Són les Xarxes Neuronals més eficients que els arbres de decisió?

Anàlisis sobre un problema d'Astronomia

Albert Ribé Costa
Màster en enginyeria informàtica
Intel·ligència Artificial

Samir Kanaan Izquierdo
Carles Ventura Royo

Desembre del 2016


Aquesta obra està subjecta a una llicència de [Reconeixement-NoComercial-SenseObraDerivada 3.0 Espanya de Creative Commons]

FITXA DEL TREBALL FINAL

Títol del treball:	*Algoritme Deep Learning per a la classificació de cossos celestes*
Nom de l'autor:	*Albert Ribé Costa*
Nom del consultor/a:	*Samir Kanaan Izquierdo*
Nom del PRA:	*Carles Ventura Royo*
Data de lliurament (mm/aaaa):	*12/2016*
Titulació o programa:	*Màster en enginyeria informàtica*
Àrea del Treball Final:	*Intel·ligència artificial*
Idioma del treball:	*Català*
Paraules clau	*Deep Learning, Astronomia, Python*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

El present treball tracta l'anàlisi del problema de la classificació d'objectes celestes des del punt de vista de l'anàlisi de dades, per fer-ho s'han realitzat dos algoritmes classificadors per operar amb dades dels datasets de Sloan Digital Sky Survey i determinar-ne la validesa de cada un.

El software desenvolupat es realitza mitjançant tècniques de Machine Learning, en concret mitjançant un Arbre de Decisió i una Feed Forward Neural Network. Està desenvolupat mitjançant el llenguatge de programació Python i amb les llibreries Theano i Lasagne.

Primer s'analitza l'entorn i dades del problema, en astronomia és molt comú el treballar amb grans quantitats de dades així que ha estat important realitzar un bon anàlisi de les mateixes, per a continuació escollir els orígens de dades òptims.

Seguidament s'ha procedit a desenvolupar els algoritmes que posteriorment s'han comparat. Per determinar quin i perquè és el millor per aquest problema s'ha realitzat execucions amb diferents premisses (configuració, nombre de registres,...) i analitzat els resultats obtinguts.

És important destacar que s'ha inclòs una secció de conclusions a mode de síntesi on a més de revisar els resultats i treball realitzat, així com els problemes més important es dedica també una secció a treballs futurs que es poden dur a terme a posteriori.

Abstract (in English, 250 words or less):

This project contains an analysis of the problem of classification of celestial objects from the point of view of the data analysis, have been made two classification algorithms for the datasets from Sloan Digital Sky Survey and determined the validity of each one.

The software is developed by using Machine Learning techniques, specifically using a Decision Tree and Feed Forward Neural Network. The development is done by using the Python programming language and Theano and Lasagne libraries.

First analyzes the data and environment of the problems, in astronomy is very common to work with huge amounts of data it has been important to make a good analysis of them and then choose the optimal data sources.

The next step was start with the development of the algorithms to compare. To determine which is the best and why, this problem has been carried out executions with different premises (configuration, number of records, ...) and has been analyzed the results.

It is important to highlight the included conclusions section, to synthesize and review the results and work done, as well as the most important problems faced and a section dedicated to a possible future works.

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

En els ultima anys la revolució de les dades massives s'ha posat en primera plana de molts mitjans de comunicació, les noves tecnologies permeten generar més dades ara que en la resta de la historia de la humanitat.

En aquest entorn composat de milers de milions de bytes d'informació en cobra especial importància el tractament per poder-los donar un ús.

Hi ha molts àmbits del coneixement humà que es beneficien del tractament de les moltes dades que generen, en aquest projecte ens centrarem en l'astronomia.

Hi ha molts estudis en l'actualitat que centren els seus esforços en l'anàlisi de cossos celestes, per exemple la busca de planetes extra solars o l'anàlisi de la historia de l'univers mitjançant el moviment de les galàxies.

Amb més de 1.000 milions d'objectes presents a la Via Làctia, el catàleg fotomètric de SDSS[1] sembla un candidat excel·lent per a la millora dels seus processos de classificació d'aquests objectes i obtenir-ne dades enriquides a partir de les seves propietats.

Ara per ara SDSS posa a disposició de tothom que ho necessiti de les dades corresponents a aquests objectes, i si bé hi ha diferents eines per automatitzar l'anàlisi de dades i fins i tot iniciatives populars per classificar els elements de SDSS que ajuden a fer-ho de forma col·laborativa, no s'aprecien gaires projectes d'anàlisi intel·ligent de les dades que permeti obtenir resultats enriquits mitjançant, per exemple, tècniques de Machine Learning.

Aquesta és la raó principal que a dut a pensar que crear un algoritme d'anàlisi per determinar de forma automàtica i certes característiques dels objectes celestes a partir d'altres propietats pot ser una contribució interessant en la recerca astronòmica.

1.2 Objectius del Treball

Vist el context exposat, es determina que l'objectiu del present treball és el de crear un sistema d'automatització i aprenentatge automàtic per poder classificar els objectes celestes corresponents als data sets de SDSS i permetre la classificació de nous elements que s'introduïssin a la col·lecció inicial, així doncs podria ampliar-se la classificació a objectes tant del projecte inicial i de la via làctia com fins i tot elements d'altres galàxies, si s'escau.

En detall els objectius que es persegueixen es poden classificar en tres valors diferents, els que corresponen a l'objectiu del projecte a desenvolupar, els que corresponen a la demostració d'habilitats adquirides durant el màster i en concret a les assignatures d'IA i les habilitats que es volen desenvolupar en el transcurs del projecte:

- Determinar quines dades s'utilitzen del data set, realitzar un anàlisi de les opcions, exposar-ne els resultats i escollir-ne un.
- Desenvolupar un algoritme mitjançant mètodes propis de l'aprenentatge automàtic / profund i en concret l'ús d'una Feed Forward Neural Network (FFNN), aquest algoritme ha de servir per classificar els objectes escollits de SDSS mitjançant les seves propietats.
- Desenvolupar un algoritme d'arbre de decisió que serà la base per comparar la xarxa neuronal i mostrar-ne la validesa.
- Comparar els resultats d'ambdós algoritmes i determinar els avantatges, si n'hi ha, del algoritme DL en termes de precisió i temps d'entrenament per a la classificació d'objectes celestes del dataset escollit.
- Al ser un treball fonamentalment per a consolidar l'aprenentatge obtingut al llarg del màster en enginyeria informàtica, es considera també com a objectiu aplicar les diferents tècniques apresses al llarg del mateix, com per exemple la planificació o distribució de tasques d'un projecte que es va aprendre a "Gestió avançada de projectes".
- Finalment es persegueix aprofundir en les tècniques i mètodes de l'aprenentatge automàtic per tal d'aplicar-lo a projectes futurs.

Per dur a terme aquests objectius es realitza una tasca de recerca on primer es determinarà exactament les dades a utilitzar, així doncs aquesta fase tot i no formar part del desenvolupament és vital i se li dedica l'esforç corresponent.

Les tecnologies i mètodes utilitzats són força innovadors de manera que una part important del desenvolupament del projecte es centrarà en la recerca i formació en aquestes tècniques i tecnologies innovadores.

1.3 Enfocament i mètode seguit

Al iniciar la recerca d'informació per a realitzar el projecte es va veure que hi ha certes iniciatives d'automatització i classificació en el mon de l'astronomia, tot i així no es va trobar cap projecte que es pugues continuar o millorar considerant l'abast possible del projecte, tampoc sembla que s'escaigui per tal d'acomplir tots els objectius marcats.

És per això que projecte es planteja com un producte original (l'algoritme de classificació) però amb l'ajuda d'altres projectes que s'han estudiat, un exemple clar n'és el de Pavel Hála de la MASARYK University de Brno (2014) [2], que té un objectiu semblant. De totes maneres es recalca que la solució és original, ja que el caire del present projecte és més enfocat a la part de l'algoritme y Deep Learning a diferència del projecte citat, que correspon a un treball de màster d'astrofísica on es fa més èmfasi als aspectes científics del projecte relacionats amb l'astrofísica. Es destaca també el treball de Lawrence O. Hall, Xiaomei Liu, Kevin W. Bowyer, Robert Baneld de la Univ. of South Florida [3] que compara els dos mètodes aplicats a la Bio-Informàtica.

Sobre el mètode de desenvolupament del projecte s'ha seguit fonamentalment un model salt d'aigua, la raó principal és que l'equip ha estat format per una persona cosa que fa que una gestió Agile no tingui massa sentit, a més la data d'entrega, la capacitat estaven fixats i només s'ha pogut fer alguns canvis en l'abast.

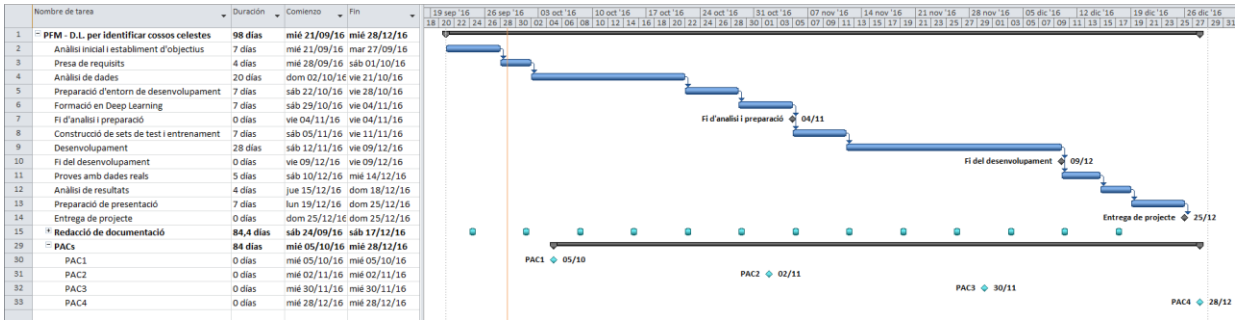
1.4 Planificació del Treball

1.4.1 Planificació

La planificació realitzada a estat a alt nivell, indicant les tasques principals i incloent les fites més importants així com les diferents entregues a realitzar (PACs).

S'ha intentat definir una planificació força acurada donant temps suficient per l'anàlisi de dades i l'aprenentatge, ja que ambdós són conceptes força nous i complexos, tot i que tampoc s'ha d'oblidar de donar temps suficient per desenvolupar l'algoritme.

Alhora, donat que no es té massa experiència en les tecnologies i conceptes a emprar, s'ha intentat reservar temps suficient per imprevistos, corresponent, sobretot, a la dedicació de proves i anàlisi de resultats durant 9 dies, en que es podria resoldre també possibles defectes de l'algoritme creat.



	Nombre de tarea	Duración	Comienzo	Fin
1	**PFM - D.L. per identificar cossos celestes**	**98 días**	**mié 21/09/16**	**mié 28/12/16**
2	Anàlisi inicial i establiment d'objectius	7 días	mié 21/09/16	mar 27/09/16
3	Presa de requisits	4 días	mié 28/09/16	sáb 01/10/16
4	Anàlisi de dades	20 días	dom 02/10/16	vie 21/10/16
5	Preparació d'entorn de desenvolupament	7 días	sáb 22/10/16	vie 28/10/16
6	Formació en Deep Learning	7 días	sáb 29/10/16	vie 04/11/16
7	Fi d'analisi i preparació	0 días	vie 04/11/16	vie 04/11/16
8	Construcció de sets de test i entrenament	7 días	sáb 05/11/16	vie 11/11/16
9	Desenvolupament	28 días	sáb 12/11/16	vie 09/12/16
10	Fi del desenvolupament	0 días	vie 09/12/16	vie 09/12/16
11	Proves amb dades reals	5 días	sáb 10/12/16	mié 14/12/16
12	Anàlisi de resultats	4 días	jue 15/12/16	dom 18/12/16
13	Preparació de presentació	7 días	lun 19/12/16	dom 25/12/16
14	Entrega de projecte	0 días	dom 25/12/16	dom 25/12/16
15	**Redacció de documentació**	**84,4 días**	**sáb 24/09/16**	**sáb 17/12/16**
29	**PACs**	**84 días**	**mié 05/10/16**	**mié 28/12/16**
30	PAC1	0 días	mié 05/10/16	mié 05/10/16
31	PAC2	0 días	mié 02/11/16	mié 02/11/16
32	PAC3	0 días	mié 30/11/16	mié 30/11/16
33	PAC4	0 días	mié 28/12/16	mié 28/12/16

Figura 1

Sobre l'horari i les dates

Les duracions de les tasques s'han establert amb dies de 4 hores, comptant els 7 dies de la setmana, això correspon a setmanes de 28 hores, temps estimat de dedicació màxima que es podrà dedicar. Això correspon a una dedicació mitja de 3 hores de dilluns a divendres i 13 els caps de setmana.

Per altre banda s'ha establer com inici del projecte el primer dia del màster (21 de setembre) i marcat el 28 de desembre com a data límit ja que correspon al final del semestre i entrega de la ultima PAC, això vol dir que s'ha intentat acotar l'abast per poder assolir les dates.

Sobre les tasques

Com es pot veure, les dates d'entrega de les PAC concideixen parcialment amb la finalització de diferents fases:

PAC1: aproximadament al final de la presa de requisits.

PAC2: amb la fi de l'anàlisi i just abans de la construcció dels conjunts de dades a utilitzar.

PAC3: hauria de contenir una fase força avançada del desenvolupament.

A continuació es pot consultar una descripció detallada de les tasques:

- *Anàlisi inicial i establiment d'objectius*: Recerca inicial d'informació sobre el tema del treball, a partir de les primeres informacions s'estableixen els objectius.
- *Presa de requisits*: Detallar les fites del projecte i els objectius.
- *Anàlisi de dades*: Cerca exhaustiva d'informació sobre les dades a utilitzar en el projecte, donat que els data set de SDSS són variats i molt grans s'haurà de triar quin és el més adequat.
- *Preparació d'entorn de desenvolupament*: Configurar la màquina de desenvolupament amb el software necessari per a la implementació del algoritme, també s'ha de preparar per poder obtenir i consultar les dades d'SDSS.
- *Formació en Deep Learning*: Obtenir informació, tutorials i ajuda per aprendre els principals trets distintius de Deep Learning que seran necessaris per a la realització del algoritme. En aquesta tasca serà necessari disposar de l'entorn preparat per tal de crear i executar exemples i proves.
- *Construcció de sets de test i entrenament*: Descarregar i preparar les dades a utilitzar en el treball i determinades en la tasca d'anàlisi de dades..
- *Desenvolupament*: Procés de desenvolupament del algoritme de Deep Learning per classificar els objectes.
- *Proves amb dades reals*: Realitzar diferents bateries de proves i preparar-ne les dades resultants.
- *Anàlisi de resultats*: Mitjançant les dades obtingudes en les proves es realitzarà un anàlisi per veure si s'ha assolit l'objectiu de fiabilitat de les classificacions.
- *Preparació de presentació*: Preparació i realització de la presentació del treball.
- *Redacció de documentació*: Tasca periòdica en la que es redacta la present documentació i s'estén durant tot el projecte, permet realitzar la documentació de forma evolutiva i realitzar les diferents entregues de les PAC.
- *PACs*: Correspon a les entregues de les diferents PACS.

1.4.2 Recursos

El projecte es realitza en dues fases força diferenciades, la primera correspon a un anàlisi que només requerirà d'eines d'ofimàtica com Word, Excel o MsProject per a la construcció dels diagrames de Gantt o connexió a internet per consultar la bibliografia. En aquesta fase també es realitza la preparació per afrontar el desenvolupament per tant requerirà també d'eines de programació (detallades més endavant).

La segona fase correspon al desenvolupament i anàlisi de resultats, aquí s'haurà de disposar d'un entorn de programació i execució adequats, per això s'ha pensat en utilitzar el llenguatge de programació **Python** [4] versió 3 per al desenvolupament ja que és l'eina utilitzada a l'assignatura d'Intel·ligència Artificial Avançada i el grau de familiarització és superior, addicionalment s'ha utilitzat el paquet de llibreries **Anaconda** [5] per dotar al entorn de més flexibilitat per desenvolupar.

Pel que fa a les llibreries a utilitzar per el desenvolupament de l'algoritme de Deep Learning s'ha decidit treballar amb **Lasagne** [6], una llibreria per construir i entrenar xarxes neuronals en Python i que a més utilitza **Theano** [7] que li permet realitzar els entrenaments utilitzant una GPU.

En quant a l'editor s'ha decidit utilitzar **Visual Studio Code** [8] juntament amb l'extensió de Python per ser un eina gratuïta i d'ampli creixement en l'actualitat amb desenes d'extensions que simplifiquen el desenvolupament a més de la possibilitat de depurar el codi.

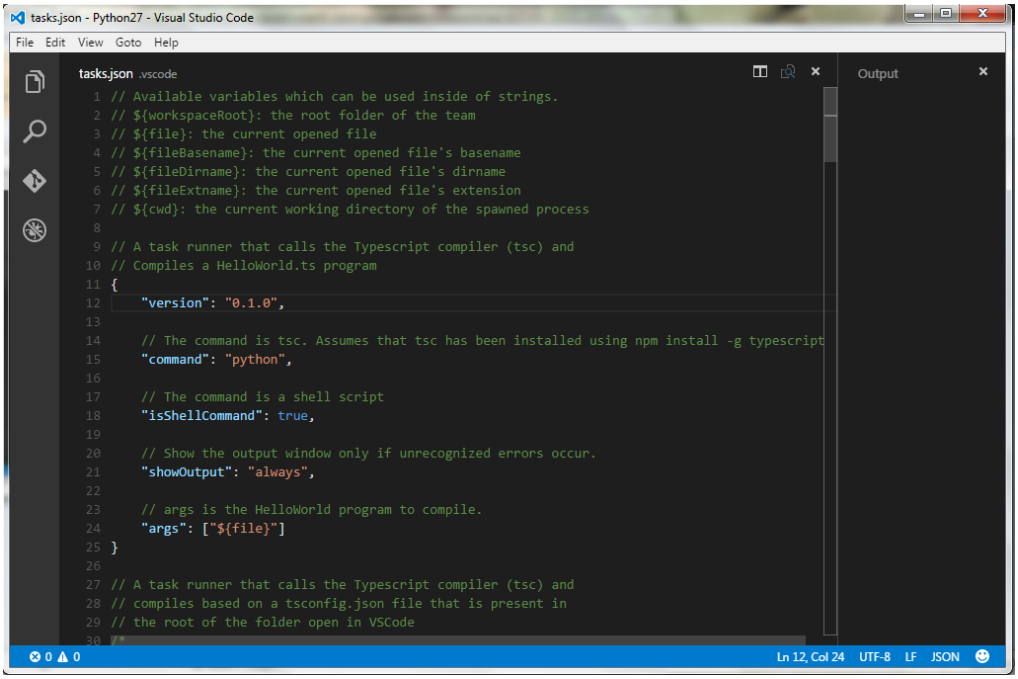


Figura 2

Addicionalment, com que les dades d'origen dels datasets d'astronomia solen ser fitxers fits que poden contenir tant imatges com dades, serà necessari tant llibreries corresponents per utilitzar-los en Python (**astropy.io.fits** [9]) com un visualitzador per poder-les consultar, en aquest cas s'ha escollit l'eina gratuïta Fv: The Interactive FITS File Editor [10] proporcionada per la NASA.

