Estudio y análisis de las técnicas del *pipeline* de OCA aplicadas a datos simulados de la misión GAIA

Trabajo Fin de Máster

presentado por **D. Juan Gabriel Pérez Liñana**

Máster en *Inteligencia Artificial Avanzada*. **UNED**.

Directores:

Dr. D. Luis Manuel Sarro Baro

Dr. D. Miguel García Torres

Septiembre de 2012

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Quisiera agradecer a todas aquellas personas que me han ayudado en el desarrollo del presente Trabajo Final de Máster.

En primer lugar al director del trabajo, el Dr. D. Luis Manuel Sarro Baro, profesor del Departamento de Inteligencia Artificial de la UNED y miembro de OCA *(Object Clustering Analysis)*, por sus consejos y dirección, especialmente por guiarme cuando los objetivos iniciales del proyecto cambiaron por causas ajenas. En segundo lugar quisiera agradecer al codirector, Dr. D. Miguel García Torres, profesor de la Universidad Pablo de Olavide y miembro también de OCA, sus numerosas aportaciones, desde el acceso para algunas experimentaciones al cluster de computación del Centro Informático Científico de Andalucía hasta los consejos sobre la utilización de *LaTex* como herramienta para la composición del presente trabajo.
Para concluir, me gustaría dedicar esta memoria a mi mujer Maria Esteller, por todo el apoyo y comprensión recibidos y por soportar innumerables fines de semana y períodos de vacaciones mi ausencia, trabajando frente al ordenador en vez de estar junto a ella.

Resumen

Este Trabajo de Fin de Máster se enmarca dentro de la misión GAIA de la Agencia Europea del Espacio, cuyo lanzamiento está previsto para finales de 2013. Debido al elevado volumen de datos que se generará, se ha decidido crear un consorcio para el procesamiento y análisis de datos, llamado *Gaia Data Processing and Analysis Consortium* (DPAC). Dicho consorcio se encuentra dividido en 10 unidades de coordinación de las cuales, la octava (CU8), está enfocada en la estimación de parámetros astrofísicos a partir de las observaciones. El presente trabajo pertenece a los estudios preliminares llevados a cabo dentro del bloque de trabajo *Object Clustering Algorithm* (OCA) de CU8. El objetivo principal de OCA es desarrollar las herramientas adecuadas para el análisis de los datos recibidos desde la perspectiva del agrupamiento.

Los principales retos de OCA son estudiar la técnicas más adecuadas de agrupamiento para la naturaleza de los datos de GAIA y ser capaz de escalar dichas técnicas para poder tratar el gran volumen de datos que se recibirá a lo largo de la misión. A lo largo de este trabajo se presentarán las metodologías propuestas en la literatura para poder escalar algoritmos de agrupamiento así como la descripción de las técnicas elegidas debido a sus propiedades.

La investigación desarrollada se puede dividir en diferentes partes:

- Estudio de los datos que nos permita seleccionar una metodología adecuada para su preprocesamiento.

- Estudio de distintos atributos así como la generación de nuevos atributos. Mediante el análisis de los grupos obtenidos con el algoritmo de agrupamiento seleccionado con los distintos conjuntos de datos, se

podrán concluir la adecuación de éstos para su inclusión en la herramienta final.

- Estudio del algoritmo de agrupamiento frente la presencia de ruido.
- Análisis de las distintas técnicas que empleará OCA. Su importancia radica en que permitirá saber la idoneidad y viabilidad de estas técnicas con el diseño planteado para escalar el algoritmo de agrupamiento. Se incluirá la implementación de alguna de las técnicas del diseño planteado.

Por lo tanto este trabajo desarrolla el conjunto de pruebas preliminares del paquete de trabajo OCA.

Abstract

This Final Master Work is part of the GAIA mission of the European Space Agency, which is scheduled for late 2013. Due to the high volume of data being generated, it was decided to create an international consortium in charge of Gaia data processing, called *Gaia Data Processing and Analysis Consortium* (DPAC). It is formed around a set of 10 Coordination Units, the eighth (CU8), is focused on the estimation of astrophysical parameters. This work is part of the preliminary studies carried out within the working group *Object Clustering Algorithm* (OCA) of CU8.

