s'encarregarà d'executar les comandes *shell* sobre els arxius xml per l'extracció d'aquests valors. Un cop tenim els valors calcularem totes les mètriques (*TotesLesMetriquesPossibles*) amb la funció corresponent *calculaMetrica()* que tindrà el mateix nom que la mètrica que calcula i els resultats seran guardats a la instància corresponent de la classe DadesArxiu. Finalment, fixarem que les mètriques han sigut calculades amb el métode *setMetricsCalculades(true)*.

4. Disseny

DS2.2 Calcular les mètriques seleccionades per l'usuari

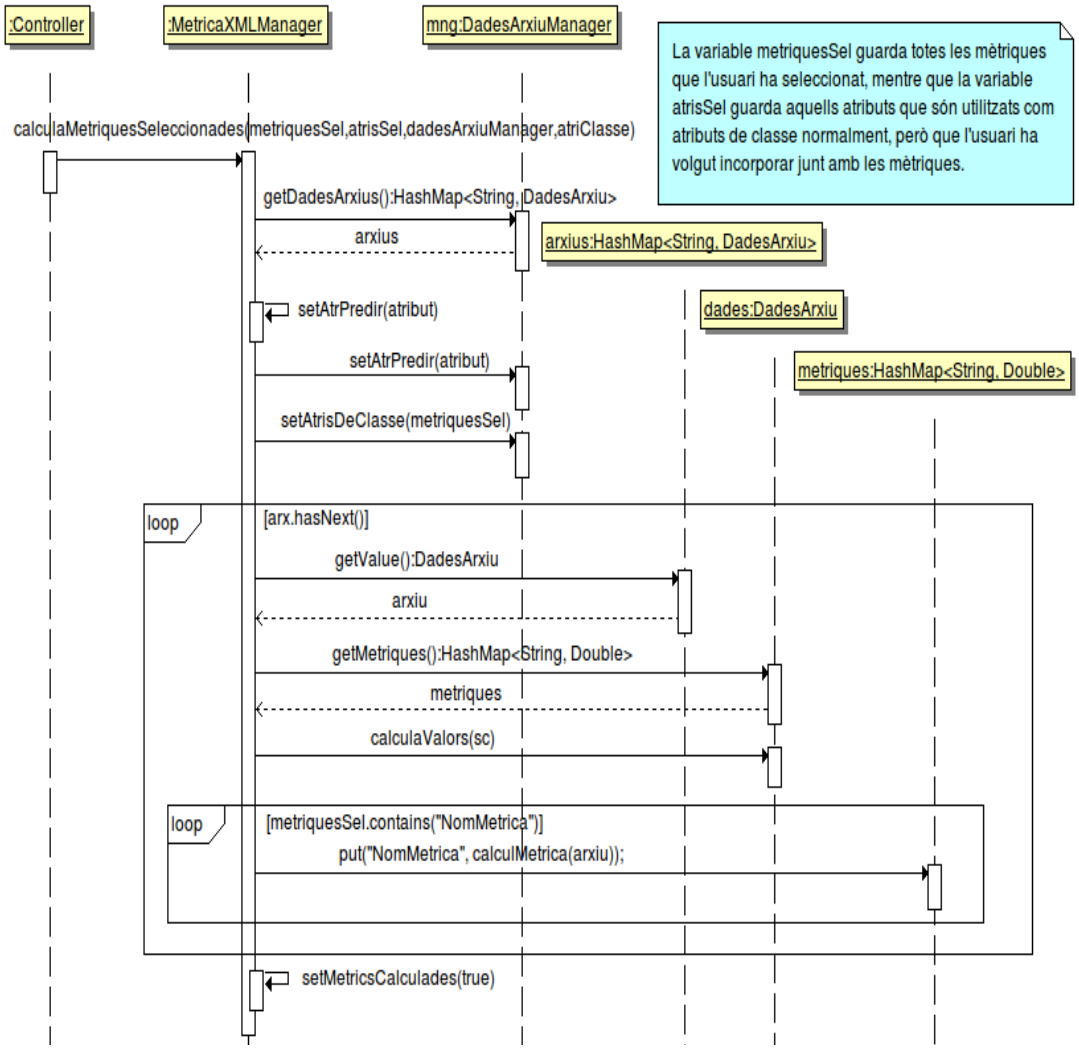

Figura 7: DS2.2 Calcular les mètriques seleccionades per l'usuari

Explicació: igual que el diagrama DS2.1 amb la diferència que, en comptes de calcular totes les mètriques, es calculen les que l'usuari ha seleccionat a través de l'entorn gràfic. Com a arguments per la funció *calculaMetriquesSeleccionades()* tenim la variable *metriquesSel* que guarda la llista de mètriques seleccionades per l'usuari i la variable *atriSel* que guarda la llista del atributs que són utilitzats com a atributs de classe i que l'usuari ha inclòs junt amb les mètriques (si es donés el cas). D'aquesta manera i, per cada instància de la classe *DadesArxiu* es comprova si les mètriques apareixen a la llista de les mètriques escollides (*metriquesSel*) per l'usuari i en cas que hi siguin es calculen i guarden els resultats.

4. Disseny

DS3. L'usuari realitza una prova de classificació amb les mètriques que ha calculat prèviament.

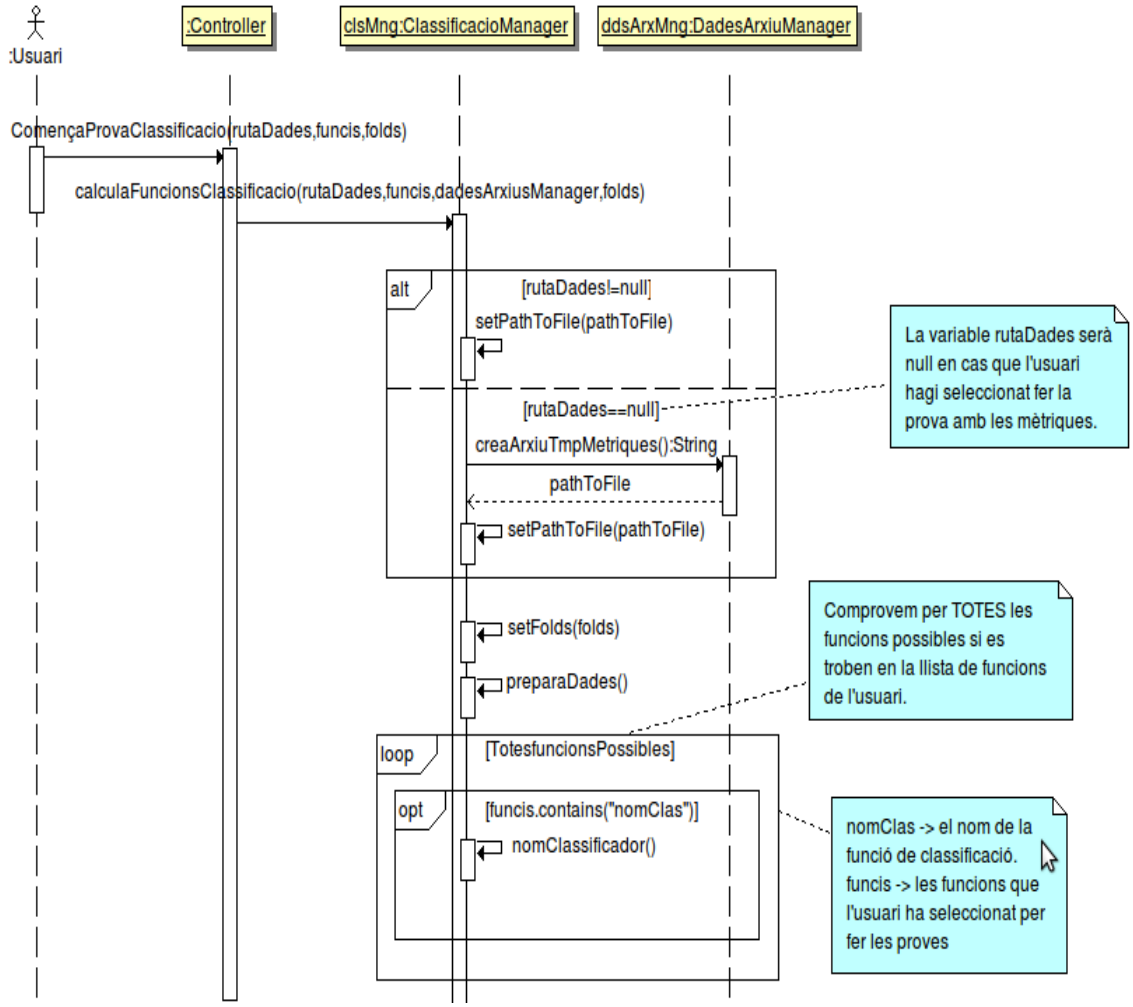

Figura 8: DS3. L'usuari realitza una prova de classificació amb les mètriques

Explicació: L'usuari realitza una prova de classificació on, primerament, escull les funcions de classificació que vol, de les opcions, el número de *folds* (pel *cross-validation*), l'origen de les dades, és a dir, si vol utilitzar les dades de les mètriques, ja calculades, o un arxiu concret (cas en el qual haurà d'indicar la ruta d'accés) i, finalment, el directori on vol guardar els resultats de la prova. Amb el valor dels arguments introduïts per l'usuari, el controlador crida a la funció *calculaFuncionsClassificacio()* de la classe *ClassificacióManager* que s'encarrega de inicialitzar els valors per la prova, preparar les dades i, tot seguit, per cadascuna de les funcions de classificació escollides crida a la funció corresponent dins la classe, que té el mateix nom que el classificador i aquesta s'encarrega de crear el classificador i fer el *cross-validation*, cridant a la funció *makeCrossValidation(Classificador, nomFunciónClassificació)* on els arguments són l'objecte *Classifier* (de la funció de classificació en qüestió) i el nom de la funció de classificació. Després del càlcul dels 10 percentatges del cross-validation per cada classificador es posen els resultats dins el *HashMap cvRes* junt amb el nom del classificador utilitzat.

4. Disseny

DS4. L'usuari realitza una prova de selecció d'atributs amb les mètriques com a conjunt de dades.

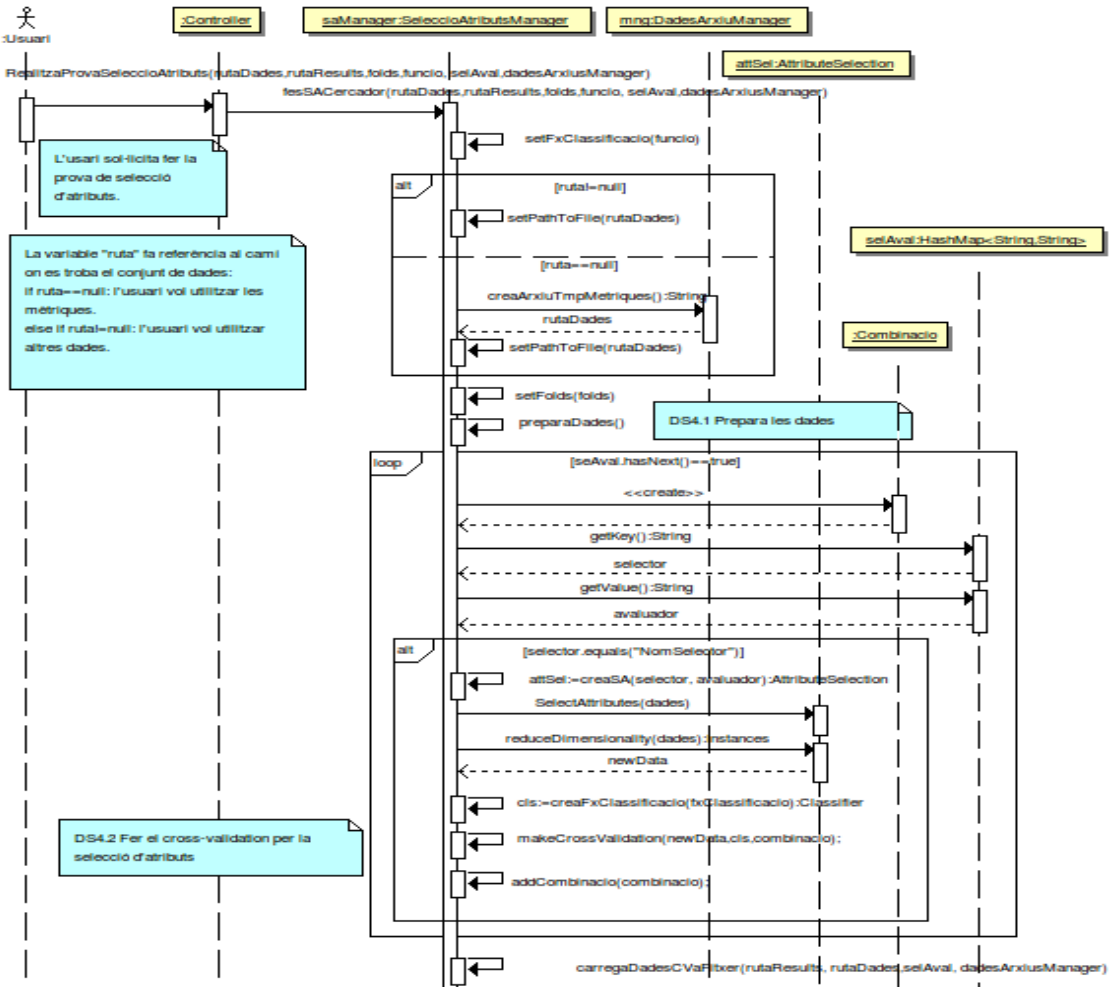

Figura 9 : DS4. L'usuari realitza una prova de selecció d'atributs amb les mètriques com a conjunt de dades.