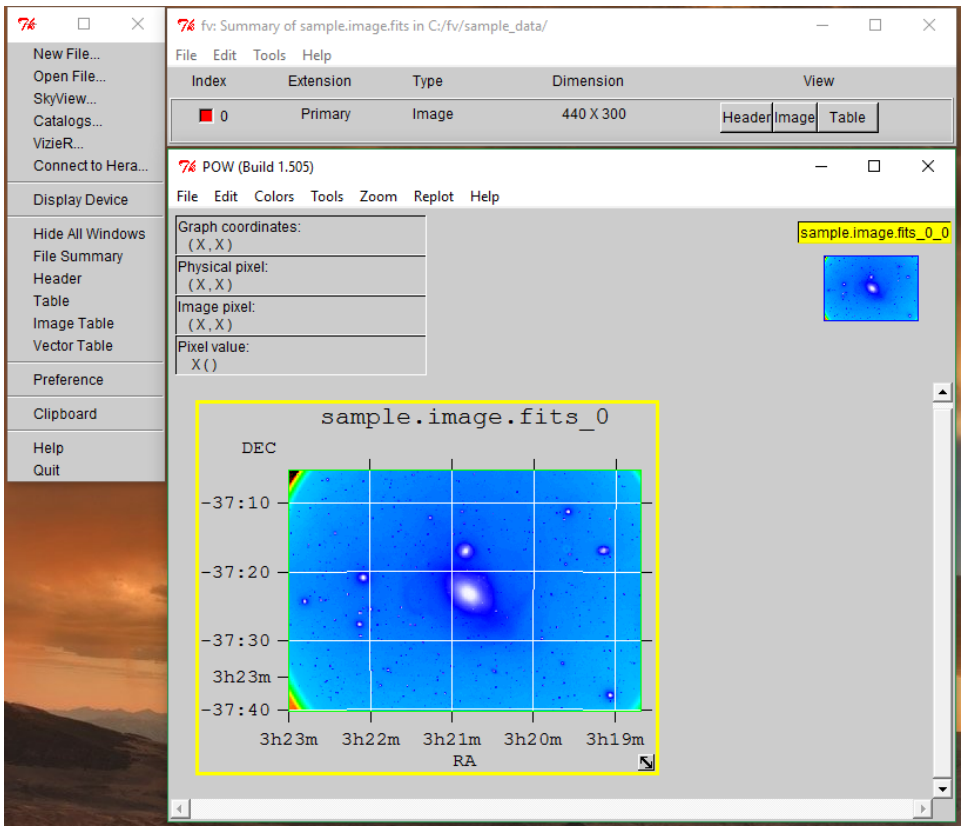


Figura 3

Finalment per consultar les dades és possible que sigui necessari l'ús d'eines pròpies de SDSS per descarregar o tractar-les, això es pot seguir a l'apartat "Descripció de les dades", subsecció "Eines de consulta i accés a les dades".

Es destaca que s'han intentat utilitzar eines de lliure distribució, com en el cas de l'editor o el llenguatge de programació, pel que fa a les eines d'ofimàtica s'acudeix a les versions que Microsoft ofereix als estudiants de la universitat.

1.5 Productes obtinguts

Documentació:

- El present document completa la memòria del projecte.
- Documents de seguiment de les PACs 2, 3 i 4 on s'especifica l'evolució del projecte al llarg de les diferents fites.
- Document markdown amb informació bàsica del software i fitxers proveïts (Lleguir.md).

Software

- DecisionTree.py: Script d'un arbre de decisió.
- FilesManager.py: Gestor de fitxers per la FNN.
- MLP.py: Script corresponent al algorisme FNN desenvolupat.

Fitxers de dades

- Sets de dades amb informació real en format fits.

1.6 Descripció dels capítols

Introducció: El present capítol, correspon a una descripció inicial de l'estat de l'art, així com la motivació del treball, també inclou una breu descripció de la resta del treball.

Descripció de les dades: Un anàlisi exhaustiu de les fonts de dades analitzades i quines es decideixen utilitzar per a desenvolupar el projecte.

Catàleg fotomètric: Descripció detallada del catàleg escollit com se n'obtenen les dades i un resum del que conté.

Deep learning: Introducció a Machine Learning i la tècnica de Deep learning que es vol utilitzar, la Feed Forward Neural Network.

Solució proposada: Descripció detallada dels programes creats i com executar-los, en aquesta secció es descriu tant l'algoritme d'arbre de decisió com el desenvolupament de la FFNN.

Anàlisi de resultats: Dades dels diferents experiments i comprovació dels resultats, inclou el grau de fiabilitat dels càlculs automàtics.

Conclusions: Conclusions final del projecte, assoliment d'objectius, reflexió sobre el seguiment de la planificació i comentari sobre la validesa de la solució proposada així com possibles treballs futurs.

2. Descripció de les dades

L'objectiu principal d'aquest apartat és el de establir un marc per seleccionar un conjunt de dades (o dataSet) per a la realitzar del projecte.

Ja que la temàtica del projecte és la classificació de galàxies / objectes estel·lars, i per tal de disposar d'un banc de dades prou extens i actualitzat s'ha buscat entre projectes actius que disposessin de dades publiques que a més fossin accessibles per a realitzar les classificacions automàtiques corresponents al projecte.

2.1 Descripció SDSS

Com ja s'ha anunciat, d'entre tots els projectes analitzats s'ha escollit SDSS, Sloan Digital Sky Survey (http://www.sdss.org/), que és un projecte d'investigació que va començar utilitzant imatges multi filtre i de corriment al vermell estereoscòpic d'un telescopi dedicat de 2.5m gran angular que està situat a Apache Point (APO) a Nou Mèxic, USA [11]. El projecte està en part finançat per la Alfred P.Sloan Foundation [12] i té com a objectiu principal cartografiar parts de l'univers observable i que utilitza dades d'un terç del cel.

Una particularitat d'aquest projecte és que la classificació d'objectes celestes es pot fer mitjançant enquestes a les que es pot contribuir, per exemple Galaxy Zoo (imatge) utilitza una pagina web on la població pot contribuir especificant característiques dels objectes que es veuen.

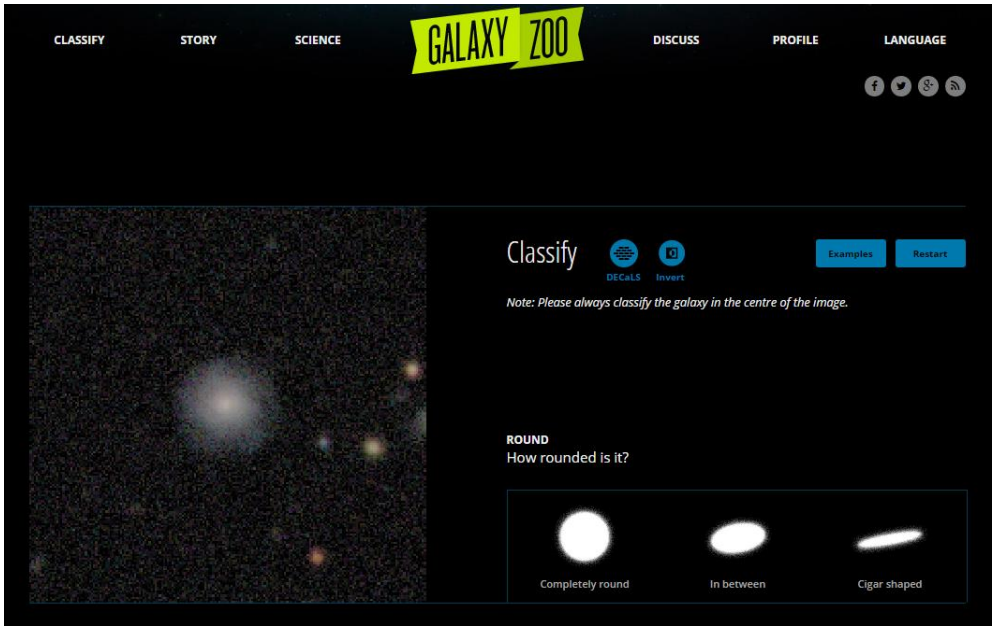


Figura 4

El projecte està en marxa des de l'any 2000, des de llavors s'ha passat per 4 fases diferents SDSS-I (2000 - 2005), SDSS-II (2005 - 2008), SDSS-III (2008 - 2014), SDSS IV (2014 - 2020). En quan a les dades, posa a disposició les de totes les fases per tant s'ha decidit utilitzar les de la ultima, en concret els conjunts de dades de la Data Release 13, la primera del SDSS IV. Aquesta release observacions realitzades al juliol de 2015, però tot i així els data sets són incrementals i sempre contenen les dades de les releases anteriors.

A més de les dades SDSS posa a disposició un conjunt d'eines que fan molt senzill el tractament i exportació de les dades.

2.2 Conjunts de dades i estudis més importants

Així doncs en aquesta recent release es centren els esforços en 3 estudis principals:

- **APOGEE-2**: Estudi espectroscòpic de la Via Làctia que conté dos components: la nord que agrupa les dades obtingudes per APO i la sud amb dades del telescopi de també 2.5m du Pont Telescope.

  En aquest experiment s'utilitza el registre de centenars de milers d'estrelles per explorar la formació de la Via Làctia. Per fer-ho analitza els moviments de les estrelles i la seva composició química obtinguda a través d'espectrografies. També mesura la abundància de elements com carboni, oxigen, nitrogen i ferro en estrelles que tenen planetes orbitant per determinar com aquests elements contribueixen en la formació de planetes.

- **eBOSS** (Baryon Oscillation Spectroscopic Survey): Aquest estudi identifica quàsars i galàxies que alhora conté dos subprogrames; **TDSS** per inspeccionar objectes variables i **SPIDERS** per fonts de rajos X.

  Es vol mesurar la historia de la expansió de l'univers a llarg del 80% de la seva historia, des de quan l'univers tenia menys de 3000 milions d'anys i aprofundir en els estudis sobre com l'energia obscura per determinar l'acceleració de l'expansió de l'univers.

  eBOSS observa galàxies y quàsars d'una rang de distàncies, de corriment al vermell que fins ara han estat ignorats per altres mapes de l'univers, al analitzar aquests objectes es crearà l'estudi de major volum l'univers dels realitzats fins ara.

- **MaNGA**: En aquest cas també s'utilitza el telescopi d'APO i es dedica a explorar l'estructura interna del voltant de 10.000 galàxies utilitzant espectroscòpica resolta en espai.

  L'objectiu principal és el de entendre la historia de les galàxies actuals a partir de les pistes deixades en el seu naixement i construcció i a través del ser creixement i fusió fins a la seva mort per refredament.

L'estudi utilitzarà mapes bidimensionals amb diferents característiques com la velocitat estel·lar i dispersió, la mitjana d'edat estel·lar o la historia de formacions. I utilitza galàxies seleccionades per abastar intervals de massa estel·lar de 3 ordres de magnitud.

2.3 Tipus de dades

Pel que fa a les dades que es posen a disposició durant aquesta fase s'inclou el següent, identificat per als components de la release actual i d'altres corresponents a l'anterior com a resultat de la característica acumulativa de SDSS:

- Espectre òptic de galàxies i quàsars del component BOSS com part del Sloan Extended Quasar (ELG) i l'estudi LRG (SEQUELS).
- Recent espectre reduït de galàxies de BOSS.
- Recent espectre estel·lar infraroig reduït per al component APOGEE.
- Abundància de determinat elements per a APOGEE.
- Cubs de dades per unitats de camp integral, observacions de galàxies properes del component MaNGA.
- Imatges re processades d'anteriors fases.

2.4 Model de dades

Al esquema que es troba tot seguit es mostra el model de dades de DR13 i que ha estat utilitzat per ajudar a escollir el sets de dades adients:

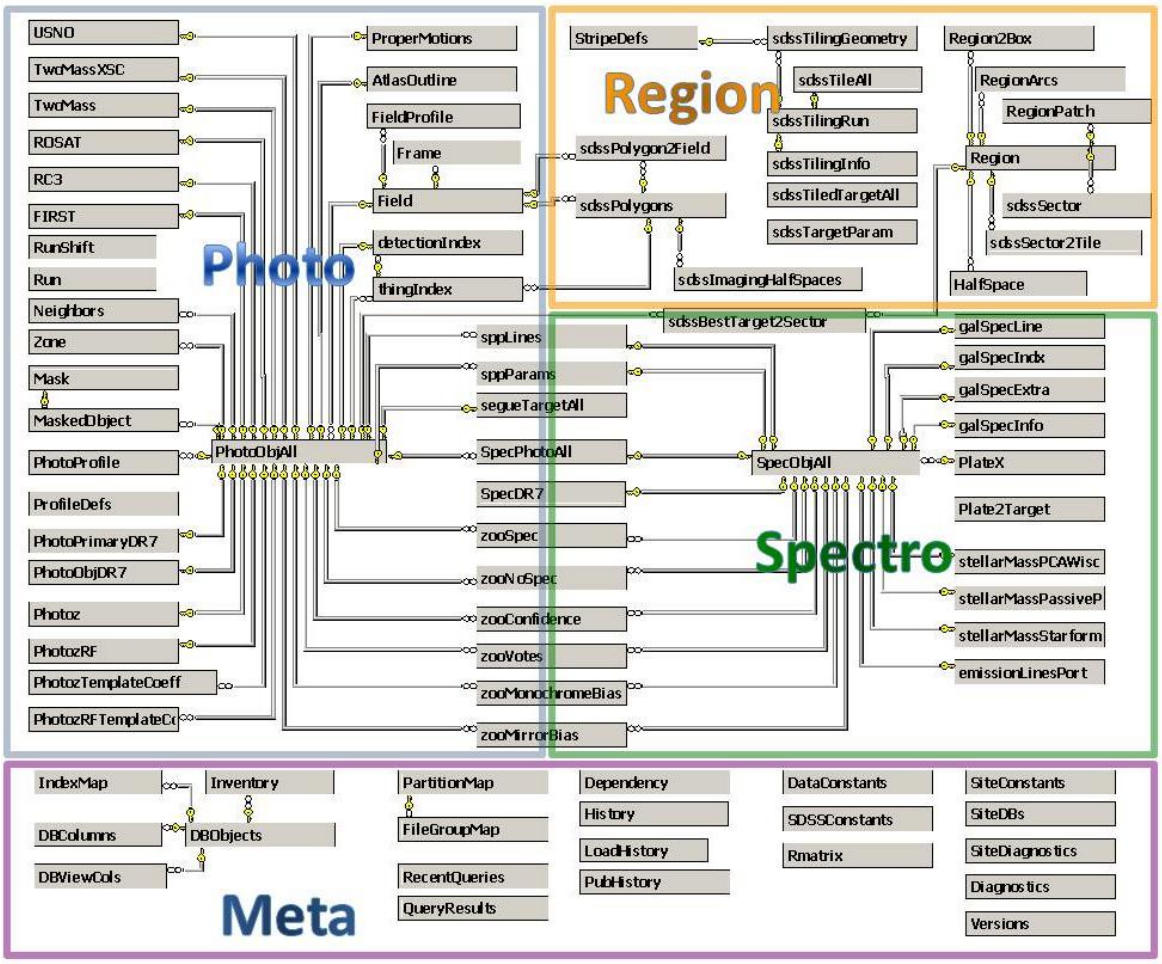


Figura 5

2.5 Eines de consulta i accés a les dades

Com s'ha anunciat SDSS posa a disposició les dades dels seus estudis mitjançant diferents eines i webs a continuació es fa una breu descripció d'ells, així doncs a més dels esmentats recursos per desenvolupar el projecte també s'utilitzaran les següents:

- Documentació: www.sdss.org/dr13
- Servidor arxiu de ciència (SAS), contindrà imatges i una aplicació interactiva per consultar-les (encara no està disponible).
- Detalls de SAS, estructura, format dels fitxers i continguts: https://data.sdss.org/datamodel
- Descarrega directe de DR13: https://data.sdss.org/sas/dr13
- Aplicació web per accedir a la base de dades del catàleg (CAS): http://skyserver.sdss.org/dr13
- **CASJobs**, aplicació web per realitzar consultes a CAS mitjançant una interfície basada en SQL i que permet accedir a les dades de totes les releases, alhora també des de CASJobs es posa a disposició **CASJobs Command Line Tool**, una aplicació Java de consola que permet fer les mateixes operacions que la web. Aquesta eina s'ha utilitzat al projecte i s'inclou als paquets resultants.

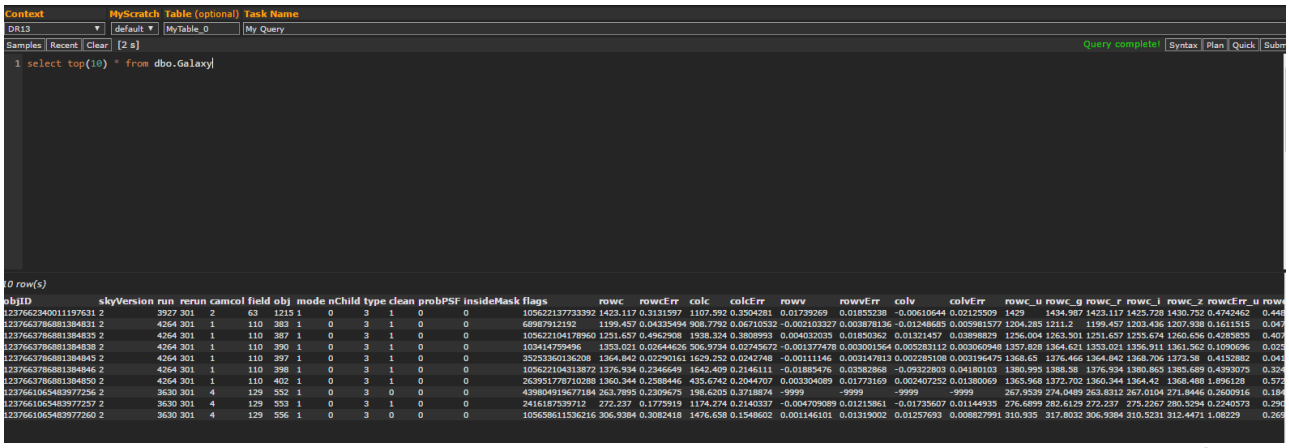


Figura 6

3. Catàleg fotomètric

Una vegada analitzades les dades i estudis creats a SDSS, s'ha determinat que un bon punt de partida serà el catàleg fotomètric.

La fotometria és una branca de l'astronomia que mesura la brillantor dels objectes per determinar-ne el tipus. La tècnica és molt antiga ja que va ser establerta al 190 AC per Hiparc de Nicea. En l'actualitat mitjançant les càmeres CCD s'obté una millora de precisió fins a deumil·lèsimes de magnitud, per exemple, el telescopi Kepler es va llançar al espai amb un sensor capaç de detectar canvis de 20 parts per milió.

Utilitzant les diferents eines a la nostra disposició s'ha determinat que dins del catàleg fotomètric de SDSS les taules i vistes més importants són les següents:

PhotoObjAll: És la taula principal, conté tots els objectes del catàleg i una entrada per detecció, la quantitat d'elements de la taula és de 1.231.051.050.

PhotoObj: Vista de PhotoObjAll, conté tots els objectes, tant primaris com secundaris.

PhotoPrimary: Tots els objectes considerats primaris, és a dir la millor versió de les dades d'un objecte donat corresponent a una observació.

Star: Objectes primaris que són estrelles.

Galaxy: Objectes primaris que són galàxies.

Sky: En aquesta vista hi ha mostres corresponents a fragments de cel.

Unknown: Objectes que no són cap dels anteriors.

PhotoSecundary: Segones deteccions d'objectes.

PhotoFamily: Objectes que no corresponen ni al primari ni al secundari.

Com es pot veure a l'annex 11.2, la quantitat de columnes del catàleg és molt gran (509) al llarg del projectes es provarà a executar l'algoritme amb diferents nombres de variables per tal de veure'n el rendiment, també es veurà que hi ha certes columnes innecessàries com ara l'id de la taula.

A l'estudi utilitzarem **PhotoPrimary** tal com es recomana en cas de només voler una sola detecció de cada objecte, la llista de tots els elements és de 469.050.976. També ens valdria utilitzar PhotoObj en cas que no ens importés tenir més d'una detecció de cada element, en aquest cas s'ha vist que el set tenia gairebé el doble d'elements 794.328.715, és a dir tindríem copies de molts d'ells.