The main objective of OCA is to develop appropriate tools for analyzing data received from the clustering perspective. OCA's main challenges are to study clustering techniques suitable to the nature of the GAIA data and be able to scale these techniques to deal with the large volume of data to be received over the mission. The methodologies proposed in the literature to be able to scale clustering algorithms, as well as descriptions of the techniques chosen due to their properties will be presented throughout this work.

The developed research can be divided into different parts:

- A study of the data that allows us to select a methodology appropriate to its preprocessing.
- Study of different attributes and the generation of new attributes. By analyzing the groups obtained with the selected clustering algorithm with different sets of data, we can conclude the suitability of these for inclusion in the final tool.
- Clustering algorithm study with noise in data.

- Analysis of the different techniques used by OCA. Its importance lies in knowing the suitability and feasibility of these techniques with the proposed design to scale the clustering algorithm. We'll implement some of the techniques of the proposed design.

Therefore, this paper develops the preliminary test suite of the OCA work package.

Capítulo 1

Introducción

Este capítulo presenta el contexto del proyecto desarrollado e información básica sobre el trabajo realizado.

Comienza con la localización del trabajo dentro de la misión espacial Gaia, continuamos explicando la razón por la que se propone su realización y los objetivos, tanto iniciales como su revisión durante el desarrollo del proyecto. Finalmente explicamos la estructura de la memoria.

1.1 Contexto del trabajo

GAIA (*Global Astrometric Interferometer for Astrophysics*) es una misión espacial de la Agencia Espacial Europea cuyo lanzamiento está previsto para finales de 2013. El objetivo de la misión es recopilar un catálogo de objetos estelares con medidas astrométricas y fotométricas. Esto permitirá trazar el mayor y más preciso mapa tridimensional creado hasta la fecha así como un mapa de los movimientos de dichos objetos. Esta información permitirá mejorar nuestro conocimiento sobre la composición, formación y evolución de nuestra galaxia.

Durante la misión, cada objeto estelar será monitorizado, en promedio, alrededor de 70 veces durante la misión, que se espera dure 5 años. Se obtendrán medidas de las posiciones, distancias, movimientos y cambios en el brillo de cada objeto. Considerando que se espera que la misión dure 5 años,

se estima que los datos recopilados ascenderán a 200 TB.

Analizar esta ingente cantidad de información es un desafío para las técnicas de minería de datos y en concreto para las técnicas de clustering o análisis de conglomerados, donde se engloba nuestro proyecto.

Este trabajo se localiza dentro del grupo de trabajo Object Clustering Analysis (OCA) del *Gaia Data Processing and Analysis Consortium* (DPAC); un consorcio internacional de más de 400 científicos e ingenieros de software creado para el procesamiento y análisis de la información que recogerá el satélite. DPAC está organizado en diferentes unidades de coordinación responsables de diferentes aspectos claves en el procesado de datos. OCA se engloba dentro de la unidad de trabajo 8 (CU8), la cual tiene el cometido de estimar los parámetros astrofísicos de los objetos observados y de llevar a cabo tareas de clasificación.

OCA realizará tareas de agrupamiento con el objetivo de obtener los grupos naturales de los distintos objetos y poder descubrir nuevos tipos. Además los resultados que obtenga podrán contribuir a mejorar los clasificadores mediante la incorporación de las nuevas clases.

Dado que aun no ha sido lanzado el satélite Gaia, el trabajo ha sido desarrollado con datos semiempíricos provenientes del proyecto *Sloan Digital Sky Survey* (SDSS) [15].

1.2 Objetivos perseguidos

El objetivo inicial del trabajo era la implementación y adaptación del algoritmo HMAC [29] en el marco de trabajo de Gaia y estudio de su comportamiento y resultados. Sin embargo, en una temprana fase del proyecto, con una implementación inicial del algoritmo realizada, encontramos que HMAC se encontraba ya implementado en el proyecto global a mayor escala donde se localiza nuestro trabajo.