Explicació: L'usuari realitza una prova de selecció d'atributs. Indica el classificador, el conjunt de dades, el número de *folds*, totes les combinacions de selectors amb avaluadors que vol fer i el fitxer on vol guardar els resultats. Si el conjunt de dades escollit és el de les mètriques la classe SeleccioAtributsManager generarà un arxiu temporal amb els valors de les mètriques cridant la funció *creaArxiuTmpMetriques*(). A més, aquesta classe guardarà el número de *folds* i prepararà les dades pel cross-validation amb *preparaDades()*. Després per cadascuna de les combinacions escollides per l'usuari el sistema crearà un objecte Combinació que guardarà el nom del selector, del avaluador, un identificador i tots els resultats generats per aquesta combinació. De cada combinació el sistema crearà un objecte *AttributeSelection* al qual li assignarà el selector i l'avaluador de la combinació actual. Aquest realitzarà la fase de selecció d'atributs amb la funció *selectAttributes(dades)* i redimensió de les dades amb els atributs seleccionats, *reduceDimensionality()*. Tot seguit, la classe *SeleccioAtributsManager* crearà el classificador amb la funció *creaFxClassificació(nomFxClassificació)*, cridarà la funció *makeCrossValidation(newData, classificador, combinacio)*, guardarà l'objecte Combinacio dins la llista de Combinacions i, finalment, cridarà a la funció *carregaDadesCVaFitxer()* que s'encarregarà de crear el fitxer amb els resultats del *cross-validation*.

4. Disseny

DS4.1 Prepara les dades

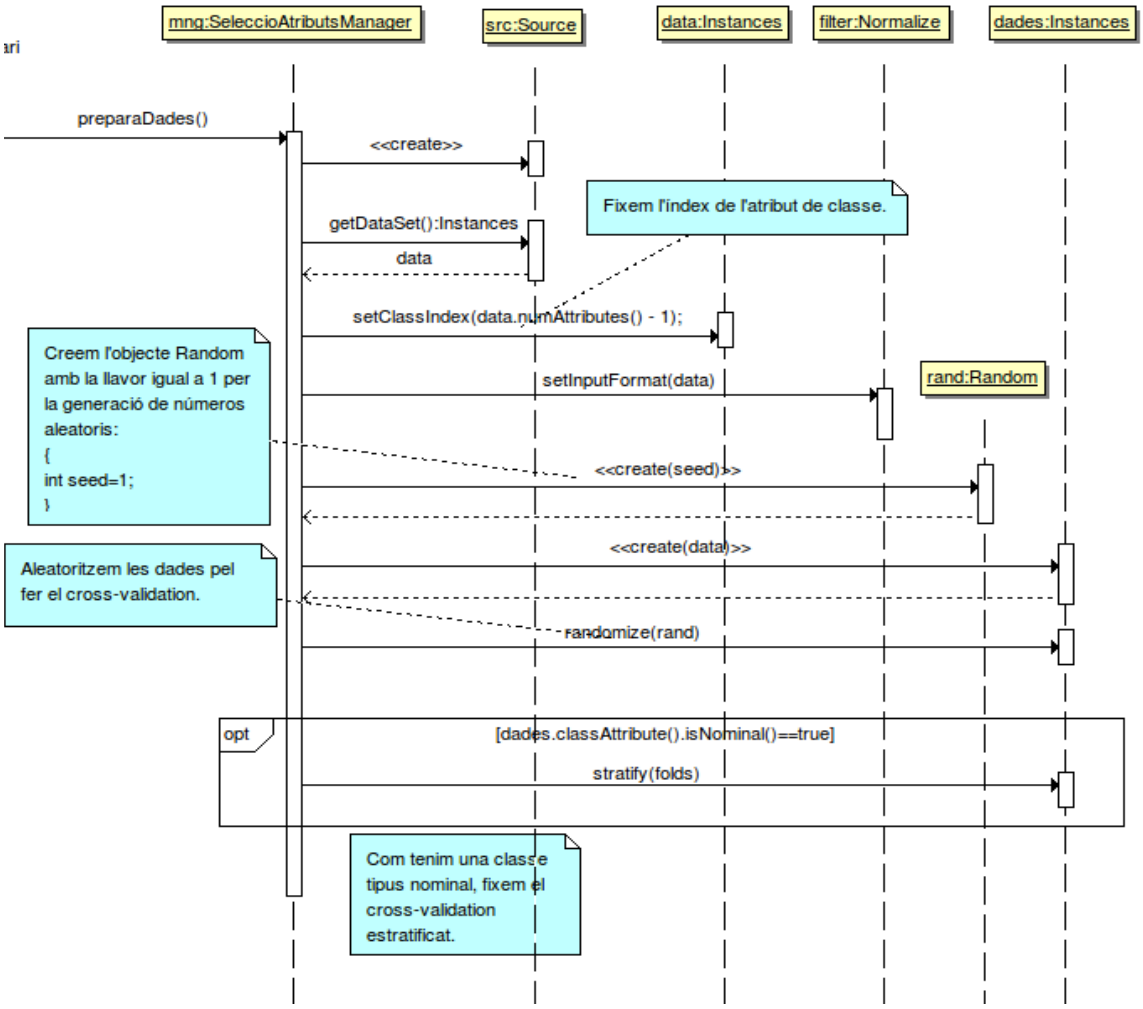

Figura 10 : DS4.1 Preparar les dades

Explicació: la funció *preparaDades()* s'encarrega de carregar des d'un fitxer (*pathToFile*) les dades amb les quals es farà la prova. Les passa a format de l'objecte *Instances.* Un cop les dades es troben carregades al format de l'objecte *Instances*, es crida a la funció *setClassIndex(data.numAttributes() - 1)* d'aquesta classe per fixar l'atribut de classe. Després normalitza els valors de les dades creant un filtre de la classe *Normalize* i cridant a la funció *setInputFormat(Instances data)* d'aquest filtre. Tot seguit, aleatoritzem les dades per fer el *cross-validation* creant un objecte de la classe Random al qual li passem com a llavor un 1 i, a continuació, es crida a la funció *randomize()* de la classe Instances passant-li com a argument la instància rand de la classe Random creada. Finalment, "estratifiquem" les dades per poder fer el *cross-validation* amb l'atribut de classe nominal.

4. Disseny

DS4.2 Fer el cross-validation per la selecció d'atributs

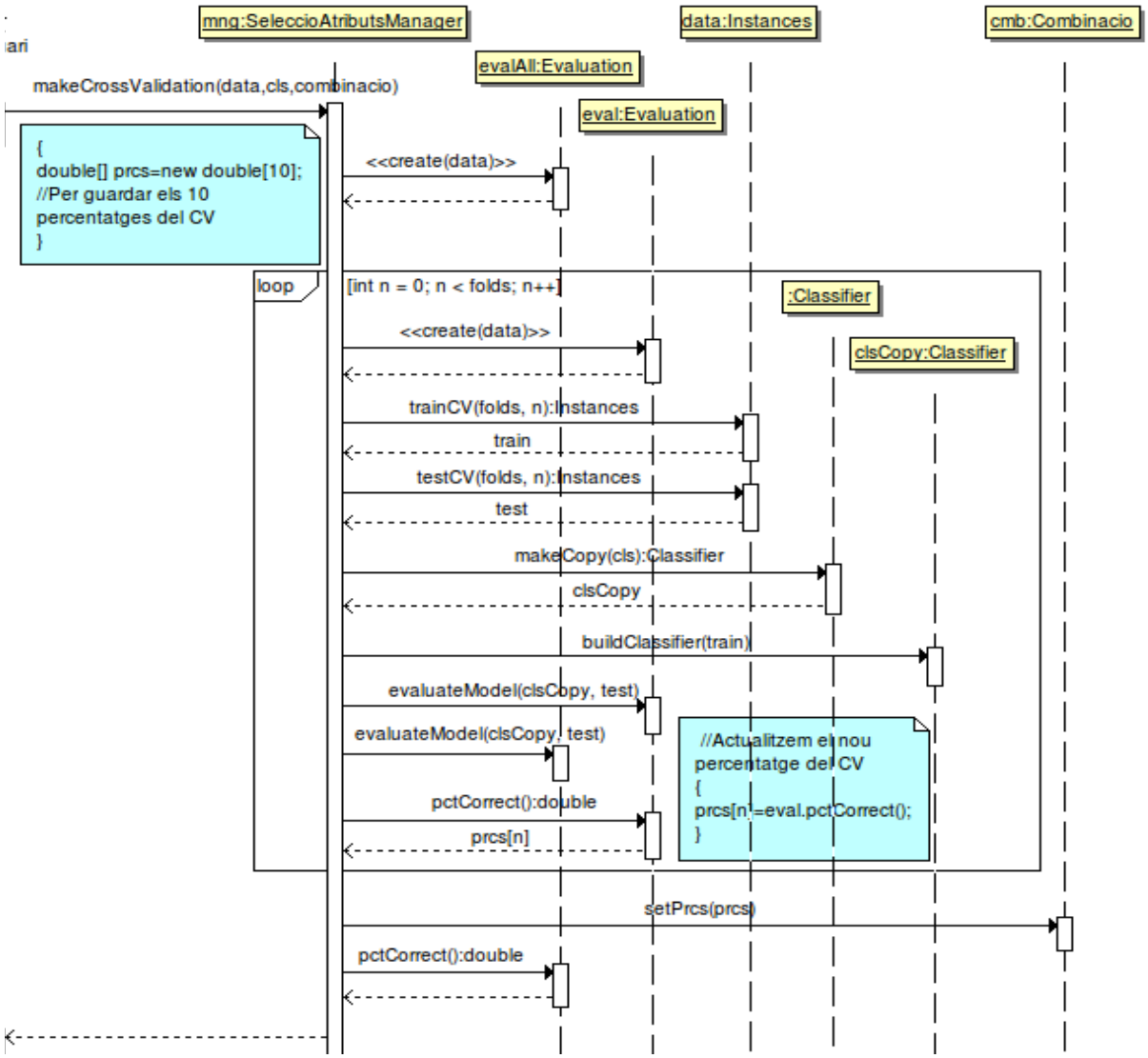

Figura 11: DS4.2 Fer el *cross-validation* per la selecció d'atributs

Explicació: el mètode *makeCrossValidation(data,cls,combinacio)* s'encarrega de fer el cross-validation per les dades (data) i el classificador que li passem com a arguments. La funció realitza 10 plecs (*folds)* en cadascun dels quals genera el conjunt de dades d'entrenament i test segons el plec en el qual es troba. En cada plec s'entrena el classificador amb les dades d'entrenament, cridant la funció *buildClassifier(train)* de la classe *Classifier* i després s'avalua el classificador cridant a la funció *evaluateModel()* de la classe *Evaluation*. Es crida la funció *pctCorrect()* de la classe *Evaluation* per obtenir el percentatge d'encert del plec actual després d'haver fet l'entrenament. Un cop s'han realitzat tots els plecs i guardat tots els percentatges d'encert de cadascun, aquests són guardats en l'objecte *combinacio* de la classe *Combinacio* que se li havia passat a la funció com a argument. Al final, després de la realització de tots els plecs, la funció retorna el percentatge promig de tots els plecs cridant la funció pctCorrect() de la instància evalAll de la classe Evaluation.