Es destaca que en aquest set de dades s'utilitzen unitats definides per SDSS en algunes columnes, com per exemple el nanomaggies, utilitzat per representar els fluxos en unitat linear, no obstant, això no ha d'afectar el bon funcionament de l'algoritme.

3.1 Pretractament de dades

Com s'ha vist, l'origen de dades escollit es més que enorme, 509 columnes, de les quals se n'ha eliminat algunes, a continuació la relació de columnes excloses i el motiu:

- La columna [probPSF] en la majoria d'elements té el mateix valor que la pròpia etiqueta i el fet d'incloure-la ens porta a classificacions directes que no aprofitarien el potencial del algoritme com no ho faria utilitzar la pròpia etiqueta,
- L'identificador de la taula, no aporta cap mena de valor a les dades més que la identificació del registre.
- [obj] és un altre identificador de l'objecte que no aporta informació rellevant.

Per aquest estudi s'ha decidit deixar aquests valors fora de l'anàlisi. Així doncs s'ha realitzat un subset amb les columnes que s'utilitzaran i que ens deixa amb algunes columnes menys (a l'annex 10.2 apareixen marcades en negre les columnes ignorades). Addicionalment els algoritmes desenvolupats s'ha analitzat les columnes per determinar si tenen valors constants, en cas de ser així s'ha eliminat aquetes columnes del càlcul per tal d'evitar errors.

De les dades restants la corresponent a la columna Type, és el valor que etiqueta l'objecte i per tant el que més ens interessa ja que representa el tipus de l'objecte, l'objectiu de l'algoritme es determinar el valor del tipus per les mostres del conjunt de test després de realitzar l'entrenament pertinent, pel que fa a la resta es tracta, en qualsevol cas, de dades numèriques.

Pel que fa a la resta de dades, s'ha considerat que són vàlides ja que no es disposa del coneixement en astronomia necessari per determinar si n'hi ha més de superflus.

3.2 Conjunts d'entrenament i test

Una vegada aclarit les dades a utilitzar es descriu els diferents conjunts de dades que s'utilitzaran al llarg del projecte.

Ens seran necessaris 2 tipus de conjunts, els d'entrenament que aniran convenientment etiquetats i els de test que farem servir per classificar com si d'objectes nous es tractés.

Com que l'origen de dades és molt gran s'ha decidit crear conjunts de diferents mides per tal de poder accelerar les proves durant el desenvolupament i l'anàlisi. Per aquesta raó tenim el següent:

Nom del conjunt	Tipus	Nombre de registres
Training_Small	Conjunt petit	1.000
Test_Small		100
Training_Medium	Conjunt mitjà i principal punt de comparació entre els 2 algoritmes	10.000
Test_Medium		1.000
Training_Big	Conjunts amb el màxim de dades que s'han extret de SDSS	300.000
Test_Big		50.000

És important destacar que en ambdós algoritmes, la selecció d'elements d'entrenament i test es fa mitjançant la configuració de certs paràmetres, ja que el data set original és un únic fitxer, de manera que l'anterior taula no és fixa i es pot alterar a voluntat abans de l'execució dels algoritmes.

Es destaca que s'ha obtingut al voltant de 350.000 registres del datasource original de SDSS, són els màxims que s'han pogut obtenir donades les restriccions tècniques (ordinador i connexió a internet) utilitzats per realitzar el projecte.

4. Deep learning

Es coneix com a Deep Learning o aprenentatge profund a un conjunt de tècniques i algoritmes propis de l'aprenentatge automàtic que utilitzen arquitectures composades de varies capes per realitzar extracció i transformació de noves característiques en abstraccions d'alt nivell, estan composades per múltiples transformacions que poden ser lineals o no lineals. A més, els algoritmes de DL poden ser supervisats o no.

De forma simplificada podem dir que els algoritmes DL simulen el funcionament bàsic del cervell i les neurones. On els nodes del sistema corresponen a les neurones.

Alguns dels usos principals del DL són el reconeixement de veu, la visió artificial o el reconeixement d'imatges.

4.1 Xarxes neuronals

Actualment els conceptes de Deep Learning són pràcticament sinònims de xarxa neuronal, com s'ha dit l'objectiu d'aquestes tècniques es la d'emular en cert grau, el cervell per tal de donar solució a problemes no lineals. Si un sistema de xarxa neuronal està ben entrenat és molt útil per sistemes de classificació i identificació d'informació desconeguda

Així doncs més concretament les xarxes neuronals poden ser vistes com un graf:

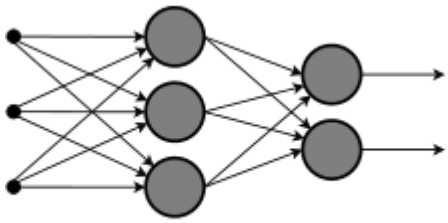


Figura 7

On tenim nodes, es a dir, neurones, que estan connectats entre ells, cada entrada a una neurona té un pes associat, i si cada pes multiplicat per l'entrada i sumat al de totes les entrades sobrepassa cert llindar el node s'activa, aquesta serà la funció d'activació del node.

Inicialment s'utilitzava força com funció d'activació la corba sigmoïdal [13] que aporta no linealitat en els càlculs de la xarxa i delimita el nivell d'activació entre 1 i 0. Però actualment el més comú és utilitzar les Rectified Liner Unit (ReLu), aquest projecte n'és un exemple, que ha demostrat ser més efectiva ja que presenta alguns avantatges com ara: la plausibilitat biològica, el menor grau d'activació, el gradient es propaga de forma eficient, realitza únicament

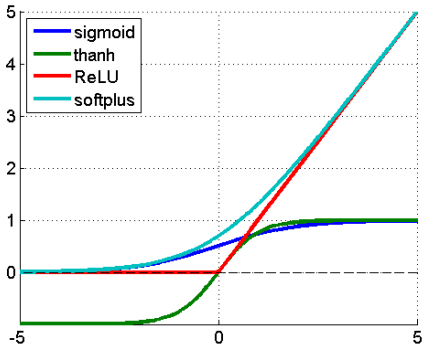


Figura 8

comparacions, sumes i productes cosa que es tradueix en una millora computacional finalment, disposa d'escala invariant on: max(0,ax) = amax(0,x).

Un dels tipus més comuns de xarxa neuronal i el que s'ha utilitzat en aquest projecte és la Feedforward [14], també anomenat MultiLayer Perceptron. En aquesta xarxa els nodes estan organitzats en múltiples capes, les connexions es formen connectant cada node d'una capa a totes les neurones de la següent capa, així tots els nodes d'una capa estan connectats a tots els de la següent.

En una xarxa Feedforward podrem trobar usualment, una capa d'entrada, una, o vàries capes ocultes i una capa de sortida, el processament real de la informació es realitza principalment a la capa oculta i també a la de sortida, i la capa d'entrada només es dedica a transferir la informació a la primera capa oculta, amb aquest flux no es permet la retroalimentació.

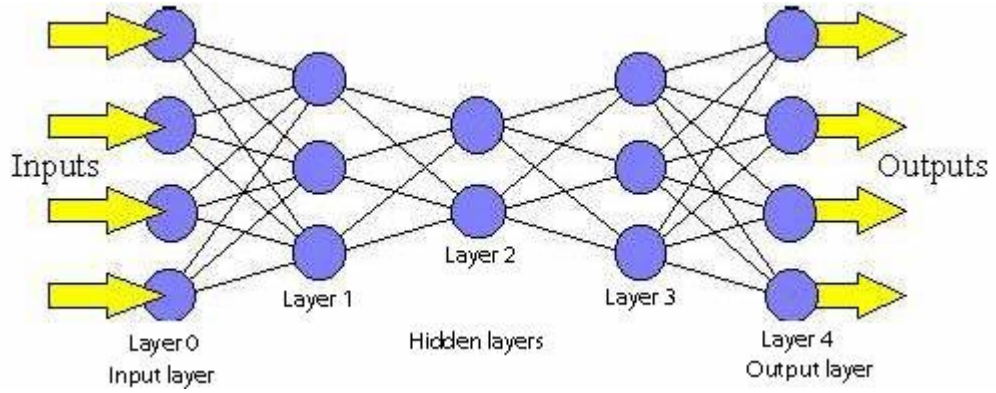


Figura 9

Pel que fa a l'entrenament aquest tipus de xarxes utilitza l'anomenada Backpropagation, una vegada obtinguts valors a la sortida, aquests es comparen amb la resposta correcte i se n'obté un valor d'error. Aquest error es propaga cap a la xarxa de manera que es poden ajustar els càlculs de cada connexió per reduir el valor d'error. A base de repetir aquest procés la xarxa arribarà a un estat on l'error en el càlcul serà petit. En aquest cas s'ha d'anar amb compte al repetir aquest procés per no caure en un sobreajustament en les dades d'entrenament que després afectarà negativament al resultat.

Un altre apunt important en el desenvolupament de les xarxes neuronals és l'ús de les anomenades funcions de drop_out, que eliminen unitats després de, per exemple crear les capes d'entrada o en cada capa oculta, eliminar cert percentatge d'unitats ajuda també a reduir el sobreentrenament prevenint coadaptacions complexes en les dades d'entrenament.

Un altre tipus de xarxa neuronal, que no és tractada en aquest projecte, és la Convolucional [15], aquestes emulen les neurones del còrtex visual i que són molt efectives en visió artificial per exemple en classificació d'imatges.

5. Solució proposada

El desenvolupament realitzat es compon de dos algoritmes, un correspon a un classificador tradicional implementat mitjançant un arbre de decisió i l'altre a una xarxa neuronal Feedforward. L'arbre s'utilitza com a base per comparar-ne els resultats amb els obtinguts amb la xarxa neuronal i així poder-ne determinar els pros i contres.

5.1 Lector de fitxers

Ja que es treballa amb grans quantitats de dades s'ha estimat oportú, utilitzar, per a la xarxa neuronal els fitxers fits [16], que són de mida menor i amb els que es pot treballar de forma eficient tot i que siguin molt grans.

Els fitxers fits tot i ser utilitzats per emmagatzemar imatges per astronomia, també és utilitzat en el mateix cap per dades. Cada fitxer conté almenys 2 índexs, un per imatges i el segon per taules de dades:

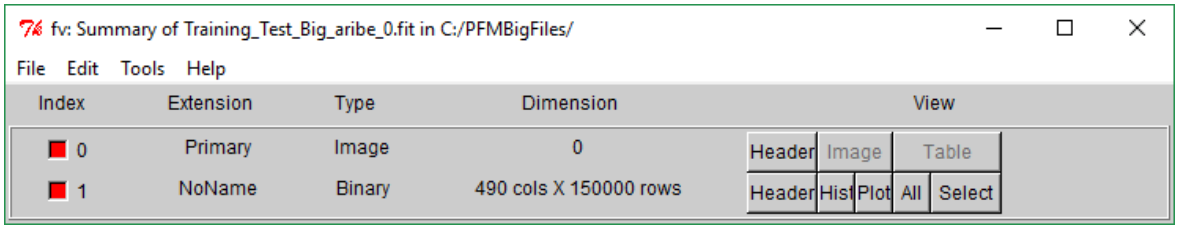


Figura 10

Tot i ser molt útils i àgils per treballar amb ells s'ha d'importar la llibreria ja esmentada *astropy.io.fits,* que ens permet obrir els fitxers de forma asíncrona i ajuda al tractament de les seves dades sense haver de mantenir en memòria tot el fitxer, que pot arribar a ser molt gran.

Per tal de treballar amb aquest format de fitxers s'ha creat un script Python que els llegeix i a més s'encarrega de generar els conjunts d'enterament i test segons una ruta de fitxers fits i les mides desitjades.

La gestió de fitxers fits no és l'únic àmbit de l'script, addicionalment també s'encarrega de tot el relacionat amb fitxers que s'utilitzen per executar la FNN i l'arbre de decisió, obre el fitxer de configuració o genera i llegeix els fitxers de metadades de la FNN, s'anomena, FilesManager.py, i conté els següents mètodes:

Gestió de fitxers FITS:

def openFitsFile(file): Obra un fitxer fits i en retorna un objecte HDUList amb les capçaleres del fitxer.
def closeFitsFile(hdulist): Tanca el fitxer fits prèviament obert, rep l'objecte HDUList.

def getDatasetList(file, size = 100, index = 0): A partir d'una ruta de fitxer una mida i un index retorna una col·lecció d'elements.
def getTrainingTestSets(*arg): És un mètode genèric que s'utilitza per retornar diferents col·leccions de diferent mida a partir d'un datasource original.
def getDTNormalizedLines(trainingSize, testSize, getMetadata = None): Utilitzat per al arbre de decisió, a partir de les mides i la manera en que s'obtenen les metadades retorna les col·leccions d'entrenament i test.
def normBatch(inputData, metaData): Normalitza un conjunt de dades (batch) mitjançant un llistat de metadades passat per paràmetre.
Donat que les dades són força disperses s'ha decidit normalitzar les dades a més d'eliminar les columnes que tinguin valors constants.
def getTrainingTestLists(traiSize = 100, valSize = 10, testSize = 10, getMetadata = None): A partir del fitxer i de les mides dels conjunts d'entrenament, validació i test, retorna els 3 conjunts, els 3 llistat d'etiquetes i les metadades necessàries per executar l'algorisme.
def getFitsFromSDSS(tableName = "MyTable", traiSize = 100, testSize = 10): Aquest mètode permet descarregar un data set directament de SDSS, per això s'utilitza el CasJobs Command Line Tool.

Gestió de fitxers de metadades:

def getMetadataFromFile(): Es recuperen les metadades dels fitxers corresponents.
def calculateMetadata(): Es calculen les metadades i es guarden als fitxers corresponents.

Gestió de la configuració i altres:

def getConfigDate(filename = 'config.cfg'): Es recupera la configuració completa a partir del fitxer especificat, que per defecte és config.cfg.
def printAndSave(msg, dt=True): Escriu un missatge passat en un fitxer de log, per defecte ho farà al fitxer de log del arbre de decisió.

5.2 Classificador mitjançant un arbre de decisió

L'objectiu de l'algoritme és determinar el tipus d'un objectes segons les seves propietats. Per tal de dur a terme la tasca s'han generat dos conjunts de dades a partir de PhotoObj que tenen etiquetats els tipus d'objectes. Un correspondrà al conjunt d'entrenament i l'altre al de test per veure si la classificació es realitza correctament.

Per realitzar aquest algoritme s'ha partit del treball realitzat a la PAC3 de l'assignatura d'Intel·ligència Artificial Avançada (curs 2015/16) que tenia un funcionament semblant, però fent-li els canvis necessaris perquè funcioni amb les dades d'aquest projecte (fitxers fits) i les restriccions tècniques que n'esdevenen.

5.2.1 Anàlisi del codi

Com s'ha comentat aquest algoritme està basat en el realitzat a la PAC3 d'IAA de manera que la seva estructura és molt semblant, tot i així s'han fet alguns canvis importants respecte l'original, com el tractament automatitzat dels noms de les capçaleres enlloc d'escriure-les manualment, amb prop de 500 columnes el codi augmentava molt i es transformava en poc mantenible i poc escalable.

També s'ha canviat la manera en que es generen els conjunt d'entrenament i test, ara s'obtenen directament i no es calculen, això accelera una mica l'execució del programa, a continuació es descriuen les funcions més importants:

***def runTests(instance_list, dectree)**:* Amb un arbre passat per paràmetre, intenta classificar els elements del conjunt de test, finalment en calcula la precisió.

def generateDT(training_set, max_depth, min_accuracy): Construeix l'arbre, per defecte s'ha deixat una profunditat de 10 i una precisió mínima de 0.8

def classify(instance, dectree): Donat un arbre i un element, l'intenta classificar.

def getMajorityClass(items): Obté la classe majoritària en funció dels 4 tipus d'elements possibles, Star, Galaxy, Unkown i Sky.

def removeDup(seq): Elimina els valors duplicats d'una seqüència, operació necessària per calcular la classe majoritària.

def goodness(classes,attr_values): Funció de bondat, genera els punts de tall i per cada un en calcula les classes majoritàries, posteriorment en calcula la bondat.

5.2.2 Execució, dades de prova i conclusions prèvies

Per poder executar l'algorisme és necessari:

- Tenir l'entorn configurat amb els recursos explicats al punt 1.4.2 i preferiblement amb les versions llistades a l'annex 10.3.
- Accedir a una consola y executar la comanda:
  python decisionTree [path del set d'entrenament] [path del set de test].
- Disposar a la mateixa carpeta d'execució d'un fitxer de configuració "Config.cfg" que tingui els següent format:

  [Config]
  trainingSize = 1000
  validationSize = 100
  batchSize = 50
  testDataSize = 100
  nHiddenLayers = 10
  num_epochs = 10
  fits_file = C:\\PFMBigFiles\\DataSet_Small\\TrainingTest_Small.fit
  metadata_file = metadata.dat
  getFromMetadataFile = True
  colsToDeleteFile = colsToDelete.dat

  I del que se n'utilitzaran els camps trainingSize, testDataSize, el fitxer fits i les ubicacions dels fitxers de metadades.
- Disposar també d'un fitxer fits com a base per construir els diferents sets, en cas de no disposar d'un d'aquests fitxers utilitzar el mètode "getFitsFromSDSS" de FilesManager.

Ben aviat s'ha observat que intentar executar aquest algoritme amb els conjunts de dades mitjà o gran esdevenia en uns temps de procés alarmantment alts, cosa que s'analitzarà en futures seccions, per ara, es destaca que la majoria de proves s'han fet amb els conjunts Training_small i Test_small.

5.3 Feedforward neural network

5.3.1 Algorisme FNN

L'algorisme principal és l'anomenat MLP.py, i conté tot el necessari per crear, entrenar i executar test en una Xarxa Neuronal de tipus Feed Forward. A part de les llibreries utilitzades inclou la importació dels filesManager per tal de gestionar l'entrada i sortida a fitxers i conjunts de dades i de metadades. A continuació es descriuen els mètodes de l'algorisme així com els seu funcionament bàsic.

> ***def buid_MLP(input_var=None, depth=2, width=1000, drop_input=.2, drop_hidden=.5, nCols = None):*** Es construeix una xarxa neuronal de tipus Feedforward a partir dels paràmetres de configuració establerts als arguments.
> ***def iterate_minibatches(list,list_targets, batchsize, metadata, colsToRemove):*** Es creen petits blocs de longitud especificada a la configuració per enviar com a entrada a la FNN, addicionalment es normalitzen les dades de cada bloc segons els paràmetres de metadades prèviament calculats. Es important tenir en compte que fins arribar a aquest mètode es treballa amb objectes Fits_rec que es carregen sota demanda i evitant sobrecarregar la memòria de l'ordinador quan trobem grans quantitats de dades, en aquest mètode es on al obtenir un bloc petit aquest es transforma en un array numpy que serà més fàcil de tractar i enviar com input de la NN. Aquest procés millora tant la carrega en memòria com el rendiment de l'aplicació.
> ***def main():*** Funció principal, prepara les dades i organitza els mètodes per construir la NN, entrenar-la, executar-ne validacions i els tests.