En este momento se revisaron los objetivos del proyecto y se centraron alrededor de la estructura, en fase de diseño, conocida como *pipeline*, cuyo objetivo es tratar de escalar el problema para que pueda utilizarse de forma

eficiente y precisa el análisis de conglomerados en GAIA.

Por ello nuestro trabajo adquirió una vertiente más experimental, con el análisis, estudio y comprensión de los datos disponibles con diversos objetivos:

- Estudio del preprocesamiento necesario en los datos, con aspectos claves como la disminución de dimensionalidad, elección de atributos, etc.
- Adelantarnos a los problemas que puedan surgir respecto al tratamiento de los datos y sus soluciones, antes de la llegada de las muestras recogidas por Gaia.
- Análisis del coste temporal y viabilidad de la aplicación de las diferentes técnicas que utilicemos, especialmente HMAC.
- Validar experimentalmente si la estructura diseñada para resolver el problema será capaz de realizar su trabajo con el tamaño del problema esperado.
- Investigar el comportamiento de los algoritmos frente al ruido. Caracterización de los resultados.

Además del carácter empírico, el trabajo no estuvo exento de la implementación de algunas de las técnicas sugeridas para mejorar el rendimiento del análisis de conglomerados.

1.3 Estructura del trabajo

Para conseguir nuestros objetivos, el trabajo comienza realizando una revisión del área, que centraremos en las diferentes metodologías para escalar los problemas en el análisis de conglomerados y en el estudio de los nuevos modelos para el trabajo con grandes bases o repositorios de datos.

Posteriormente, en el capítulo 3 comenzamos con el estudio de los datos disponibles, centrándonos en la procedencia de los diferentes corpus de datos

y su división principal en datos completamente sintéticos y muestras provenientes de simulaciones realizadas sobre observaciones reales extraidas de repositorios existentes. Revisaremos la información contenida en las muestras y realizaremos el análisis de estos datos estudiando y proponiendo soluciones a los diferentes problemas encontrados.

Relataremos también el preprocesado realizado, extensible a los datos de Gaia cuando estén disponibles, las razones que nos llevan a elegir una determinada normalización, técnicas aplicadas para la reducción de la dimensionalidad, etc.

En el capítulo cuarto, describiremos las técnicas más relevantes manejadas durante el proyecto, los algoritmos K-means++ y K-means#, el algoritmo de clustering HMAC/MAC y finalizamos el capítulo con la estructura conocida como árbol kd.

El capítulo 5 introduce la estructura del *pipeline*, ideada para manejar de forma escalable los datos, para que el coste temporal y espacial del algoritmo de clustering crezca de forma lineal con el número de datos, o al menos, para que el crecimiento no sea tal que el problema se vuelva intratable.

La segunda parte de este capítulo presentará experimentos que nos permitirán extraer importantes conclusiones, como las pruebas de escalado de los algoritmos K-means++ y K-means#, pruebas de rendimiento del algoritmo de reducción de dimensionalidad, los experimentos sobre el coste espacial y temporal de HMAC que demuestran la necesidad de plantear una metodología para reducir el tamaño del problema, o las pruebas y conclusiones sobre la variación del algoritmo de agrupación utilizando estructuras de árbol kd.

El capítulo sexto está dedicado a los experimentos iniciales del algoritmo MAC, con los datos y los atributos disponibles, donde los malos resultados nos obligarán a plantear un análisis de posibles soluciones para mejorar la eficacia del clustering.

El siguiente capítulo presenta la segunda fase de la experimentación donde desarrollamos las diferentes líneas de mejora, destacando los resultados obtenidos con una de las líneas de investigación, la incorporación de información derivada de los momentos estadísticos de los espectros de los objetos, que es una idea original del presente trabajo.