4.-Disseny

4.2 Diagrama de classes

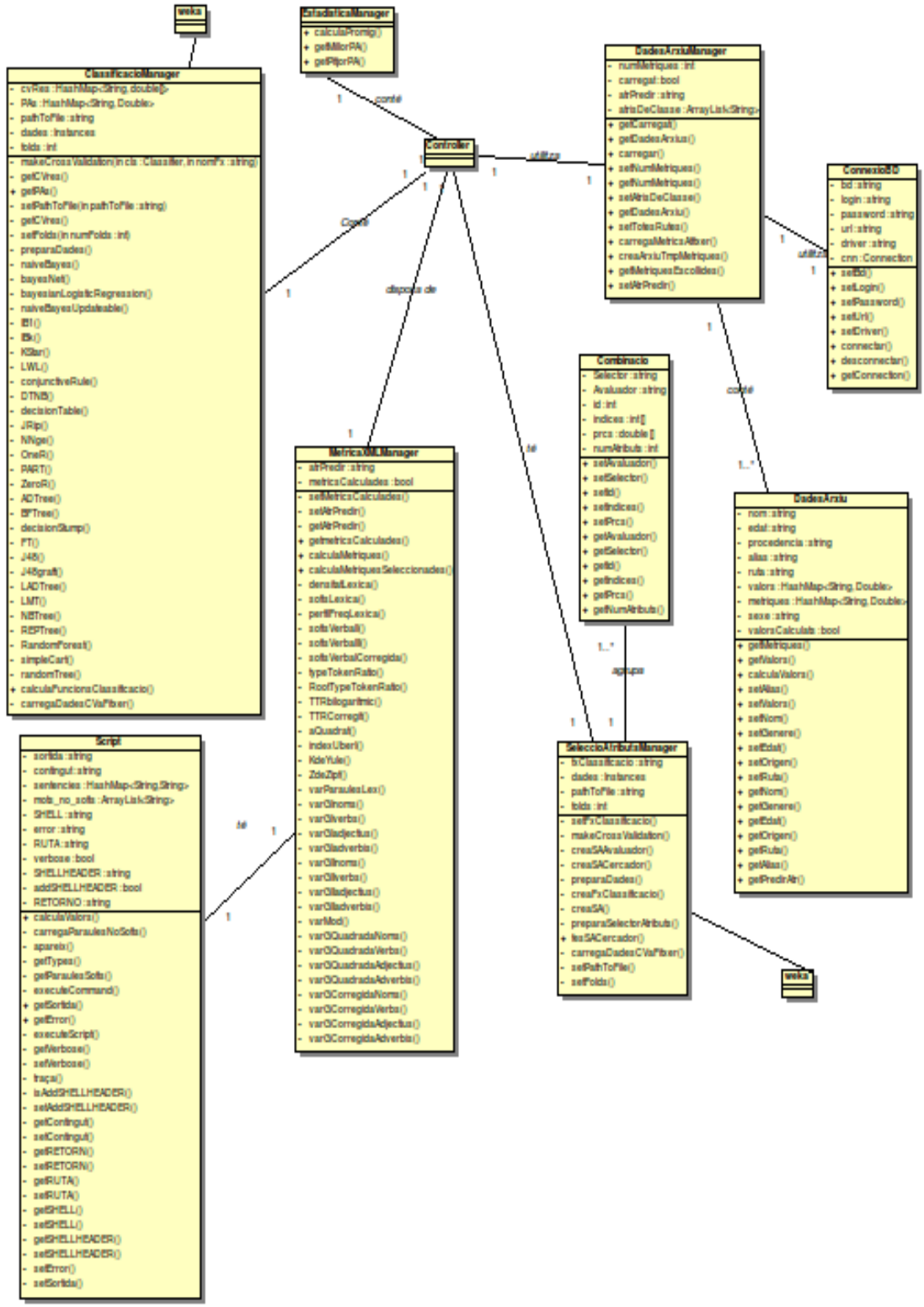

Figura 12 : Diagrama de classes de l'eina ATOp

4.-Disseny

4.2 Diagrama de classes

Anem a veure amb detall cadascuna de les classes amb els seus atributs i métodes.

DadesArxiu

Atributs:

-nom: nom de l'autor del text d'opinió
-edat: edat de l'autor del text d'opinió
-procedència: origen de l'autor del text d'opinió.
-sexe: sexe de l'autor del text d'opinió.
-alias: àlies de l'autor del text d'opinió.
-ruta: camí on es troba l'arxiu *xml* amb el qual farem l'anàlisi -valors: valors necessaris pel càlcul de les mètriques.
-valorsCalculats: per determinar si els valors han sigut ja calculats.
-metriques: valors de les mètriques.

Métodes:

-getMetriques: retorna els valors de les mètriques calculades.
-getValors: retorna els valors utilitzats en el càlcul de les mètriques.
-calculaValors: calcula els valors que seran utilitzats per obtenir les mètriques.
-setAlias: assigna l'àlies de l'autor del text.
-setValors: assigna els valors amb els quals es calculen les mètriques -setNom: assigna el nom de l'autor del text.

-setGenere: assigna el sexe de l'autor del text.

DadesArxiuManager

Atributs:

-arxiusDades: tots els DadesArxiu cadascun amb la informació sobre un text d'un autor.
-cn: per establir la connexió amb la base de dades.
-numMetriques: número de mètriques escollides per fer les proves -carregat: per conèixer si les dades de referència sobre els textos han sigut carregades.
-atrPredir: l'atribut de classe que es vol predir (edat, sexe o origen) en un moment donat.
-atrisDeClasse: característiques de l'autor que es volen incorporar junt amb les mètriques.

Métodes:

-getCarregat: retorna si les dades han sigut ja carregades o no encara.
-getDadesArxius: retorna tots els DadesArxius que hagin sigut creats.
-carregar: fa la càrrega a memòria de les dades de referència sobre els textos.
-setNumMetriques: assigna el número de mètriques escollides per fer l'anàlisi.
-getNumMetriques: retorna el número de mètriques escollides per fer l'anàlisi.
-setAtrisDeClasse: assigna les característiques de l'autor junt amb les mètriques.
-getDadesArxiu: retorna tots els *DadesArxiu* creats amb informació dels textos.
-setTotesRutes: assigna a tots els *DadesArxiu* el directori on es troben els arxius xml dels textos.
-carregaMetricsAfitxer: escriu les mètriques calculades dins un fitxer *csv* o *arff*.

4.-Disseny

DadesArxiuManager

-creaArxiuTmpMetriques: escriu les mètriques dins un fitxer temporal amb extensió *arff*.
-getMetriquesEscollides: retorna la llista de les mètriques escollides per l'usuari.
-setAtrPredir: assigna l'atribut de classe.

ConnexioBD

Atributs:
-bd: nom de la base de dades amb la que connectarem.
-login: nom de l'usuari autoritzat per l'accés.
-password: contrasenya per accedir a la base de dades.
-url: adreça completa per permetre la connexió amb la base de dades.
-driver: nom del controlador pel Sistema Gestor de Bases de Dades.
-adreça: adreça *ip* de la màquina on es troba la base de dades.
-conn: objecte *Connection* que permetrà la connexió amb la base de dades.

Métodes:
-setBd: assigna el nom de la base de dades (hopinion).
-setLogin: assigna el nom de l'usuari autoritzat.
-setPassword: assigna la contrasenya.
-setUrl: assigna l'adreça completa a partir de l'ip i controlador.
-setDriver: assigna el controlador pel SGBD.
-connectar: estableix la connexió amb la base de dades.
-desconnectar: tanca la connexió amb la base de dades.
-getConnection: recupera la connexió.

MetricaXMLManager

Atributs:
-atrPredir: l'atribut de classe a predir.
-sc: objecte de la classe *Script* per executar les comandes *shell* des de java.
-metricsCalculades: per saber si les mètriques han sigut calculades o no.

Métodes:
-setMetricsCalculades: assigna si les mètriques han sigut calculades.
-setAtrPredir: assigna l'atribut de classe.
-getAtrPredir: retorna l'atribut de classe.
-getmetricsCalculades: retorna si les mètriques han sigut calculades o no.
-calculaMetriques: calcula totes les mètriques.
-calculaMetriquesSeleccionades: calcula les mètriques escollides per l'usuari.
-densitatLexica: calcula la Densitat Lèxica.
-sofisLexica: calcula la Sofisticació Lèxica.
-perfilFreqLexica: calcula el Perfil de Freqüència Lèxica.
-sofisVerbalI: calcula la Sofisticació Verbal I.
-sofisVerbalII: calcula la Sofisticació Verbal II.
-sofisVerbalCorregida: calcula la Sofisticació Verbal Corregida.

4. Disseny

MetricaXMLManager

-typeTokenRatio: calcula el Type/Token Ratio.
-RootTypeTokenRatio: calcula el Root Type/Token Ratio.
-TTRbilogaritmic: calcula el Type Token Ratio Bilogarítmic.
-TTRCorregit: calcula el Type Token Ratio Corregit.
-aQuadrat: calcula a^2.
-indexUberI: calcula l'índex d'Úber I.
-KdeYule: calcula la K de Yule.
-ZdeZipf: calcula la Z de Zipf.
-varParaulesLex: calcula la Variació de Paraules Lèxiques.
-varGInoms: calcula la Variació G I de noms.
-varGIverbs: calcula la Variació G I de verbs.
-varGIadjectius: calcula la Variació G I d'adjectius.
-varGIadverbis: calcula la Variació G I d'adverbis.
-varGIInoms: calcula la Variació G II de noms.
-varGIIverbs: calcula la Variació G II de verbs.
-varGIIadjectius: calcula la Variació G II d'adjectius.
-varGIIadverbis: calcula la Variació G II d'adverbis.
-varMod: calcula la Variació Mod.
-varGQuadradaNoms: calcula la Variació G Quadrada de noms.
-varGQuadradaVerbs: calcula la Variació G Quadrada de verbs.
-varGQuadradaAdjectius: calcula la Variació G Quadrada d'adjectius.
-varGQuadradaAdverbis: calcula la Variació G Quadrada d'adverbis.
-varGCorregidaNoms: calcula la Variació G Corregida de noms.
-varGCorregidaVerbs: calcula la Variació G Corregida de verbs.
-varGCorregidaAdjectius: calcula la Variació G Corregida d'adjectius.
-varGCorregidaAdverbis: calcula la Variació G Corregida d'adverbis.

ClassificacioManager

Atributs:

-dades: les dades (en format Instances) amb les quals farem les proves d'entrenament.
-pathToFile: ruta a l'arxiu on es troba el conjunt de dades (les mètriques).
-folds: número de *folds* pel cross-validation.
-Pas: percentatge promig d'encert per cada classificador.
-cvRes: percentatges d'encert dels *10-fold CV* per cada classificador.

Métodes:

-getCVres: retorna els percentatges d'encert del *cross-validation* per classificador.
-getPAs: retorna els percentatges promig per classificador.
-setPathToFile: assigna la ruta a l'arxiu on es troba el conjunt de dades.
-setFolds: assigna el número de *folds* pel *CV* -preparaDades: carrega les dades de l'arxiu en l'objecte Instances i li dóna format.
-makeCrossValidation: realitza la prova de *cross-validation* pel classificador donat.

4.-Disseny

ClassificacioManager

-naiveBayes: crea el classificador NaiveBayes i crida a makeCrossValidation.
-bayesNet: crea el classificador BayesNet i crida a makeCrossValidation.
-bayesianLogisticRegression: crea el classificador BayesianLogisticRegression.
-naiveBayesUpdateable: crea el classificador NaiveBayesUpdateable.
-IB1: crea el classificador IB1 i crida a makeCrossValidation.
-Ibk: crea el classificador Ibk i crida a makeCrossValidation.
-Kstar: crea el classificador Kstar i crida a makeCrossValidation.
-LWL: crea el classificador LWL i crida a makeCrossValidation.
-ConjunctiveRule: crea el classificador ConjunctiveRule i crida a makeCrossValidation.
-DTNB: crea el classificador DTNB i crida a makeCrossValidation.
-DecisionTable: crea el classificador DecisionTable i crida a makeCrossValidation.
-Jrip: crea el classificador Jrip i crida a makeCrossValidation.
-Nnge: crea el classificador Nnge i crida a makeCrossValidation.
-OneR: crea el classificador OneR i crida a makeCrossValidation.
-PART: crea el classificador PART i crida a makeCrossValidation.
-Ridor: crea el classificador Ridor i crida a makeCrossValidation.
-ZeroR: crea el classificador ZeroR i crida a makeCrossValidation.
-ADTree: crea el classificador ADTree i crida a makeCrossValidation.
-BFTree: crea el classificador BFTree i crida a makeCrossValidation.
-decisionStump: crea el classificador DecisionStump i crida a makeCrossValidation.
-FT: crea el classificador FT i crida a makeCrossValidation.
-J48: crea el classificador J48 i crida a makeCrossValidation.
-J48graft: crea el classificador J48graft i crida a makeCrossValidation.
-LADTree: crea el classificador LADTree i crida a makeCrossValidation.
-LMT: crea el classificador LMT i crida a makeCrossValidation.
-NBTree: crea el classificador NBTree i crida a makeCrossValidation.
-REPTree: crea el classificador REPTree i crida a makeCrossValidation.
-RandomForest: crea el classificador RandomForest i crida a makeCrossValidation.
-simpleCart: crea el classificador SimpleCart i crida a makeCrossValidation.
-randomTree: crea el classificador RandomTree i crida a makeCrossValidation.
-calculaFuncionsClassificacio: fa l'entrenament per les funcions escollides utilitzant *CV*.
-carregaDadesCVaFitxer: escriu els resultats del *CV* de tots els classificadors en un fitxer.

