Facultat d'Informàtica de Barcelona
Treball de final de grau

Memòria

Algorismes d'entrenament per xarxes neuronals basades en similitud

Adrià Jorquera Codina

Director:
Lluís A. Belanche Muñoz

21 de juny de 2016

Índex

1 Introducció

En aquest treball es tracta el tema d'aprenentatge per similitud, una àrea poc explotada de l'aprenentatge automàtic. L'aprenentatge automàtic és una disciplina que té com a meta fer que un computador sigui capaç d'extreure patrons d'un conjunt de dades i poder predir events futurs. A diferència d'això, l'aprenentatge basat en similitud no utilitza directament les dades sinó que primer les transforma mitjançant una funció de similitud. D'aquesta manera, assimila el coneixement en base a com de semblants són les observacions entre sí.

Concretament es tractarà el cas de les xarxes neuronals basades en similitud (Similarity Neural Network o SNN en anglès). És un tipus de modelat estadístic que utilitza una funció de similitud a les seves entrades i després busca les observacions més significatives per aprendre. Els principals avantatges de la seva utilització són la capacitat de tractar diferents tipus de variables i l'acceptació de valors perduts.

1.1 Problema i objectius

El problema plantejat en aquest treball és l'inexistència d'un model acabat de SNN. Existeixen algunes implementacions[1] prèvies al nostre projecte i tot i que només es tracta d'una implementació bàsica ja mostra un rendiment millor que alguns altres mètodes.

També existeix la creixent necessitat del mercat de nous i millors mètodes de predicció. Molts camps mèdics, laborals i socials s'aprofiten de sistemes informàtics i es pot trobar una alta demanda de models estadístics que puguin aportar millores al seu funcionament.

Els objectius que s'han proposat per solucionar aquests problemes són els següents:

- Implementar una funció de similitud que sigui capaç de tractar valors perduts i diversos tipus de variables de manera eficient.

- Programar cinc implementacions diferents de SNNs seguint diferents tipus de construcció.

- Trobar la millor versió segons l'valuació de les implementacions sobre problemes complexos en tres àmbits diferents: error, simplicitat i temps d'execució. A més a més, es posaran en contra de mètodes més reconeguts com Generalized Linear Model[2], Random Forest[3] i Support Vector Machine[4] per comprovar la seva eficàcia.

1.2 Abast

El projecte té les seccions funció de similitud, implementacions i problemes:

Funció de similitud: La funció de similitud serà limitada a un conjunt de tipus de variables. Es tractaran els més comuns com variables quantitatives, nominals, binàries simètriques, binàries asimètriques, ordinals i circulars. Es deixaran fora les variables *Fuzzy* doncs no s'han trobat problemes que les tractin.

Implementacions: Existeixen diverses maneres de programar una SNN però en l'àmbit del treball només se'n tractaran cinc. Les implementacions tractades són basades en els mètodes més coneguts en el món de l'estadística. S'han deixat fora nombroses implementacions per centrar esforços.

Problemes: Per comprovar l'eficàcia de les nostres implementacions s'ha escollit un petit conjunt de problemes. Els problemes escollits es consideren complicats doncs no tenen solució actualment i contenen nombrosos tipus de variables i valors perduts. A més, són de mida moderada (menors de 1.500 observacions) per evitar grans temps de computació. A més, s'ha provat un de complexitat computacional alta (40.000 observacions) i de gran dificultat (classificació desvalancejada, preprocés considerable) per tenir un cas real i modern d'anàlisi de dades.

1.3 Estructura del document

El document està estructurat seguint un esquema de context, conceptes d'aprenentatge automàtic, implementació, material utilitzat, recollida de resultats, conclusions i finalment el resultat de la planificació:

Context: S'expliquen conceptes bàsics relacionats amb el treball per poder entendre el funcionament d'aquest.

Conceptes d'aprenentatge automàtic: Aquí es troben les tècniques i métodes més tècnics relacionats directament amb l'aprenentatge automàtic.

Implementació: La descripció de les diferents implementacions i els seus components es troba en aquesta secció.

Material utilitzat: Es descriuen els problemes utilitzats i les seves propietats.

Recollida de resultats: Conté els resultats d'executar les implementacions sobre els problemes esmentats.

Conclusions: L'anàlisi dels resultats i la decisió de la millor implementació.

Planificació: Resumeix temps i recursos utilitzats i la sostenibilitat del procés.

2 Context

En aquesta secció s'expliquen els principis bàsics per a poder entendre millor els termes i procediments fets servir en la resta del treball.

2.1 Xarxes neuronals

Les xarxes neuronals[5] (Neural networks o NN en anglès) són un tipus de model estadístic que s'inspira en el funcionament del sistema nerviós central dels animals, el cervell. Un model d'aquest tipus es forma a partir d'un conjunt de neurones interconectades entre si que es transmeten informació.

La forma clàssica d'una NN està organitzada en tres capes: neurones d'entrada, neurones ocultes i neurones de sortida. Existeixen nombrosos tipus de xarxes però normalment la transmissió d'informació és unidireccional de capa d'entrada a capa oculta i finalment a capa de sortida. En la figura 1 es pot veure una construcció clàssica de xarxa neuronal.

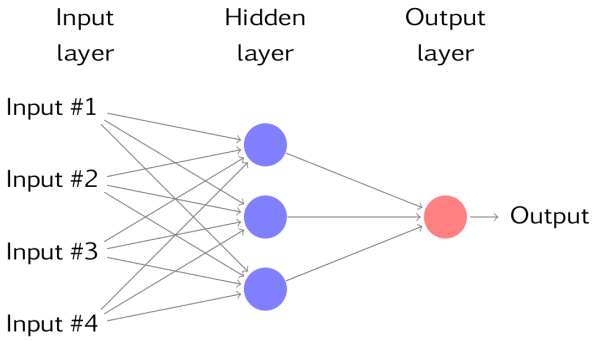

Figura 1: Exemple bàsic d'una xarxa neuronal

L'informació es recull en la capa d'entrada on cada variable està conectada a totes les neurones ocultes. Després cada neurona oculta calcula una combinació no lineal de les entrades i seguidament la capa de sortida calcula una altre combinació amb les dades de la capa oculta. Aquest resultat és el valor de sortida predit.

2.2 Mesures de similitud

Una mesura de similitud és una funció simètrica que expressa com de semblants són dues observacions. Primer es desenvolupen mesures de similitud específiques per cada tipus de variable, definides en el codomini $[0, 1]$. Es fa servir s_{ijk} per denominar $s_k(x_{ik}, x_{jk})$ la semblança entre les observacions x_i i x_j per una variable k. Finalment s'agreguen en s_{ij}, la semblança final entre x_i i x_j.

Alternativament, existeix la definició de distància que expressa la diferència entre dues observacions. S'expressa com $d_{ij} = \sqrt{1 - s_{ij}}$ i es troba en el codomini $[0, 1]$. En aquest cas, 0 significaria que són totalment iguals i 1 que no tenen res en comú. Es pot entendre com una transformació de la similitud.

Expressar la relació de les observacions en distàncies serà la decisió presa en aquest projecte. Les principals raons són dues: popularitat i representabilitat. En primer lloc, les distàncies són més utilitzades que les similituds en el món de la estadística i, per tant, existeix més documentació i recursos. En segon lloc, és més fàcil representar distàncies en gràfics.

2.3 Format de les dades

En aquest treball, s'abarcaran els problemes amb les dades en forma de matriu $X_{n \times k}$. Cada una de les n files suposa una observació, un cas aillat amb les seves propietats. Les columnes es divideixen en k variables independents i una variable dependent. També es pot entendre com que es tenen k predictors que descriuen una situació que ha desencadenat en el resultat de la variable resposta. Depenent del tipus de la variable independent, un problema es considera de regressió si es vol predir valors numèrics o classificació si es vol predir la pertinença a una classe.

Les dades també poden contenir valors perduts (Missing value, not available o NA en anglès). Això significa que per una variable d'una observació pot haver-hi un valor impossible, il·lògic o simplement un buit. Es pot donar la situació si el procés de recolecció d'informació és incomplet, es cometen errors a la hora d'introduir els valors en una base de dades o si l'autor del problema va considerar un criteri no estandaritzat.

Les variables poden ser de diversos tipus que descriuen de manera diferent la informació d'una observació. Cadascún dels tipus de variables té les seves propietats i manera única d'analitzar. Els casos que es poden trobar són els següents:

- **Quantitativa:** Pren valors numèrics reals. Pot contenir valors infinits i no tenen cap ordre o propietat especial a part d'expressar una magnitud.
  Exemple: Edat, pes, altura, etc.

- **Nominal:** Pren valors finits. Poden ser tant cadenes de caràcters com valors numèrics enters, però no descriuen una magnitud concreta. Els diferents valors d'aquest tipus de variables es divideixen en classes i són distingibles entre sí.
  Exemple: Color d'ulls, menjar preferit, etc.

- **Ordinal:** Pren valors finits. Poden ser tant numèrics com cadenes de caràcters. Té la característica especial de que entre les classes existeix un ordre.
  Exemple: Educació (primaria $<$ secundària $<$ universitat), temperatura, etc.

- **Binaria asimètrica:** Pren dos valors (0 o 1). És de tipus lògica i indica la existència o no d'una propietat. La importància d'un 1 és major que la d'un 0 doncs el fet de tenir la propietat descriu concretament la observació.
  Exemple: És alcohòlic?, pateix insomni?, etc.

- **Binària simètrica:** Pren dos valors. Cada observació ha de prendre un dels dos i és exclusiu. Contrariament a les asimètriques, aquestes no tenen prioritat entre sí i ambdues classes tenen el mateix valor.
  Exemple: Sexe, casat, etc.

- **Circular:** Pren valors finits. Poden ser tant numèrics com cadenes de caràcters. Té la característica especial de que entre les classes existeix un ordre. Aquí però, es forma un anell entre les classes de la variable. Això significa que el primer valor és contigu a l'últim.
  Exemple: Dia de la setmana, punt cardinal, etc.

També existeix el tipus de variable *Fuzzy* que denota en el interval [0,1] un grau de certesa. Però quedarà fora del abast del projecte doncs l'ús d'aquesta variable és bastant escàs i no s'han trobat problemes que l'utilitzin.

3 Conceptes d'aprenentatge automàtic

L'aprenentatge automàtic[6] (Machine Learning o ML en anglès) és un camp de l'inteligència artificial que dóna la capacitat a les màquines d'aprendre i evolucionar. La finalitat és crear programes capaços d'extreure patrons de dades per tal de poder generalitzar comportaments. El ML està estretament relacionat amb el camp de l'estadística.

Els programes que es crean en el procés d'aprenentatge s'anomenen models. Comencen com una construcció buida que es fa evolucionar mitjançant una etapa d'entrenament. En aquesta etapa se'l presenta un conjunt de dades i intenta adaptar patrons. El resultat és un model capaç d'analitzar noves dades i predir o classificar propietats.

A continuació es descriuran les nocions, els mètodes i tècniques utilitzats relacionats amb aquesta disciplina.

3.1 Entrenament i testeig

Quan s'entrena un model interessa poder comprovar com rendeix amb noves observacions. La divisió del problema en dos conjunts anomenats set d'entrenament (més conegut com training set) i set de test (més conegut com testing set) és la pràctica més usual.

El set d'entrenament és el que podem suposar que el coneixem totalment i, per tant, podem tenir en compte la variable dependent. Aquest és el set que proporcionarem al model i del que podrà aprendre. El set de testeig és el que adjudicarem com a noves dades. En teoria no coneixem el resultat fins després d'intentar predir amb el model. Aquest procés és el que es farà servir per avaluar el rendiment de futurs experiments. Normalment es repeteix varies vegades per evitar casos anòmals i es fa la mediana com a eficàcia esperada.

3.2 Cross Validation

Cross validation és una tècnica de comprovació de la generalització de resultats per a un estudi estadístic. Consisteix en definir p particions del training set i efectuar p fases d'entrenament. A cada fase, s'agafen p-1 subsets i s'utilitzen per entrnar un model que es valida amb el subset restant. Al final es retorna un promig dels errors com a rendiment final.

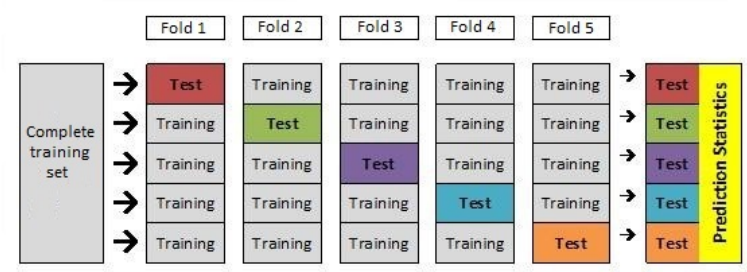

Figura 2: Exemple de Cross Validation

L'utilitat d'aquest mètode resideix en l'afinament de paràmetres i l'evasió del *overfitting*. Mitjançant un remostreig es pot generalitzar els resultats i trobar els valors òptims dels paràmetres del model per al problema en qüestió. L'overfitting és conegut com el fenòmen on un model apren massa bé el training set i no generalitza bé respecte a noves observacions.

La notació utilitzada per descriure el Cross Validation és $(X \times Y)CV$ significant que es fan X execucions de Y divisions.

3.3 Preprocessat

El preprocessat és una tècnica que consisteix en tractar un conjunt de dades per poder fer-lo servir en un sistema que normalment no podria. Molts sistemes tenen restriccions de com ha de ser l'informació que reben a l'entrada i l'única manera d'aconseguir executar-los és duent a terme un preprocessat.

En el nostre projecte la majoria de problemes que es volen abarcar contenen variables de diversos tipus i certa quantitat de valors perduts, cosa que alguns models no són capaços d'acceptar. Random forests, GLMs i SVMs no permeten valors perduts a l'entrada i, a més a més, aquests dos últims no reconeixen valors no numèrics. Un preprocessat soluciona aquests dos problemes.

El pas més important és el tractament de valors perduts. S'han de substituïr per uns que siguin utilitzables. Ens podem trobar dues situacions per a un valor perdut: variable de valors numèrics i variable de valors discrets. El preprocessat decidit per als nostres problemes és el més bàsic i intuïtiu:

- **Valor perdut numèric:** Suposarem que en aquestes variables els valors perduts no surten del rang de la mostra i que el cas més probable és que tinguin

un valor proper a la mediana dels valors dels que disposem. Com a resultat, els valors perduts seran substituits per la mediana de la variable on es troba.

- **Valor perdut discret:** Suposarem que en aquestes variables els valors perduts no suposen noves classes (S'ha perdut un cas reconeixible) i que el cas més probable és que tinguin un valor semblant a la moda dels valors dels que disposem. Com a resultat, els valors perduts seran substituits per la moda de la variable on es troba.