Finaliza esta sección con el análisis de los conglomerados, caracterizando la información perteneciente a los clusters. Con este objetivo, se desarrolla una herramienta que a través de servicios web, consulta los datos de los objetos estelares originales de cuyas muestras se generaron las simulaciones de los datos utilizados, de forma que mostramos cómo se puede realizar un análisis de la información útil en el dominio del problema, a la vez que facilitamos una herramienta para realizar este cometido.

El capítulo 8 presenta los resultados del algoritmo de agrupación, cuando las muestras contienen ruido, bien sea ruido real ya que utilizamos muestras de simulaciones sobre datos de repositorios existentes donde podemos controlar el nivel de ruido o bien por la introducción de ruido sintético.

Por último se recopilan las conclusiones más relevantes y trabajo futuro.

Capítulo 2

Revisión del área

Este capítulo se centrará en los algoritmos de agrupamiento o análisis de conglomerados en grandes bases de datos así como en los marcos de trabajo específicos para trabajar con datos masivos.

En primer lugar presenta los distintos paradigmas de agrupamiento para escalar los algoritmos de agrupamiento. A continuación describe los algoritmos más relevantes en cada paradigma indicando los motivos por los que no son adecuados en nuestros datos. Finalmente se revisan dos marcos de trabajo para poder trabajar con datos masivos.

Para escalar las distintas técnicas de agrupamiento hay que tener en cuenta que todos los datos deben ser visitados y analizados en una o más ocasiones. Esto conlleva que al alto coste espacial se le una un coste temporal que dependerá de la técnica empleada. Para abordar esto, se han propuestos diversos paradigmas [44]:

- Divide y vencerás.
- Metodología incremental.
- Paralelización.

El escalado de un algoritmo de agrupamiento puede implicar el uso de uno o varios paradigmas.

2.1 Divide y vencerás

La estrategia divide y vencerás está basada en plantear el problema original de tal modo que se descomponga en varios subproblemas de menor tamaño que pasan a resolverse de manera independiente. Finalmente se combinan las soluciones obtenidas para construir la solución del problema original.

Los algoritmos K-medias [30] y CURE (Clustering Using REpresentatives) [22] son dos algoritmos que han sido adaptados a este paradigma.

Una adaptación del K-medias propuesto por [26] divide los datos en K subconjuntos y ejecuta el algoritmo en cada uno de ellos. Finalmente fusiona los grupos resultates. Esta estrategia se presta de forma natural a su ejecución en paralelo.

El amplio uso de este algoritmo se debe a su simpleza y eficiencia; de modo que es adecuado para su aplicación en grandes conjuntos de datos. Sin embargo los mayores inconvenientes son su alta dependencia en una buena inicialización para obtener buenos resultados, su escasa robustez frente a casos anómalos y su limitación para encontrar grupos no esféricos o con distinto tamaño y/o densidad.

En nuestros datos este algoritmo no es adecuado debido a que la distribución de los grupos no es esférica. Además, los distintos tipos de objetos forman grupos heterogeneos así como se espera que haya presencia de casos anómalos. Otro motivo por el cual no es un buen candidato es que existe mucha confusión entre grupos. Por ejemplo, las características de algunas galaxias hacen que sean confundidas con estrellas. Esto se corroboró aplicando dicho algoritmo a los datos en estudio.

El algoritmo CURE (Clustering Using REpresentatives) [22] trabaja con una muestra aleatoria obtenida a partir de los datos originales. Tal y como demuestran los autores, una muestra de datos de tamaño moderado preserva la información sobre la geometría de los grupos del conjunto original. Para escalarlo a grandes conjuntos de datos, el conjunto original de datos se divide en K particiones y se aplica el algoritmo en cada partición. En una fase posterior se fusionan los grupos obtenidos en cada partición.

Es un algoritmo robusto frente a casos anómalos y permite identificar

grupos de forma no esférica así como de diferentes tamaños. Sin embargo no trabaja bien en caso de que haya grupos con distintas densidades. El coste espacial es $\mathcal{O}(n)$; mientras que el temporal es $\mathcal{O}(n^2 \log(n))$.