SeleccioAtributsManager

Atributs:

-fxClassificacio: nom del classificador utilitzat per l'anàlisi.
-dades: les dades (en format Instances) amb les quals farem les proves d'entrenament.
-pathToFile: ruta a l'arxiu on es troba el conjunt de dades (les mètriques).
-folds: número de *folds* pel cross-validation.
-combinacions: llista de combinacions de selectors + avaluadors.

Mètodes:

-setFxClassificacio: assigna el nom del classificador per la prova.
-makeCrossValidation: fa l'entrenament per un classificador i una combinació -

4.-Disseny

SeleccioAtributsManager

-creaSAAvaluador: crea l'avaluador.
-creaSACercador: crea el selector.
-preparaDades: carrega les dades de l'arxiu en l'objecte Instances i li dóna format.
-creaFxClassificacio: crea el classificador.
-creaSA: crea l'objecte *AttributeSelection* per fer la selecció d'atributs.
-preparaSelectorAtributs: prepara l'objecte *AttributeSelection*
-fesSACercador: fa selecció d'atributs per les combinacions de selectors + avaluadors + *cls*.
-carregaDadesCVaFitxer: escriu els resultats del *CV* de totes les combinacions en un fitxer.
-setPathToFile: assigna la ruta al fitxer on es troben el conjunt de dades.
-setFolds: assigna el número de *folds*.

Combinacio

Atributs:

-avaluador: nom de l'avaluador.
-selector: nom del selector.
-id: l'identificador de la combinació.
-indices: els índex dels atributs seleccionats donada la combinació.
-prcs: els percentatges d'encert resultants del *cross-validation*.
-numAtributs: número d'aributs seleccionats.

Métodes:

-setAvaluador: assigna el nom de l'avaluador.
-setSelector: assigna el nom del selector.
-setId: assigna l'identificador de la combinació.
-setIndices: assigna els índex dels atributs seleccionats i el número d'atributs seleccionats.
-setPrcs: assigna els percentatges del *CV*.
-getAvaluador: retorna l'avaluador.
-getSelector: retorna el selector.
-getId: retorna l'identificador.
-getIndices: retorna els índex dels atributs seleccionats.
-getPrcs: retorna els percentatges d'encert del *CV.*
-getNumAtributs: retorna el número d'atributs seleccionats.

4.-Disseny

Script

Atributs:

-contingut: contingut del script a executar.
-SHELL: intèrpret de comandes.
-sortida: resultat de l'execució de la comanda.
-error: errors de l'execució de la comanda.
-RUTA: camí per defecte per fitxers temporals.
-verbose: per imprimir o no traça (informa estat execució).
-SHELLHEADER: capçalera per *shell* scripts.
-addSHELLHEADER: indica si s'ha d'afegir l'intèrpret als *shell scripts* que es generen.
-RETORN: caràcter de retorn.
-sentencies: sentències o comandes *shell* que s'han d'executar per extreure els valors.
-mots_no_sofis: llista de mots considerats com no sofisticats.
-calculaValors: extreu un valor d'un arxiu xml a partir de la comanda i la ruta a l'arxiu.
-carregaParaulesNoSofis: carrega a memòria des d'un fitxer els mots no sofisticats.
-apareix: indica donat un mot i una llista de mots, si el mot apareix a la llista.

Métodes:

-getTypes: donada una llista de mots retorna els types de mots que conté.getParaulesSofis: donada una llista de mots retorna aquelles que són sofisticades.
-executeCommand: executa una *comanda shell*.
-getError: recupera l'error d'execució de la comanda.
-executeScript: executa un script creant un fitxer .sh amb la comanda i executant-lo.
-getVerbose: retorna el valor del *flag* per mostrar la traça o no.
-setVerbose: assigna el valor del *flag* per mostrar la traça o no.
-traça: imprimeix un missatge de traça, d'estat, durant l'execució d'una comanda.
-isAddSHELLHEADER: retorna si cal afegir l'intèrpret als *shell scripts* que es generen.
-setAddSHELLHEADER: assigna si cal afegir l'intèrpret als *shell scripts* que es generen.
-getContingut: retorna el contingut de la comanda a executar.
-setContingut: assigna el contingut de la comanda a executar.
-getRETORN: retorna el caràcter de retorn.
-setRETORN: assigna el caràcter de retorn.
-getRUTA: retorna la ruta o camí per defecte dels fitxers temporals.
-setRUTA: assigna la ruta o camí per defecte dels fitxers temporals.
-getSHELL: retorna l'intèrpret de comandes.
-setSHELL: assigna l'intèrpret de comandes.
-getSHELLHEADER: retorna la capçalera per *shell scripts.*
-setSHELLHEADER: assigna la capçalera per *shell scripts.*
-setError: assigna un error d'execució.
-setSortida: assigna la sortida de la comanda.

4.-Disseny

EstadisticaManager

Métodes:

-calculaPromig: calcula el promig dels percentatges d'encert d'un conjunt de prova.

-getMillorPA: retorna el PA més alt i el nom de la corresponent funció de classificació.

-getPitjorPA: retorna el pitjor PA i el nom de la corresponent funció de classificació.

-getPercentatgesUsAtributsSeleccionats: calcula el percentatge d'aparició d'un atribut en la prova de selecció d'atributs

4. Disseny

4.3 Principals algorismes

Les mesures de la riquesa lèxica, anomenades mètriques constitueixen els atributs amb els quals caracteritzem els textos d'opinió en llenguatge natural amb la finalitat d'utilitzar-les com a coneixement en el procés d'aprenentatge automàtic. Anem a veure en detall aquestes característiques i aspectes més concrets de la seva implementació.

Dim.	Id	Métrica	Etiqueta	Fórmula	Referencia
DenLex	1	Densidad léxica	DL	$\frac{N_{lex}}{N}$	(Engber, 1995)
SofLex	2	Sofisticación léxica	SL	$\frac{N_{slex}}{N_{lex}}$	(Linnarud, 1986; Hyltenstam, 1988)
	3	Perfil de Frecuencia Léxica	PFL5	$\frac{T_s}{T}$	(Laufer y Nation, 1995)
	4	Sofisticación Verbal I	SV-I5	$\frac{T_{vs}}{N_v}$	(Harley y King, 1989)
	5	Sofisticación Verbal II	SV-II5	$\frac{T^2_{vs}}{N_v}$	(Chaudron y Parker, 1990)
	6	Sofisticación Verbal Corregida	SVC5	$\frac{T_{vs}}{\sqrt{2N_v}}$	(Wolfe-Quintero, Inagaki, y Kim, 1998)
VarLex	7	Type/Token ratio	TTR	$\frac{T}{N}$	(Templin, 1957)
	8	Root TTR	RTTR	$\frac{T}{\sqrt{N}}$	(Guiraud, 1960)
	9	TTR Bilogarítmico	TTRB	$\frac{LogT}{LogN}$	(Herdan, 1960)
	10	TTR Corregido	TTRC	$\frac{T}{\sqrt{2N}}$	(Carroll, 1964)
	11	a^2	ac	$\frac{logN - logT}{log^2N}$	(Maas, 1972; Tweedie y Baayen, 1998)
	12	Indice de Uber I	UI	$\frac{(logN)^2}{logN - logT}$	(Dugast, 1979; Tweedie y Baayen, 1998)

Figura 13 : Mètriques utilitzades pel càlcul de la riquesa lèxica I

4. Disseny

4.3 Principals algorismes

13	K de Yule	YuleK	$10^4 \times (\sum i^2 T_i - N_{lex})/N_{lex}^2$	(Yule, 1944; Smith y Kelly, 2002; Miranda-García y Calle, 2005; Tweedie y Baayen, 1998)
14	Z de Zipf	ZIPF	$\frac{Z \times N \times log(N/Z)}{(N-Z)log(p \times Z)}$	(Smith y Kelly, 2002; Tweedie y Baayen, 1998)
15	Variación de Palabras Léxicas VPL		$\frac{T_{lex}}{N_{lex}}$	(Engber, 1995)
$16_{n,v,a,r}$	Variación G I	VG-I	$\frac{T_G}{N_{lex}}$	(McClure, 1991; Harley y King, 1989)
$20_{n,v,a,r}$	Variación G II	VG-II	$\frac{T_G}{N_G}$	
24	Variación Mod.	VM	$\frac{(T_a + T_r)}{N_{lex}}$	
$25_{n,v,a,r}$	Variación G Cuadrada	VG2-II	$\frac{T_G^2}{N_G}$	(Wolfe-Quintero, Inagaki, y Kim, 1998)
$29_{n,v,a,r}$	Variación G Corregida	VGC-II	$\frac{T_G}{\sqrt{2N_G}}$	

Convenciones:

N = tokens	T = types	$_{lex}$ = unidades léxicas
$_s$ = unidades sofisticadas	$_G$ = categoría gramatical (n, v, a, r)	$_n$ = nombre
$_v$ = verbo	$_a$ = adjetivo	$_r$ = adverbio
T_i = número de Types léxicos que ocurren i veces	Z = una medida de la riqueza léxica	p = Token más frecuente dividido por la longitud del texto

Figura 14 : Mètriques utilitzades pel càlcul de la riquesa lèxica II

En la figures 13 i 14 es mostren les 32 mètriques amb les quals treballem. Es divideixen en 3 dimensions que són la densitat (Den-Lex), la sofisticació (SofLex) i la variació (VarLex).

El càlcul de les mètriques es troba dividit en dues fases:

1. Obtenció dels *valors*.
2. Càlcul de les mètriques amb els *valors*.

4. Disseny

Conceptes previs:

- **Token:** aparició concreta d'una paraula en un text.
- **Type:** unitat abstracta que engloba totes les aparicions d'una mateixa paraula en un text.

ETIQUETA: <wd>

Exemple:

“El gos sense amo no es gos ni amo”.

-> gos: dos *token* i un *type*.

-> amo: dos *token* i un *type* .

- lex: les unitats o paraules lèxiques s'oposen a les gramaticals. Aquestes són els noms, els verbs, adjectius i adverbis.

ETIQUETA: <v>, <a>, <n>, <r>

- G: les unitats o paraules gramaticals són els pronoms, determinants, preposicions, conjuncions i interjeccions.

ETIQUETA: <p>, <d>, <s>, <c>,<i>

- s: unitats o paraules sofisticades són les menys utilitzades.

Definició dels valos

Els valors que cal extreure són els següents:

-**N**: número de tokens (mots).

-**Nlex**: tokens d'unitats lèxiques

-**Nv**: tokens verbals

-**Nn:** tokens noms

-**Na:** tokens adjectius

-**Nr:** tokens adverbials

-**Nslex:** tokens unitats lèxiques sofisticades

-**T**: types

-**Tlex**: types unitats lèxiques

-**Ts:** types unitats sofisticades

-**Tvs**: types verbals sofisticats

-**Tv:** types verbals

-**Tn:** types noms

-**Ta:** types d'adjectius

4. Disseny

Definició dels valos

-**Tr:** types d'adverbis

-**Z:** una mesura de la riquesa -> TTR

-**p:** token més freqüent/longitud del text

-**freqTypes:** llista de types lèxics que ocorren *i* cops.

Aquests valors són guardats en la variable de classe *valors* de *DadesArxiu* per poder ser utilitzats en el càlcul de les mètriques. Aquests valors s'extreuen dels arxius *xml* a partir de les ETIQUETES.