Ara que totes les dades tenen un valor definit, fa falta que passem totes les variables a numériques. Així tots els mètodes amb els que volem comparar podran fer servir els problemes descrits. La manera de procedir és la següent:

- S'identifiquen els valors possibles que té la variable discreta.
- S'adjudica un enter a cada classe. En el cas de variables ordenades o circulars, els enters seran ordenats igualment segons la prioritat.
- A cada observació s'adjudicarà el enter que li pertoqui.

3.4 Avaluació de l'error

Per avaluar el rendiment dels diversos models sobre els problemes es faran servir tres mesures: error de predicció, *sparsity* i temps d'entrenament.

Error de predicció
En els problemes de classificació es farà servir el percentatge mig d'error (Missclassification Error o ME en anglès). Pren valors en l'interval [0,1] significant 1 equivocació en totes les prediccions i 0 si no s'ha comés cap error. Si tenim $\hat{\mu}$ el vector de prediccions i y els resultats de les observacions l'error es calcula mitjançant

$$ME = \frac{1}{n}\sum_i \mathbb{1}(y_i \neq \hat{\mu}_i)$$

$$\mathbb{1}(B) = \begin{cases} 0, & si \quad B \\ 1, & si \quad \neg B \end{cases}$$

En els problemes de regressió es farà servirla fracció de variança inexplicada (Fraction of Variance Unexplained o FVU en anglès). Pren valors en l'interval [0,1] significant 0 la predicció perfecta del resultat i 1 que la predicció és totalment contraria respecte a la mitja. Si tenim $\hat{y}$ el vector de prediccions, y els resultats de les observacions i $\bar{y}$ la notació de la mitjana l'error es calcula mitjançant

$$FVU = \frac{SS_{err}}{SS_{tot}}$$

$$SS_{err} = \sum_i (y_i - \hat{\mu}_i)^2$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$

Sparsity
Sparsity traduït literalment com dispersió, s'entén com la proporció de coeficients del model que són diferents de zero. Es valua en el domini [0,1] i és calculat com

$$Sparsity = \frac{number\ of\ coefficients\ used}{total\ number\ of\ coefficients}$$

Una *sparsity* propera a 0 és sinònim de ràpida avaluació, robustesa, interpretabilitat i reduït ús de memòria. Quant més es redueixen el nombre d'entrades, més es pot sintetitzar l'informació d'entrada necessària per avaluar el model agilitzant el procés de predicció. Un model és robust quan totes les seves entrades són significatives per la qual cosa un menor nombre elimina redundància. Es pot veure fàcilment l'importància de les entrades amb un model molt dispers doncs són els úincs que tenen coeficients diferents de 0. Finalment, un model dispers ha de tenir en compte pocs factors i fa pocs càlculs reduïnt el espai que ocupa la seva utilització.

Com a punt afegit, els problemes escollits en aquest treball (i la majoria de problemes reals) tenen un nombre elevat d'observacions. Una bona *sparsity* significaria una bona sintetització del problema.

Temps d'entrenament
Mesurat com els segons necessàris per aprendre un problema. Aquest és un indicador directe de l'agilitat que té un model per aprendre les dades d'entrada. Serà calculat com els segons que tarda desde que es presenta el conjunt de dades d'entrada fins que acaba l'entrenament.

4 Xarxes neuronals basades en similitud

Les SNNs són una variant de les conegudes xarxes neuronals artificials que utilitza una funció de similitud sobre les entrades. Gracies a això, existeix la capacitat d'addició de coneixement previ mitjançant la definició del tipus de variable. El model, també, és capaç de tractar directament amb valors perduts sense problemes.

La manera d'aprendre d'una SNN és buscant prototips, observacions del conjunt de dades d'entrada que representen un clúster d'observacions semblants a aquest. Noves dades són comparades amb els prototips i la predicció es fa d'acord amb aquestes interaccions.

Suposem $X_{n\times k}$ una matriu que conté n observacions de k variables i Y un vector de n respostes a la matriu X. Els passos que segueix una SNN en el seu procés d'entrenament són els següents:

- **Càlcul de distàncies** Mitjançant aquest procés, a partir de X s'extreu $S_{n\times n}$, una matriu simètrica amb la diagonal a 0 que conté les distàncies entre observacions.

- **Selecció de prototips** En la fase de selecció de prototips es transforma S en $S'_{n\times d}$ on $d << n$. Es busca trobar les d observacions més significatives per aconseguir la simplificació del problema i l'eliminació d'informació redundant o irrellevant.

- **Modelat** El pas final on es crea el model a partir de S' com a predictors i Y com a resposta.

El procés de predicció de noves observacions té dues etapes: càlcul de distàncies amb els prototips i predicció amb el model. Tenint m el nombre de noves observacions i d el nombre de prototips, es calcula $P_{m\times d}$ que conté les distàncies amb els prototips. Aquestes entrades es passen al model per adquirir les prediccions finals.

L'algorisme utilitzat en la primera fase és l'anomenada distància de Gower[7], en la segona LASSO, Random forset i Support Vector Machine i per al modelat Generalized Linera Model, Random forest i Support Vector Machine.

4.1 Distància de Gower

La distància de Gower és una extensió del popular coeficient general de similitud de Gower[7]:

$$s_{ij} = \frac{\sum_{k=1}^{n} s_{ij}\delta_{ijk}w_k}{\sum_{k=1}^{n} \delta_{ijk}w_k}$$

on n és el nombre de variables, s_{ijk} és la similitud entre les observacions i i j per la variable k, δ_{ijk} és igual a 0 si la variable k per alguna de les dues observacions és un valor perdut 1 contrariament i w_k és el pes de la variable. Es pot entendre llavors que la similitud de dues observacions és la mitja ponderada de la similitud de les variables que estan disponibles. Per passar a distància aquesta similitud tenim $d_{ij} = \sqrt{1 - s_{ij}}$ que podem transmetre al factor s_{ijk}.

La primera variable tractada és la **quantitativa** on Gower fa servir la distància Manhattan que calcula el valor absolut de la diferència. Per normalitzar es divideix entre el rang de la variable denotat com R_x. Així tenim l'equació resultant:

$$d_{ijk} = \sqrt{\frac{\mid x_{ik} - x_{jk} \mid}{R_x}}$$

$$R_x = max(x_k) - min(x_k)$$

Per a variables **nominals** i **binàries simètriques** Gower fa servir l'operació d'igualtat. Tenint x_{ik} el valor de l'observació i per a la variable nominal o binària simètrica k:

$$d_{ijk} = \begin{cases} 0, & si \quad x_{ik} = x_{jk} \\ 1, & si \quad x_{ik} \neq x_{jk} \end{cases}$$

L'últim tipus de variable que Gower usa en el seu coeficient general de similitud és la **binària asimètrica** (o dicotòmica). Tenint x_{ik} el valor de l'observació i per a la variable binària asimètrica k:

$$d_{ijk} = \begin{cases} 0, & si \quad x_{ik} = 1 \quad i \quad x_{jk} = 1 \\ 1, & altrament \end{cases}$$

La distància per a variables **ordinals** no ha estat definida per Gower, proposarem una mesura per extendre el mètode de distàncies. Una variable ordinal està distribuida en m nivells, pren valors enters entre 1 i m, i podem denominar el rang com R_x. Tenint x_{ik} el valor de l'observació i per a la variable ordinal k s'han d'escalar llavors els valors de la variable per finalment tenir:

$$d_{ijk} = \frac{\mid z_{ik} - z_{jk} \mid}{max(z_k) - min(z_k)}$$
$$z_{ik} = (x_{ik} - 1)/R_x$$
$$R_x = max(x_k) - 1$$

La distància per a variables **circulars** no ha estat definida per Gower, es farà servir l'extensió descrita en l'article[8]. Una variable circular està distribuida en m nivells, pren valors enters entre 1 i m. Tenint x_{ik} el valor de l'observació i per a la variable circular k, la distància és:

$$d_{ijk} = \begin{cases} \sqrt{1 - \mid 1 - 2 \mid \frac{x_{ik}}{m} - \frac{x_{jk}}{m} \mid \mid}, & si \quad m \quad parell \\ \sqrt{1 - \mid 1 - \frac{2m}{m-1} \mid \frac{x_{ik}}{m} - \frac{x_{jk}}{m} \mid \mid}, & si \quad m \quad senar \end{cases}$$

4.2 Modelat

El modelat és el proces mitjançant el qual s'entrena la construcció que després servirà per predir. Els mètodes escollits són un model lineal genralitzat (Generalized lineal model o GLM en anglès), arbres aleatoris (Random Forests o RF en anglès) i màquines de vectors suport (Support Vector Machine o SVM en anglès).

4.2.1 Generalized lineal model

Un model lineal prediu el valor esperat d'una quantitat desconeguda com una combinació lineal de les entrades (les variables independents). Això implica que canvis en les entrades suposen canvis proporcionals en la resposta.

GLM es comporta de manera diferent depenent de si el problema és de regressió o classificació. En el primer cas, executa regressió lineal on la combinació lineal de les entrades dóna un valor numèric que és directament la resposta. Però per classificació, fa una regressió logística que calcula la probabilitat de pertinença a una classe.

Tenint $X_{n \times k}$ la matriu d'entrada i x_j el vector de valors per la variable j, GLM

fa una combinació lineal de les variables segons els coeficients $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_k$ per calcular prediccions $\hat{\mu}$. La forma d'un GLM es pot representar com

$$f(\hat{\mu}) = \sum_{j=1}^{k} x_j \hat{\beta}_j$$

on f és una funció apropiada que s'anomena funció d'enllaç.

L'entrenament d'un GLM es basa en trobar els valors de $\hat{\beta}_j$ que minimitzen l'error quadràtic. Tenint Y el vector resposta i y_i la resposta per la observació i l'objectiu d'optimització és minimitzar

$$E = \sum_{i=1}^{n} (y_i - \hat{\mu}_i)^2 = (Y - \hat{\mu})^2$$

Del que podem deduir l'error per una seqüència de coeficients $\hat{\beta}$

$$E(\hat{\beta}) = (Y - \sum_{j=1}^{k} x_j \hat{\beta}_j)^2$$

Per poder trobar els millors valors de $\hat{\beta}_j$ que minimitzen l'error E, es calculen les derivades parcials $\bigtriangledown E(\hat{\beta}_j)$ per construir un sistema d'equacions i finalment es soluciona amb la tècnica de *gradient descent*. *Gradient descent* comença amb una solució de l'equació amb tots els coeficients a 0, calcula el gradient de l'error i ofereix una nova solució. Normalment aquest procés es repeteix fins que convergeix una solució o s'arriba a un màxim d'iteracions.

4.2.2 RandomForest

Un *Random Forest* és un tipus de model d'ensamblatge de models més petits. La premisa bàsica és que construir un petit arbre de decisions amb poques variables és computacionalment fàcil. Així que si construim nombrosos arbres petits i els juntem en una formació més gran, obtenim un model robust i eficaç. Un arbre de decisió és un tipus especial d'arbre on les fulles descriuen un resultat i cada node és una condició que discrimina en dues situacions.

Els Random Forests són considerats com un dels millors mètodes de modelat estadístic. Les raons sent que, tot i contenir gran grau d'aleatorietat, han demostrat ser robustos i eficaços. En addició a això, tenen altres propietats interessants com la capacitat de consultar la importància de les variables de les dades d'entrada.

Tenint un Random Forest ja entrenat, noves dades es prediuen segons les respostes dels diversos arbres. Si el problema és de classificació, es prediu la classe que tingui la majoria de vots. En regressió es fa la mediana dels arbres.

Algorisme 2 Random Forest

Precondició: Entrada $S := (\vec{x}_1, y_1), ..., (\vec{x}_n, y_n)$, variables V i nombre d'arbres B
Funció: Random Forest(S, B)
    H $\leftarrow \emptyset$
    for: i $\in 1, ..., B$
        $S^{(i)} \leftarrow$ mostra aleatòria de S
        $V^{(i)} \leftarrow$ mostra aleatòria petita de V
        $h_i \leftarrow$ RandomizedTreeLearn($S^{(i)}$, $V^{(i)}$)
        $H \leftarrow H \cup h_i$
    end for
    return H
end function
function RandomizedTreeLearn(S , V)
    A $\leftarrow$ arbre buit amb tants nodes com variables en V
    Per cada node de A:
        $v_i \leftarrow$ la variable de V que millor divideix S
        $V \leftarrow V \cap v_i$
        node actual $\leftarrow$ divisió segons v_i
        if (node conecta a fulles) adjudica valors a les fulles
    return A
end function

4.2.3 Support Vector Machine

Les màquines de vectors suport (Support Vector Machine o SVM en anglès) són un tipus de modelat que representa les observacions d'un conjunt de dades com a punts en l'espai i busca hiperplans lineals que els separi en regions segons el seu resultat. Els hiperplans que busca també estan ajustats al màxim per separar de manera òptima les regions. Una SVM és no probabilística, cosa que significa que un punt pertany de manera absoluta a una regió. Noves dades són representades en el mateix espai i la predicció resulta de la regió on es trobi.

Les SVMs han estat disenyades com a classificadors lineals que divideixen les dades en dos categories de manera lineal, però mitjançant un *kernel* existeix la

possibilitat de mapejar les dades en una major dimensionalitat. Un *kernel* és una funció $K : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ que calcula una relació entre dos observacions $\vec{u}$ i $\vec{v}$. Els kernels més coneguts són:

Kernel lineal

$$K(u, v) =< u, v >$$

on $<,>$ és el producte escalar.

Kernel polinomial

$$K(u, v) = (< u, v > +\gamma)^d$$

amb paràmetres $\gamma \in \mathbb{R}$ i $d \in \mathbb{N}$.

Kernel Gaussià , també conegut com Radial Basis Function o RBF

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

amb paràmetre $\sigma \in \mathbb{R}$.

Kernel Sigmoidal

$$K(u, v) = \tanh(\alpha < u, v > +r)$$

amb paràmetres $\alpha > 0$ i $r < 0$, on tanh és la tangent hiperbòlica.

Els grans avantatges de fer servir SVMs són:

- Les SVMs estan basades en la minimització de risc.

- No pateixen de mínims locals i tenen pocs paràmetres, normalment C i el Kernel.

- Les SVMs són resistents al overfitting (però no del tot).

El kernel RBF és l'escollit per els experiments del nostre treball doncs és el més popular en l'entrenament de SVMs. Això és degut a que té rang entre 1 (x = x') i 0 (valors totalment contraris) i , per tant, s'aproxima a una mesura de similitud.