Los datos con los que trabajamos forman grupos de distintas densidades. Esto, junto al alto coste temporal hacen que esta técnica haya sido descartada.

A pesar de la poca idoneidad de las técnicas mostradas, el paradigma divide y vencerás es una estrategia adecuada para poder agrupar datos masivos. Por lo tanto, y tal y como veremos más adelante, será la solución que aplicaremos para poder agrupar nuestros datos.

2.2 Metodología incremental

La idea central del agrupamiento incremental es agrupar una muestra suficientemente significativa de datos y evaluar los nuevos datos a partir de los grupos obtenidos.

Dos de los algoritmos más populares de este paradigma son BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [56] y DBSCAN (Density-Based Spatial Clustering of Application with Noise) [14].

BIRCH empieza contruyendo un árbol balanceado que almacena información relativa a los datos sin que éstos tengan que estar en memoria. Posteriormente la información de dicho árbol se va actualizando de forma incremental a medida que va recibiendo datos nuevos.

Dicha estrategia es adecuada para grupos con forma esférica y es robusta frente a casos anómalos. Sin embargo no es adecuado si existen grupos de diferentes tamaños. Este inconveniente y la heterogeneidad de los grupos presentes en nuestros datos hacen que sea descartado para nuestro propósito.

DBSCAN se basa en el concepto densidad para llevar a cabo el agrupamiento. Para ello define la densidad como el número de puntos dentro de un radio determinado y el procedimiento consiste en conectar regiones con una densidad mayor que el considerado por medio de un parámetro.

Las mayores ventajas son que no es necesario especificar el número de

grupos, es robusto frente al ruido y casos anómalos y puede trabajar en caso de que los grupos tengan formas distintas. Como inconvenientes, requiere definir una métrica para calcular distancias y no trabaja bien si los grupos tienen densidades distintas.

Considerando que en nuestros datos los grupos tendrán distintas densidades, DBSCAN no es una buena elección para nuestros datos.

Otro algoritmo que sigue este paradigma, es CLARA [25] *(Clustering LARrge Applications)*, que trata de mejorar el coste temporal del algoritmo PAM [25] (similar en concepto a K-medias pero utilizando elementos centrales de grupo en lugar de centroides), la mejora que presenta CLARA respecto a PAM, se basa en la elección de un número bajo de puntos en las muestras, que aproximen los puntos centrales representativos del conjunto completo de datos. Los inconvenientes que presenta frente a nuestro problema son, en primer lugar, que la utilización de un subconjunto de los datos puede no ser representativo de las clases en nuestro caso, ya que se puede perder la información de grupos naturales con pocos datos y en segundo lugar el algoritmo subyacente, PAM, funciona para grupos naturales de forma esférica.

Existen mejoras más recientes de este algoritmo como MCMRS Sampling Scheme (*MultiCentroid, Multi-Run Sampling Scheme*) [9], que observa experimentalmente que existe una alta probabilidad de que los mejores elementos centrales de los grupos se encuentren cerca de los centroides y utiliza esta información para generar un subconjunto de datos para la muestra, acelerando el cálculo. Sin embargo la técnica de agrupamiento continúa funcionando en base a distancias a elementos centrales, generando grupos con formas esféricas, por lo tanto no se presenta adecuado para nuestro problema.

2.3 Paralelización

La paralelización pretende reducir el tiempo de cómputo distribuyendo la carga de trabajo entre los procesadores disponibles. Para poder adaptar un código a dicho paradigma se requiere:

- Independencia de datos. Deben identificarse tareas independientes den-

tro del código, de tal forma que no exista dependencias de datos que puedan obstruir la paralelización.

- Identificar los cuellos de botella de rendimiento. La paralelización de estas regiones permite mejorar considerablemente el rendimiento del algoritmo.