Procés d'obtenció dels arxius xml a partir dels textos d'opinió

Per tal d'obtenir aquests arxius *xml*, primer tenim els arxius amb els textos d'opinió en format *txt*. Aquests són anotats utilitzant una llibreria, que conté un analitzador morfològic (*tokenizer*), anomenada Freeling [11], de manera que els arxius *txt* són transformats a arxius format txt_tagged on a cada paraula del text se li assigna una categoria morfològica. El procés es pot veure a la figura 15**.**

Un cop tenim els textos d'opinió en llenguatge natural amb la seva anotació morfològica en arxius de format txt_tagged, utilitzem l'eina, que prèviament integrem en l'entorn de programació Eclipse, AnCoraPipe [23] que ens permet passar aquests arxius txt_tagged a xml per poder extreure els valors de manera més senzilla i mitjançant consultes. Aquest procés es pot veure a la figura 16 .

Procés d'obtenció dels arxius xml a partir dels textos d'opinió

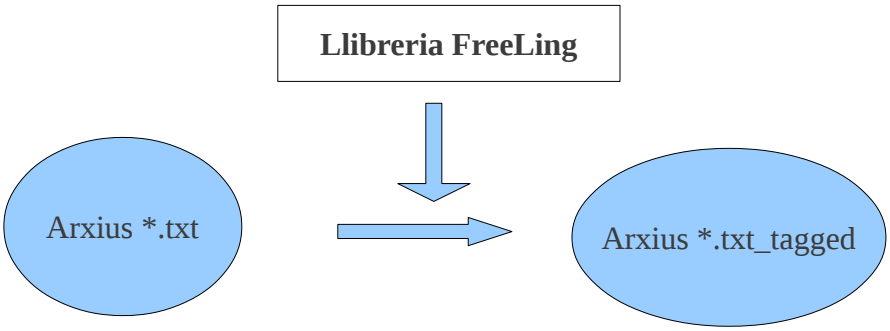

Figura 15 : Anotació morfològica dels textos utilitzant la llibreria FreeLing

4. Disseny

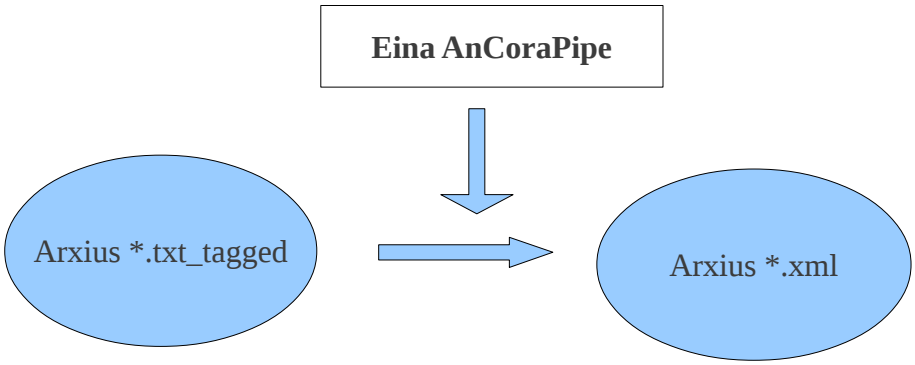

Figura 16: Pas d'arxius txt_tagged a xml utilitzant l'entorn AnCoraPipe.

Un cop tenim els arxius xml poder obtenir els valors que necessitem pel càlcul de les mètriques. Per fer-ho utilitzem l'eina XmlStarlet [22] que a través de comandes *shell* executades des del codi Java permeten fer consultes sobre els arxius i recollir els resultats. Dins la classe Script hi ha una variable anomenada *sentencies* que conté la llista de comandes necessàries d'aquesta eina per obtenir tots els valors. Després, amb l'ús de funcions auxiliars implementades dins aquesta classe obtenim la resta de valors que en calen. Anem cada funció i algun detall de la seva implementació:

- **frequenciaPerToken(String[] tokens):** calcula la freqüència de cada *token*. A partir de la comanda corresponent obtenim una llista de *tokens* o mots que conté l'arxiu xml. Aquesta llista és la que passem com argument a aquesta funció que per cada mot/*token* calcularà el número de cops que apareix guardant-ho en la variable *freqTokens* que és un *Hashmap* on per cada *token* és té el número de cops que apareix.
- **frequenciaPerTypes():** ens retorna un *Hashmap* on la clau és el número *i* de vegades que ocorre un *type* i el valor serà el número de *types* que ocorren *i* vegades. Aquesta funció es val de la variable *freqTokens* que guarda la freqüència per cada *token.* Per dir-ho d'alguna manera, el número de vegades que apareix un mot equival al valor *i*, mentre el número de mots que apareixen el mateix número de vegades i és el valor de Ti (*i->Ti*). Els valors *i*, T_i, obtinguts en aquesta funció són utilitzats, concretament, en la mètrica 13 de la figura 17 , que és la fórmula *K de Yule*.

$$10^4(\Sigma i^2 * T_i - N_{lex})/N^2_{lex}$$

Figura 17: Formula de K de Yule

4. Disseny

- **getTokenMesFrequent():** calcula el *token* que apareix més vegades repetit al text. Es val de la llista *freqTokens*, obtinguda anteriorment en el codi amb la funció *frequenciaPerToken*(), i retorna la freqüència més alta de entre tots els *tokens*. Aquesta freqüència es necessita en el càlcul del valor *p* que és igual el *token* més freqüent dividit per la longitud del text que és el valor de N. En la fórmula de la figura 18 i representa aquesta freqüència més alta que retorna aquesta funció.

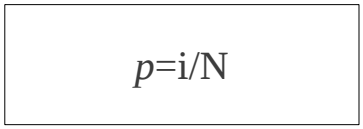

Figura 18: Fórmula valor de *p*

- **apareix(String mot, ArrayList<String> mots)**: funció que a partir d'un mot i una llista de mots ens retorna si el mot apareix en la llista.
- **getTypes(String[] words)**: ens retorna el número de *types* que conté la llista que li passem com a argument. Recordem que un *type* és un "mot únic". Per tant, el número de *types* és el número de paraules diferents. La funció recorre una llista on a cada iteració utilitza la funció *apareix()* per veure si el mot actual ja havia aparegut.
- **carregaParaulesNoSofis():** carrega el fitxer de nom "_es5000rae.txt" a memòria. Aquest fitxer conté una llista de 5000 paraules considerades més freqüents en l'idioma. Per tant, les unitats o mots sofisticats són aquells que no apareixen en aquesta llista. La crida a aquesta funció es fa en el constructor de la classe *Script*. Si hi hagués algun problema al carregar el fitxer es mostraria un missatge d'error en l'aplicació.
- **getParaulesSofis(ArrayList<String> words)**: donada la llista de paraules que li passem com a argument ens retorna la llista d'aquestes que són sofisticades. Per fer-ho utilitza la variable *mots_no_sofis* de la classe *Script* que conté la llista de les 5000 paraules no sofisticades, carregades anteriorment del fitxer amb la funció *carregaParaulesNoSofis*() i la funció *apareix*(), de manera que per cada mot de la llista que es passa a la funció mira si apareix en *mots_no_sofis*.
- **calculaValors(String nomArxiu):** és la funció principal pel càlcul de tots els valors i que utilitza per fer-ho les comandes *shell* que ens proporciona l'eina Xmlstarlet [22] i totes les funcions auxiliars que s'acaben de mencionar.

Un cop calculats tots aquests valors es procedeix al càlcul de les mètriques. D'això s'encarrega la classe MetricaXMLManager. Dins d'aquesta es troben implementades totes les funcions de les mètriques d'acord amb la seva expressió algebraica que apareix en la **figura i que tenen el mateix nom que la mètrica que calculen**. Cadascuna d'aquestes funcions rep com a argument una instància de l'objecte DadesArxiu a partir del qual pot recuperar els valors obtinguts prèviament per poder calcular les mètriques.

5.-Implementació i resultats

Prova d'anàlisi: classificació

Es fa una prova de classificació amb les mètriques seguint els passos corresponents i visualitzem tot seguit els resultats:

Condicions inicials:

-> **Número de instàncies** (serien el número de textos considerats): 1911 mostres

-> **Número de *folds* pel *cross-validation***: *10*

-> **Llista de classificadors utilitzats**: [BayesNet, IBk, KStar, NNge, JRip, PART, ZeroR, J48graft, FT, LADTree]

-> **Mètriques calculades**: 'SVC5', 'SV-II5', 'SL', 'SV-I5' , 'PFL5', i l'atribut de classe ('class') és el sexe que pot tenir dos valors -> {'H', 'M'}

1. Primer es fa la càrrega de les dades de referència sobre els textos:

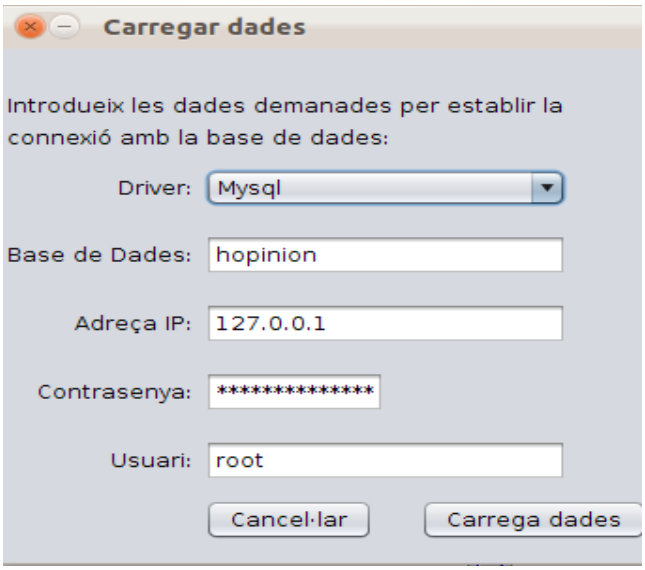

Figura 19 : Pas Carrega dades

5.-Implementació i resultats

2. Es seleccionen les mètriques que es volen calcular i l'atribut de classe a predir.

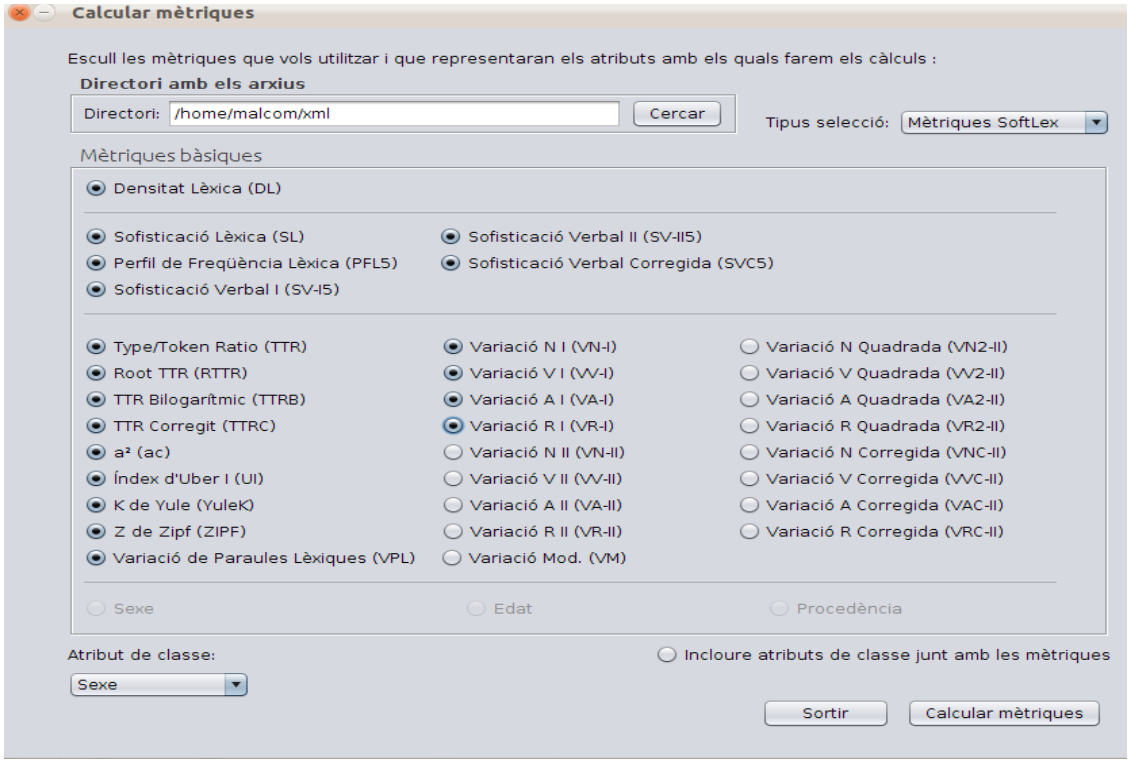

Figura 20: Pas escollir mètriques i atribut de classe.