SVMs lineals

Considerem un conjunt de dades $X = \{\vec{x}_1, \ldots, \vec{x}_n\}$ etiquedates en dues classes ω_1, ω_2 com $z_1, \ldots, z_n$ amb $z_i = +1$ si $\vec{x}_i \in \omega_1$ i $z_i = -1$ si $\vec{x}_i \in \omega_2$. Si declarem una funció afí $g(\vec{x}) = \vec{w} \cdot \vec{x} + b$, llavors tenim un discriminant lineal $sgn(g(\vec{x}))$, per el qual ens agradaria:

$$\begin{aligned} \vec{w} \cdot \vec{x}_i + b > 0 \qquad & \vec{x}_i \in \omega_1 \, (z_i = +1) \\ \vec{w} \cdot \vec{x}_i + b < 0 \qquad & \vec{x}_i \in \omega_2 \, (z_i = -1) \end{aligned}$$

En resum, $z_i(\vec{w}\cdot\vec{x}_i+b)>0$, traduible com $z_i\,g(\vec{x}_i)>0$, per tot $1\leq i\leq n$. Donat el hiperplà $\pi : g(\vec{x})=0$, la distància perpendicular de $\vec{x}$ a π és $d(\vec{x},\pi)=\frac{|g(\vec{x})|}{\|\vec{w}\|}$. Els *vectors suport* són aquells $\vec{x}$ més propers al hiperplà. Reescalant $\vec{w},b$ tal que $|\vec{w}\cdot\vec{x}+b|=1$ per aquests punts més propers, s'obté $|\vec{w}\cdot\vec{x}+b|\geq 1$. Els *vectors suport* ara son $\{\vec{x}_i \;/\; |\vec{w}\cdot\vec{x}_i+b|=1\}$.

El *marge* $m(\pi)$ d'un pla π pot ser escrit com el doble de la seva distància amb qualsevol vector suport: $m(\pi)=2\,d(\vec{x}_{\mathrm{SV}},\pi)=\frac{2}{\|\vec{w}\|}$, on $|g(\vec{x}_{\mathrm{SV}})|=1$. Per maximitzar el marge, s'ha de minimitzar $\|\vec{w}\|$ subjecte a $z_i\,(\vec{w}\cdot\vec{x}_i+b)\geq 1$, per tot $1\leq i\leq N$.

En el cas on un hiperplà que separi correctament les dades d'entrada no existeixi, un conjunt devariables *flexibles* son introduïdes per permetre petites violacions del marge, donant la matriu:

$$z_i(\vec{w}\cdot\vec{x}_i+b)+\xi_i\geq 1 \qquad i=1,...,n$$

on $\xi_i\geq 0$. Per que hi hagi un error, el corresponent ξ_i ha de superar la unitat, així $\sum_i\xi_i$ és el marge superior del nombre d'errors en l'entrenament. El hiperplà òptim pot ser trobat com a solució de la norma L_1 quadràtica:

$$\min_{\vec{w},\xi} \qquad \tfrac{1}{2}||\vec{w}||^2+C\sum_{i=1}^{N}\xi_i$$

$$s.t. \quad z_i(\vec{w}\cdot\vec{x}_i+b)\geq 1-\xi_i,\ i=1,\ldots,N$$

La solució d'aquest problema d'optimització correspon al punt de sella de la seva Lagrangiana associada:

$$\frac{||\vec{w}||^2}{2}-\sum_{i=1}^{N}\alpha_i(z_i(\vec{w}\cdot\vec{x}_i+b)-1+\xi_i)+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\mu_i\xi_i$$

on $\alpha_i,\mu_i\geq 0$ for $i=1,...,n$.

Un cop solucionat el problema, el vector solució $\vec{w}^*$ pot ser expressat com una extensió lineal sobre els vectors suport:

$$\vec{w}^*=\sum_{i=1}^{N}\alpha_i^* z_i\vec{x}_i$$

Els vectors suport són exactament aquells $\vec{x}_i\in D$ pels quals $\alpha_i^*>0$.

SVMs no lineals

Assumim ara que no es possible separar el set d'entrenament amb un hiperplà, però la SVM pot projectar les entrades x_i a un major espai dimensional de característiques z_i fent servir una projecció no lineal on el conjunt és casi separable.

Primer es mapeja el conjunt d'entrenament a una altre espai dimensional $\mathcal{H}$, fent servir el mapeig $\phi : \mathbb{R}^d \to \mathcal{H}$. El problema d'optimització és ara de la forma

$$\max_{0 \leq \alpha_i \leq C} -\frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j < \phi(x_i), \phi(x_j) > + \sum_{i=1}^{n} \alpha_i = (w^*, \alpha^*)$$

Ara, l'algorisme d'entrenament només dependria del conjunt d'entrenament a través de productes escalars en $\mathcal{H}$, per exemple, en funcions de la forma $< \phi(x_i), \phi(x_j) >$. Ara, si hi hagués un kernel tal que $K(x_i, x_j) =< \phi(x_i), \phi(x_j) >$, només necessitariem fer servir K en l'algorisme d'entrenament i no necessitariem saber ϕ. En el cas no separable linealment, si es reemplaça $< x_i, x_j >$ per $K(x_i, x_j)$ a tot arreu en l'algorisme d'entrenament, l'algorisme produirà una SVM que resideix en $\mathcal{H}$. Finalment, un classificador en $\mathcal{H}$ és

$$f(x) = sgn(< w^*, x > + b^*) = sgn\left(\sum_{i=1}^{n} \alpha_i^* y_i K(x_i, x) + b^*\right).$$

4.3 Selecció de prototips

La selecció de prototips consisteix en trobar les observacions més descriptives del problema. A partir de la matriu de distàncies $S_{n\times n}$ s'arriba a la matriu d'entrenament $S'_{n\times p}$ que es passarà al model per fer l'aprenentatge. L'objectiu és identificar les observacions que representen millor els possibles clústers que es formen en les dades i prendre les distàncies respecte aquests com variables. Com S és simètrica, podem entendre que tenim n observacions amb n variables i que mètodes que seleccionin variables serviran com a seleccionadors de prototips. O també podem extreure les files més importants segons algun criteri en $S^2_{p\times n}$ i transposar la matriu.

4.3.1 LASSO

LASSO[9] (least absolute shrinkage and selection operator) és un mètode de regressió que realitza selecció de variables i regularització per millorar la precissió i interpretabilitat del model. El seu funcionament és molt semblant a l'entrenament d'un GLM però amb certes restriccions. Suposant $X_{n\times k}$ la matriu d'entrada i x_j el vector de valors per la variable j, LASSO busca coeficients $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_k$ per calcular prediccions $\hat{\mu}$,

$$\hat{\mu} = \sum_{j=1}^{k} x_j \hat{\beta}_j$$

L'ús de LASSO es basa en trobar els valors de $\hat{\beta}_j$ que minimitzen l'error quadràtic i la seva norma L_1 no supera un llindar λ. Tenint Y el vector resposta i y_i la resposta per la observació i l'objectiu d'optimització és minimitzar

$$min\left\{ \sum_{i=1}^{n} (y_i - \hat{\mu}_i)^2 \right\} \quad subjecte\ a \quad \sum_{j=1}^{k} \mid \hat{\beta}_j \mid \leq \lambda$$

La notació $\mid \hat{\beta}_j \mid$ és la norma L_1, la suma dels valors absoluts del vector $\hat{\beta}_j$. D'aquesta manera, el terme del que depèn error quadràtic limita la quantitat de termes majors que zero. El millor valor de λ és sensible al problema. Per trobar-lo s'ha de declarar una seqüència de valors i provar-los en LASSO mitjançant Cross Validation.

Algorisme 4 LASSO

Precondició: Variables d'entrada $X := (x_1, ..., x_k)$, variable de sortida Y
Funció: LASSO(X, Y, λ)
    B $\leftarrow$ seqüència de k 0s
    R $\leftarrow$ Y
    for: i $\in 1, ..., maxiteracions$ & (l'algorisme no ha convergit)
        $v_j \leftarrow$ variable de X més correlacionada amb R
        incrementaB_j en la direcció de la correlació entre v_j i R sense superar λ
        R $\leftarrow Y - \hat{\mu}$
    end for
end function

4.3.2 RandomForest

Una propietat que tenen els Random Forests és la capacitat d'identificar l'importància d'una variable en el procés d'entrenament. Durant l'aprenentatge, cada arbre de decisió registra el error de predicció que comet i l'informació que aporta cada variable. Amb aquestes dades, el model final disposa d'una valoració de com útil és cada variable el qual es pot fer servir per realitzar una selecció de variables.

El procés que ens interessa és trobar les p columnes més importants de $S_{n\times n}$ per arribar a $S'_{n\times p}$. Mitjançant Random Forest, el procés és entrenar un model, extreure les importàncies de les variables i eliminar del conjunt d'entrada les que tinguin menor importància. El problema que se'ns presenta és el desconeixement de p.

La solució presentada és monitoritzar l'error d'entrenament a mesura que s'eliminen variables per trobar el punt óptim (veure figura 3). Primer entrenem un model inicial, avaluem l'error d'entrenament, eliminem les m variables menys importants i l'entrenem de nou amb les $n-m$ variables. El valor m es decideix com el nombre de variables per sota del 10-quantil de la distribució de les importàncies de les variables. Així fins que trobem el subconjunt de variables p que dóna el menor error d'entrenament (veure algorisme 5).

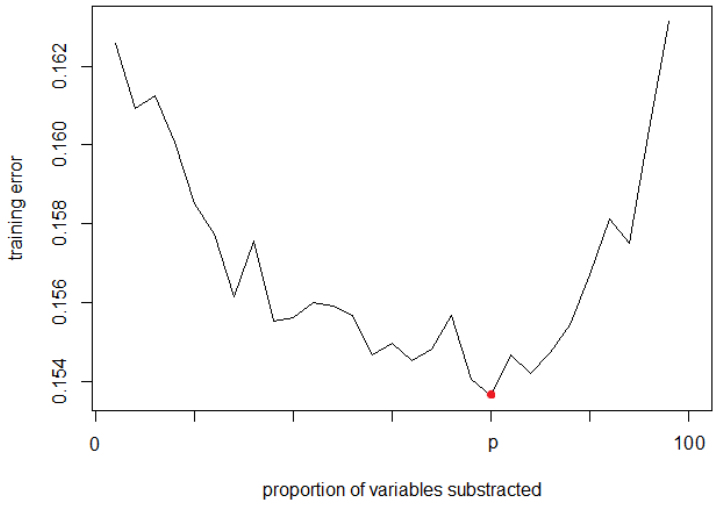

Figura 3: Training error vs proporció de variables extretes

Algorisme 5 Selecció de variables mitjançant RF

Precondició: Variables d'entrada $S := (s_1, ..., s_n)$, variable de sortida Y
Funció: RFselect(X, Y)
    rf ← RandomForest(S,Y)
    I ← rf.importance
    E ← rf.trainerror
    while: (S conté variables)
        S ← variables amb més importancia que el 10-quantil de I
        rf ← RandomForest(S,Y)
        E2 ← rf.trainerror
        if (E2 > E) **break**
        else E ← E2
        I ← rf.importance
    end for
    return S
end function

4.3.3 Support Vector Machine

La definició d'una SVM és que busca observacions en el conjunt d'entrada per considerar-los vectors suport per classificar tot el espai de possibles observacions. Es pot entendre com que busca les observacions més representatives per aprendre.

Tenint $S_{n\times n}$ com a dades d'entrada, es pot entrenar una SVM per buscar els vectors suport. El resultat serà un nombre d'observacions sv que podem traduir a una matriu $S'_{sv\times n}$. Si entenem aquestes sv observacions com les més importants del problema i el fet de que $S_{n\times n}$ és simètrica, podem construir la matriu $S_{n\times sv}$ i considerar-la com una selecció de prototips.

4.4 Construccions avaluades

Per decisió de disseny, totes les construccions de SNN tenen com a càlcul de distàncies la distància de Gower. $S_{n\times n}$ és calculada mitjançant la distància de Gower a partir de $X_{n\times k}$ Les diferents construccions programades es basen en diferents formes de seleccionar prototips i els mètodes de modelat un cop escollits els prototips.

A l'hora de predir, totes les implementacions de SNN procedeixen igual. Les noves dades $X2_{m\times k}$ són posades contra $X_{p\times n}$ per calcular les distàncies $S2_{m\times p}$. S2 és llavors predita segons el model.

Algunes construccions han quedat apartades doncs intuitivament no suposaven combinacions aptes. LASSO és un mètode lineal que selecciona en base a correlació, un criteri del que es poden aprofitar gairebé tots els models existents. En canvi, Random Forest i SVM tenen el seu propi procediment. Així s'ha decidit que la selecció de prototips amb aquests dos mètodes només es farà servir sobre modelats d'ells mateixos. Les construccions finalment avaluades són:

LASSO - glm Sobre la matriu de distàncies $S_{n\times n}$ s'aplica *lasso* per trobar les columnes que ofereixen més informació sobre Y per obtenir $S'_{n\times p}$. S' és llavors entregada a *glm* per modelar la SNN.
Intuició: LASSO i GLM són tots dos de tipus lineal i els seus fonaments són semblants. S'espera que GLM aprofiti bé la selecció que efectua LASSO.

LASSO - Random Forest En aquest cas Random Forest funciona millor amb major quantitat d'informació. És per això que *lasso* s'executa sobre $S_{n\times n}$ tantes vegades com nombre de voltes de CV i s'agafen tots els prototips seleccionats per formar $S'_{n\times p}$. S' és llavors entregada a Random Forest per modelar la SNN.
Intuició: La selecció de prototips que fa LASSO és de tipus lineal, però es

vol comprovar si un model no lineal com Random Forest es pot aprofitar de la selecció de LASSO.

LASSO - SVM Per proporcionar la major dimensionalitat a SVM per tenir més grau de separabilitat es procedirà de manera semblant a LASSO-RandomForest. És per això que *lasso* s'executa sobre $S_{n\times n}$ tantes vegades com nombre de voltes de CV i s'agafen tots els prototips seleccionats per formar $S'_{n\times p}$. S' és llavors entregada a SVM per modelar la SNN.
Intuició: La selecció de prototips que fa LASSO és de tipus lineal, però es vol comprovar si un model no lineal com SVM es pot aprofitar de la selecció de LASSO.

Random Forest - Random Forest Tal com s'ha descrit en el seu propi apartat, s'executarà un seguit de vegades el modelat de Random Forest per eliminar les variables poc importants. D'aquesta manera desde la matriu de distàncies $S_{n\times n}$ s'obté $S'_{n\times p}$. S' és llavors entregada a Random Forest per modelar la SNN.
Intuició: El millor model que es pot aprofitar d'una selecció de prototips amb Random Forest s'espera que sigui ell mateix.