L'script primer recupera i prepara les dades i posteriorment les entrena, per acabar executant el conjunt de test i mostrar la precisió de les proves, a continuació es mostra aquest procés d'execució en les principal parts del codi:

1. Es recupera la configuració cridant a FilesManager, i es guarden els valors necessaris per la configuració i execució de la NN.

```
trainingSize,validationSize,batchSize,testDataSize,nLayer,num_epochs, getFromFile =
        getConfigDate()
```

2. S'obtenen els conjunts de dades per l'entrenament, validació i test.

```
train, trainTargets, val, valTargets, test, \
testTargets, metadata, colsToRemove = \
getTrainingTestLists( traiSize = trainingSize,
        valSize = validationSize,
        testSize = testDataSize,
        getMetadata =  getMetadata)
```

3. Es declaren les variables Theano per l'entrada (inputs) de tipus matrius i les etiquetes (targets) que serà un vector.

```
input_var = T.matrix('inputs')
target_var = T.ivector('targets')
```

4. Es construeix la NN.

```
network = buid_MLP(input_var = input_var, depth = nLayer, width =
    500,drop_input=.2,drop_hidden=.5, nCols = len(metadata))
```

El procés de construcció es compon de 3 parts:

a. Creació de la capa d'entrada i drop d'unitats especificades, que s'han fixat a un 20%:

```
network_ = lasagne.layers.InputLayer(shape=(None, nCols), input_var=input_var)
if drop_input:
    network_ = lasagne.layers.DropoutLayer(network_, p=drop_input)
```

b. Creació de capes ocultes i drop d'un 50% de les unitats:

```
for _ in range(depth):
    network_ = lasagne.layers.DenseLayer(
        network_, num_units = width, nonlinearity=lasagne.nonlinearities.rectify)
    if drop_hidden:
        network_ = lasagne.layers.DropoutLayer(network_, p=drop_hidden)
```

c. Creació de capa de sortida i retorn:

```
l_out = lasagne.layers.DenseLayer(
    network_, num_units=7,
    nonlinearity=lasagne.nonlinearities.softmax)

return l_out
```

5. Es creen les funcions Theano i expressions:
   a. La predicció a partir de la sortida del a NN.

```
prediction = lasagne.layers.get_output(network)
```

   b. La pèrdua de tipus crossentropy ja que la classificació serà per més de dues classes.

```
loss = lasagne.objectives.categorical_crossentropy(prediction, target_var)
loss = loss.mean()
```

   c. La funció d'updates per determinar com es realitzarà l'actualització de dades en cada pas de l'execució de la NN.

```
params = lasagne.layers.get_all_params(network, trainable=True)
updates = lasagne.updates.momentum(
    loss, params, learning_rate=0.1, momentum=0.9)
```

   d. Es determina la predicció i pèrdua per els tests i validació, en aquest cas la predicció evita fer drop-out activant el mode deterministic per passar a través de tota la xarxa.

```
test_prediction = lasagne.layers.get_output(network, deterministic=True)
test_loss = lasagne.objectives.categorical_crossentropy(test_prediction,target_var)
test_loss = test_loss.mean()
```

   e. Es crea una funció Theano per a la precisió en els tests i validació.

```
test_acc = T.mean(T.eq(T.argmax(test_prediction, axis=1), target_var),
        dtype=theano.config.floatX)
```

f. Finalment es declara la funció Theano per l'entrenament, que retornarà la pèrdua i utilitzarà la funció d'updates anteriorment declarada i també es crea la funció per la validació que retornarà la pèrdua i la precisió.

```
train_fn = theano.function([input_var, target_var], loss, updates=updates,
        name="TrainingFunc", mode=NanGuardMode(nan_is_error=True,
        inf_is_error=True, big_is_error=True))
val_fn = theano.function([input_var, target_var], [test_loss, test_acc],
        name="ValidationFunc")
predict_fn = theano.function([input_var], prediction)
```

6. Amb la NN configurada i totes les funcions Theano preparades s'inicia l'entrenament, que es divideix amb un nombre preconfigurat de iteracions on, en cada una, s'executarà l'entrenament de totes les dades del set.

```
for epoch in range(num_epochs):
  start_time = time.time()
  train_err = 0
  train_batches = 0
  for batch in iterate_minibatches(train,trainTargets, batchSize, metadata,
    colsToRemove):
    inputs, targets = batch
    tmp = train_fn(inputs, targets)
    train_err += tmp
        train_batches += 1
```

a. La funció iterate_minibatches genera les diferents col·leccions d'elements a entrenar, una de les tasques principals és la de normalitzar les dades per poder treballar amb conjunts de dades on elements siguin petits i uniformes, evitant també les columnes amb valors constants o 0, de manera que s'eviten errors o resultats no desitjats.

   Per normalitzar s'utilitzen els valors mínims i màxims de totes les columnes del set de dades que es calculen prèviament, i guarden a un fitxer .dat per poder evitar-ne el re càlcul, que pot ser molt costos.

```
def iterate_minibatches(list,list_targets, batchsize, metadata, colsToRemove):
  for start_idx in range(0, len(list) - batchsize + 1, batchsize):
    batchLines = list[start_idx: start_idx + batchsize]
    batchTargets = list_targets[start_idx: start_idx + batchsize]
    targets = batchTargets
    targets = targets.astype(int)
    tmp = np.array(batchLines.tolist())
    inputs = np.delete(tmp,colsToRemove,1)
    # s'estandaritzen les dades mitjançant les metadades calculades
    inputs = normBatch(inputs, metadata)
    yield inputs, targets
```

7. Dins de cada una de les iteracions es realitza també la validació d'aquell entrenament.

```
for batch in iterate_minibatches(val,valTargets, batchSize, metadata, colsToRemove):
    inputs, targets = batch
    err, acc = val_fn(inputs, targets)
    val_err += err
    val_acc += acc
    val_batches += 1
```

8. Per acabar s'executarà la funció de validació per el conjunt de dades de test.

```
for batch in iterate_minibatches(test,testTargets, batchSize, metadata, colsToRemove):
    inputs, targets = batch
    err, acc = val_fn(inputs, targets)
    test_err += err
    test_acc += acc
        test_batches += 1
```

5.3.2 Execució

Per poder executar l'script ens caldrà un conjunt de fitxers prèviament preparats:

- Tenir l'entorn configurat, amb Theano i Lasagne instal·lats tal com es descriu al punt 1.4.2 i preferiblement amb les versions llistades a l'annex 10.3.
- Disposar a la mateixa carpeta d'execució d'un fitxer de configuració "Config.cfg" que tingui els següent format:

```
[Config]
trainingSize = 1000
validationSize = 100
batchSize = 50
testDataSize = 100
nHiddenLayers = 10
num_epochs = 10
fits_file = C:\\PFMBigFiles\\DataSet_Small\\TrainingTest_Small.fit
metadata_file = metadata.dat
getFromMetadataFile = True
colsToDeleteFile = colsToDelete.dat
```

- Disposar també d'un fitxer fits com a base per construir els diferents sets, en cas de no disposar d'un d'aquests fitxers utilitzar el mètode "getFitsFromSDSS" de FilesManager.
- Disposar dels fitxers metadata.dat i colsToDelete.dat, en cas contrari es pot configurar l'script perquè els generi posant *getFromMetadataFile = False* al fitxer de configuració.
- Finalment per executar l'script només cal escriure la següent comanda:

```
python MLP.py
```

Una vegada realitzades les primeres execucions s'ha pogut comprovar que l'algorisme és molt sensible als canvis de configuració, ens trobem amb 5 elements de configuració bàsics que influeixen directament a l'entrenament: Mida del set, mida dels blocs de dades de

l'entrenament, nombre de hidden layers, mida de cada layer, nombre d'iteracions. Una tasca vital és la de trobar la millor combinació d'aquests valors per maximitzar la precisió de l'entrenament, reduir-ne el temps d'execució i evitar el sobre entrenament.

Sobre les capes ocultes, en diferents articles es parla de tècniques per determinar-ne el nombre idoni, un de molt interessant és aquest: http://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw , que explica els casos en que es suficient amb una única hidden layer però calculant la seva mida en funció de la mitjana de neurones entre l'entrada i la sortida. Però la majoria de fonts revisades semblen d'acord en que l'assaig/error és una de les pràctiques més habituals per trobar la configuració òptima, i aquesta és la que s'ha seguit per a dur a terme les proves de l'algorisme.

Es destaca que en aquest algoritme, a diferència de l'arbre de decisió, normalitza les dades per cada bloc del procés iterate_minibatches, s'ha decidit fer així per aprofitar al màxim la potencia dels fitxers fits, que permeten llegir les dades sota petició, de manera que el sistema no hagi de mantenir en memòria grans quantitats de dades, d'aquesta manera amb aquest algoritme es poden processar volums de dades molt grans sense haver de preocupar-se de sobrepassar el límit de memòria principal.

6. Anàlisi de resultats

6.1 Arbre de decisió

Per a avaluar l'arbre de decisió s'han realitzat execucions de l'algoritme utilitzant diferents mides de sets d'entrenament i test fins que s'ha obtingut la informació necessària per comparar amb la xarxa neuronal.

Per evitar el sobreentrenament s'ha escollit les següents dades de configuració, després de realitzar múltiples proves amb més i menys capes aquesta sembla que dona una bona relació de precisió i temps:

Profunditat máxima	Precisió límit[1]
10	1

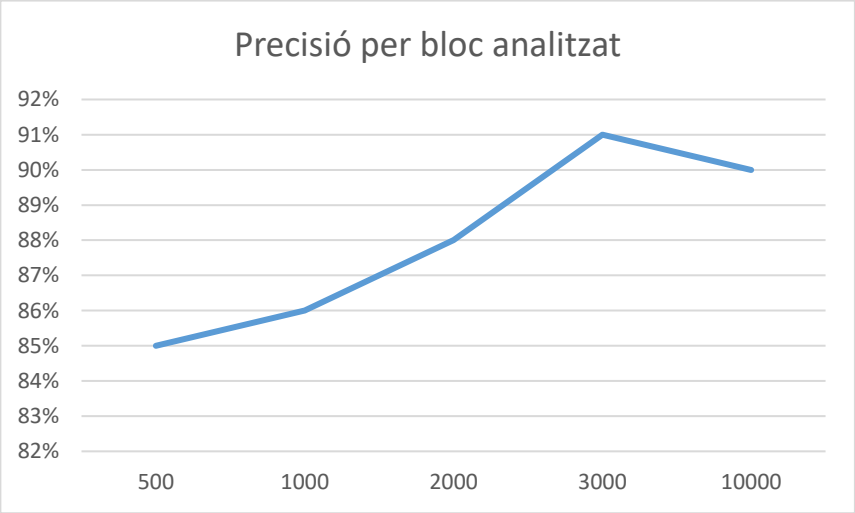


Figura 11

Tot i que només s'han inclòs els resultats de 5 execucions (500, 1000, 2000, 3000 i 10.000 registres amb 500 de test cada un), ja es pot veure certa tendència. Des de la primera execució l'arbre ja dona una precisió força bona, sobrepassant el 80%, en les subsegüents la cosa es manté amb certs augments conforme afegim més registres, arribant amb un màxim de 91% amb 3000 registres que sembla que tendeix a estabilitzar-se en la de 10.000 que n'obté 90%.

És clar que afegint més registres o augmentant la profunditat màxima podríem anar millorant una mica la precisió (els nodes generats no eren excessivament especialitzats per tant no es produïa sobreentrenament), però aquesta millora seria a costa d'augments molt elevats en els temps d'execució i força difícils de manejar, també sembla que arriba un moment que l'algorisme arriba al límit de precisió que ronda al 91%.

Així doncs, el punt més crític en l'execució ha estat el temps, que ens ha impedit sobrepassar el 10.000 registres i pel qual no s'han realitzat execucions de 4.000 a 9.000 elements per optimitzar les execucions. Recordem que l'arbre en cada execució ha de

[1] Es pot veure aquests valors al mètode "generateDT(training_set, max_depth=10, min_accuracy=1)" on apareixen com a valor per defecte dels paràmetres.

recórrer tots els elements de la llista varies vegades, partint d'un node inicial que el revisa tots i posteriorment partint-lo en nodes de menor mida i així recursivament fins a arribar als nodes finals.

Per dur a terme aquest procés, en cada node, l'arbre ha d'analitzar els valors de totes les columnes del seu bloc d'element per tal de trobar-ne la bondat, i aquí estem utilitzant unes col·leccions de dades que disposen del voltant de ***500*** columnes.

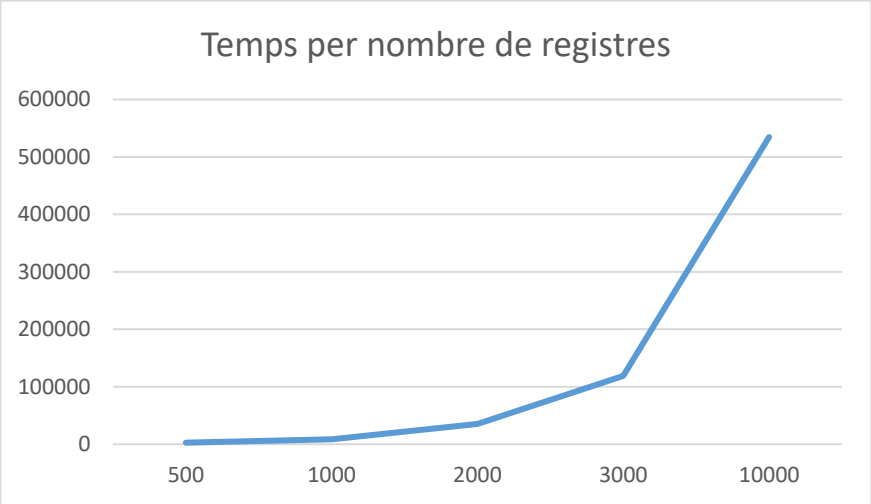


Figura 12

Això ens dona com a resultat, que amb un data set de 10.000 elements, només pel primer node l'arbre trigava 109.886 segons en processar-lo, és a dir 30 hores i un total de 534.629 segons, unes 148 hores, és a dir més de 6 dies, en l'execució completa dels 10 nivells de l'arbre. Si ho revisem respecte les altres execucions veurem que per cada 1.000 elements afegits a la col·lecció a processar el temps d'execució gairebé es multiplica per 4.

El principal handicap en aquest projecte és la immensa quantitat de columnes, a més de la possibilitat de disposar de milions d'elements en el data set, i queda clar que utilitzar un arbre afecta molt en el temps d'execució.

6.2 Feedforward neural network

Fixem-nos ara amb els resultats d'execució de la Xarxa Neuronal. En aquest cas s'ha establert una configuració inicial i executat amb diferents mides de dataset. Per cada bloc s'han realitzat múltiples execucions amb les mateixes dades, donant resultats molt semblants en totes elles, aquí s'en presenta una mostra.

Nombre de capes ocultes	Iteracions de l'entrenament	Mida dels blocs
3	10	50

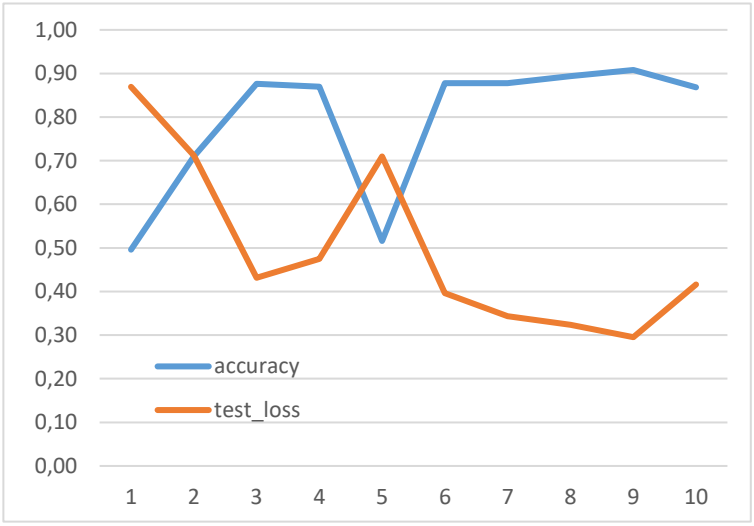


Figura 13

Tot i que totes aquestes execucions han estat molt més ràpides, la de 10.000 elements trigava uns 12 minuts, en general la precisió en el test millorava poc entre els diferents blocs d'entrenament, s'observa una petita millora quan més registres d'entrenament fem servir.

En el cas de les execucions de 5.000 i 10.000 elements s'ha observat que a les ultimes iteracions s'entrava en sobreentrenament raó per la qual la precisió en el test es redueix tant. Podem veure, també, que s'arriba al màxim de 90,8% a l'execució amb 9.000 registres i si ens fixem en l'error en el test podem veure que segueix una evolució oposada a la de la precisió, reduint-se amb precisions altes i elevant-se quan n'hi ha de baixes.

Ara bé, després d'aquestes primeres proves, i veient que no queda molt clara la utilitat de la FNN que ha donat precisions irregulars i no gaire més altes que a l'arbre, s'ha decidit entrar en més detall en una de les execucions. Per exemple, s'ha agafat la de 3.000 registres, que amb la configuració "estàndard" obtenia un 87,6%, i se n'ha ajustat els paràmetres per millorar el resultat.

És interessant utilitzar aquest bloc ja que és un dels utilitzats en les proves amb l'arbre, així ens servirà per comparar els dos algoritmes en termes de precisió i temps.

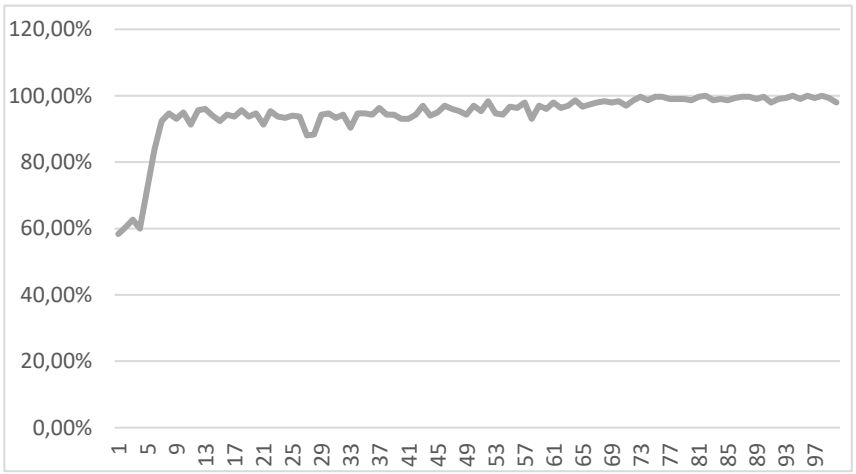


Figura 14

Amb 100 iteracions obtenim una precisió del 97.33% al test, 10 punts superior a l'obtinguda amb el mateix nombre de registres a l'anterior execució i també força superior a l'arbre que amb 3000 registres va obtenir un 91%. Pel que fa a l'error al test és de 0.16, inferior al 0,43 obtingut a l'execució prèvia o al 0,29 que és el mínim de totes les anteriors execucions. Tot i que aquesta és la millor d'un seguit d'experiments realitzats amb els mateixos conjunts de dades, els resultats dels altres eren molt semblants a aquest.

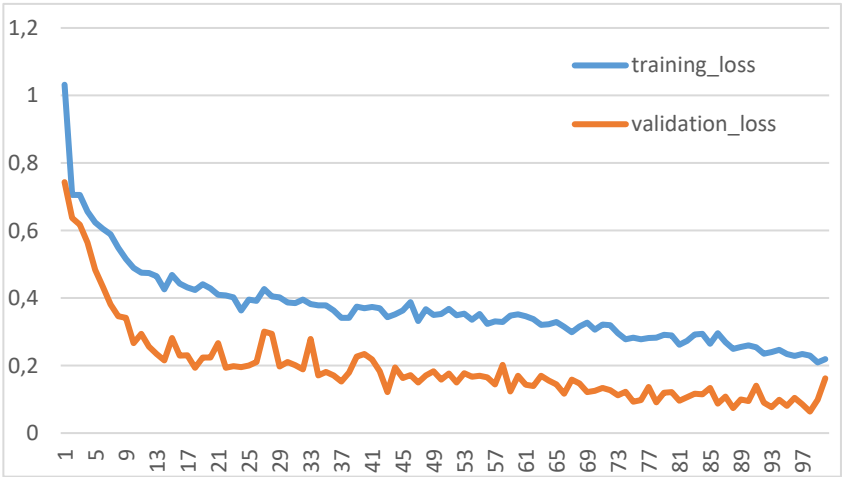


Figura 15

Pel que fa als errors veiem que en tot moment tenen un tendència a la baixa, cosa que no impediria el realitzar més iteracions per intentar augmentar la precisió. De fet s'ha realitzat una execució amb 500 iteracions del mateix data set i amb la mateixa configuració i al acabar s'ha pogut observar que al voltant de la iteració 220 comença a produir-se el sobreentrenament, moment que marcaria quan hem d'aturar l'entrenament.