3. En la finestra classificació s'introdueix tota la informació (conjunt de dades seran les mètriques) i es comença el procés:

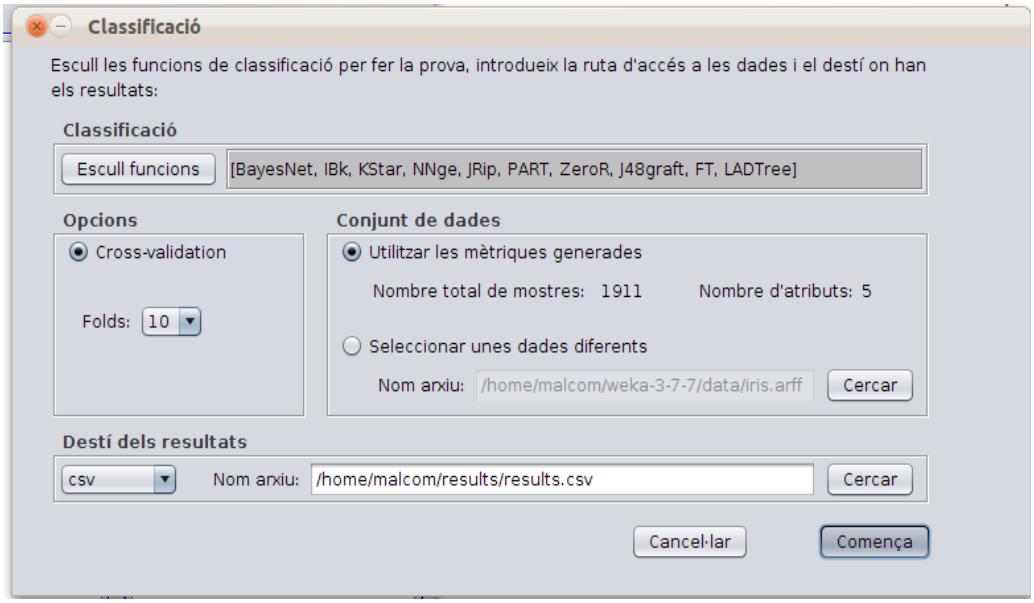

Figura 21: Pas condicions classificació

5.-Implementació i resultats

4. Visualització dels resultats estadístics

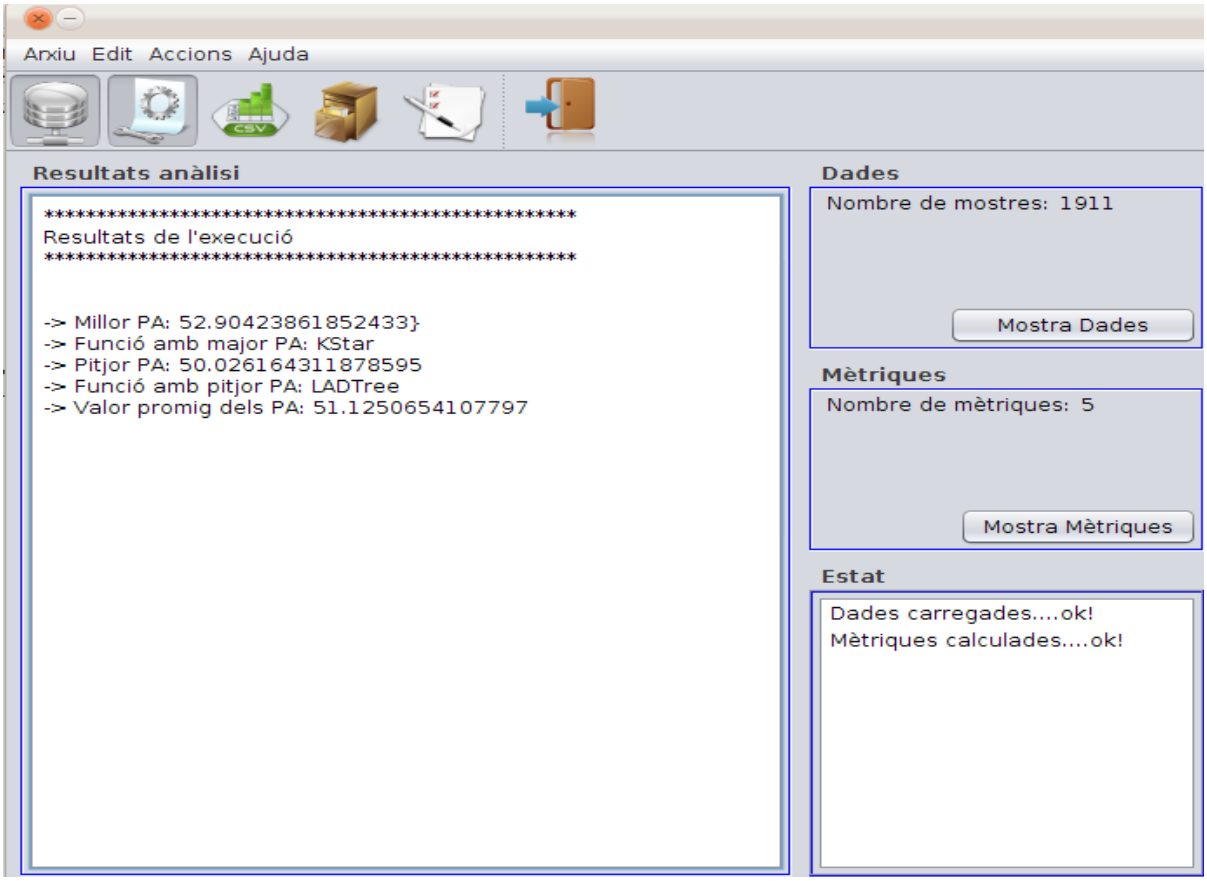

Figura 22: Visualització dels resultats estadístics

Resultats estadístics

En els resultats estadístics de la prova d'anàlisi realitzat s'indica quin és el percentatge d'encert més alt obtingut i a quin classificador pertany, el percentatge d'encert més baix i a quin classificador pertany i, finalment, la mitjana de tots els obtinguts.

5.-Implementació i resultats

Resultats de l'entrenament amb cross-validation

També es genera un fitxer en format *.csv, com a resultat de la prova d'anàlisi, al directori indicat en la finestra classificació en el pas corresponent, on s'ha guardat els 10 percentatges d'encert del *cross-validation*, per cadascun dels classificadors seleccionats.

Conjunt de dades: Mètriques										
Atributs: 1.SVC5	2.SV-II5	3.SL	4.SV-I5	5.PFL5						
Funció classificació	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10
KStar	53.12	51.83	51.83	47.64	49.74	54.45	52.88	56.02	57.59	53.93
ZeroR	50.52	50.26	50.26	50.26	50.26	50.26	50.26	50.26	50.79	50.79
IBk	55.21	47.12	50.79	45.03	50.26	53.40	50.79	60.73	51.31	53.93
NNge	55.73	51.83	50.79	45.55	45.55	46.07	50.26	49.74	58.64	56.02
FT	50.00	47.64	52.36	51.31	51.31	53.40	50.79	50.26	52.88	50.79
LADTree	46.88	47.12	50.26	48.17	52.88	53.40	48.17	52.36	49.74	51.31
JRip	53.65	49.21	54.45	49.21	53.40	50.79	50.26	49.21	48.17	58.12
J48graft	50.52	49.21	50.26	49.74	50.26	50.26	50.79	50.26	50.26	50.79
PART	50.52	49.21	50.26	49.74	50.26	50.26	50.79	50.26	50.26	50.79
BayesNet	53.65	49.21	53.40	53.40	48.69	51.31	52.88	54.45	49.74	51.83

Figura 23: resultats del CV per cadascun dels classificadors seleccionats

Prova d'anàlisi: selecció d'atributs

Fem una prova de selecció d'atributs amb les mètriques calculades seguint els passos corresponents i visualitzem tot seguit els resultats:

Condicions inicials:

-> **Número de instàncies** (serien el número de textos considerats): 1911 mostres

-> **Número de *folds* pel *cross-validation***: *10*

-> **Classificador utilitzat**: NaiveBayes

-> **Mètriques calculades**: 'SVC5', 'SV-II5', 'SL', 'SV-I5' , 'PFL5', i l'atribut de classe ('class') és el sexe que pot tenir dos valors -> {'H', 'M'}

-> **Parelles de selectors + avaluadors seleccionades:**

Ranker ----> InfoGainAttributeEval

BestFirst ----> CfsSubsetEval

GreedyStepwise ----> ConsistencySubsetEval

RankSearch----> ClassifierSubsetEval

5.-Implementació i resultats

Els passos 1 i 2 per la càrrega de dades i el càlcul de les mètriques són els mateixos que els de la primera prova d'anàlisi amb classificació realitzada.

3. En la finestra Selecció d'atributs s'introdueix tota la informació (conjunt de dades seran les mètriques) i es comença el procés

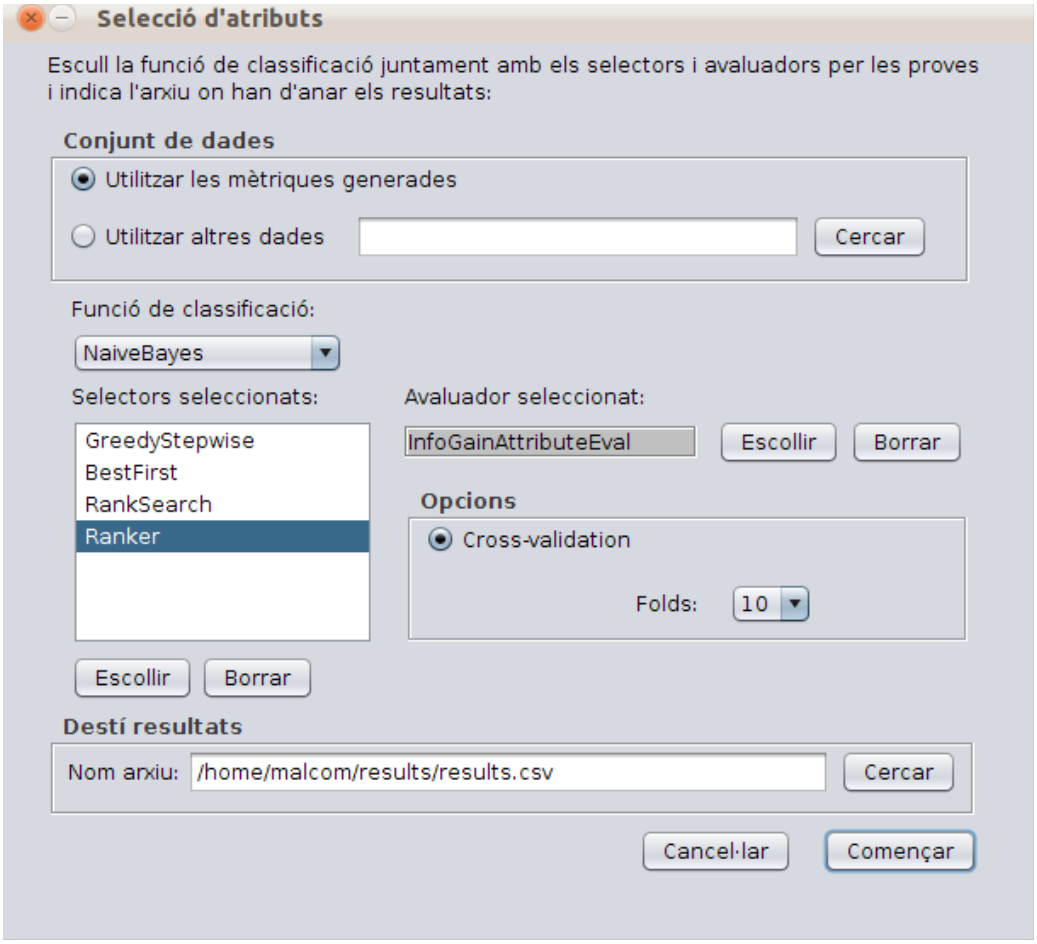

Figura 24: Pas selecció atributs

Escollim utilitzar les mètriques com a conjunt de dades, la funció de classificació de la llista, el número de *folds* i el directori destí on han d'anar els resultats.