SVM - SVM Sobre la matriu de distàncies $S_{n\times n}$ es modela una SVM per trobar els vectors suport de S. S'interpreten aquests com prototips i s'utilitzen com variables per obtenir $S'_{n\times p}$. S' és llavors entregada a SVM per modelar la SNN.
Intuició: El millor model que es pot aprofitar d'una selecció de prototips amb SVM s'espera que sigui ell mateix.

5 Problemes modelats

En aquesta secció es troben especificats els problemes que s'utilitzaran per comprovar l'efectivitat dels models fets servir durant el treball. Tots ells han estat recollits des del repositori online UCI[10]. En total s'han provat vuit problemes dels quals Bank Marketing es considera extremadament difícil doncs conté nombrosos valors perduts, té variables circulars i és de classificació desvalancejada.

S'han buscat problemes reals de complexitat elevada per evaluar la veritable capacitat de les implementacions de SNNs. La complexitat es tradueix a que els mètodes de modelat actual no han donat resultats prou satisfactoris per als autors dels respectius. També tenen diversos tipus de variables i valors perduts.

5.1 Problemes clàssics de benchmarking

Els problemes trobats en aquesta secció són els que han estat fets servir com a proves de comparació. A continuació la llista dels problemes amb una petita descripció:

- **Horse colic:** Es tracten els casos de cavalls que pateixen de còlic biliar. Diferents variables sobre la condició física són donats per intentar predir si la lesió ha estat quirúrgica o no. El problema és bastant complex doncs disposa de variades variables i un alt nombre de valors perduts. A més, les dades contenen valors estranys com un 3 en una variable binària. Aquests valors són considerats perduts a falta de documentació al respecte.

- **Hepatitis:** Un interessant problema que descriu pacients d'hepatitis. La condició i característiques de la persona són denotats en les variables. L'objectiu és predir si el pacient viu o mor a causa de la hepatitis, aspecte de gran importància en el camp mèdic.

- **Credit approval:** Per qüestions de protecció de confidencialitat, no s'especifica què signifiquen les variables. L'única dada coneguda és que es vol predir si una aplicació de crèdit ha de ser acceptada o no. El desconeixement parcial del significat de les variables fa el problema més difícil. A més te un nombre considerable de observacions.

- **Heart:** Molt semblant al problema d'hepatitis però amb problemes del cor. El pacient és descrit en les variables. En aquest cas, es vol predir si el pacient pateix de problemes del co o no.

- **Audiology:** Un altre cas de predicció mèdica aquesta vegada de la condició de l'oída. Aquest problema és especial doncs tracta classificació de més de dues

classes. Té un total de 24 classes que han estat resumides en 4 per qüestions de similitud i simplicitat.

- **Contraceptive:** Aquest set de dades consisteix en un estudi dús de mètode anticonceptiu en dones d'Indonèsia. Cada observació és una dona casada i les seves dades demogràfiques i socio-econòmiques. El objectiu de predicció és el tipus de mètode anticonceptiu que utilitza. És un problema veritablement curiós doncs es pretén predir el mètode anticonceptiu a partir de variables aparentment no correlacionades.

- **Servo:** Interessant problema de regressió que tracta servomotors. Només té quatre variables, cosa que pot semblar que complica el problema. S'intenta predir el temps de reacció del sistema a partir de les quatre parts que el composa.

- **Automobile:** Segon problema de regressió en el qual es tracten casos de subhastes de cotxes. Les variables denoten les propietats i valoracions del cotxe que s'està subhastant. Es vol predir el preu de venta que resultarà el cotxe en qüestió.

Data set	Tipus	Mida	Variables	Perduts?
Horse colic	C	368	21 (7Q,6N,8O)	28%
Hepatitis	C	155	19 (6O,13N)	6%
Credit approval	C	690	15 (6O,9N)	5%
Heart	C	270	13 (6Q,3S,3N,1O)	0%
Audiology	CM	226	31 (7O,1N,23A)	2%
Contraceptive	CM	1473	10 (2Q,4N,3O)	0%
Servo	R	167	4 (2Q,2N)	0%
Automobile	R	201	25 (14Q,11N)	3%

Detalls tècnics dels problemes: Tipus inclou Classificació binomial(C), Classificació multinomial(CM) i Regressió(R). Llegenda de tipus de variable: (Q)uantitativa, (N)ominal, (O)rdinal, binària (S)imètrica, binària (A)simètrica i (C)ircular.

5.2 Bank Marketing

Bank marketing és el problema escollit per realitzar proves exhaustives. És d'alta complexitat doncs recull gran quantitat de dificulatats que es poden trobar en el món de l'estadística. Tracta d'una campanya de màrketing d'un bank on es truquen possibles clients que podrien estar interessats en un dipòsit a termini. Es vol predir si una persona és un potencial client interessat en contractar els serveis del banc.

El problema està definit per un total de 20 variables de diferents tipus. Les variables són 10 quantitatives, 4 nominals, 1 ordinal, 2 binàries asimètriques, 1 binària simètrica i 2 circulars. Però d'aquest conjunt se n'han eliminat dues: *duration* doncs és una dada a posteriori que conté la duració de la trucada i *default* que descriu si la persona té deutes personals i només 3 observacions del total tenen el valor a 1.

Un total del 10% dels valors són perduts. A més, la variable pdays compta el nombre de dies entre la trucada objectiu i l'última efectuada i si és un client nou, el valor és 999. Aquesta decisió de diseny es comptarà com si 999 es un valor perdut doncs altera la naturaleza de la variable.

Finalment, la distribució de classes del objectiu és altament irregulartenint una proporció de 11.26% del "si" respecte al "no". Per qualsevol model, això suposarà que després del entrenament sempre dirà "no". Però com el que ens interessa és poder predir el "si", es rebalancejaran les classes. Dintre del set d'entrenament, s'eliminaran observacions amb resultat "no" per igualar la quantitat d'observacions negatives amb les positives per igualar les classes. D'aquesta manera els models aprendran correctament ambdues classes.

6 Resultats

En aquesta secció exposarem els experiments realitzats amb les diferents implementacions de les SNNs respecte als mètodes clàssics GLM, Random forest i SVM. També s'ha avaluat una versió de SNN on no es fa selecció de prototips (denotat com *dist glm*) per propòsits de comparació.

Els mètodes clàssics no permeten els problemes complicats escollits per culpa dels valors perduts i alguns tipus de variables. Per aquesta raó, s'ha executat el preprocés ja descrit per poder executar-los.

Els resultats d'errors, sparsity i temps d'entrenament s'han representat en forma de taula i gràfics individuals. Les taules mostren només les medianes per tenir un resum inicial, els gràfics pretenen ilustrar la distribució de cada apartat. S'han obtingut mitjançant 50 execucions per problema en les quals s'ha fet la divisió entre set d'entrenament i set de testeig de manera aleatòria. La proporció entre els dos sets és del 50%.

En totes les execucions d'entrenament de models s'ha fet servir $(3\times5)CV$. L'objectiu és trobar els millors paràmetres per cada model i extreure el millor rendiment.

Els punts d'avaluació són l'error (ME en cas de classificació, FVU en cas de regressió), la *sparsity* i el temps d'execució en segons. En el cas especial de *Bank Marketing* com existeix especial interés en predir els "yes" també es detalla el error de predicció de "yes".

Horse colic:

Mètode	ME	Sparsity	temps (s)
dist-glm	0,2692	1	10
lasso-glm	0,1704	**0,065**	67
RF-RF	0,162	0,184	40
lasso-RF	0,1714	0,113	74
SVM-SVM	0,1712	0,538	5
lasso-SVM	0,1721	0,121	38
lasso	0,182	0,438	**2**
RF	**0,1494**	1	**2**
SVM	0,1533	1	3

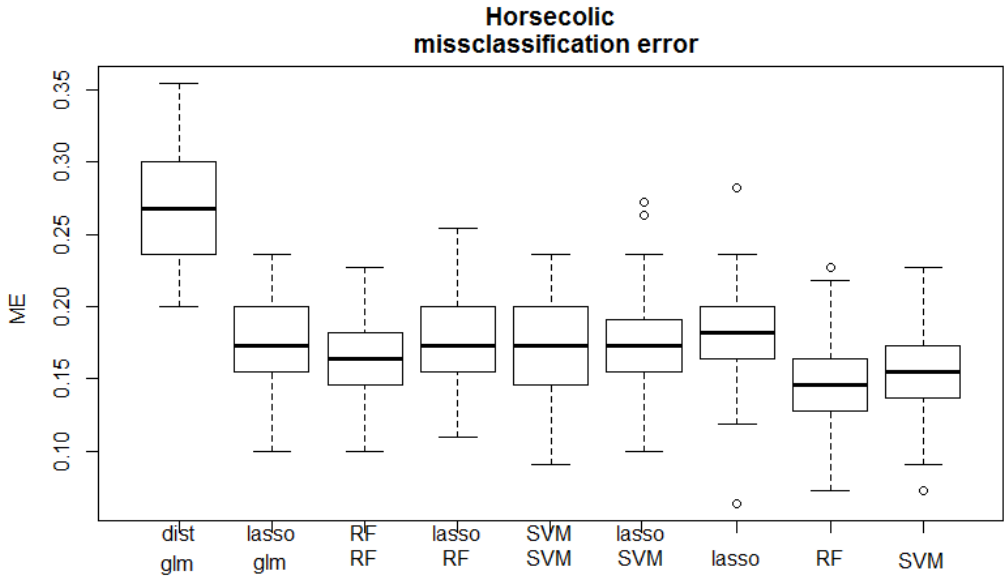

Figura 4: ME per Horse colic

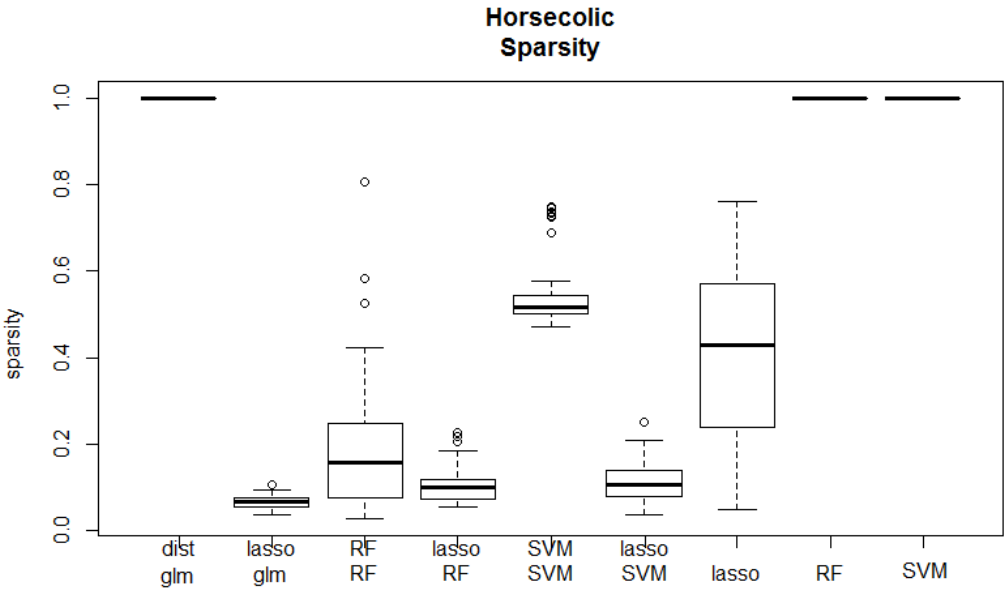

Figura 5: Sparsity per Horse colic

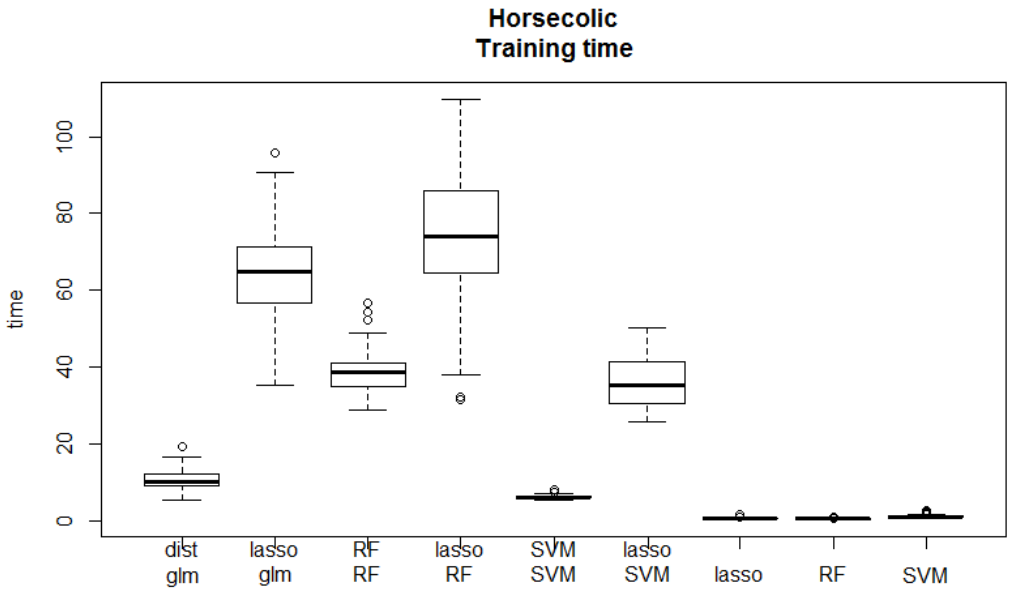

Figura 6: Temps d'entrenament per Horse colic

Hepatitis:

Mètode	ME	Sparsity	temps (s)
dist-glm	0,1923	1	1,5
lasso-glm	0,179	**0,117**	1,16
RF-RF	0,1923	0,1428	1,885
lasso-RF	0,179	**0,117**	1,235
SVM-SVM	0,179	0,5195	0,865
lasso-SVM	0,179	0,1818	2,63
lasso	0,1667	0,3684	0,61
RF	**0,1538**	1	**0,08**
SVM	0,179	0,1446	0,365