Los algoritmos paralelos desarrollados son adaptaciones de algoritmos que pertenecen a alguno de los pradigmas anteriores visto. Las principales estrategias de paralelización [50, 27] son:

- Paralelización independiente. Donde cada procesador accede al conjunto de datos pero no se comunican entre si.
- Paralelización de la tarea. Cada procesador realiza diferentes tareas en los datos particionados o en el conjunto de datos.
- SMPD (*Single Program Multiple Data*). Diferentes procesadores ejecutan el mismo algoritmo en diferentes subconjuntos de datos e intercambian resultados parciales para cooperar en el proceso que conduzca a una solución final.

La mayoría de los algoritmos de agrupamiento paralelos siguen una combinación de SMPD y paralelización de la tarea, con una arquitectura maestro-esclavo.

De los algoritmos vistos anteriormente, el K-medias se presta de forma natural a la paralelización, tanto para el cáculo de las distancias al centroide como en la asignación de los grupos. Por ejemplo, en [24] usan un modelo de paso de mensajes y en [57] una arquitectura maestro-esclavo.

Una versión paralela de BIRCH, PBIRCH [17], utiliza el paso de mensajes entre nodos de computación para distribuir los datos entre cada procesador para que construyan su propia estructura de datos y luego intercambian información para actualizar los conglomerados.

DBSCAN es otro algoritmo del que existen varias propuestas paralelas. En [6] se propone un modelo maestro-esclavo, el cual divide la entrada de

los datos en varias particiones, de modo que cada partición es asignada a un procesador que los agrupa mediante una versión concurrente de DBSCAN. Finalmente, se añade un último paso de unión de los grupos encontrados en cada procesador para generar los grupos globales.

Las versiones paralelas de los algoritmos presentados mejoran la eficiencia, pero presentan los mismos inconvenientes que los de su versión concurrente. Dichos inconvenientes se resumen en:

- Limitado a grupos esféricos: K-medias y BIRCH.
- Dificultad en agrupar grupos con distinto tamaño: K-medias y BIRCH.
- Dificultad en agrupar grupos con distintas densidades: K-medias, CURE y DBSCAN.

En la Tabla 2.1 se resumen los paradigmas y algoritmos presentados en este capítulo. De cara al desarrollo de nuestro trabajo nos interesará que el diseño permita, en la medida de lo posible, una futura paralelización.

El análisis de los diferentes algoritmos nos ayuda a entender las decisiones tomadas en el diseño de la estrategia de agrupamiento considerada para abordar grandes conjuntos de datos. También permite entender por qué la solución propuesta no incluye ninguno de los algoritmos presentados en este capítulo.

2.4 Trabajo con grandes bases de datos

Para el manejo del repositorio de datos se hace necesario la introducción de nuevas técnica de acceso a los datos; existen nuevos modelos y marcos de trabajo especialmente dedicados a ello, como *MapReduce* [10] introducido por Google para dar soporte a la computación paralela sobre grandes colecciones de datos en grupos de computadoras.

Su funcionamiento se basa en disgregar los procesos en operaciones de *map*() y *reduce*(), los datos de entrada son particionados y procesados independientemente por estas operaciones.

Tabla 2.1: Resumen de los paradigmas y algoritmos analizados.

Paradigma	Algoritmo
Divide y venceras	K-medias [26]
	Cure [22]
Metodologia incremental	BIRCH [56]
	DBSCAN [14]
	CLARA [25]
	MCMRS Sampling Scheme [9]
Paralelizacion	Parallel K-means [24, 57]
	PBIRCH [17]
	Parallel DBSCAN [6]

Con map se toma una función de entrada y una secuencia de valores aplicándose la función a cada valor en la secuencia. Con tareas *reduce* se procesan independientemente las listas y de forma colectiva crean la salida final.

Podemos entender *MapReduce* desde el punto de vista de la arquitectura o del programador:

- Desde la arquitectura, un nodo maestro divide la entrada en problemas de menor identidad, y los distribuye a otros nodos que pueden subdividir los problemas a su vez dando lugar a una estructura en árbol, procesándose el problema y devolviendo la respuesta al nodo maestro.