En quant al temps, aquesta execució amb aquestes 100 iteracions només ha trigat 2.307 segons en ser entrenada, és a dir menys de 40 minuts, molt menys que les 33 hores del arbre amb els mateixos registres.

La configuració utilitzada ha estat amb 2 capes ocultes, que si bé no son masses, si són suficients per obtenir una bona precisió amb el set utilitzat. Estem utilitzant un nombre de registres força reduït de manera que al reutilitzar les dades moltes vegades dins la xarxa pot causar resultats inesperats, raó per la que s'ha reduït el nombre de capes ocultes.

6.2.1 Proves amb datasets més grans

La principal característica del data set de SDSS és la enorme quantitat de dades, de manera que per continuar amb les proves amb la Xarxa neuronal s'ha procedit a realitzar-ne amb sets de dades més grans i així podrem veure que la potencia de la xarxa neuronal pren un altre sentit.

Registres	Nombre de capes ocultes	Iteracions de l'entrenament	Mida dels blocs
200.000	10	100	500

Amb aquesta configuració ja hem utilitzat un nombre més elevat de mida de bloc, ja que estem treballant amb força més dades, els blocs són les particions mínimes que es fan de les dades originals abans de passar-les per la FNN, i un bloc ha de tenir com a màxim el nombre de registres total del set a entrenar, validar o testejar.

S'ha augmentat també nombre de capes ocultes per tal de millorar la precisió amb més capes que iterin les dades, i finalment s'han utilitzat 100 iteracions, per així poder observar l'evolució de l'entrenament.

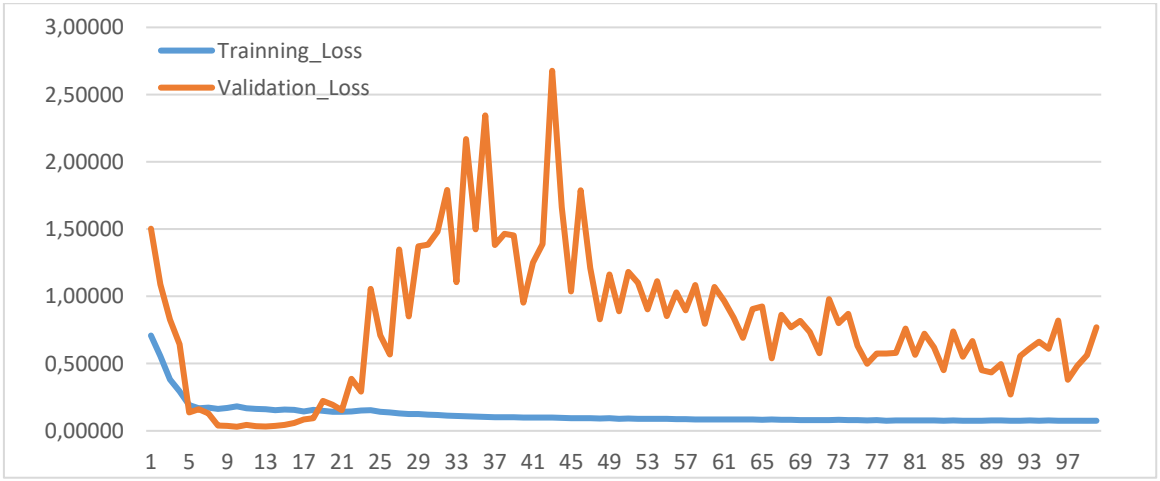


Figura 16

A primer cop d'ull ja es pot veure que la precisió augmenta molt utilitzant aquest set, en aquest exemple es mostren 100 iteracions amb un conjunt de 200.000 elements (més 50.000 per validació i test), a part de la bona precisió obtinguda aquest entrenament només ha trigat 17,5 hores, una gran diferència amb els resultats i temps obtinguts amb l'arbre de decisió on requeria més d'un dia per únicament 3.000 registres.

Si ens fixem en aquests resultats veurem que l'error en l'entrenament tot i que es va reduint al llarg de les 100 iteracions, ho fa de forma molt subtil a partir de les iteracions 8 a 10 quan tendeix a estabilitzar-se.

Pel que fa a la pèrdua en la validació passa una cosa semblant, fins a la iteracions 9-10 arriba al mínim, tot i que després passa a estabilitzar-se fins aproximadament a la 15 quan comença a tornar a augmentar amb un màxim cap a la 45 i després tornant a reduir-se i estabilitzar-se de mica en mica.

Així doncs podem afirmar que cap a la iteració 19 (quan l'error en la validació comença a augmentar i en aquest cas fins i tot supera l'error en el training) es produeix overfitting i seria el millor moment per aturar l'entrenament. És a dir, que amb 18 iteracions ja tindríem la xarxa entrenada per aquest conjunt de dades per tal d'obtenir una bona precisió.

Si ens fixem en els resultats de la precisió veure'm que acompanyen a l'anterior afirmació:

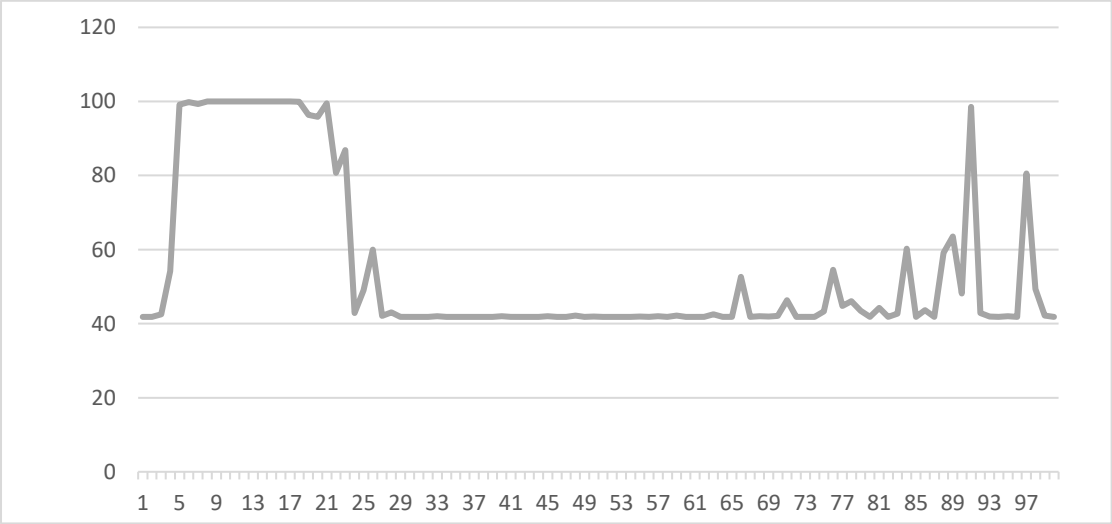


Figura 17

Arribem al màxim de precisió força ràpid, entre la iteració 8 i 10, a partir d'aquí s'estabilitza un temps i comença a reduir-se de forma evident cap a la 19. Així doncs, la millor opció sembla que és entrenar a la xarxa amb 10 primeres iteracions per al conjunt de dades utilitzat i amb la configuració establerta.

Podem observar el resultat d'executar l'algorisme amb únicament 10 iteracions en el següent registre de log:

```
Loading data...
Calculating metadata...
****************************************************
Starting training...
Epoch 1 of 10 took 514.684s
  training loss:              0.705333
  validation loss:            1.363318
  validation accuracy:              41.83 %
Epoch 2 of 10 took 515.311s
  training loss:              0.557132
  validation loss:            1.056276
  validation accuracy:              41.83 %
Epoch 3 of 10 took 518.309s
  training loss:              0.383574
  validation loss:            0.973544
  validation accuracy:              41.83 %
Epoch 4 of 10 took 516.505s
  training loss:              0.324998
```

validation loss: 0.832125
validation accuracy: 45.46 %
Epoch 5 of 10 took 516.721s
training loss: 0.206690
validation loss: 0.462493
validation accuracy: 69.10 %
Epoch 6 of 10 took 517.701s
training loss: 0.156720
validation loss: 0.202742
validation accuracy: 96.24 %
Epoch 7 of 10 took 518.137s
training loss: 0.150508
validation loss: 0.140284
validation accuracy: 98.68 %
Epoch 8 of 10 took 515.826s
training loss: 0.160898
validation loss: 0.031455
validation accuracy: 99.98 %
Epoch 9 of 10 took 515.784s
training loss: 0.150781
validation loss: 0.022945
validation accuracy: 99.99 %
Epoch 10 of 10 took 514.869s
training loss: 0.147939
validation loss: 0.026299
validation accuracy: 99.98 %
Training in 5164.12800002
Final results:
test loss: 0.025665
test accuracy: 99.99 %
Tests in 55.1600000858

Es pot veure que el temps d'entrenament ha estat de 5.164 segons, que correspon a 1 hora i 26 minuts, i d'1 minut per al test, a més, la precisió en el test ha estat d'un 99.99%, a diferència del 40% obtinguda a l'execució de 100 iteracions (o s'ha produït overfitting). Podem determinar, així, que aquesta és una de les millors configuracions per al conjunt de 200.000 registres d'eternament.

Execució amb diferents conjunts de dades

S'han realitzat experiments també amb 200.000 registres de training i 50.000 pel test i validació però amb conjunts de dades diferents. En concret s'han seleccionat les dades de la següent manera:

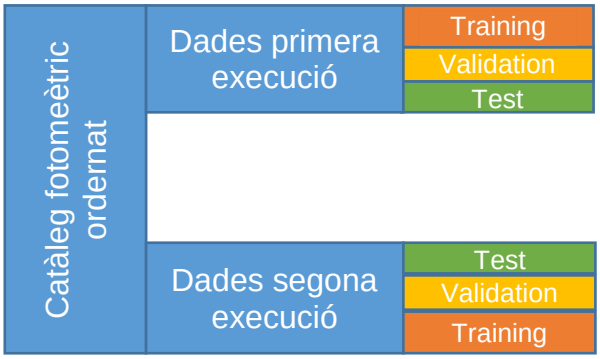


Figura 18

En el primer experiment de 200 mil registres s'han escollit les dades del inici del catàleg (recordem que conté 700 milions de registres), i en el segon, que ara es presenta, s'han escollit del final. Una característica important és que aquest catàleg es va crear cap a l'any 2.000 i des de llavors periòdicament si afegeixen nous registres, ara està a la release 13 i és molt possible que les tècniques i eines hagin millorat o com a mínim canviat, per exemple modificacions en als telescopis que en millorin la capacitat d'observació o fins i tot ús de nous telescopis, aquest canvis poden resultar en registres diferents i ens permetran veure com de precís amb altres dades, tot mantenint els mateixos paràmetres de configuració.

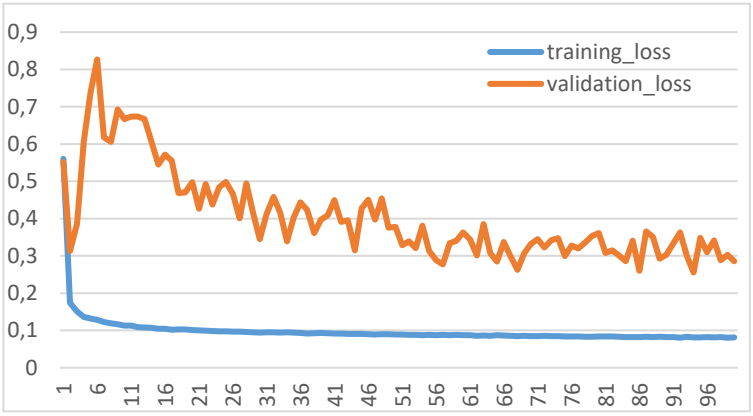


Figura 19

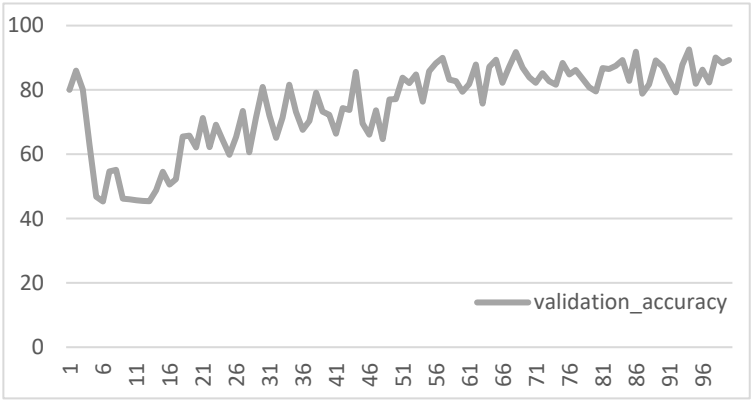


Figura 20

Podem veure que aquest cop el resultat és força diferent de l'anterior, arribem a precisions altes força més tard, fins a la iteració 57 on s'arriba a un 90%. Sembla clar que la xarxa estava en aquest cas configurada per obtenir el millor resultat amb les dades anteriors.

A més els resultats són força més inestables, partint d'una precisió molt alta a les primeres 3 iteracions que després cau i procedeix a ascendir novament. De la mateixa manera que en la precisió, l'error de la validació té un comportament erràtic, a les primeres iteracions descendeix ràpidament per, de sobte, augmentar i procedir a un descens en les següents. Això, aparentment, ens porta a 2 possibles escenaris, considerar que això és sobreentrenament i aturem la xarxa amb 3 iteracions, o bé, com que a partir de la 4a la tendència és a la baixa podríem allargar una mica més l'execució i veure fins a quin nou mínim arriba, aquesta segona opció sembla més interessant ja que amb 100 ja es poden veure que molts valors d'error inferiors als primers.

Per altre banda el que queda clar és que abans de res s'haurien d'ajustar els valors de configuració de la xarxa per tal de millorar els resultats i obtenir-ne de tant bons com en el primer experiment o millor encara, utilitzar molts més elements d'entrenament, els 200 mil registres emprats només conformen un 6% del total del catàleg.

Anàlisi del progrés de l'error en test

A continuació ens fixarem en un altre paràmetre, l'error en el test, si fins ara s'analitzava el grau d'aprenentatge mitjançant la reducció en l'error del training i la validació, ara s'ha afegit el càlcul del test en cada una de les iteracions, de manera que es pot observar quina hagués estat la precisió si s'hagués aturat l'execució en una determinada iteració.

La raó per realitzar aquest anàlisi és veure en quina mesura l'optimització de la configuració de la xarxa respecte les dades d'entrenament i validació pot resultar en valors pitjors respecte les dades de test o amb possibles nous valors, per tant com més semblants siguin millor entrenada considerarem la xarxa. En el següent gràfic es mostra l'error de validació i en el test de l'anterior experiment.

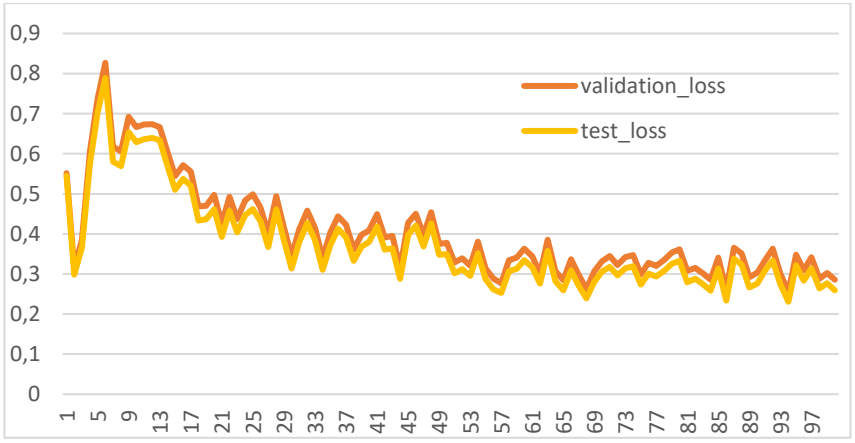


Figura 21

Es pot veure que la progressió és semblant, però amb certa diferència, on l'error de validació es superior al de test, això sembla contrari al que es pretenia demostrar. Revisant la precisió es pot veure un comportament equivalent:

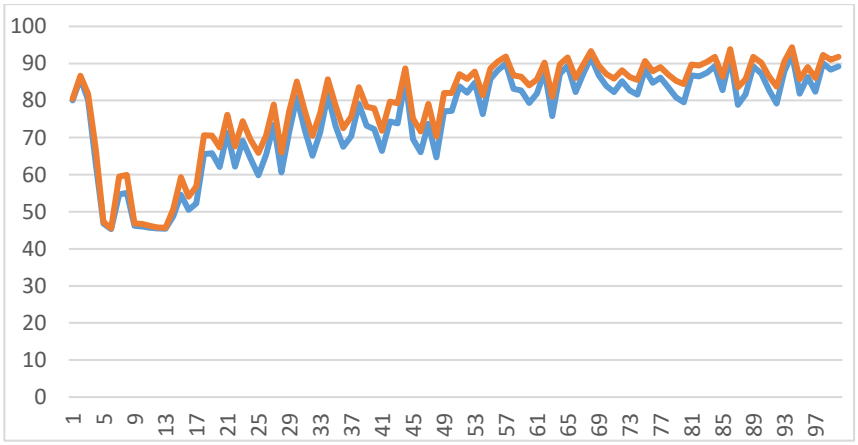


Figura 22

Aquests resultats semblen erronis i poc concloents ja que parteixen d'un experiment que ja s'ha vist no és òptim, s'ha entrenat amb dades que no han estat les utilitzades per optimitzar els valors de configuració i per això els resultats de l'entrenament són pitjors. A continuació mostrarem la diferència d'error en validació i test respecte les dades del primer experiment que si està optimitzat:

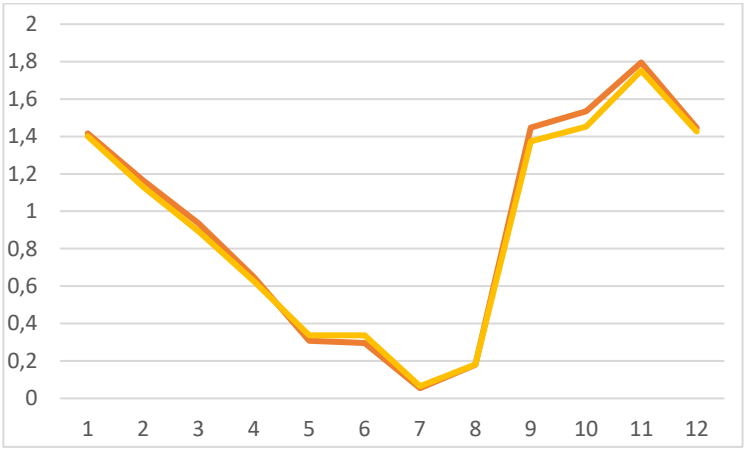


Figura 23

Se'n mostren les primeres iteracions, fins a la 12, ja que a partir de la 7 es comença a sobreentrenar. S'hi aprecia una diferència encara més petita (només es veu una diferència una mica superior a les iteracions 9 i 10, però on la xarxa ja està sobreentrenada), la configuració de la xarxa està optimitzada per obtenir els millors resultats amb aquest conjunt i sembla que les dades d'entrenament i test no són excessivament diferents, fins i tot, en algun cas els valors d'error són pràcticament iguals o s'inverteixen lleugerament.