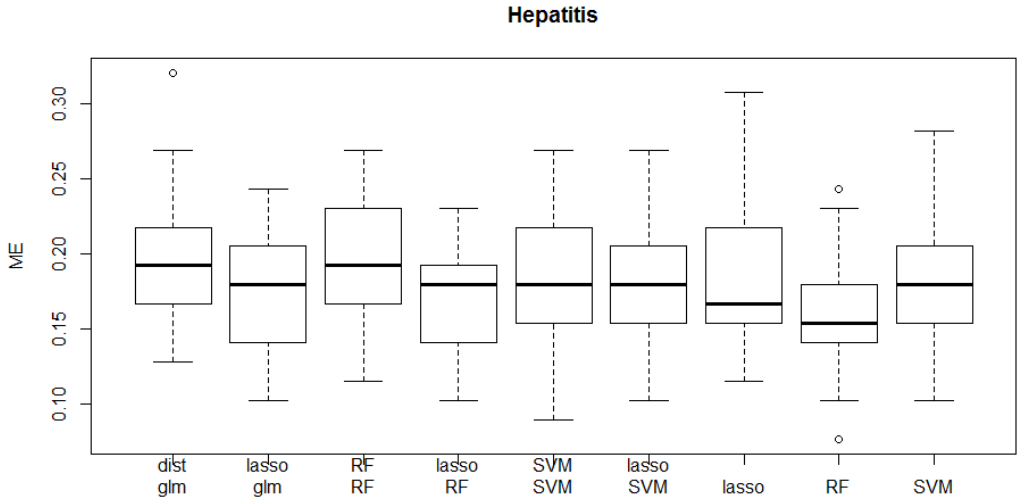

Figura 7: ME per Hepatitis

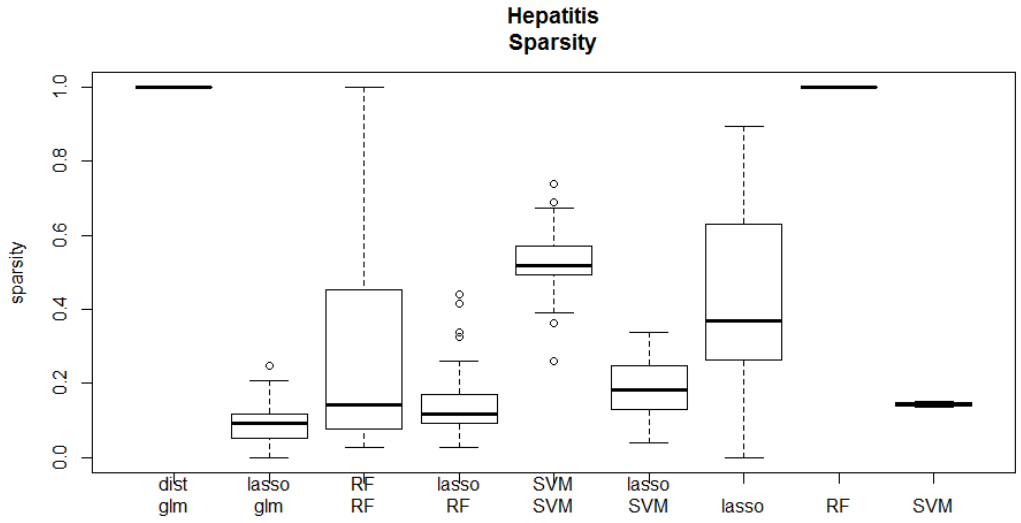

Figura 8: Sparsity per Hepatitis

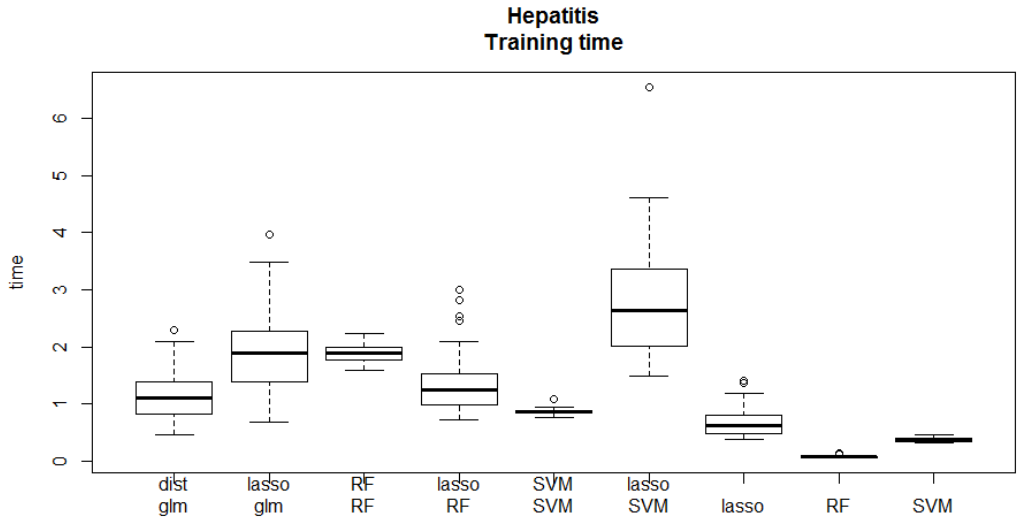

Figura 9: Temps d'entrenament per Hepatitis

Credit approval:

Mètode	ME	Sparsity	temps (s)
dist-glm	0,2	1	29,975
lasso-glm	0,145	**0,05**	18,135
RF-RF	0,1565	0,51	31,59
lasso-RF	0,145	0,055	16,57
SVM-SVM	0,14	0,635	5,87
lasso-SVM	0,142	**0,05**	16,25
lasso	0,1478	0,6	0,56
RF	**0,1333**	1	**0,37**
SVM	0,1362	0,469	0,97

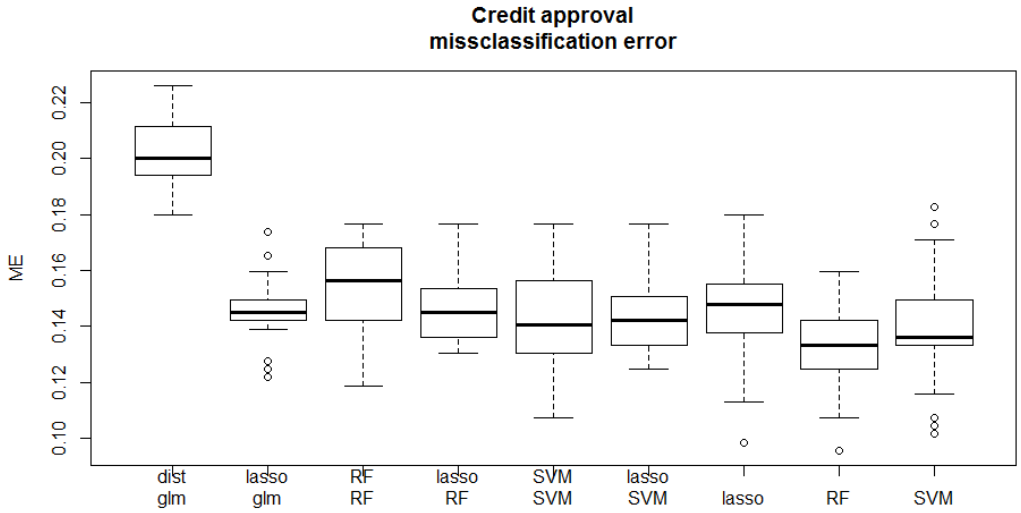

Figura 10: ME per Credit approval

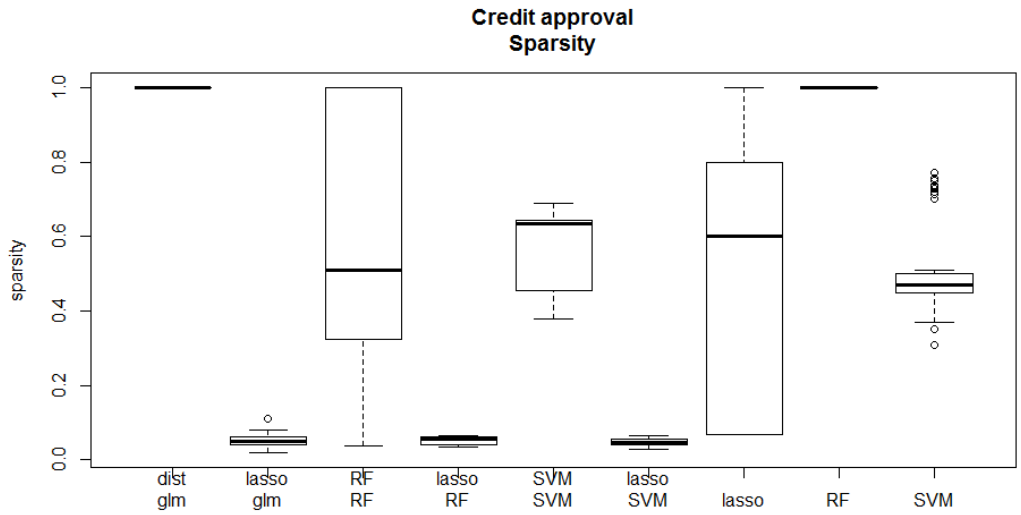

Figura 11: Sparsity per Credit approval

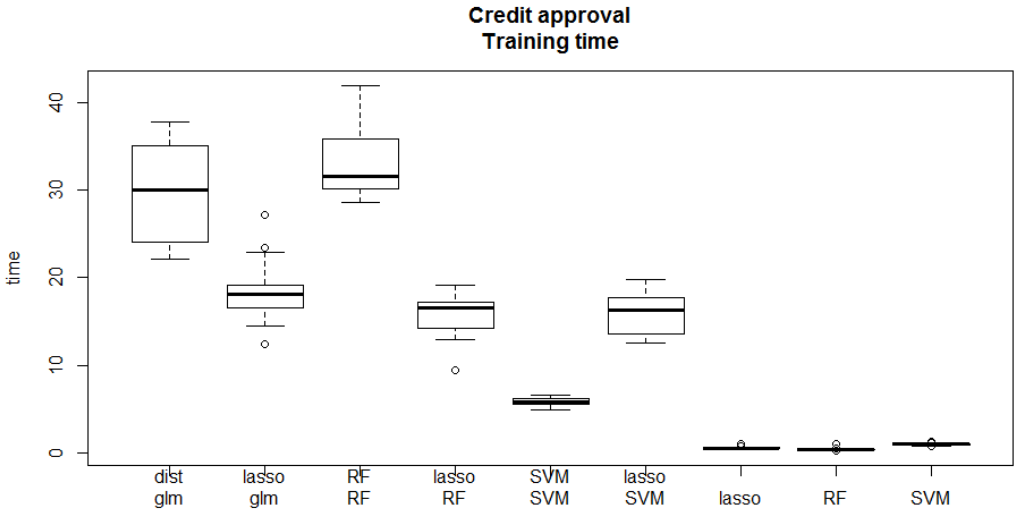

Figura 12: Temps d'entrenament per Credit approval

Heart:

Mètode	ME	Sparsity	temps (s)
dist-glm	0,259	1	6,39
lasso-glm	**0,163**	0,078	2,64
RF-RF	0,17	0,237	4,43
lasso-RF	0,19	0,048	2,3
SVM-SVM	0,2037	0,478	1,6
lasso-SVM	0,17	**0,037**	4,33
lasso	0,1778	0,769	0,43
RF	0,17	1	**0,16**
SVM	0,1778	0,6	0,455

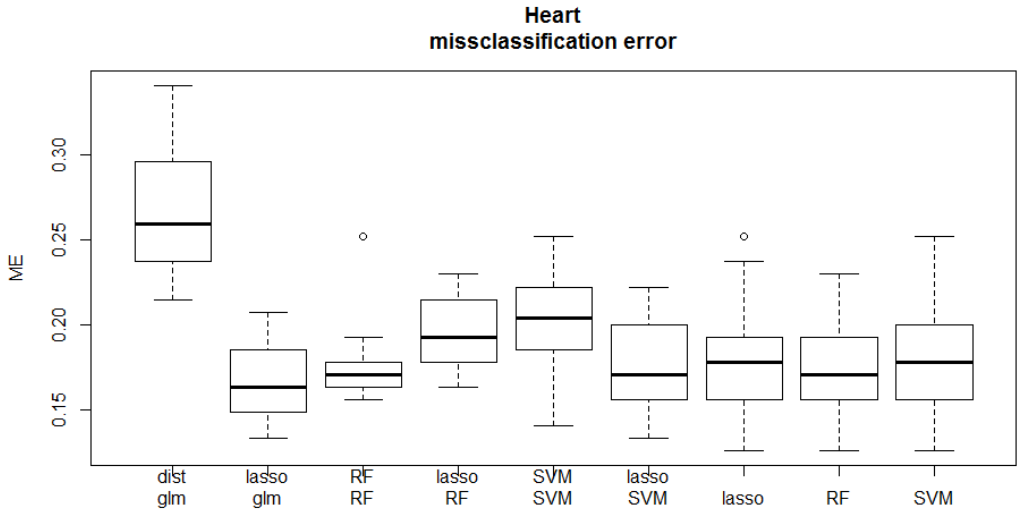

Figura 13: ME per Heart

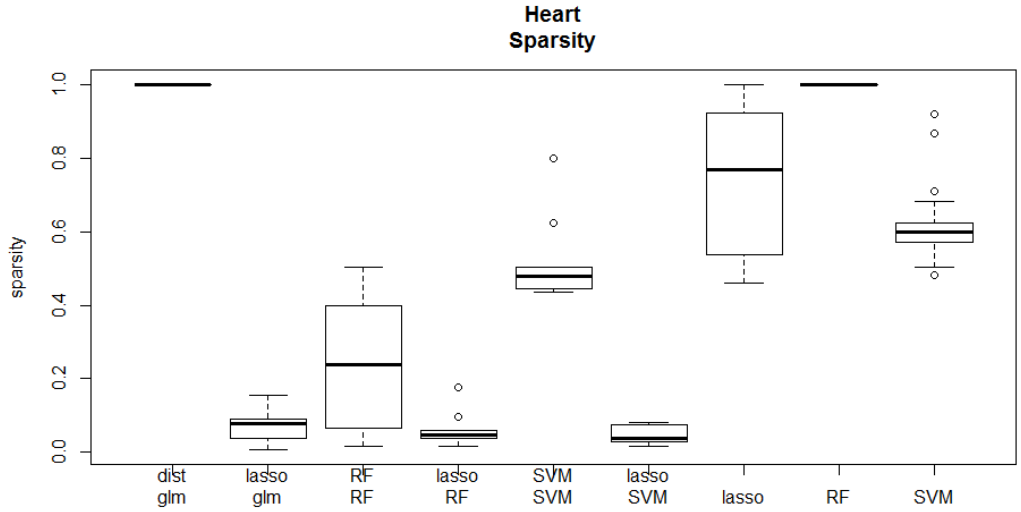

Figura 14: Sparsity per Heart

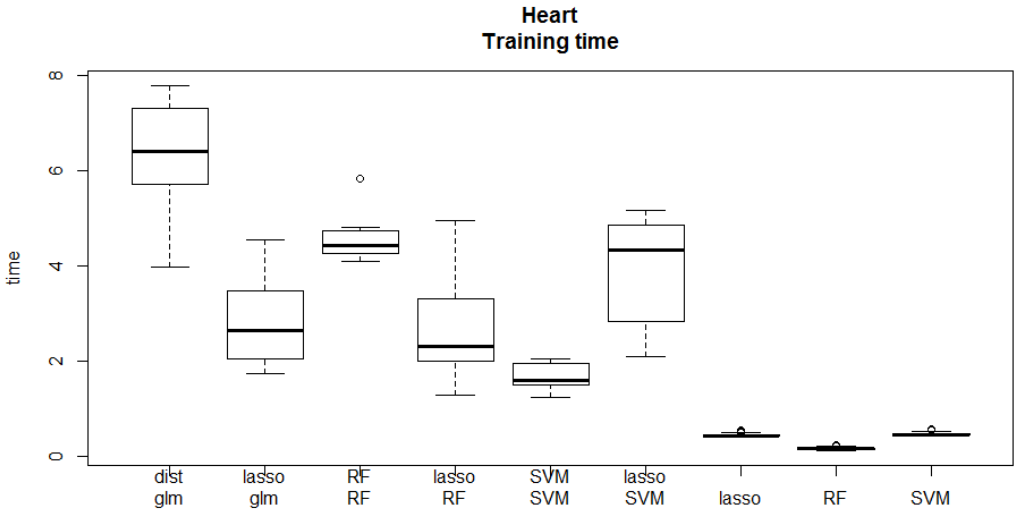

Figura 15: Temps d'entrenament per Heart

Audiology:

Mètode	ME	Sparsity	temps (s)
dist-glm	0.159	1	2.515
lasso-glm	**0.132**	0.278	3.995
RF-RF	0.154	0.252	5.43
lasso-RF	0.17	0.267	3.94
SVM-SVM	0.148	**0.18**	2.295
lasso-SVM	**0.132**	0.252	5.455
lasso	0.154	0.855	4.28
RF	0.154	1	**0.3**
SVM	0.154	0.674	1.285

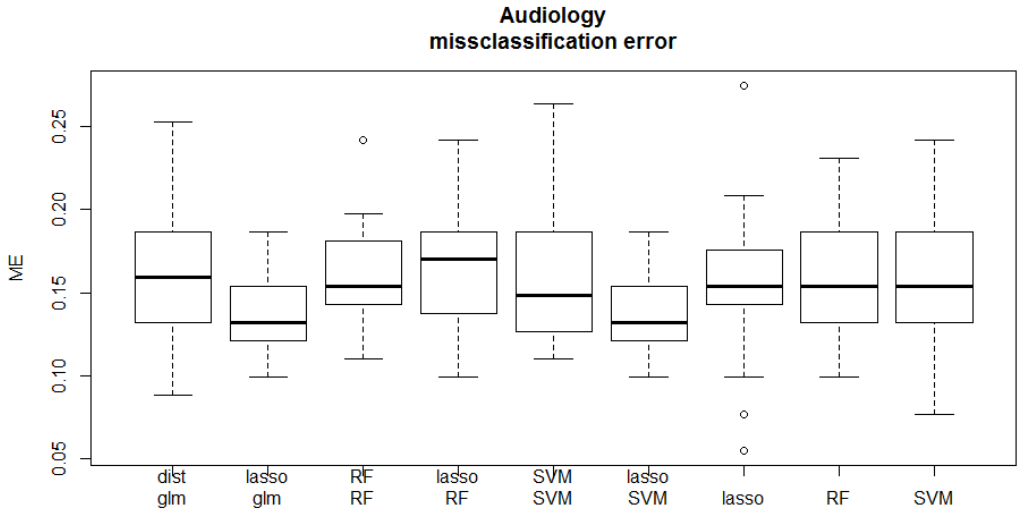

Figura 16: ME per Audiology

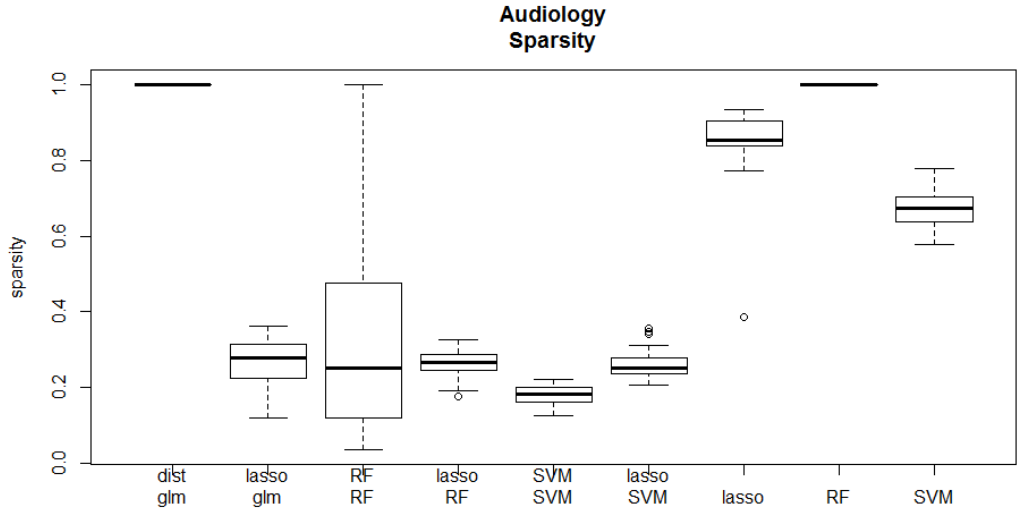

Figura 17: Sparsity per Audiology

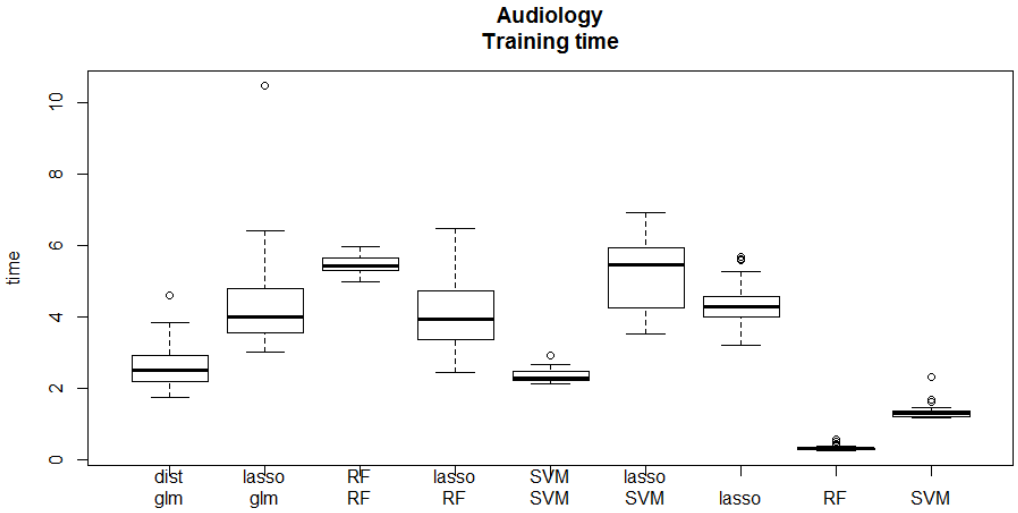

Figura 18: Temps d'entrenament per Audiology

Contraceptive:

Mètode	ME	Sparsity	temps (s)
dist-glm	0,4859496	1	9,08
lasso-glm	**0,4670543**	0,08843537	48,23
RF-RF	0,5406977	**0,0600907**	84,41
lasso-RF	0,5343992	0,08730159	50,975
SVM-SVM	0,5373062	0,4557823	16,09
lasso-SVM	0,5067829	0,08956916	60,7
lasso	0,4937016	1	2,3
RF	0,4680233	1	**0,5**
SVM	0,4781977	0,9047619	3,555

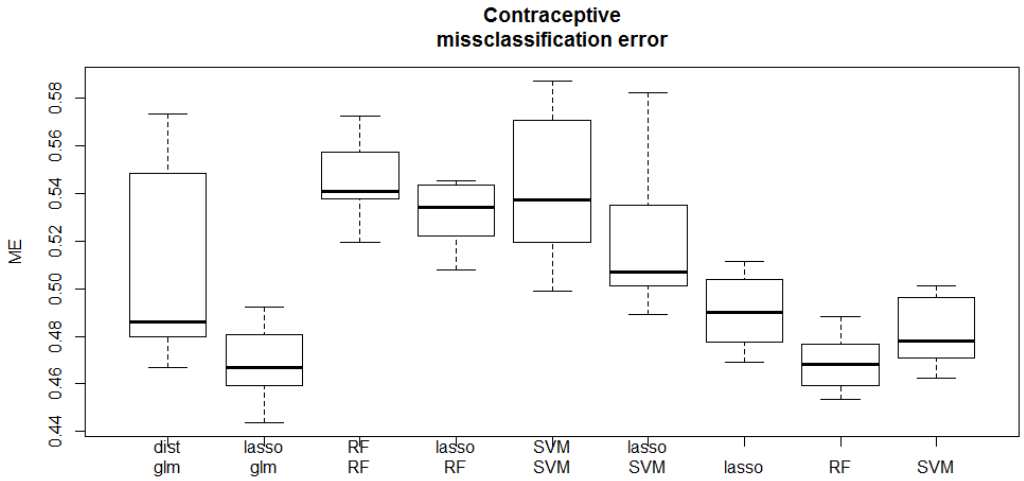

Figura 19: ME per Contraceptive

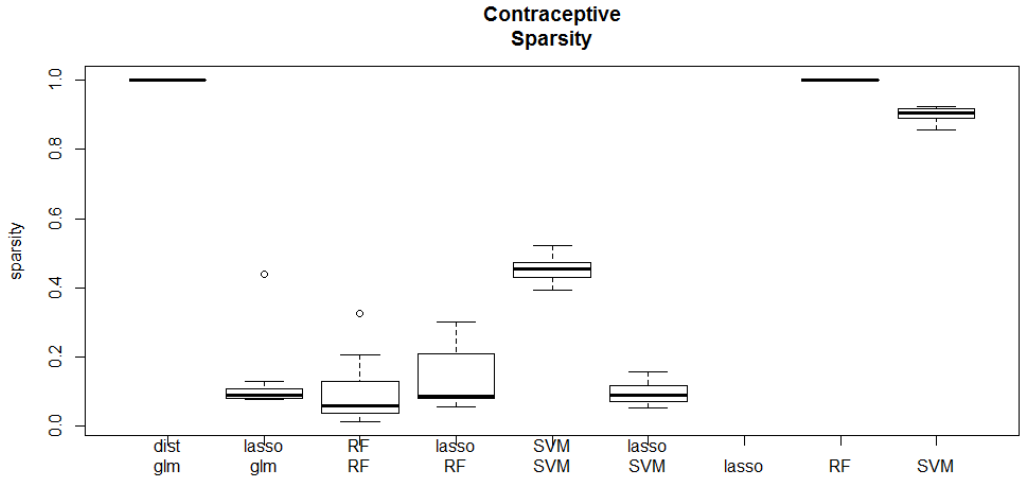

Figura 20: Sparsity per Contraceptive

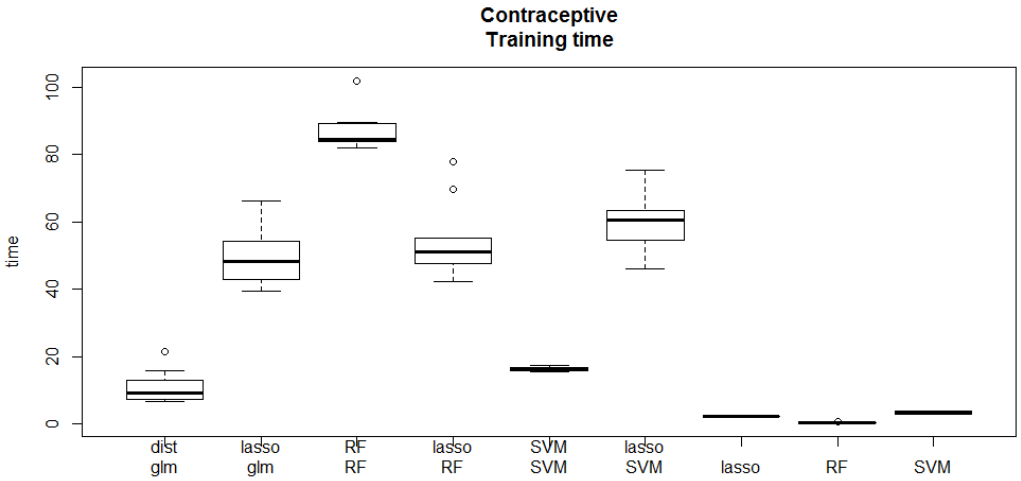

Figura 21: Temps d'entrenament per Contraceptive

Servo:

Mètode	FVU	Sparsity	temps (s)
dist-glm	0,318	1	**0,05**
lasso-glm	0,316	0,195	0,32
RF-RF	0,246	0,25	7,485
lasso-RF	0,243	0,205	0,64
SVM-SVM	0,21	0,705	1,3
lasso-SVM	**0,189**	**0,165**	0,665
lasso	0,66	1	0,42
RF	0,3898	1	0,11
SVM	0,5487	0,68	0,85