- Para el programador, el paradigma *MapReduce* es una gran abstracción, que se centra en la forma de resolver un problema utilizando sólo funciones map y reduce. Parte de la abstracción incluye la partición de los datos, asignación y monitorización de tareas y distribución de los datos a través de las maquinas utilizando un sistema de archivos distribuido.

El paradigma *MapReduce*, trata de solventar una de las limitaciones en un entorno de procesamiento en paralelo, que es el transporte de grandes

ficheros entre ordenadores debido al limitado ancho de banda entre ellos, dado que la función *map*() utiliza una memoria intermedia de carácter local de forma agregada y ordenada y la función *reduce*() recibe una lista de valores asociados a una única clave, debido a la latencia en la comunicación entre los ordenadores, cualquier reducción de la cantidad de datos intermedios, realizada de forma local, incrementará la eficiencia de los algoritmos.

El entorno es tolerante a fallos, incluso ante fallos hardware de nodos. El nodo maestro vigila el comportamiento del resto y ante una caída, cualquier función *map*() o *reduce*() que se encuentre en progreso durante el fallo, es asignada a otro nodo, convirtiéndose en la elección para la compresión de datos masivamente paralelos en un esquema tolerante a fallos.

MapReduce no es adecuado para todo tipo de problemas, sólo para los que su esquema de mapeo y reducción de datos sea adecuado, podemos ver en [19] cómo la clave es utilizar algoritmos que de alguna forma lleven implícita la función $y = \sum f(x)$,donde $f(x)$ es realizada por *map*() y $\sum$ por *reduce*().

Como ejemplo, podemos ver cómo algoritmos utilizados en minería de datos que realizan una única iteración y cuyas claves estarían uniformemente distribuidas, como es el caso de Naïve Bayes, es adecuado para este esquema, o cómo algoritmos que realizan múltiples iteraciones con poca información intercambiada o sincronizada entre las iteraciones, como el caso de K-Medias son también adecuados, frente a algoritmos como SVM (Support Vector Machines) o HMM (Hidden Markov Model) que no son adecuados por utilizar mucha información compartida y por la fina granularidad necesaria en la sincronización.

En [19] podemos encontrar también el pseudocódigo y análisis de la implementación de K-Medias utilizando el modelo *MapReduce*.

La implementación más popular del paradigma *MapReduce* es *Hadoop*, que es una implementación de código libre de Apache, con un API nativa en Java.

Algunas particularidades de esta implementación son que utiliza su propio sistema de archivos distribuido, HDFS (Hadoop Distributed File System), escalable y portátil escrito en Java.

Este sistema de archivos almacena los archivos grandes a través de múlti-

ples máquinas consiguiendo la fiabilidad a través del replicado de datos en múltiples máquinas sin requerir almacenamiento RAID.

Otra mejora introducida por esta implementación es que cada sistema de archivos debe conocer y proporcionar su ubicación: el nombre del rack/switch donde está el nodo trabajador.

Las aplicaciones *Hadoop* pueden usar esta información para ejecutar trabajo en el nodo donde están los datos y, en su defecto, en el mismo rack/switch, reduciendo así el tráfico de red troncal y aumentando el tráfico entre máquinas del mismo armario, conectados a mayor velocidad. El sistema de archivos HDFS usa esto cuando replica datos, para intentar conservar copias diferentes de los datos en armarios diferentes. El objetivo es reducir el impacto ante el fallo no sólo de una máquina, sino de un armario completo de máquinas. Otras alternativas al paradigma *MapReduce* son:

- Percolator [37]: Se trata de un sistema para manejar las actualizaciones en grandes conjuntos de datos de forma continua. El sistema se desarrolló para optimizar el indexado de páginas web y no se muestra adecuado en nuestro caso.