6.2.2 Revisió dels diferents paràmetres de configuració

Per obtenir aquests resultats s'ha realitzat diferent variacions dels paràmetres de configuració i s'han determinat:

Capes ocultes: Com més capes menys iteracions seran necessàries per entrenar, però es cert que amb un nombre elevat de capes ocultes és més fàcil arribar a un sobreentrenament, sobretot depenen del nombre de registres utilitzat.

Iteracions: Com s'ha vist en els diferents experiments, s'ha d'anar amb compte i procurar aturar l'entrenament abans que iteracions excessives comencin a provocar sobreentrenament.

Mida del bloc: Amb mida del bloc ens referim als petits blocs que s'envien a la FNN partint el conjunt inicial en fragments més petits. Canviar aquest valor afecta a com es realitza l'entrenament, si utilitzem blocs molt petits, el mètode Iterate_minibatches haurà de fer moltes més iteracions i per tant pot alentir l'execució de cada epoch.

Unitats d'entrenament de cada capa: aquest paràmetre s'ha deixat fixat a 500 unitats ja que amb ell ja hem trobat una molt bona configuració.

7. Conclusions, lliçons apreses i treballs futurs

Tal com hem anat veien al llarg de l'anàlisi sembla que es confirma la Xarxa neuronal com a millor opció en el nostre problema, es a dir alhora de generar un classificador per les dades fotomètriques de SDSS.

Aquest data set té unes característiques que el fan idoni per la xarxa, com la gran quantitat de registres, n'hem utilitzat 300.000 però el set original disposa de més de 470 milions de registres únics. D'altre banda, per cada element n'hem analitzat la majoria de columnes, unes 506, tot i que les que tenen dades repetides s'han eliminat del càlcul aquestes en general són menys de 10, de manera que seguim treballant amb elements molt grans.

Tenint en compte que l'arbre analitza les columnes de forma intensiva aquest pot ser un dels motius del seu pobre rendiment. El càlcul dels punts de tall ha resultat ser una operació d'allò més costosa, hem de tenir en compte que per cada node es calculen els punts de tall d'entre tots els elements per cada una de les columnes requerides en aquell moment, per això en la primera iteració es quan més triga ja que ho fa per totes les columnes i totes les files del set. En les següents es va reduint de mica en mica el càlcul per la divisió d'elements en cada node, però això també afecta la possibilitat d'afegir profunditat a l'arbre.

Si per altre banda ens fixem en la precisió obtinguda, també podem veure que hi ha una millora de precisió per part de la FNN (amb 97,3% amb 3.000 registres) davant de l'arbre (amb 91% amb els mateixos 3.000 registres). L'arbre té certa tendència ascendent en les execucions fetes i potser amb més registres encara augmentaria una mica però l'important és que amb el mateix nombre de registres la FNN obté millor precisió.

Totes aquestes dades ens demostren que la Xarxa Neuronal resulta més eficient que l'Arbre de decisió com a classificador dels datasets escollits. I que tot i que la dificultat en configurar la Xarxa en ocasions suposa un handicap, permet personalitzar-la per trobar resultats més òptims que en el cas d'un arbre.

7.1 Comentaris sobre les dades

Si ens fixem en les etiquetes de cada registre, a les especificacions de les dades explica que els elements poden ser de 3 tipus Estrella, Galàxia i Cel, però si revisem el data set complet (el dels 470 milions de registres) ens trobarem el següent:

select count(1) from photoprimary where type = '3' : 208476804
select count(1) from photoprimary where type = '6' : 260562240
select count(1) from photoprimary where type = '0' : 11932

El elements amb tipus 3 i 6 (galàxia i estrella respectivament) representen el 99.997% dels registres, deixant com anecdòtica la presencia d'elements cel (0), aquesta és una de les raons per la qual en algunes iteracions la precisió es redueix de forma sobtada i l'error augmenta quan es troba amb elements cel, és important per tant intentar seleccionar sets d'entrenament que continguin tots les tipus existents.

Per altre banda la variabilitat ens els valors d'algunes columnes és molt alta. S'han detectat columnes amb un mínim de 0 o fins i tot negatius i màxims de 100.000 i entre ells valors de totes les escales i variacions. D'altres en canvi només variaven entre unes poques desenes de valors o decimals.

7.2 Comentaris i problemes del desenvolupament i sobre les tecnologies utilitzades

Sobre el desenvolupament

A l'execució de la FNN s'ha decidit deixar com a fixes alguns dels valors de configuració, un és el nombre d'unitats, els altres corresponen al drop d'elements en les capes que s'ha establert a 20% en la d'entrada i 50% a les ocultes. Els dos últims són el learning_rate i momentum de l'expressió d'updates. Els valors escollits són després de realitzar diferents proves i semblen els més adients per la majoria de configuracions. Ha requerit d'especial atenció el learning_rate ja que amb valors molt grans augmenta la mida dels passos d'update i resultava amb reduccions a la precisió de cada iteració.

Aquesta cura respecte els valors de configuració ha estat perquè han suposat un handicap durant tot el desenvolupament, la alteració indeguda d'algun dels molt valors de configuració sol resultar en valors dispars i erronis a la sortida de la xarxa, per això s'ha decidit fixar els que en general menys s'han hagut de tocar i que valen per totes les execucions fetes de la xarxa. De fet una part molt important del desenvolupament va ser la de trobar aquest conjunt de valors de configuració que donessin resultats coherents i que mostressin aprenentatge, com s'apunta, la xarxa neuronal és molt sensible al canvis de configuració.

Durant aquest projecte s'han desenvolupat dos algoritmes, i tot i que la FNN és molt més ràpida en executar els experiments també triga hores en finalitzar els que contenen més registres, això ha suposat haver d'invertir més d'un centenar d'hores d'espera per finalitzar experiments, veure'n el resultat, potser alterar la configuració i tornar-los a executar per obtenir-ne valors diferents. I tot i que l'impacte podria semblar superior en l'arbre que triga més en finalitzar, on més temps s'ha dedicat ha estat a la FNN per la comentada característica dels múltiples valors de configuració.

Sobre les tecnologies

El desenvolupament d'una nova FNN utilitzant les tecnologies descrites ha suposat tot un repte tecnològic, tant Theano, com sobretot Lasagne són llibreries força noves i en les que la comunitat encara esta creixent amb el que molts dels problemes trobats era complicat disposar de material de suport. Lasagne per exemple està tot just a la primera versió estable 0.1 llençada a l'estiu del 2015.

Aquest estadi inicial de Lasagne fa en ocasions complicada la troballa d'un error concret, en la majoria de situacions en que hi havia un problema només apareixia un resultat "nan" a la sortida de la FNN i no especificava si hi havia error o quin era. Després de dedicar moltes hores a investigar es va acudir a un conjunt de funcions de de troubleshooting (com ara el NanGuardMode) que han ajudat a posar controls d'errors i evitar futurs resultats incoherents.

Finalment es destaca la gran utilitat de la llibreria astropy per al tractament de fitxers fits a la xarxa neuronal. Utilitzar-lo ha permès treballar amb grans quantitats de dades sense problemes de sobrecarrega de memòria i la seva implementació és senzilla i clara amb molt de suport de la comunitat. Només s'ha de tenir en compte el càlcul de les metadades que si suposa una carrega de memòria inusual, però s'ha preparat per fer-ho únicament un cop per data set ja que els valors es guarden en un fitxer, i així deixant al usuari l'opció de recrear-lo o no segons valors del fitxer de configuració.

Comentaris finals

Més enllà del desenvolupament dels algoritmes que ja s'ha comentat, aquest projecte a requerit d'un molt important esforç d'anàlisi, una vegada escollit SDSS com a datasource s'ha requerit un anàlisi profund de tots els seus estudis i dades fins a trobar una part on es pogués desenvolupar un projecte de classificació que tingues utilitat real, aquest esforç queda reflectit a la planificació on si ha dedicat una bona porció del projecte.

Per altre banda, continuant amb la planificació, es destaca que s'ha pogut seguir en la majoria de casos, tot i així s'ha hagut de fer algun canvi al respecte en el desenvolupament i en l'anàlisi dels resultats, tal com s'ha comentat als documents de seguiment, per evitar entrar en un retard es va augmentar el nombre d'hores dedicades cada dia i d'aquesta manera es va poder arribar a la fita.

7.3 Treballs futurs

En fases futures d'aquest projecte es podrien millorar certs aspectes o afegir noves funcionalitats a continuació se'n detallen alguns:

- Activació de l'ús de la GPU i analitzar-ne els resultats respecte el obtinguts sense. Poder recórrer a la GPU de ben segur agilitzaria la FNN permetent fer més execucions.
- S'ha vist que en cada execució Theano triga força estona en iniciar el procés, això és perquè cada vegada ha de compilar-se en C, si el desenvolupament es realitzar amb, per exemple, Tensorflow, podríem millorar aquest punt.
- Obtenir l'anàlisi d'un expert, en astronomia o en els datsets de SDSS per eliminar de l'anàlisi algunes columnes que puguin ser innecessàries.
- Pel que fa a l'arbre es podria observar la precisió després de podar-lo per veure si millora, tot i que això augmentaria encara més el seu temps de creació.
- Seria interessant comprovar el rendiment de la FNN amb totes les dades del dataset, en el present projecte s'ha pogut treballar només amb un 6% del total.
- Aquest no representa un producte comercial llest per l'ús, per posar-lo a disposició d'usuaris convindria fer certs ajustos, com la millora del tractament d'errors en punts com la prevenció de dades errònies a la configuració. També seria necessària la creació d'una interfície més amigable que possibilites realitzar classificacions d'elements nous, ja que ara no es permet donat que l'objectiu es determinar la validesa de la xarxa cosa que s'ha fet amb l'entrenament i dades de test.

8. Glossari

SDSS: Sloan Digital Sky Survey, és un projecte d'investigació l'objectiu del qual és cartografiar els objectes de l'espai mitjançant imatges de l'espectre visible i del desplaçament al roig

Theano: És un llibreria Python que permet avaluar expressions matemàtiques utilitzant arrays multidimensionals de forma eficient. És utilitzada sobretot per desenvolupar xarxes neuronals, moltes vegades, con el present projecte, en conjunció amb Lasagne.

Lasagne: Es tracta d'una llibreria Python especialitzada en la creació de xarxes neuronals utilitzant Theano i que al ser dedicada a aquesta tasca en simplifica molt el desenvolupament.

Arbre de decisió: És un tipus d'algoritme de classificació que genera un arbre basat en regles i que genera nodes amb elements afins, es comença amb un node principal amb tots els elements i es va dividint en nodes fins arribar a una profunditat màxima o precisió requerides.

NN, Neural network: correspon a un tipus de sistema que inspirat en el sistema nerviós biològic intenta resoldre problemes propis del processament automàtic. L'estructura bàsica doncs, està composada per neurones interconnectades que col·laboren per produir una sortida.

Perceptró: En el camp de les xarxes neuronals sol tenir dues accepcions, la de neurona o un tipus de xarxa neuronal. En la segona definició es tracta de xarxes simples que permeten separar els elements de forma lineal, entre els que compleixen un criteri i els que no.

MLP, Multilayer perceptron: un perceptró multicapa és una xarxa neuronal composada per múltiples capes. A diferència del perceptró simple, aquest permet resoldre problemes que no són linealment separables.

FNN: Feedforward neural network, és el tipus de xarxa neuronal que primer es va crear, una de les principals característiques és que les diferents neurones que la composen con conformen un cicle. Hi ha FNN de tipus Perceptró i de tipus perceptró multicapa.

Convolutional neural network: Les xarxes convolucionals estan especialitzades en emular les neurones del còrtex visual, utilitza dos tipus de neurones, les convolucionals que s'utilitzen en una primera fase d'extracció de característiques juntament amb neurones de reducció del mostreig, en aquesta fase es va reduint la dimensionalitat de les dades que en la segona fase seran classificades per neurones de tipus perceptró.

Tensorflow: És una llibreria per a Machine learning creada per Google i que és més ràpida que Theano alhora de compilar.

9. Bibliografia

[1] Pàgina oficial de SDSS: http://www.sdss.org/dr13/irspec/ (25/09/2016)

[2] Master's thesis, Spectral classification using convolutional neural networks, Pavel Hála: https://arxiv.org/pdf/1412.8341.pdf (25/09/2016)

[3] Why are Neural Networks Sometimes Much More Accurate than Decision Trees: An Analysis on a Bio-Informatics Problem, Lawrence O. Hall, Xiaomei Liu, Kevin W. Bowyer, Robert Baneld, Univ. of South Florida: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.445.7&rep=rep1&type=pdf (11/12/2016)

[4] Python: https://www.python.org/ (28/09/2016)

[5] Anaconda: https://www.continuum.io/downloads (25/10/2016)

[6] Lasagne: http://lasagne.readthedocs.io/en/latest/index.html (26/10/2016)

[7] Theano: http://deeplearning.net/software/theano/index.html (28/10/2016)

[8] Visual Studio Code: https://code.visualstudio.com (28/09/2016)

[9] Astropy: http://docs.astropy.org/en/stable/index.html (25/10/2016)

[10] The Interactive FITS File Editor: http://heasarc.gsfc.nasa.gov/ftools/fv/ (28/09/2016)

[11] Apache Point Observatory: http://www.apo.nmsu.edu/ (30/09/2016)

[12] Sloan Foundation: http://www.sloan.org/ (30/09/2016)

[13] Article a Wikipedia de la funció Sigmoidea: https://es.wikipedia.org/wiki/Funci%C3%B3n_sigmoide (26/10/2016)

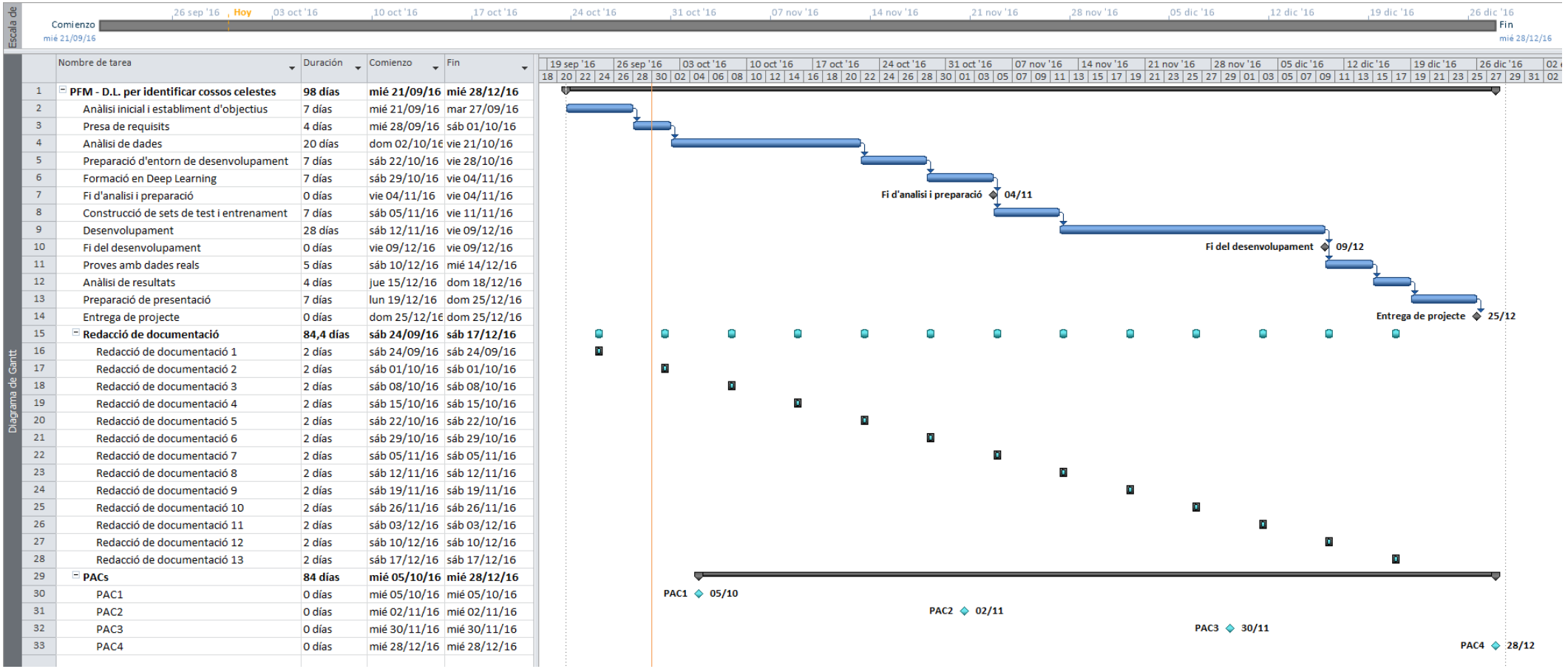
[14] Feedforward neural networks. What is a feedforward neural network?, Paul Boersma: http://www.fon.hum.uva.nl/praat/manual/Feedforward_neural_networks_1__What_is_a_feedforward_ne.html (26/10/2016)

[15] Convolutional Neural Network, Stanford University: http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/ (28/10/2016)

[16] Article a Wikipedia sobre fitxers Fits: https://en.wikipedia.org/wiki/FITS (01/11/16)

10. Annexos

10.1 Diagrama de Gantt complet



	Nombre de tarea	Duración	Comienzo	Fin
1	**PFM - D.L. per identificar cossos celestes**	**98 días**	**mié 21/09/16**	**mié 28/12/16**
2	Anàlisi inicial i establiment d'objectius	7 días	mié 21/09/16	mar 27/09/16
3	Presa de requisits	4 días	mié 28/09/16	sáb 01/10/16
4	Anàlisi de dades	20 días	dom 02/10/16	vie 21/10/16
5	Preparació d'entorn de desenvolupament	7 días	sáb 22/10/16	vie 28/10/16
6	Formació en Deep Learning	7 días	sáb 29/10/16	vie 04/11/16
7	Fi d'analisi i preparació	0 días	vie 04/11/16	vie 04/11/16
8	Construcció de sets de test i entrenament	7 días	sáb 05/11/16	vie 11/11/16
9	Desenvolupament	28 días	sáb 12/11/16	vie 09/12/16
10	Fi del desenvolupament	0 días	vie 09/12/16	vie 09/12/16
11	Proves amb dades reals	5 días	sáb 10/12/16	mié 14/12/16
12	Anàlisi de resultats	4 días	jue 15/12/16	dom 18/12/16
13	Preparació de presentació	7 días	lun 19/12/16	dom 25/12/16
14	Entrega de projecte	0 días	dom 25/12/16	dom 25/12/16
15	**Redacció de documentació**	**84,4 días**	**sáb 24/09/16**	**sáb 17/12/16**
16	Redacció de documentació 1	2 días	sáb 24/09/16	sáb 24/09/16
17	Redacció de documentació 2	2 días	sáb 01/10/16	sáb 01/10/16
18	Redacció de documentació 3	2 días	sáb 08/10/16	sáb 08/10/16
19	Redacció de documentació 4	2 días	sáb 15/10/16	sáb 15/10/16
20	Redacció de documentació 5	2 días	sáb 22/10/16	sáb 22/10/16
21	Redacció de documentació 6	2 días	sáb 29/10/16	sáb 29/10/16
22	Redacció de documentació 7	2 días	sáb 05/11/16	sáb 05/11/16
23	Redacció de documentació 8	2 días	sáb 12/11/16	sáb 12/11/16
24	Redacció de documentació 9	2 días	sáb 19/11/16	sáb 19/11/16
25	Redacció de documentació 10	2 días	sáb 26/11/16	sáb 26/11/16
26	Redacció de documentació 11	2 días	sáb 03/12/16	sáb 03/12/16
27	Redacció de documentació 12	2 días	sáb 10/12/16	sáb 10/12/16
28	Redacció de documentació 13	2 días	sáb 17/12/16	sáb 17/12/16
29	**PACs**	**84 días**	**mié 05/10/16**	**mié 28/12/16**
30	PAC1	0 días	mié 05/10/16	mié 05/10/16
31	PAC2	0 días	mié 02/11/16	mié 02/11/16
32	PAC3	0 días	mié 30/11/16	mié 30/11/16
33	PAC4	0 días	mié 28/12/16	mié 28/12/16