5.-Implementació i resultats

4. Visualització dels resultats estadístics de la prova d'anàlisi de selecció d'atributs.

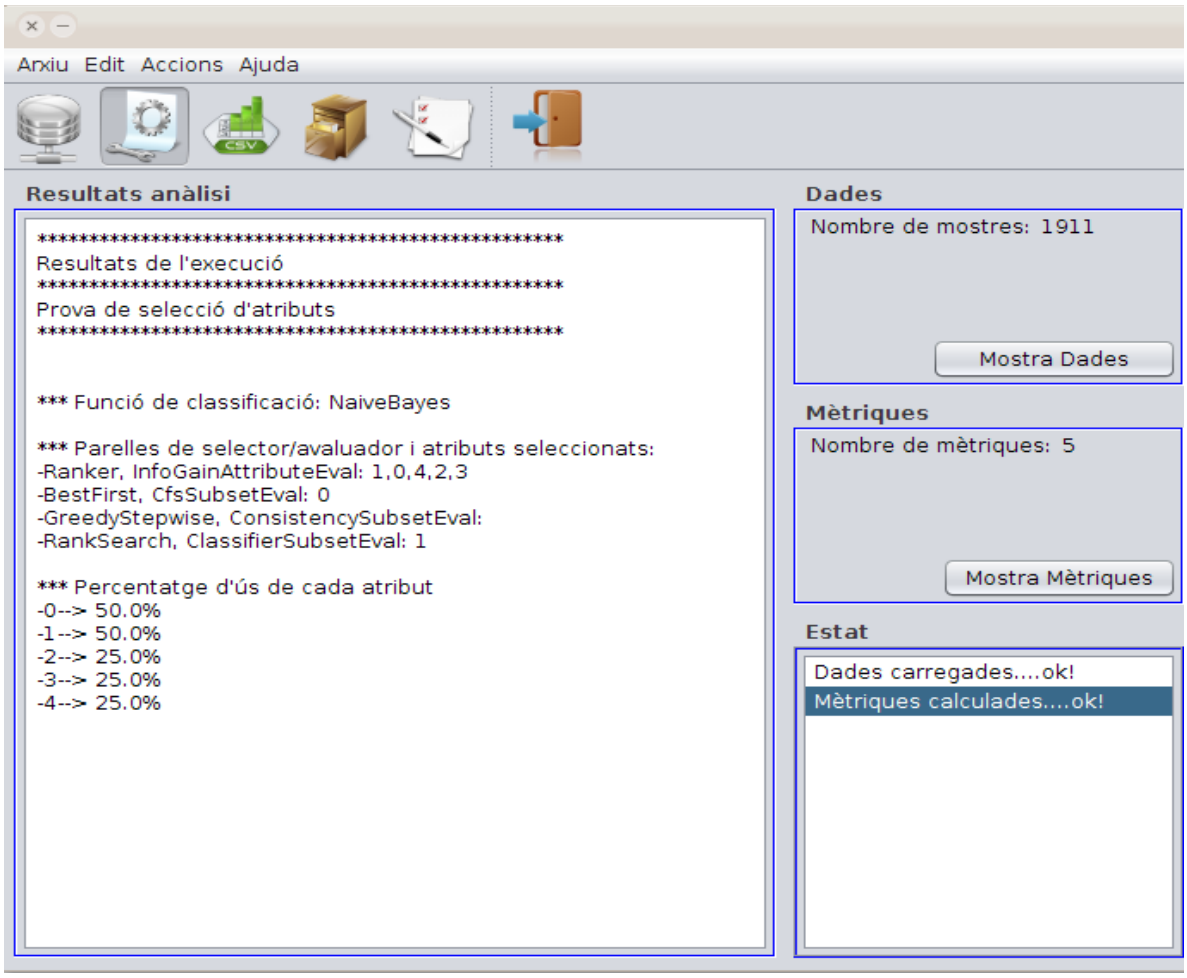

Figura 25: Pas 4 visualització dels resultats estadístics de la prova selecció d'atributs

Resultats estadístics

En els resultats obtinguts es mostra informació referent a la prova: el classificador seleccionat, les parelles de selectors i avaluadors junt amb els atributs seleccionats per aquestes combinacions i els percentatges d'ús de cada atribut per les combinacions fetes.

5.-Implementació i resultats

Resultats de l'entrenament amb cross-validation

També es genera un fitxer en format *.csv, com a resultat de la prova d'anàlisi, al directori indicat en la finestra selecció atributs en el pas corresponent, on s'ha guardat els 10 percentatges d'encert del *cross-validation*, per cadascuna de les combinacions de classificador+selector+avaluador.

Conjunt de dades:	Mètriques												
Atributs: 1.SVC5	2.SV-I5	3.SL	4.SV-I5	5.PFL5									
Classificador	Selector	Avaluador	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	Fold 9	fold 10	#atributs	atributs
NaiveBayes	Ranker	InfoGainAttributeEval	53.65	49.21	53.40	53.40	48.69	51.31	52.88	54.45	49.74	51.83	5	1,0,4,2,3
NaiveBayes	BestFirst	CfsSubsetEval	55.73	49.74	57.07	48.69	54.45	53.40	50.26	55.50	50.26	56.02	1	0
NaiveBayes	GreedyStepwise	ConsistencySubsetEval	50.52	50.26	50.26	50.26	50.26	50.26	50.26	50.26	50.79	50.79	0	
NaiveBayes	RankSearch	ClassifierSubsetEval	51.56	49.21	53.40	51.31	52.36	51.83	50.26	57.07	49.21	53.40	1	1

Figura 26: resultats CV per la prova de selecció atributs

A més dels resultats del CV també es guarda al fitxer el número d'atributs seleccionats per cada combinació i l'índex dels mateixos. Per saber de quin atribut es tracta a la capçalera del fitxer trobem una llista on tenim l'índex de l'atribut i el seu nom.

6. Valoració econòmica

6.1 Anlisi del temps de realització del projecte

La realització del projecte es divideix en diferents apartats:

-Formació: temps dedicat a la recerca d'informació sobre el tema a tractar.

-Anàlisi: temps destinat als requisits, casos d'ús, etc.

-Disseny: temps destinat al disseny del sistema.

-Implementació: temps dedicat a la programació del sistema i la interfície gràfica.

-Testeig temps dedicat a les proves.

Aquestes són les hores dedicades a cada apartat:

-Estudi i formació: 130 hores

-Anàlisi: 65 hores

-Disseny: 40 hores

-Implementació: 210 hores

-Testing: 80 hores

En total suposen 525 hores de dedicació

Figura 27: Gràfic de les hores de dedicació per apartats

6. Valoració econòmica

6.2 Valoració del cost econòmic

Es suposa que per la realització d'aquest projecte han calgut un analista i un programador. Es considera que el preu que cobra l'analista a l'hora són 40€ i un programador són 30€. El cost d'aquesta aplicació hagués sigut:

- Analista: (130h+65h+40h)*40€/h=9400€
- Programador: (210h+80h)*30€/h=8700€
- Total: 9400+8700=18100€

Donat que les tecnologies utilitzades en el projecte són totes *Open Source*, i que l'entorn de desenvolupament utilitzat, *Eclipse*, és també gratuït, el cost d'aquestes tecnologies és 0€. Així doncs, el cost total és el mencionat anteriorment, 18100€.

7.Conclusió

Com s'ha vist al llarg del projecte l'estudi del comportament verbal dels autors de textos d'opinió per obtenir alguns dels seus atributs demogràfics és una tasca complexa que requereix de tècniques pel Processament del llenguatge natural combinades amb tècniques d'Intel·ligència Artificial. Actualment hi ha molts investigadors que treballen en aquesta matèria, entre d'altres, perquè es disposa d'una quantitat molt gran d'informació per poder analitzar gràcies a l'era digital en la que vivim i, al mateix temps, és una manera no intrusiva d'obtenir dades sobre els autors de textos que després es puguin utilitzar en altres sistemes informàtics actuals, com els Recomanadors.

Tot i els avenços en aquesta matèria, planteja algunes dificultats pel que fa al tractament del llenguatge natural, el qual és inherentment ambigu a diferents nivells com per exemple, a nivell lèxic una mateixa paraula pot tenir diferents significats o a nivell referencial es requereix d'un cert coneixement sobre les estructures pròpies d'un llenguatge. En aquest sentit cal destacar, la gran importància que tenen els mètodes matemàtics en aquest estudi que mitjançant l'ús de transformacions algebraiques tracta d'obtenir mesures més exactes de determinades propietats o característiques lingüístiques d'un text.

En quant a l'eina ATOp implementada en aquest projecte podríem dir que es tracta d'una eina estable encaminada a la realització de proves d'anàlisi sobre textos d'opinió extrets de la web, de manera automàtica i que, per tant, ofereix les funcionalitats necessàries pel tractament i estudi d'aquests. El treball futur està dirigit cap a l'ús de noves mètriques o característiques extretes dels textos mitjançant transformacions algebraiques per ser incorporades en les proves d'anàlisi per millorar el rendiment dels classificadors i, per tant, els resultats d'aquestes proves.

Personalment, puc dir que em sento satisfet del treball realitzat, tot i que, inicialment, desconeixia les tècniques del Processament del Llenguatge Natural i com s'aplicaven. M'ha paregut interessant poder elaborar una eina que agrupés i implementés tant tècniques pel Processament del llenguatge natural com de la IA per permetre la realització de proves d'anàlisi sobre els textos.

8.-Referències bibliogràfiques

[1] weka: col·lecció d'algorismes per tasques de mineria de dades implementats en llenguatge java, http://www.cs.waikato.ac.nz/ml/weka/.

[2] PAYRATÓ, LL. *Et alii* (1996): Corpus, corpora, Barcelona, Universidad de Barcelona.

[3] ALVAR EZQUERRA, M. J. BLANCO RODRÍGUEZ i F. PÉREZ LAGOS (1994): "Diseño de un corpus español en el marco de un corpus europeo".

[4] BIBER, D.,S. CONRAD i R. REPPEN (1998): *Corpus linguistics: investigating language structure and use, Cambridge University Press.*

[5] McEnery, T. I A. Wilson (1996:36), Corpus linguistics. Edinburgh, Edinburgh University Press.

[6] Aijmer i Altenberg (1991:1) : English Corpus Linguistics. London. Longman.

[7] Leech (1992, 1993): "Corpora and theories of linguistics performance", en J. SVARTVIK (ed.), pp.105-134

[8] Pérez Guerra 1999: "Estándares de anotación en lingüística del corpus", Revista Española de Lingüística Aplicada, Volúmen Monográfico, 25-52

[9] Moure T. I J. Llisterri (1996): "Lenguaje y nuevas tecnologías: el campo de la lingüística computacional", en M. Fernández Pérez (coord.) (1996), pp. 147-227

[10] Berber T. (1999): Corpus Linguistics. Document electrònic: www.tonyberber.f2s.com

[11] Llibreria Freeling: consisteix en una llibreria que proveeix de serveis per l'anàlisi del llenguatge (com anàlisi morfològic, reconeixement de dates, *PoS tagging*, etc.), http://nlp.lsi.upc.edu/freeling/.

[12] Bindi, R. *Et alii* (1994:29): "*Corpora and Computational Lexica. Integration of different methodologies of lexical knowledge acquisition*", *Literary and Linguistic Computing*, 9, 1, 29-46.

[13] Ostler, N. (1992:2): "*Corpus Design Criteria*", *Literary and Linguistic computing*, 7, 1, 1-16.

[14] Payrató (1998:108): De profesión, lingüista (Panorama de la lingüística aplicada), Barcelona, Ariel.

[15] Grishman, R. (1991:15): *Introducción a la lingüistica computaciona, Madrid, Visor.*

[16] Allen, J. (1987): Natural Language Understanding, Redwood City, Benjamin/Cummings.

[17] Moreno Sandoval, A. (1998): Lingüísta Computacional, Madrid, Síntesis.
[18] Moreno Sandoval, A. (1998:14-15): Lingüística Computacional, Madrid, Síntesis.
[19] *Gazdar i Mellish (1989:16): Natural Language Processing in PROLOG. An Introduction to Computacional Lingüistics, Wokingham, Addison Wesley Publishing Company.*
[20] Gómez Guinovart, J. (1999:7-8): "*Introducción*", *Revista Española Aplicada, Volumen Monográfico, 7-9.*

8.-Referències bibliogràfiques

[21] Meya, M. I W. Huber (1986): *Lingüística computacional*, *Barcelona*, *Teide*.

[22] XmlStarlet: eina que reuneix un conjunt d'utilitats de línia de comandes, que es poden utilitzar per transformar, consultar, validar i editar documents i arxius XML utilitzant un conjunt de senzilles comandes shell -> http://xmlstar.sourceforge.net/overview.php.

[23] AncoraPipe: és un entorn gràfic, que es pot inegrar a Eclipse, que permet la creació, edició i anàlisi de còrpora lingüística i lexicons.