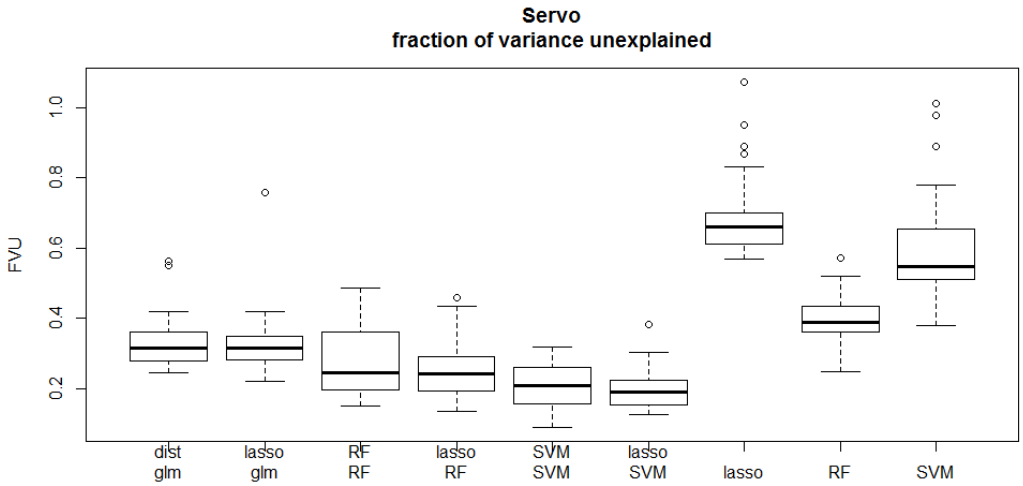

Figura 22: ME per Servo

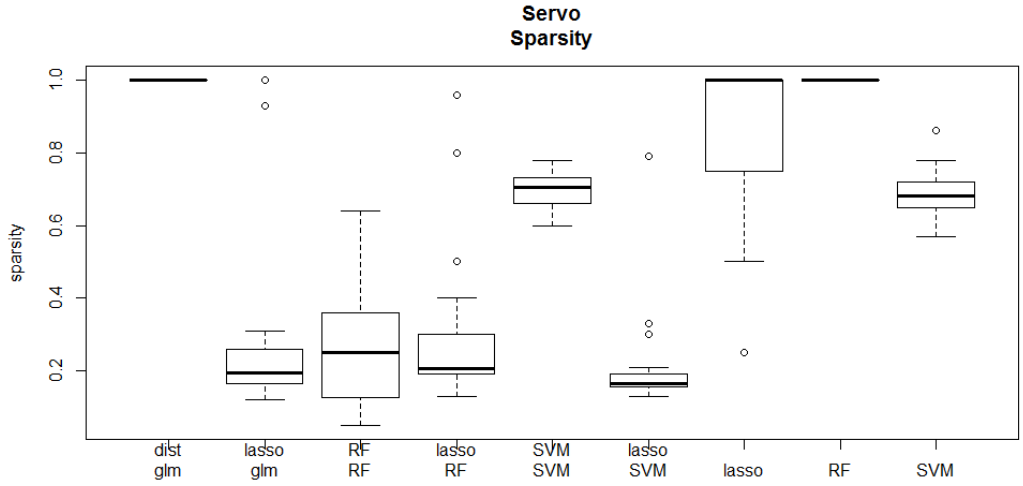

Figura 23: Sparsity per Servo

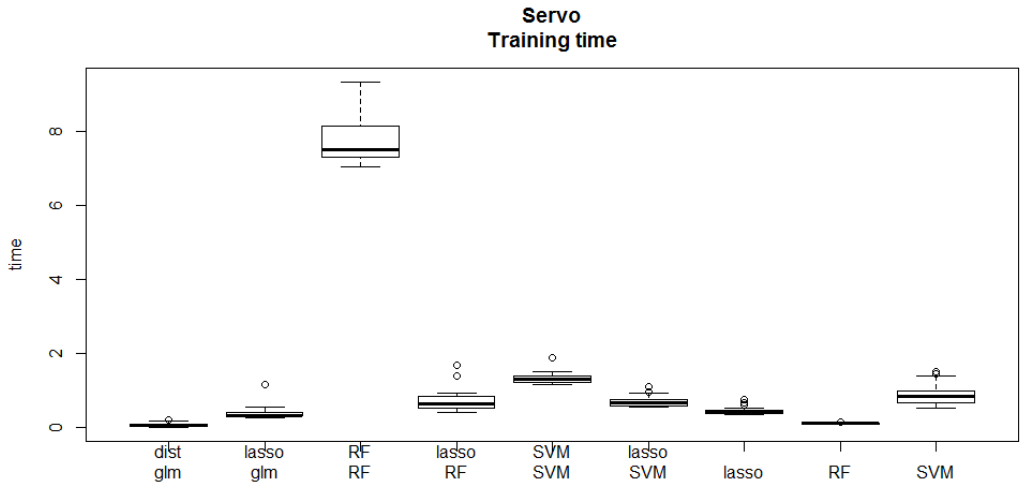

Figura 24: Temps d'entrenament per Servo

Automobile:

Mètode	FVU	Sparsity	temps (s)
dist-glm	0,1058	1	**0,22**
lasso-glm	**0,093**	0,9458	0,81
RF-RF	0,2225	**0,283**	12,3
lasso-RF	0,197	0,942	2,32
SVM-SVM	0,116	0,642	1,455
lasso-SVM	0,1316	0,95	1,53
lasso	0,148	1	0,44
RF	0,0954	1	0,39
SVM	0,144	0,683	0,6

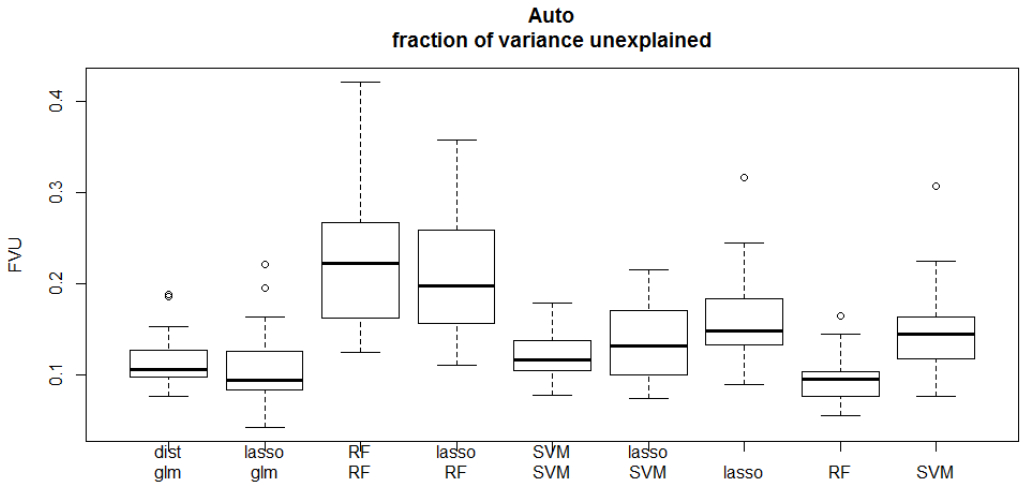

Figura 25: ME per Automobile

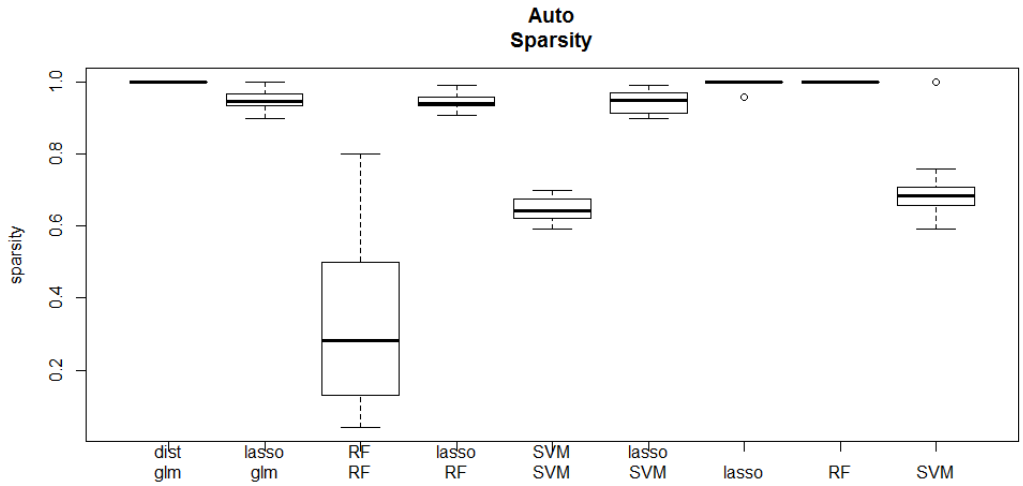

Figura 26: Sparsity per Automobile

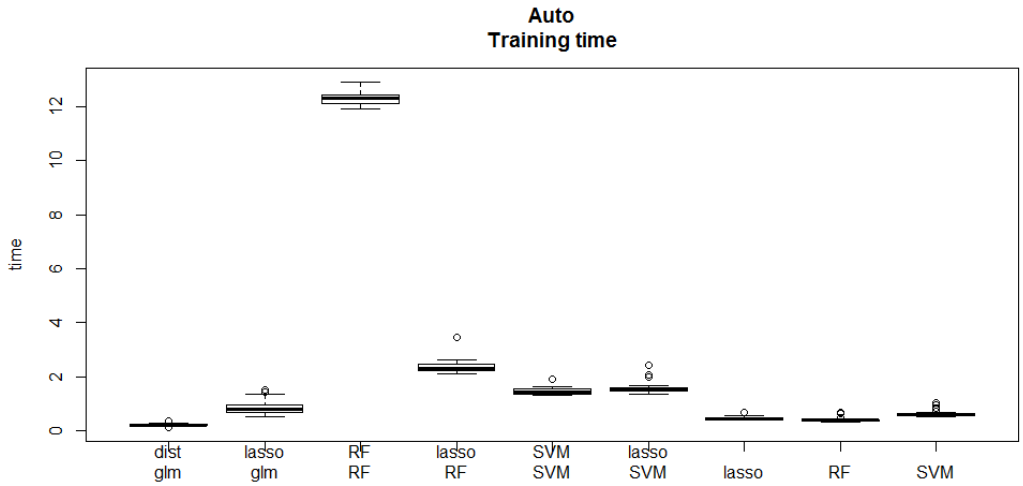

Figura 27: Temps d'entrenament per Automobile

Bank Marketing:

Mètode	ME	ME(yes)	Sparsity	temps (s)
dist-glm	0,1915	0,37	1	309
lasso-glm	**0,1704**	**0,321**	0,059	518
RF-RF	0,272	0,3594	0,322	6296
lasso-RF	0,257	0.333	**0,046**	252
SVM-SVM	0,239	0,36	0,63	1940
lasso-SVM	0,21	0,3364	0,15	2125
lasso	0,26275	0,3236	0,684	**10**
RF	0,1822	0,374	1	12
SVM	0,182	0,3738	0,1446	61