10.2 Descripció del catàleg fotomètric

A continuació la llista de columnes detallada del catàleg fotomètric de SDSS (en negre les columnes que no s'utilitzen):

Name	Type	Size	Unit	Summary
objID	bigint	8		Unique SDSS identifier composed from [skyVersion,rerun,run,camcol,field,obj].
skyVersion	tinyint	1		Layer of catalog (currently only one layer, 0; 0-15 available)
run	smallint	2		Run number
rerun	smallint	2		Rerun number
camcol	tinyint	1		Camera column
field	smallint	2		Field number
obj	smallint	2		The object id within a field. Usually changes between reruns of the same field
mode	tinyint	1		1: primary, 2: secondary, 3: other
nChild	smallint	2		Number of children if this is a composite object that has been deblended. BRIGHT (in a flags sense) objects also have nchild == 1, the non-BRIGHT sibling.
type	smallint	2		Type classification of the object (star, galaxy, cosmic ray, etc.)
clean	int	4		Clean photometry flag (1=clean, 0=unclean).
probPSF	real	4		Probability that the object is a star. Currently 0 if type == 3 (galaxy), 1 if type == 6 (star).
insideMask	tinyint	1		Flag to indicate whether object is inside a mask and why
flags	bigint	8		Photo Object Attribute Flags
rowc	real	4	pix	Row center position (r-band coordinates)
rowcErr	real	4	pix	Row center position error (r-band coordinates)
colc	real	4	pix	Column center position (r-band coordinates)
colcErr	real	4	pix	Column center position error (r-band coordinates)
rowv	real	4	deg/day	Row-component of object's velocity
rowvErr	real	4	deg/day	Row-component of object's velocity error
colv	real	4	deg/day	Column-component of object's velocity
colvErr	real	4	deg/day	Column-component of object's velocity error
rowc_u	real	4	pix	Row center, u-band
rowc_g	real	4	pix	Row center, g-band
rowc_r	real	4	pix	Row center, r-band
rowc_i	real	4	pix	Row center, i-band
rowc_z	real	4	pix	Row center, z-band
rowcErr_u	real	4	pix	ERROR Row center error, u-band
rowcErr_g	real	4	pix	ERROR Row center error, g-band
rowcErr_r	real	4	pix	ERROR Row center error, r-band
rowcErr_i	real	4	pix	ERROR Row center error, i-band
rowcErr_z	real	4	pix	ERROR Row center error, z-band
colc_u	real	4	pix	Column center, u-band
colc_g	real	4	pix	Column center, g-band
colc_r	real	4	pix	Column center, r-band
colc_i	real	4	pix	Column center, i-band
colc_z	real	4	pix	Column center, z-band

colcErr_u	real	4	pix	Column center error, u-band
colcErr_g	real	4	pix	Column center error, g-band
colcErr_r	real	4	pix	Column center error, r-band
colcErr_i	real	4	pix	Column center error, i-band
colcErr_z	real	4	pix	Column center error, z-band
sky_u	real	4	nanomaggies/arcsec^2	Sky flux at the center of object (allowing for siblings if blended).
sky_g	real	4	nanomaggies/arcsec^2	Sky flux at the center of object (allowing for siblings if blended).
sky_r	real	4	nanomaggies/arcsec^2	Sky flux at the center of object (allowing for siblings if blended).
sky_i	real	4	nanomaggies/arcsec^2	Sky flux at the center of object (allowing for siblings if blended).
sky_z	real	4	nanomaggies/arcsec^2	Sky flux at the center of object (allowing for siblings if blended).
skyIvar_u	real	4	nanomaggies/arcsec^2	Sky flux inverse variance
skyIvar_g	real	4	nanomaggies/arcsec^2	Sky flux inverse variance
skyIvar_r	real	4	nanomaggies/arcsec^2	Sky flux inverse variance
skyIvar_i	real	4	nanomaggies/arcsec^2	Sky flux inverse variance
skyIvar_z	real	4	nanomaggies/arcsec^2	Sky flux inverse variance
psfMag_u	real	4	mag	PSF magnitude
psfMag_g	real	4	mag	PSF magnitude
psfMag_r	real	4	mag	PSF magnitude
psfMag_i	real	4	mag	PSF magnitude
psfMag_z	real	4	mag	PSF magnitude
psfMagErr_u	real	4	mag	PSF magnitude error
psfMagErr_g	real	4	mag	PSF magnitude error
psfMagErr_r	real	4	mag	PSF magnitude error
psfMagErr_i	real	4	mag	PSF magnitude error
psfMagErr_z	real	4	mag	PSF magnitude error
fiberMag_u	real	4	mag	Magnitude in 3 arcsec diameter fiber radius
fiberMag_g	real	4	mag	Magnitude in 3 arcsec diameter fiber radius
fiberMag_r	real	4	mag	Magnitude in 3 arcsec diameter fiber radius
fiberMag_i	real	4	mag	Magnitude in 3 arcsec diameter fiber radius
fiberMag_z	real	4	mag	Magnitude in 3 arcsec diameter fiber radius
fiberMagErr_u	real	4	mag	Error in magnitude in 3 arcsec diameter fiber radius
fiberMagErr_g	real	4	mag	Error in magnitude in 3 arcsec diameter fiber radius
fiberMagErr_r	real	4	mag	Error in magnitude in 3 arcsec diameter fiber radius
fiberMagErr_i	real	4	mag	Error in magnitude in 3 arcsec diameter fiber radius
fiberMagErr_z	real	4	mag	Error in magnitude in 3 arcsec diameter fiber radius
fiber2Mag_u	real	4	mag	Magnitude in 2 arcsec diameter fiber radius
fiber2Mag_g	real	4	mag	Magnitude in 2 arcsec diameter fiber radius
fiber2Mag_r	real	4	mag	Magnitude in 2 arcsec diameter fiber radius
fiber2Mag_i	real	4	mag	Magnitude in 2 arcsec diameter fiber radius
fiber2Mag_z	real	4	mag	Magnitude in 2 arcsec diameter fiber radius
fiber2MagErr_u	real	4	mag	Error in magnitude in 2 arcsec diameter fiber radius
fiber2MagErr_g	real	4	mag	Error in magnitude in 2 arcsec diameter fiber radius
fiber2MagErr_r	real	4	mag	Error in magnitude in 2 arcsec diameter fiber radius
fiber2MagErr_i	real	4	mag	Error in magnitude in 2 arcsec diameter fiber radius

fiber2MagErr_z	real	4	mag	Error in magnitude in 2 arcsec diameter fiber radius
petroMag_u	real	4	mag	Petrosian magnitude
petroMag_g	real	4	mag	Petrosian magnitude
petroMag_r	real	4	mag	Petrosian magnitude
petroMag_i	real	4	mag	Petrosian magnitude
petroMag_z	real	4	mag	Petrosian magnitude
petroMagErr_u	real	4	mag	Petrosian magnitude error
petroMagErr_g	real	4	mag	Petrosian magnitude error
petroMagErr_r	real	4	mag	Petrosian magnitude error
petroMagErr_i	real	4	mag	Petrosian magnitude error
petroMagErr_z	real	4	mag	Petrosian magnitude error
psfFlux_u	real	4	nanomaggies	PSF flux
psfFlux_g	real	4	nanomaggies	PSF flux
psfFlux_r	real	4	nanomaggies	PSF flux
psfFlux_i	real	4	nanomaggies	PSF flux
psfFlux_z	real	4	nanomaggies	PSF flux
psfFluxIvar_u	real	4	nanomaggies^{-2}	PSF flux inverse variance
psfFluxIvar_g	real	4	nanomaggies^{-2}	PSF flux inverse variance
psfFluxIvar_r	real	4	nanomaggies^{-2}	PSF flux inverse variance
psfFluxIvar_i	real	4	nanomaggies^{-2}	PSF flux inverse variance
psfFluxIvar_z	real	4	nanomaggies^{-2}	PSF flux inverse variance
fiberFlux_u	real	4	nanomaggies	Flux in 3 arcsec diameter fiber radius
fiberFlux_g	real	4	nanomaggies	Flux in 3 arcsec diameter fiber radius
fiberFlux_r	real	4	nanomaggies	Flux in 3 arcsec diameter fiber radius
fiberFlux_i	real	4	nanomaggies	Flux in 3 arcsec diameter fiber radius
fiberFlux_z	real	4	nanomaggies	Flux in 3 arcsec diameter fiber radius
fiberFluxIvar_u	real	4	nanomaggies^{-2}	Inverse variance in flux in 3 arcsec diameter fiber radius
fiberFluxIvar_g	real	4	nanomaggies^{-2}	Inverse variance in flux in 3 arcsec diameter fiber radius
fiberFluxIvar_r	real	4	nanomaggies^{-2}	Inverse variance in flux in 3 arcsec diameter fiber radius
fiberFluxIvar_i	real	4	nanomaggies^{-2}	Inverse variance in flux in 3 arcsec diameter fiber radius
fiberFluxIvar_z	real	4	nanomaggies^{-2}	Inverse variance in flux in 3 arcsec diameter fiber radius
fiber2Flux_u	real	4	nanomaggies	Flux in 2 arcsec diameter fiber radius
fiber2Flux_g	real	4	nanomaggies	Flux in 2 arcsec diameter fiber radius
fiber2Flux_r	real	4	nanomaggies	Flux in 2 arcsec diameter fiber radius
fiber2Flux_i	real	4	nanomaggies	Flux in 2 arcsec diameter fiber radius
fiber2Flux_z	real	4	nanomaggies	Flux in 2 arcsec diameter fiber radius
fiber2FluxIvar_u	real	4	nanomaggies^{-2}	Inverse variance in flux in 2 arcsec diameter fiber radius
fiber2FluxIvar_g	real	4	nanomaggies^{-2}	Inverse variance in flux in 2 arcsec diameter fiber radius
fiber2FluxIvar_r	real	4	nanomaggies^{-2}	Inverse variance in flux in 2 arcsec diameter fiber radius
fiber2FluxIvar_i	real	4	nanomaggies^{-2}	Inverse variance in flux in 2 arcsec diameter fiber radius
fiber2FluxIvar_z	real	4	nanomaggies^{-2}	Inverse variance in flux in 2 arcsec diameter fiber radius
petroFlux_u	real	4	nanomaggies	Petrosian flux
petroFlux_g	real	4	nanomaggies	Petrosian flux
petroFlux_r	real	4	nanomaggies	Petrosian flux
petroFlux_i	real	4	nanomaggies	Petrosian flux
petroFlux_z	real	4	nanomaggies	Petrosian flux

petroFluxIvar_u	real	4	nanomaggies^{-2}	Petrosian flux inverse variance
petroFluxIvar_g	real	4	nanomaggies^{-2}	Petrosian flux inverse variance
petroFluxIvar_r	real	4	nanomaggies^{-2}	Petrosian flux inverse variance
petroFluxIvar_i	real	4	nanomaggies^{-2}	Petrosian flux inverse variance
petroFluxIvar_z	real	4	nanomaggies^{-2}	Petrosian flux inverse variance
petroRad_u	real	4	arcsec	Petrosian radius
petroRad_g	real	4	arcsec	Petrosian radius
petroRad_r	real	4	arcsec	Petrosian radius
petroRad_i	real	4	arcsec	Petrosian radius
petroRad_z	real	4	arcsec	Petrosian radius
petroRadErr_u	real	4	arcsec	Petrosian radius error
petroRadErr_g	real	4	arcsec	Petrosian radius error
petroRadErr_r	real	4	arcsec	Petrosian radius error
petroRadErr_i	real	4	arcsec	Petrosian radius error
petroRadErr_z	real	4	arcsec	Petrosian radius error
petroR50_u	real	4	arcsec	Radius containing 50% of Petrosian flux
petroR50_g	real	4	arcsec	Radius containing 50% of Petrosian flux
petroR50_r	real	4	arcsec	Radius containing 50% of Petrosian flux
petroR50_i	real	4	arcsec	Radius containing 50% of Petrosian flux
petroR50_z	real	4	arcsec	Radius containing 50% of Petrosian flux
petroR50Err_u	real	4	arcsec	Error in radius with 50% of Petrosian flux error
petroR50Err_g	real	4	arcsec	Error in radius with 50% of Petrosian flux error
petroR50Err_r	real	4	arcsec	Error in radius with 50% of Petrosian flux error
petroR50Err_i	real	4	arcsec	Error in radius with 50% of Petrosian flux error
petroR50Err_z	real	4	arcsec	Error in radius with 50% of Petrosian flux error
petroR90_u	real	4	arcsec	Radius containing 90% of Petrosian flux
petroR90_g	real	4	arcsec	Radius containing 90% of Petrosian flux
petroR90_r	real	4	arcsec	Radius containing 90% of Petrosian flux
petroR90_i	real	4	arcsec	Radius containing 90% of Petrosian flux
petroR90_z	real	4	arcsec	Radius containing 90% of Petrosian flux
petroR90Err_u	real	4	arcsec	Error in radius with 90% of Petrosian flux error
petroR90Err_g	real	4	arcsec	Error in radius with 90% of Petrosian flux error
petroR90Err_r	real	4	arcsec	Error in radius with 90% of Petrosian flux error
petroR90Err_i	real	4	arcsec	Error in radius with 90% of Petrosian flux error
petroR90Err_z	real	4	arcsec	Error in radius with 90% of Petrosian flux error
q_u	real	4		Stokes Q parameter
q_g	real	4		Stokes Q parameter
q_r	real	4		Stokes Q parameter
q_i	real	4		Stokes Q parameter
q_z	real	4		Stokes Q parameter
qErr_u	real	4		Stokes Q parameter error
qErr_g	real	4		Stokes Q parameter error
qErr_r	real	4		Stokes Q parameter error
qErr_i	real	4		Stokes Q parameter error
qErr_z	real	4		Stokes Q parameter error
u_u	real	4		Stokes U parameter

u_g	real	4	Stokes U parameter
u_r	real	4	Stokes U parameter
u_i	real	4	Stokes U parameter
u_z	real	4	Stokes U parameter
uErr_u	real	4	Stokes U parameter error
uErr_g	real	4	Stokes U parameter error
uErr_r	real	4	Stokes U parameter error
uErr_i	real	4	Stokes U parameter error
uErr_z	real	4	Stokes U parameter error
mE1_u	real	4	Adaptive E1 shape measure (pixel coordinates)
mE1_g	real	4	Adaptive E1 shape measure (pixel coordinates)
mE1_r	real	4	Adaptive E1 shape measure (pixel coordinates)
mE1_i	real	4	Adaptive E1 shape measure (pixel coordinates)
mE1_z	real	4	Adaptive E1 shape measure (pixel coordinates)
mE2_u	real	4	Adaptive E2 shape measure (pixel coordinates)
mE2_g	real	4	Adaptive E2 shape measure (pixel coordinates)
mE2_r	real	4	Adaptive E2 shape measure (pixel coordinates)
mE2_i	real	4	Adaptive E2 shape measure (pixel coordinates)
mE2_z	real	4	Adaptive E2 shape measure (pixel coordinates)
mE1E1Err_u	real	4	Covariance in E1/E1 shape measure (pixel coordinates)
mE1E1Err_g	real	4	Covariance in E1/E1 shape measure (pixel coordinates)
mE1E1Err_r	real	4	Covariance in E1/E1 shape measure (pixel coordinates)
mE1E1Err_i	real	4	Covariance in E1/E1 shape measure (pixel coordinates)
mE1E1Err_z	real	4	Covariance in E1/E1 shape measure (pixel coordinates)
mE1E2Err_u	real	4	Covariance in E1/E2 shape measure (pixel coordinates)
mE1E2Err_g	real	4	Covariance in E1/E2 shape measure (pixel coordinates)
mE1E2Err_r	real	4	Covariance in E1/E2 shape measure (pixel coordinates)
mE1E2Err_i	real	4	Covariance in E1/E2 shape measure (pixel coordinates)
mE1E2Err_z	real	4	Covariance in E1/E2 shape measure (pixel coordinates)
mE2E2Err_u	real	4	Covariance in E2/E2 shape measure (pixel coordinates)
mE2E2Err_g	real	4	Covariance in E2/E2 shape measure (pixel coordinates)
mE2E2Err_r	real	4	Covariance in E2/E2 shape measure (pixel coordinates)
mE2E2Err_i	real	4	Covariance in E2/E2 shape measure (pixel coordinates)
mE2E2Err_z	real	4	Covariance in E2/E2 shape measure (pixel coordinates)
mRrCc_u	real	4	Adaptive (+) (pixel coordinates)
mRrCc_g	real	4	Adaptive (+) (pixel coordinates)
mRrCc_r	real	4	Adaptive (+) (pixel coordinates)
mRrCc_i	real	4	Adaptive (+) (pixel coordinates)
mRrCc_z	real	4	Adaptive (+) (pixel coordinates)
mRrCcErr_u	real	4	Error in adaptive (+) (pixel coordinates)
mRrCcErr_g	real	4	Error in adaptive (+) (pixel coordinates)
mRrCcErr_r	real	4	Error in adaptive (+) (pixel coordinates)
mRrCcErr_i	real	4	Error in adaptive (+) (pixel coordinates)
mRrCcErr_z	real	4	Error in adaptive (+) (pixel coordinates)
mCr4_u	real	4	Adaptive fourth moment of object (pixel coordinates)
mCr4_g	real	4	Adaptive fourth moment of object (pixel coordinates)

mCr4_r	real	4		Adaptive fourth moment of object (pixel coordinates)
mCr4_i	real	4		Adaptive fourth moment of object (pixel coordinates)
mCr4_z	real	4		Adaptive fourth moment of object (pixel coordinates)
mE1PSF_u	real	4		Adaptive E1 for PSF (pixel coordinates)
mE1PSF_g	real	4		Adaptive E1 for PSF (pixel coordinates)
mE1PSF_r	real	4		Adaptive E1 for PSF (pixel coordinates)
mE1PSF_i	real	4		Adaptive E1 for PSF (pixel coordinates)
mE1PSF_z	real	4		Adaptive E1 for PSF (pixel coordinates)
mE2PSF_u	real	4		Adaptive E2 for PSF (pixel coordinates)
mE2PSF_g	real	4		Adaptive E2 for PSF (pixel coordinates)
mE2PSF_r	real	4		Adaptive E2 for PSF (pixel coordinates)
mE2PSF_i	real	4		Adaptive E2 for PSF (pixel coordinates)
mE2PSF_z	real	4		Adaptive E2 for PSF (pixel coordinates)
mRrCcPSF_u	real	4		Adaptive (+) for PSF (pixel coordinates)
mRrCcPSF_g	real	4		Adaptive (+) for PSF (pixel coordinates)
mRrCcPSF_r	real	4		Adaptive (+) for PSF (pixel coordinates)
mRrCcPSF_i	real	4		Adaptive (+) for PSF (pixel coordinates)
mRrCcPSF_z	real	4		Adaptive (+) for PSF (pixel coordinates)
mCr4PSF_u	real	4		Adaptive fourth moment for PSF (pixel coordinates)
mCr4PSF_g	real	4		Adaptive fourth moment for PSF (pixel coordinates)
mCr4PSF_r	real	4		Adaptive fourth moment for PSF (pixel coordinates)
mCr4PSF_i	real	4		Adaptive fourth moment for PSF (pixel coordinates)
mCr4PSF_z	real	4		Adaptive fourth moment for PSF (pixel coordinates)
deVRad_u	real	4	arcsec	de Vaucouleurs fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
deVRad_g	real	4	arcsec	de Vaucouleurs fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
deVRad_r	real	4	arcsec	de Vaucouleurs fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
deVRad_i	real	4	arcsec	de Vaucouleurs fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
deVRad_z	real	4	arcsec	de Vaucouleurs fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
deVRadErr_u	real	4	arcsec	Error in de Vaucouleurs fit scale radius error
deVRadErr_g	real	4	arcsec	Error in de Vaucouleurs fit scale radius error
deVRadErr_r	real	4	arcsec	Error in de Vaucouleurs fit scale radius error
deVRadErr_i	real	4	arcsec	Error in de Vaucouleurs fit scale radius error
deVRadErr_z	real	4	arcsec	Error in de Vaucouleurs fit scale radius error
deVAB_u	real	4		de Vaucouleurs fit b/a
deVAB_g	real	4		de Vaucouleurs fit b/a
deVAB_r	real	4		de Vaucouleurs fit b/a
deVAB_i	real	4		de Vaucouleurs fit b/a
deVAB_z	real	4		de Vaucouleurs fit b/a
deVABErr_u	real	4		de Vaucouleurs fit b/a error
deVABErr_g	real	4		de Vaucouleurs fit b/a error
deVABErr_r	real	4		de Vaucouleurs fit b/a error