9. Manual de l'aplicació

9.1 Manual de l'usuari

Càrrega de les dades referents als textos

Per anar a la pantalla que ens permetrà carregar les dades referents a cada text tenim el botó Carrega dades a la barra d'eines, o bé, a l'opció *Carregar dades* del menú *Arxiu*. Ens apareixerà una finestra com la de la figura x.

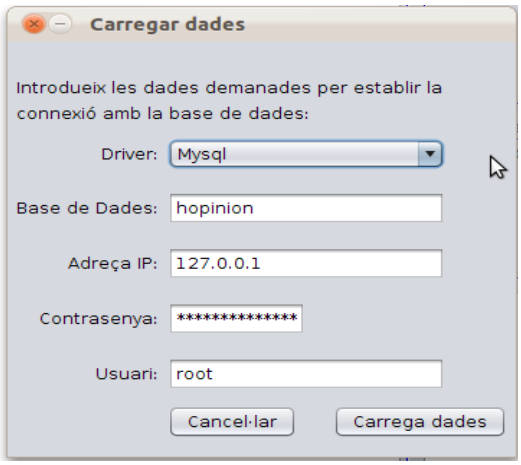

Figura 28: Finestra Càrrega de dades

En aquesta finestra l'usuari ha d'introduir la informació que permetrà la connexió amb la base de dades i càrrega de la informació. S'ha d'indicar el *driver* pel SGBD, el nom de la base de dades, en el nostre cas serà *hopinion*, l'adreça *IP* de l'ordinador on es troba la mateixa, la contrasenya i l'usuari autoritzat per l'accés. Un cop introduïda tota aquesta informació per iniciar el procés premem el botó *Carrega dades*. Si la càrrega es realitza amb èxit es tancarà aquesta finestra i es mostrarà un missatge d'estat dins el quadre *Estat* que hi ha a la finestra principal indicant que les dades s'han carregat correctament.

9. Manual de l'aplicació

Calcular les mètriques

Per tal de calcular les mètriques dels textos que constituiran els atributs per l'aprenentatge automàtic tenim el botó *Calcula Mètriques* a la barra d'eines de la finestra principal, o bé, l'opció *Calcular les mètriques* del menú *Accions.* S'obrirà una finestra com la de la figura x on trobarem totes les opcions possibles pel càlcul de les mètriques.

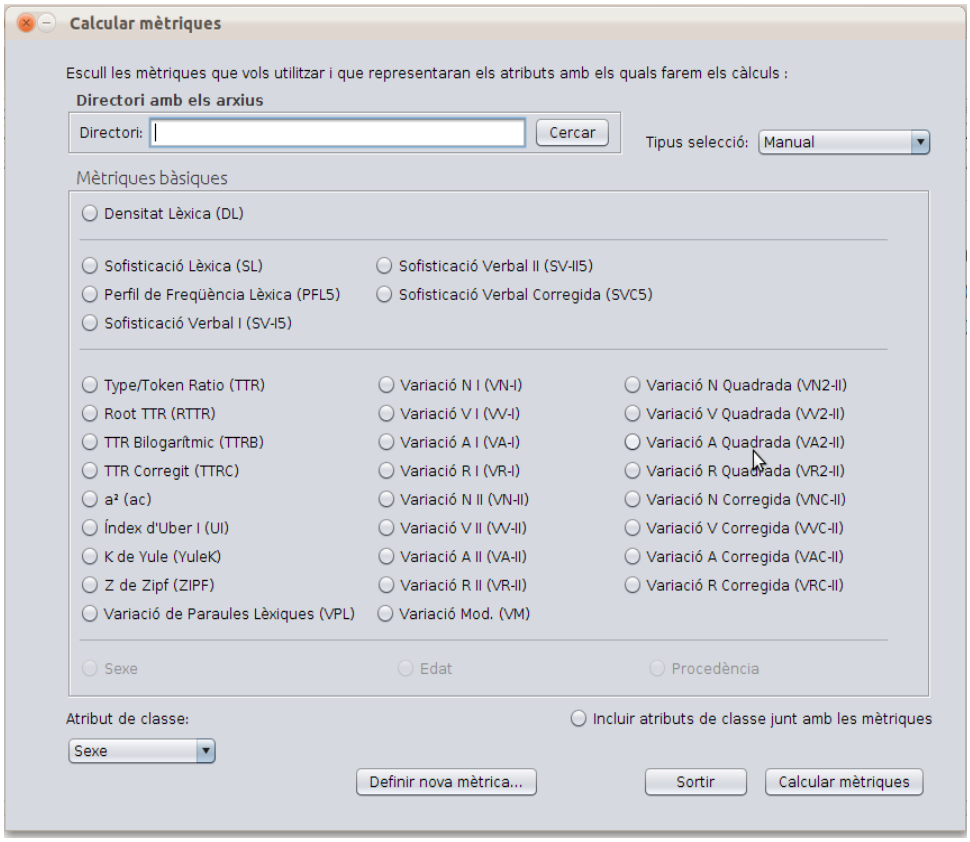

Figura 29: Finestra Calcular mètriques

Primerament, hem d'indicar el directori on es troben els arxius *xml* amb la informació dels textos d'opinió dels quals extraurem les mètriques. Podem escriure directament la ruta, o bé, prémer el botó *Cercar* per trobar el directori on es troben els arxius. Després hem de seleccionar les mètriques que volem calcular. Podem seleccionar-les una a una, o bé, seleccionar tot un grup de mètriques desplegant la llista tipus de selecció. També tenim l'opció d'incloure alguns dels atributs que podríem utilitzar com a atributs de classe dins del coneixement per la prova junt amb les mètriques.

En aquest cas, primer hauríem de marcar la casella Incloure *atributs de classe junt amb les mètriques* i ens donaria l'opció d'escollir entre dos d'aquests atributs, ja que l'altre serà l'utilitzat com a atribut de classe, pròpiament dit, per la prova.

9. Manual de l'aplicació

Un cop seleccionades totes les mètriques que volem calcular, només ens queda seleccionar de la llista *Atribut de classe* que es troba a baix a l'esquerra l'atribut que volem predir en la prova. Finalment, quan estigui tota la informació introduïda premem el botó *Calcular mètriques* i començarà el procés de càlcul de les mètriques que, un cop hagi acabat, es tancarà la finestra i, si tot ha anat bé, es mostrarà un missatge al quadre *Estat* de la finestra principal informant que les mètriques han sigut calculades satisfactòriament.

Realitzar prova de classificació

Per anar a la pantalla que ens permet configurar les opcions per fer les proves de classificació tenim el botó, o bé, l'opció *Classificació* dins el menú *Accions*. Ens apareixerà una pantalla com la de la figura x.

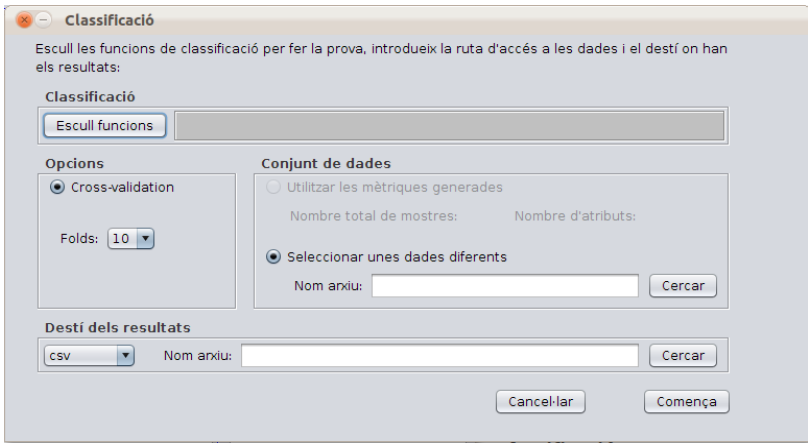

Figura 30: Finestra Classificació

En la finestra de *Classificació,* primerament, ens dóna la possibilitat d'escollir les funcions amb les quals volem realitzar les proves. Si premem el botó *Escull funcions* ens apareixerà una altra finestra on podrem anar afegint les funcions de classificació disponibles. Després entre les *opcions* trobem l'opció *cross-validation* on podem seleccionar desplegant la llista que es faci amb 10 o 5 plecs (*folds*).

Sobre l'origen de les dades que utilitzarem per fer les proves de classificació tenim, d'una banda, l'opció d'utilitzar els valors de les mètriques, que només estarà disponible en el cas que les haguem calculat prèviament i, d'altra banda, la possibilitat d'introduir la ruta a un arxiu d'extensió *csv* que contingui el conjunt de dades. Inicialment, l'opció *utilitzar les mètriques generades* apareixerà disponible només si ja han estat calculades les mètriques. En el cas d'escollir unes dades diferents haurem d'indicar la ruta d'accés a l'arxiu, o bé, obrir el quadre de diàleg *Obrir* prement el botó *Cercar*.

Finalment, hem d'indicar l'arxiu on volem guardar els resultats de la prova amb les opcions marcades en aquesta finestra. Escrivim el camí on es troba l'arxiu, o a allà on volem que es crei, o bé, obrim el quadre de diàleg *Guardar* prement el botó *Cercar* i indiquem la ruta. Per executar la prova premem el botó *Comença* i, si no es produeix cap error, es crearà l'arxiu amb els resultats en la ruta indicada i es mostraran resultats estadístics dins el quadre de text que hi ha a la pantalla principal amb el títol *Resultats anàlisi*.

9. Manual de l'aplicació

Realitzar prova de Selecció d'atributs

Per poder fer la prova de Selecció d'atributs tenim el botó *Selecció d'atributs* que es troba a la barra d'eines, o bé, l'opció *Selecció d'atributs* del menú *Accions*. Un cop escollida una d'aquestes opcions es mostra la finestra que apareix a la figura x que ens donarà la possibilitat d'introduir totes les opcions i realitzar la prova.

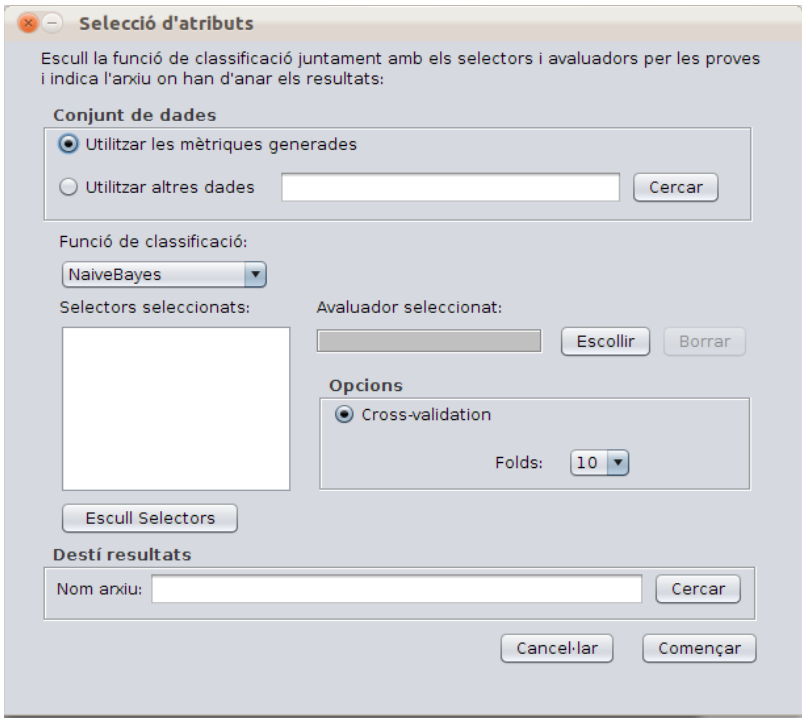

Figura 31: Finestra Selecció d'atributs

Primer, hem d'indicar totes les opcions per poder realitzar la prova. Indicarem si el conjunt de dades amb el que treballarem serà les mètriques calculades, o bé, alternativament, un arxiu que contingui les dades. La casella *Utilitzar les mètriques generades* estarà habilitada només si aquestes han sigut calculades. Després hem d'escollir una funció de classificació amb la qual farem la prova i els selectors que volem utilitzar junt amb els seus avaluadors. També podem escollir si volem fer el *cross-validation* amb 5 o 10 plecs (folds). Finalment, cal indicar on volem que es guardin els resultats del *cross-validation* per cada combinació de classificador+selector+avaluador. Un cop tenim introduïdes totes les opcions premem el botó *Començar* i iniciarà la prova. Es guardaran els resultats al lloc indicat per l'usuari i es mostrarà, al quadre *Resultats anàlisi* de la finestra principal, els resultats estadístics obtinguts.