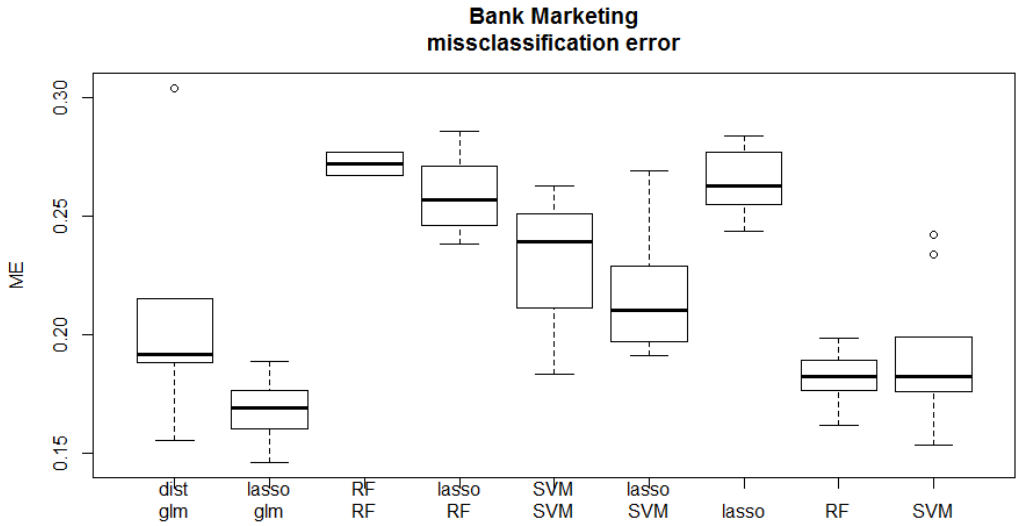

Figura 28: ME per Bank Marketing

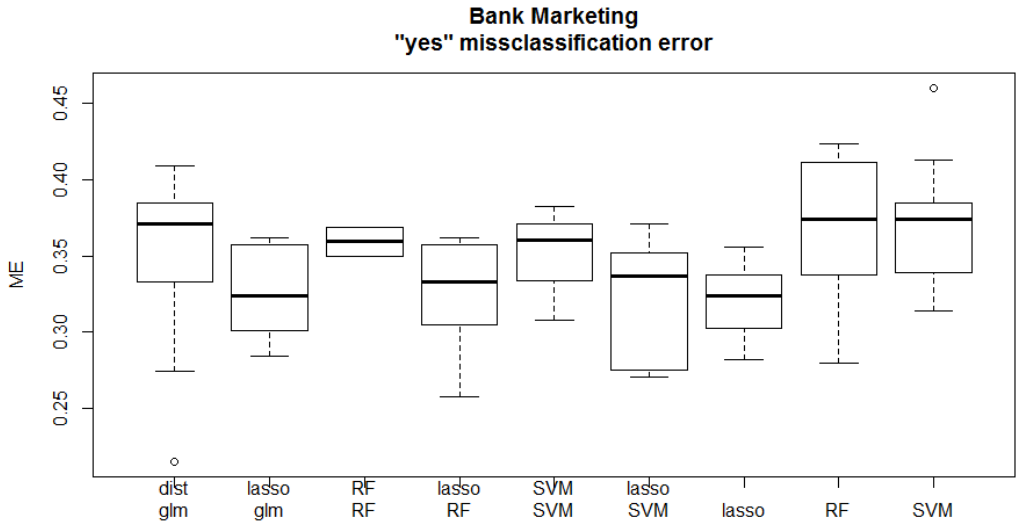

Figura 29: ME per la classe “yes” de Bank Marketing

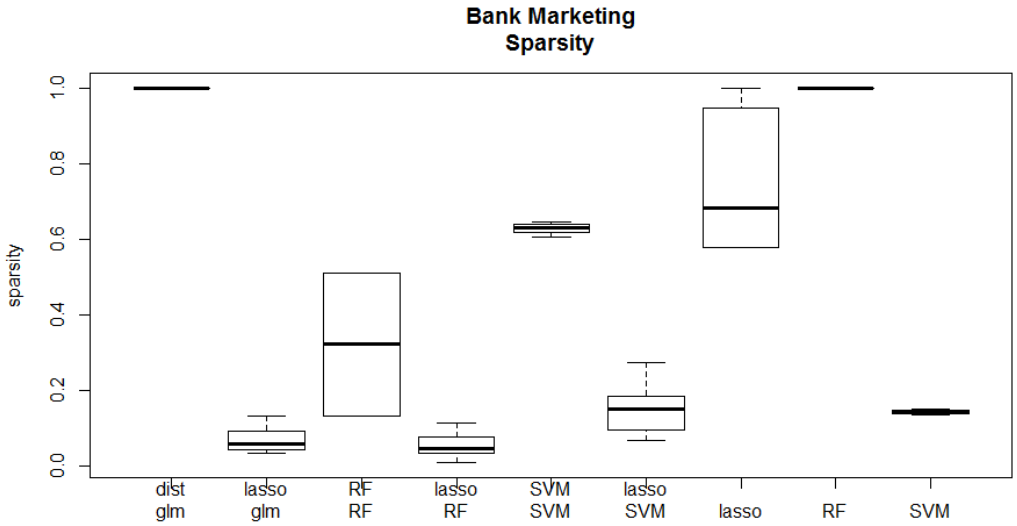

Figura 30: Sparsity per Bank Marketing

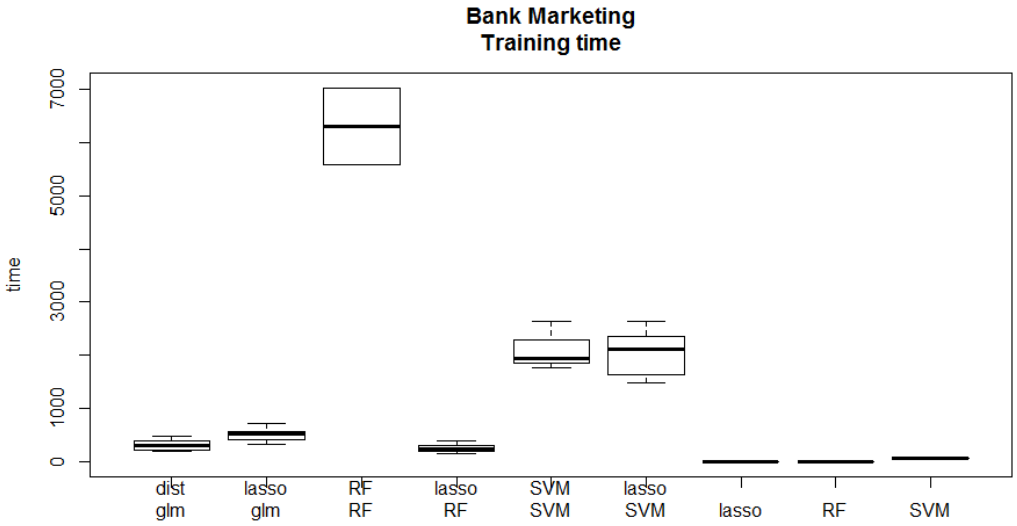

Figura 31: Temps d'entrenament per Bank Marketing

7 Discussió dels resultats

En total s'han avaluat nou tipus de modelatge sobre nou problemes reals. Per cada cas s'ha mesurat el error, sparsity i temps d'entrenament. A continuació es criticarà des de diferents punts de vista els resultats obtinguts i es farà una observació final sobre el rendiment general.

SNN vs mètodes clàssics (ús de distàncies contra no ús)

En quant a error, es pot observar que els mètodes clàssics casi sempre es troben entre els millors sistemes. Però a part de Credit approval, Hepatitis i Hores colic, en els altres problemes hi ha hagut una SNN que ha aconseguit millors resultats. Random Forest s'ha posat com millor model clàssic però en situacions s'ha vist superat i fins i tot en els casos on ha comés menys error no ha estat per gaire diferència. En general, l'ús de SNN ha demostrat certa variabilitat entre els seus diferents mètodes d'entrenament però si es fan proves previes i s'escull el millor model, una SNN té gran potencial de cometre menys error que els altres mètodes clàssics. Sobretot, la construcció lasso-glm sempre ha estat al capdavant, si no ha estat el millor, estava en segon o tercer lloc.

Tenint en compte la sparsity dels models, hi ha una clara decantació cap a la SNN. Ha demostrat que sempre aconsegueix reduir considerablement bé les dades d'entrada. Això és degut a que Random Forest sempre agafa totes les dades i les intenta aprofitar, lasso en solitari intenta reduir les entrades però sense lús de distàncies es queda a mig camí i SVM és un cas semblant. Es pot afirmar doncs que la transformació a distància de les dades d'entrada aporta molt a la sparsity del model final. Cal denotar un altre cop que lasso-glm ha estat gairebé sempre el guanyador en quant a sparsity.

De la manera que SNN és el líder de la sparsity, els models clàssics no tenen competència en quant a velocitat. Els temps d'entrenament de lasso, Random Forest i SVM són ínfims. Això es deu a que per una SNN s'han de fer tres etapes entre les quals la selecció de líders dona un cop fort al temps d'entrenament. Si no es disposa de temps per entrenar una SNN, la decisió és agafar un model classic doncs s'entrenarà relativament ràpid.

Selecció de prototips vs no selecció (dist-glm vs lasso-glm)
S'ha ilustrat en els gràfics un model addicional com és dist-glm per demostrar una cosa que ha quedat ben clara: la selecció de prototips és un pas extremadament important. No hi ha situació on dist-glm millori l'error de lasso-glm i normalment per un bon marge. La sparsity obviament no hi ha punt de comparació doncs s'a-

consegueix mitjançant la selecció de prototips. En quant a temps, aquí queda clar el preu de la selecció de prototips, dist-glm al no fer-la obté millors temps. Tot i així, per problemes de convergència, hi ha vegades que seleccionar prototips ajuda a un modelat més fluit i lasso-glm tarda menys en entrenar-se. Tenint en compte tots aquests paràmetres, no hi ha gaire discussió que en general fer selecció de líders contribueix a més del que perjudica.

Mètodes lineals vs no lineals (glm contra Random Forest i SVM)
En el camp de l'error, sorprèn veure com els models lineals rendeixen millor dintre d'una SNN en conjunt amb distàncies i prototips que fora com a model individual. Si es miren en les implementacions de les SNNs, l'opció lasso-glm (completament lineal) aconsegueix superar gairebé sempre les altres. Contrariament, si comparem lasso a soles amb models no lineals, sempre es troba en la pitjor posició. D'això podem concloure que fora d'una SNN és millor fer servir models no lineals si es vol reduir el error, pero dins, la millor opció pot ser gairebé segur la versió lineal.

La sparsity ha resultat ser el punt fort de lasso, un mètode lineal. Allà on es troba lasso, la sparsity es veu increíblement millorada. Sobretot dintre d'una SNN doncs com a mètode en solitari fa un bon treball a vegades, però no del tot. Es pot extreure d'aquí que lasso rendeix molt bé en un entorn on s'han transformat les dades a una matriu de distàncies. En general, un mètode lineal millora molt la sparsity.

Finalment, en l'apartat de temps d'execució, es troben resultats molt variats, però hi ha un model que sempre sobresurt i és RandomForest-RandomForest. Aquesta construcció ha demostrat que tarda molt en arribar a un estat de convergència. En les altres situacions es pot discutir que entre lineal o no lineal no és un factor determinant per al temps d'entrenament.

8 Conclusions i treball futur

Conclusió final
S'han provat prous problemes com per afirmar que lasso-glm té el millor rendiment en general i és el que menys error comet. Certament en Horse colic, Hepatitis i Credit approval no és el millor, però igualment dona bons resultats. Si mirem els problemes de Contraceptive i Bank Marketing, els considerats més difícils, lasso-glm no només és el que millor explica el problema sinó que de manera més estable. Si mirem els gràfics, és el que té menys variança de tots, cosa que significa que els seus resultats són fiables. En la resta de problemes, igualment demostra una gran eficàcia. En general, lasso-glm és el millor model respecte a l'error.

Si mirem la situació des del punt de vista de sparsity, s'aprecia una decantació a lasso-glm. En tots els problemes, excepte Automobile, aconsegueix una sintetització de l'informació notablement bona. En general es mou pels valors 5% - 15% cosa que significa que és un tipus de modelat molt resumit i fàcil d'interpretar.

El temps d'entrenament però no és exactament el fort de lasso-glm. No es troba entre els pitjors, però tampoc dels millors i mètodes clàssics sempre tenen una velocitat millor. Però tenint en compte que un model només s'ha d'entrenar un cop i que predir noves dades és instantani, el temps d'entrenament no és crític.

Recopilant totes les dades que s'han adquirit en la duració d'aquest treball, la decisió del millor model seria lasso-glm. Té un respectable rendiment respecte a l'error, la sparsity és de les millors i amb un temps d'entrenament acceptable.

Treball futur
Les possibilitats de futur de les SNNs són molt àmplies en gairebé tots els sentits. En aquest treball s'ha decantat l'esforç a la mesura de la distància de Gower, doncs solucionava totes les necessitats, però existeixen moltes altres que podrien donar millors resultats. Per exemple, utilitzar kernels com a mesura de similitud.

Igualment es poden trobar en el mercat nombrosos mètodes de selecció de variables i/o observacions que podrien derivar en selectors de prototips. Per exemple, es podrien provar algorismes genètics o de força bruta. Aquests s'han deixat per futur treball doncs comporten moltes hores de treball.

El modelat és potser la part més oberta de les SNNs doncs qualsevol model estadístic pot funcionar un cop aplicada una funció de similitud o distància a les entrades. La primera alternativa que es pot presentar és una xarxa neuronal artifical dintre de la xarxa neuronal basada en similitud.

9 Planificació del projecte

En aquesta secció es pot trobar els resultats de la producció del treball. Es presentaran les dades finals sobre temps i recursos utilitzats per arribar al punt final.

9.1 Resultats de la planificació temporal

El projecte s'ha dividit en vuit tasques per enfocar-lo de manera organitzada. Aquí es troba la llista de les tasques amb la seva descripció i una taula amb els valors exactes d'hores invertides.

Planificació del projecte (Fita inicial) : En aquesta secció es va organitzar com s'estructuraria el treball. L'abast, els objectius, les alternatives avaluades i el pressupost esperat es van definir en aquesta fase. Encapsula tota la matèria de gestió de projectes.

Recerca de codi (Distàncies) : La tasca portada a terme és la recolecció d'articles i llibreries que tractin de manera parcial o total el càlcul de distàncies. S'han adquirit tots els coneixements necessàris per a trobar la millor i més eficient manera d'implementar la distància de Gower.

Recerca de codi (Prototips) : Aquí s'han buscat i avaluat nombroses opcions que puguèssin fer servir per seleccionar prototips. S'ha arribat a la conclusió d'utilitzar lasso, Random Forest i SVMs.

Programació de distàncies : Un cop adquirit tot el coneixement necessàri per dur a terme el treball, primer s'ha implementat la nostra propia versió de la distància de Gower amb les seves respectives extensions.

Programació de selecció de prototips : S'han programat i refinat totes les opcions de selecció de prototips escollides amb anterioritat.

Programació dels algorismes d'entrenament de les SNNs : Ja definides totes les parts que es farien servir, les diferents versions de SNNs han estat programades de manera definitiva. Les cinc construccions de SNN i els tres mètodes clàssics per disposar de comparació han estat disposats per fer les proves finals.

Comprovació de resultats amb problemes : En aquesta tasca s'han coleccionat els diferents problemes i recollit les diferents mesures d'executar els models amb cada un d'aquests.

Tasca final (Testejos, memòria, etc.) : Finalment, s'ha ajuntat tot el coneixement adquirit en la duració del treball per recopilar-lo en l'escriptura d'aquesta memòria.

Tasca	Duració aproximada (h)
Planificació del projecte (Fita inicial)	70
Recerca de codi (Distàncies)	30
Recerca de codi (Prototips)	50
Programació de les distàncies	70
Programació de selecció de prototips	100
Programació dels algorismes d'entrenament de les SNNs	90
Comprovació de resultats amb problemes	30
Tasca final (Testejos, memòria, etc.)	60
Total	500

Figura 32: Taula amb les duracions en hores de totes les tasques

9.2 Gestió econòmica

En aquesta secció es troben desglossades totes les despeses que ha comportat l'execució del treball. Els diferents apartats tractats són els sous dels recursos humans, el preu de hardware i software, els costos indirectes i finalment una taula amb la suma total.

Rol	Hores	Preu/hora	Preu total
Cap de projecte	80 h	45 €/h	3.600 €
Dissenyador	80 h	35 €/h	2.800 €
Enginyer de software	260 h	35 €/h	9.100 €
Beta tester	90 h	30 €/h	2.700 €
TOTAL	500		18.200 €

Figura 33: Costos de recursos humans

Producte	Preu	Vida útil	Amortització
PC	500 €	5 anys	62.5 €
Total	500 €		62.5 €

Figura 34: Costos de hardware

Producte	Preu	Vida útil	Amortització
Windows 10 Professional	154,14 €	3 anys	25.69 €
R 3.2.3	0 €	3 anys	0 €
RStudio 0.98.113	0 €	3 anys	0 €
LATEX	0 €	3 anys	0 €
Total	154,14 €		25.69 €

Figura 35: Costos de software

Producte	Preu	Unitats	Cost
Electricitat	0.12 €/kWh	600 kWh	72 €
Paper	7,95 €/paquet	1 paquet	7,95 €
Total			79,95 €

Figura 36: Despeses indirectes

Concepte	Cost aproximat
Recursos humans	18.200 €
Hardware	62.5 €
Software	25.69 €
Despeses indirectes	79,95 €
TOTAL	18.368,14 €

Figura 37: Resum dels costos totals

9.3 Sostenibilitat

Aquest és un projecte basat en investigació i software per la qual cosa no té repercussions mediambientals o socials directes. Els únics problemes que pot pressentar és el consum elèctric que suposa. El camp econòmic només es veu afectat per el cost del sou dels recursos humans que és la majoría del preu. L'avaluació de sostenibilitat es pot representar amb la següent taula:

	PPP	Vida útil	Riscos
Ambiental	8	16	-4
Econòmic	9	15	-8
Social	7	13	-10
Rang de sostenibilitat	24	44	-22
	46		

Figura 38: Valoració de la sostenibilitat del projecte

Referències

[1] Lluís A. Belanche and Jerónimo Hernández. *Similarity networks for heterogeneous data.. Technical University of Catalonia, Barcelona, Spain.*. URL http://www.i6doc.com/en/livre/?GCOI=28001100967420.

[2] Andrew Gelman, Jennifer Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models.. Columbia University, New York.*. December 2006.

[3] Yu L. Pavlov. *Random Forests. Walter de Gruyter.*. March 2000.

[4] John Shawe-Taylor and Nello Cristianini. *Support Vector Machines.. Cambridge University Press.*. January 2000.

[5] Christopher M. Bishop. *Neural Networks for Pattern Recognition.*. November 1995.

[6] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).*. October 2007.

[7] Gower, J. C. 1971. *A general coefficient of similarity and some of its properties.*. Biometrics 27: 857-874.

[8] Sandrine Pavoine, Jeanne Vallet, Anne-Béatrice Dufour, Sophie Gachet and Hervé Daniel 1971. *On the challenge of treating various types of variables: application for improving the measurement of functional diversity.*. Oikos 118: 391-402.

[9] Trevor Hastie, Robert Tibshirani, Jerome Friedman. *The Elements of Statistical Learning: Data mining, Inference, and Prediction.* Second Edition, February 2009.

[10] David Aha, Arthur Asuncion and David Newman. *UCI Machine Learning Repository.. University of Massachusetts Amherst.*. URL https://archive.ics.uci.edu/ml/datasets.html.