deVABErr_i	real	4		de Vaucouleurs fit b/a error
deVABErr_z	real	4		de Vaucouleurs fit b/a error
deVPhi_u	real	4	deg	de Vaucouleurs fit position angle (+N thru E)
deVPhi_g	real	4	deg	de Vaucouleurs fit position angle (+N thru E)
deVPhi_r	real	4	deg	de Vaucouleurs fit position angle (+N thru E)
deVPhi_i	real	4	deg	de Vaucouleurs fit position angle (+N thru E)
deVPhi_z	real	4	deg	de Vaucouleurs fit position angle (+N thru E)
deVMag_u	real	4	mag	de Vaucouleurs magnitude fit
deVMag_g	real	4	mag	de Vaucouleurs magnitude fit
deVMag_r	real	4	mag	de Vaucouleurs magnitude fit
deVMag_i	real	4	mag	de Vaucouleurs magnitude fit
deVMag_z	real	4	mag	de Vaucouleurs magnitude fit
deVMagErr_u	real	4	mag	de Vaucouleurs magnitude fit error
deVMagErr_g	real	4	mag	de Vaucouleurs magnitude fit error
deVMagErr_r	real	4	mag	de Vaucouleurs magnitude fit error
deVMagErr_i	real	4	mag	de Vaucouleurs magnitude fit error
deVMagErr_z	real	4	mag	de Vaucouleurs magnitude fit error
deVFlux_u	real	4	nanomaggies	de Vaucouleurs magnitude fit
deVFlux_g	real	4	nanomaggies	de Vaucouleurs magnitude fit
deVFlux_r	real	4	nanomaggies	de Vaucouleurs magnitude fit
deVFlux_i	real	4	nanomaggies	de Vaucouleurs magnitude fit
deVFlux_z	real	4	nanomaggies	de Vaucouleurs magnitude fit
deVFluxIvar_u	real	4	nanomaggies^{-2}	de Vaucouleurs magnitude fit inverse variance
deVFluxIvar_g	real	4	nanomaggies^{-2}	de Vaucouleurs magnitude fit inverse variance
deVFluxIvar_r	real	4	nanomaggies^{-2}	de Vaucouleurs magnitude fit inverse variance
deVFluxIvar_i	real	4	nanomaggies^{-2}	de Vaucouleurs magnitude fit inverse variance
deVFluxIvar_z	real	4	nanomaggies^{-2}	de Vaucouleurs magnitude fit inverse variance
expRad_u	real	4	arcsec	Exponential fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
expRad_g	real	4	arcsec	Exponential fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
expRad_r	real	4	arcsec	Exponential fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
expRad_i	real	4	arcsec	Exponential fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
expRad_z	real	4	arcsec	Exponential fit scale radius, here defined to be the same as the half-light radius, also called the effective radius.
expRadErr_u	real	4	arcsec	Exponential fit scale radius error
expRadErr_g	real	4	arcsec	Exponential fit scale radius error
expRadErr_r	real	4	arcsec	Exponential fit scale radius error
expRadErr_i	real	4	arcsec	Exponential fit scale radius error
expRadErr_z	real	4	arcsec	Exponential fit scale radius error
expAB_u	real	4		Exponential fit b/a
expAB_g	real	4		Exponential fit b/a
expAB_r	real	4		Exponential fit b/a
expAB_i	real	4		Exponential fit b/a
expAB_z	real	4		Exponential fit b/a
expABErr_u	real	4		Exponential fit b/a
expABErr_g	real	4		Exponential fit b/a

expABErr_r	real	4		Exponential fit b/a
expABErr_i	real	4		Exponential fit b/a
expABErr_z	real	4		Exponential fit b/a
expPhi_u	real	4	deg	Exponential fit position angle (+N thru E)
expPhi_g	real	4	deg	Exponential fit position angle (+N thru E)
expPhi_r	real	4	deg	Exponential fit position angle (+N thru E)
expPhi_i	real	4	deg	Exponential fit position angle (+N thru E)
expPhi_z	real	4	deg	Exponential fit position angle (+N thru E)
expMag_u	real	4	mag	Exponential fit magnitude
expMag_g	real	4	mag	Exponential fit magnitude
expMag_r	real	4	mag	Exponential fit magnitude
expMag_i	real	4	mag	Exponential fit magnitude
expMag_z	real	4	mag	Exponential fit magnitude
expMagErr_u	real	4	mag	Exponential fit magnitude error
expMagErr_g	real	4	mag	Exponential fit magnitude error
expMagErr_r	real	4	mag	Exponential fit magnitude error
expMagErr_i	real	4	mag	Exponential fit magnitude error
expMagErr_z	real	4	mag	Exponential fit magnitude error
modelMag_u	real	4	mag	better of DeV/Exp magnitude fit
modelMag_g	real	4	mag	better of DeV/Exp magnitude fit
modelMag_r	real	4	mag	better of DeV/Exp magnitude fit
modelMag_i	real	4	mag	better of DeV/Exp magnitude fit
modelMag_z	real	4	mag	better of DeV/Exp magnitude fit
modelMagErr_u	real	4	mag	Error in better of DeV/Exp magnitude fit
modelMagErr_g	real	4	mag	Error in better of DeV/Exp magnitude fit
modelMagErr_r	real	4	mag	Error in better of DeV/Exp magnitude fit
modelMagErr_i	real	4	mag	Error in better of DeV/Exp magnitude fit
modelMagErr_z	real	4	mag	Error in better of DeV/Exp magnitude fit
cModelMag_u	real	4	mag	DeV+Exp magnitude
cModelMag_g	real	4	mag	DeV+Exp magnitude
cModelMag_r	real	4	mag	DeV+Exp magnitude
cModelMag_i	real	4	mag	DeV+Exp magnitude
cModelMag_z	real	4	mag	DeV+Exp magnitude
cModelMagErr_u	real	4	mag	DeV+Exp magnitude error
cModelMagErr_g	real	4	mag	DeV+Exp magnitude error
cModelMagErr_r	real	4	mag	DeV+Exp magnitude error
cModelMagErr_i	real	4	mag	DeV+Exp magnitude error
cModelMagErr_z	real	4	mag	DeV+Exp magnitude error
expFlux_u	real	4	nanomaggies	Exponential fit flux
expFlux_g	real	4	nanomaggies	Exponential fit flux
expFlux_r	real	4	nanomaggies	Exponential fit flux
expFlux_i	real	4	nanomaggies	Exponential fit flux
expFlux_z	real	4	nanomaggies	Exponential fit flux
expFluxIvar_u	real	4	nanomaggies^{-2}	Exponential fit flux inverse variance
expFluxIvar_g	real	4	nanomaggies^{-2}	Exponential fit flux inverse variance
expFluxIvar_r	real	4	nanomaggies^{-2}	Exponential fit flux inverse variance

expFluxIvar_i	real	4	nanomaggies^{-2}	Exponential fit flux inverse variance
expFluxIvar_z	real	4	nanomaggies^{-2}	Exponential fit flux inverse variance
modelFlux_u	real	4	nanomaggies	better of DeV/Exp flux fit
modelFlux_g	real	4	nanomaggies	better of DeV/Exp flux fit
modelFlux_r	real	4	nanomaggies	better of DeV/Exp flux fit
modelFlux_i	real	4	nanomaggies	better of DeV/Exp flux fit
modelFlux_z	real	4	nanomaggies	better of DeV/Exp flux fit
modelFluxIvar_u	real	4	nanomaggies^{-2}	Inverse variance in better of DeV/Exp flux fit
modelFluxIvar_g	real	4	nanomaggies^{-2}	Inverse variance in better of DeV/Exp flux fit
modelFluxIvar_r	real	4	nanomaggies^{-2}	Inverse variance in better of DeV/Exp flux fit
modelFluxIvar_i	real	4	nanomaggies^{-2}	Inverse variance in better of DeV/Exp flux fit
modelFluxIvar_z	real	4	nanomaggies^{-2}	Inverse variance in better of DeV/Exp flux fit
cModelFlux_u	real	4	nanomaggies	better of DeV+Exp flux
cModelFlux_g	real	4	nanomaggies	better of DeV+Exp flux
cModelFlux_r	real	4	nanomaggies	better of DeV+Exp flux
cModelFlux_i	real	4	nanomaggies	better of DeV+Exp flux
cModelFlux_z	real	4	nanomaggies	better of DeV+Exp flux
cModelFluxIvar_u	real	4	nanomaggies^{-2}	Inverse variance in DeV+Exp flux fit
cModelFluxIvar_g	real	4	nanomaggies^{-2}	Inverse variance in DeV+Exp flux fit
cModelFluxIvar_r	real	4	nanomaggies^{-2}	Inverse variance in DeV+Exp flux fit
cModelFluxIvar_i	real	4	nanomaggies^{-2}	Inverse variance in DeV+Exp flux fit
cModelFluxIvar_z	real	4	nanomaggies^{-2}	Inverse variance in DeV+Exp flux fit
aperFlux7_u	real	4	nanomaggies	Aperture flux within 7.3 arcsec
aperFlux7_g	real	4	nanomaggies	Aperture flux within 7.3 arcsec
aperFlux7_r	real	4	nanomaggies	Aperture flux within 7.3 arcsec
aperFlux7_i	real	4	nanomaggies	Aperture flux within 7.3 arcsec
aperFlux7_z	real	4	nanomaggies	Aperture flux within 7.3 arcsec
aperFlux7Ivar_u	real	4	nanomaggies^{-2}	Inverse variance of aperture flux within 7.3 arcsec
aperFlux7Ivar_g	real	4	nanomaggies^{-2}	Inverse variance of aperture flux within 7.3 arcsec
aperFlux7Ivar_r	real	4	nanomaggies^{-2}	Inverse variance of aperture flux within 7.3 arcsec
aperFlux7Ivar_i	real	4	nanomaggies^{-2}	Inverse variance of aperture flux within 7.3 arcsec
aperFlux7Ivar_z	real	4	nanomaggies^{-2}	Inverse variance of aperture flux within 7.3 arcsec
lnLStar_u	real	4		Star ln(likelihood)
lnLStar_g	real	4		Star ln(likelihood)
lnLStar_r	real	4		Star ln(likelihood)
lnLStar_i	real	4		Star ln(likelihood)
lnLStar_z	real	4		Star ln(likelihood)
lnLExp_u	real	4		Exponential disk fit ln(likelihood)
lnLExp_g	real	4		Exponential disk fit ln(likelihood)
lnLExp_r	real	4		Exponential disk fit ln(likelihood)
lnLExp_i	real	4		Exponential disk fit ln(likelihood)
lnLExp_z	real	4		Exponential disk fit ln(likelihood)
lnLDeV_u	real	4		de Vaucouleurs fit ln(likelihood)
lnLDeV_g	real	4		de Vaucouleurs fit ln(likelihood)
lnLDeV_r	real	4		de Vaucouleurs fit ln(likelihood)
lnLDeV_i	real	4		de Vaucouleurs fit ln(likelihood)

lnLDeV_z	real	4		de Vaucouleurs fit ln(likelihood)
fracDeV_u	real	4		Weight of deV component in deV+Exp model
fracDeV_g	real	4		Weight of deV component in deV+Exp model
fracDeV_r	real	4		Weight of deV component in deV+Exp model
fracDeV_i	real	4		Weight of deV component in deV+Exp model
fracDeV_z	real	4		Weight of deV component in deV+Exp model
flags_u	bigint	8		Object detection flags per band
flags_g	bigint	8		Object detection flags per band
flags_r	bigint	8		Object detection flags per band
flags_i	bigint	8		Object detection flags per band
flags_z	bigint	8		Object detection flags per band
type_u	int	4		Object type classification per band
type_g	int	4		Object type classification per band
type_r	int	4		Object type classification per band
type_i	int	4		Object type classification per band
type_z	int	4		Object type classification per band
probPSF_u	real	4		Probablity object is a star in each filter.
probPSF_g	real	4		Probablity object is a star in each filter.
probPSF_r	real	4		Probablity object is a star in each filter.
probPSF_i	real	4		Probablity object is a star in each filter.
probPSF_z	real	4		Probablity object is a star in each filter.
ra	float	8	deg	J2000 Right Ascension (r-band)
dec	float	8	deg	J2000 Declination (r-band)
cx	float	8		unit vector for ra+dec
cy	float	8		unit vector for ra+dec
cz	float	8		unit vector for ra+dec
raErr	float	8	arcsec	Error in RA (* cos(Dec), that is, proper units)
decErr	float	8	arcsec	Error in Dec
b	float	8	deg	Galactic latitude
l	float	8	deg	Galactic longitude
offsetRa_u	real	4	arcsec	filter position RA minus final RA (* cos(Dec), that is, proper units)
offsetRa_g	real	4	arcsec	filter position RA minus final RA (* cos(Dec), that is, proper units)
offsetRa_r	real	4	arcsec	filter position RA minus final RA (* cos(Dec), that is, proper units)
offsetRa_i	real	4	arcsec	filter position RA minus final RA (* cos(Dec), that is, proper units)
offsetRa_z	real	4	arcsec	filter position RA minus final RA (* cos(Dec), that is, proper units)
offsetDec_u	real	4	arcsec	filter position Dec minus final Dec
offsetDec_g	real	4	arcsec	filter position Dec minus final Dec
offsetDec_r	real	4	arcsec	filter position Dec minus final Dec
offsetDec_i	real	4	arcsec	filter position Dec minus final Dec
offsetDec_z	real	4	arcsec	filter position Dec minus final Dec
extinction_u	real	4	mag	Extinction in u-band
extinction_g	real	4	mag	Extinction in g-band
extinction_r	real	4	mag	Extinction in r-band
extinction_i	real	4	mag	Extinction in i-band

extinction_z	real	4	mag	Extinction in z-band
psffwhm_u	real	4	arcsec	FWHM in u-band
psffwhm_g	real	4	arcsec	FWHM in g-band
psffwhm_r	real	4	arcsec	FWHM in r-band
psffwhm_i	real	4	arcsec	FWHM in i-band
psffwhm_z	real	4	arcsec	FWHM in z-band
mjd	int	4	days	Date of observation
airmass_u	real	4		Airmass at time of observation in u-band
airmass_g	real	4		Airmass at time of observation in g-band
airmass_r	real	4		Airmass at time of observation in r-band
airmass_i	real	4		Airmass at time of observation in i-band
airmass_z	real	4		Airmass at time of observation in z-band
phioffset_u	real	4	deg	Degrees to add to CCD-aligned angle to convert to E of N
phioffset_g	real	4	deg	Degrees to add to CCD-aligned angle to convert to E of N
phioffset_r	real	4	deg	Degrees to add to CCD-aligned angle to convert to E of N
phioffset_i	real	4	deg	Degrees to add to CCD-aligned angle to convert to E of N
phioffset_z	real	4	deg	Degrees to add to CCD-aligned angle to convert to E of N
nProf_u	int	4		Number of Profile Bins
nProf_g	int	4		Number of Profile Bins
nProf_r	int	4		Number of Profile Bins
nProf_i	int	4		Number of Profile Bins
nProf_z	int	4		Number of Profile Bins
loadVersion	int	4		Load Version
htmID	bigint	8		20-deep hierarchical trangular mesh ID of this object
fieldID	bigint	8		Link to the field this object is in
parentID	bigint	8		Pointer to parent (if object deblended) or BRIGHT detection (if object has one), else 0
specObjID	bigint	8		Pointer to the spectrum of object, if exists, else 0
u	real	4	mag	Shorthand alias for modelMag
g	real	4	mag	Shorthand alias for modelMag
r	real	4	mag	Shorthand alias for modelMag
i	real	4	mag	Shorthand alias for modelMag
z	real	4	mag	Shorthand alias for modelMag
err_u	real	4	mag	Error in modelMag alias
err_g	real	4	mag	Error in modelMag alias
err_r	real	4	mag	Error in modelMag alias
err_i	real	4	mag	Error in modelMag alias
err_z	real	4	mag	Error in modelMag alias
dered_u	real	4	mag	Simplified mag, corrected for extinction: modelMag-extinction
dered_g	real	4	mag	Simplified mag, corrected for extinction: modelMag-extinction
dered_r	real	4	mag	Simplified mag, corrected for extinction: modelMag-extinction
dered_i	real	4	mag	Simplified mag, corrected for extinction: modelMag-extinction
dered_z	real	4	mag	Simplified mag, corrected for extinction: modelMag-extinction
cloudCam_u	int	4		Cloud camera status for observation in u-band
cloudCam_g	int	4		Cloud camera status for observation in g-band

cloudCam_r	int	4		Cloud camera status for observation in r-band
cloudCam_i	int	4		Cloud camera status for observation in i-band
cloudCam_z	int	4		Cloud camera status for observation in z-band
resolveStatus	int	4		Resolve status of object
thingId	int	4		Unique identifier from global resolve
balkanId	int	4		What balkan object is in from window
nObserve	int	4		Number of observations of this object
nDetect	int	4		Number of detections of this object
nEdge	int	4		Number of observations of this object near an edge
score	real	4		Quality of field (0-1)
calibStatus_u	int	4		Calibration status in u-band
calibStatus_g	int	4		Calibration status in g-band
calibStatus_r	int	4		Calibration status in r-band
calibStatus_i	int	4		Calibration status in i-band
calibStatus_z	int	4		Calibration status in z-band
nMgyPerCount_u	real	4	nmgy/count	nanomaggies per count in u-band
nMgyPerCount_g	real	4	nmgy/count	nanomaggies per count in g-band
nMgyPerCount_r	real	4	nmgy/count	nanomaggies per count in r-band
nMgyPerCount_i	real	4	nmgy/count	nanomaggies per count in i-band
nMgyPerCount_z	real	4	nmgy/count	nanomaggies per count in z-band
TAI_u	float	8	sec	time of observation (TAI) in each filter
TAI_g	float	8	sec	time of observation (TAI) in each filter
TAI_r	float	8	sec	time of observation (TAI) in each filter
TAI_i	float	8	sec	time of observation (TAI) in each filter
TAI_z	float	8	sec	time of observation (TAI) in each filter

10.3 Relació de versions dels recursos i llibreries utilitzats

Recurs	Versió	Enllaç
Visual Studio Code	1.7	https://code.visualstudio.com/
The Interactive FITS File Editor	5.2	http://heasarc.gsfc.nasa.gov/ftools/fv/
Python	2.7	https://www.python.org/
Distribució Anaconda	2-4.2.0	https://www.continuum.io/downloads
Theano	0.8.2	http://deeplearning.net/software/theano/index.html
Lasagne	0.1	http://lasagne.readthedocs.io/en/latest/index.html
Astropy	1.2.1	http://www.astropy.org/