- Apache Hama [43, 8, 36]: Construido sobre HDFS *(Hadoop Distributed File System)* trabaja sobre *Hadoop* aportando el paradigma de desarrollo de aplicaciones usando el paso de mensajes como medio de comunicación y garantiza la imposibilidad de bloqueos o colisiones en el mecanismo de comunicación.

  Una ventaja adicional sobre el modelo MapReduce es que éste, a diferencia de Hama, no preserva los datos de forma local en operaciones consecutivas.

  Apache Hama se muestra eficaz cuando:

  1. La localidad de los datos es importante.
  2. Procesamos datos con relaciones complejas.
  3. Usamos muchas iteraciones y recursiones.

El principal defecto es que la implementación actual presenta una deficiente tolerancia a fallos respecto las demás alternativas.

- Pregel [32]: Similar a Apache Hama, ambos se basan en técnicas BSP *(Bulk Synchronous Parallel)* óptimas especialmente para cálculos científicos masivos (donde predominan operaciones sobre matrices, grafos, etc.), pero centrándose en que el desarrollador escriba un algoritmo usando grafos como estructuras, utilizando una función que envía y recibe mensajes a grafos, abstrayendo de los detalles de la distribución de la información y la tolerancia a fallos. El problema de esta aproximación es que está más centrada y se debe desarrollar más estrictamente siguiendo su modelo, frente a alternativas como Apache Hama, que siguiendo la misma técnica BSP presenta una mayor generalidad.

Capítulo 3

Estudio de los datos

3.1 Descripción de los datos utilizados

El presente trabajo utiliza datos simulados debido a que no habrá datos reales disponibles hasta unos meses después de que el satélite Gaia sea lanzado.

La etapa de desarrollo de las simulaciones está dividida en ciclos; de modo que cada ciclo proporciona nuevas simulaciones que incorporan nuevos tipos de objetos, nuevas medidas, solventa errores y/o carencias encontrados en simulaciones anteriores.

Las simulaciones proporcionadas son de tipo sintéticos o semiempíricos. Los primeros provienen de modelos teóricos y lo segundos parten de datos reales del catálogo SDSS (*Sloan Digital Sky Survey*) que son tratados para aproximar sus medidas a los valores que observaría Gaia. Como el SDSS cubre un espectro menor que Gaia, tuvieron que extenderse los espectros reales mediante espectros sintéticos. En la Figura 3.1 se muestra un esquema de la generación de los datos semiempíricos. En [52] se describe la selección de los espectros del SDSS así como el procedimiento seguido para extender los datos.

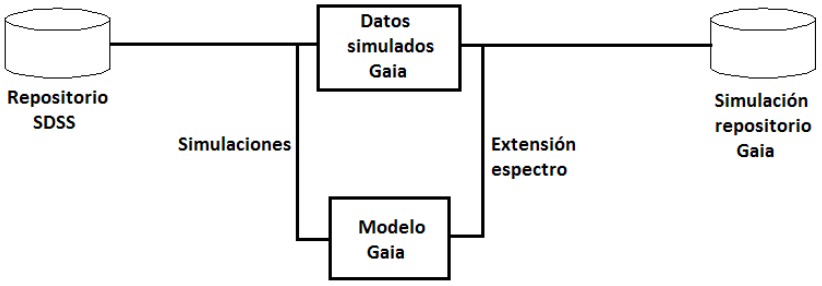


Figura 3.1: Esquema de la generación del corpus SDSS.

Los distintos tipos de objetos disponibles son:

- Galaxias (SDSS)
- Cuásares (SDSS)
- Estrellas:
  - SDSS
  - Phoenix
  - Tipo A
  - Tipo Wolf-Rayet
  - Ultrafrías

La etiqueta SDSS implica que los datos son semiempíricos. Definimos brevemente y de forma general cada tipo de objeto como:

- **Estrella**: En sentido general, es todo objeto astronómico que brilla con luz propia. Es una esfera de plasma que mantiene su forma gracias a un equilibrio de fuerzas entre la gravedad, que empuja la materia hacia el centro de la estrella, y la presión que ejerce el plasma hacia fuera, que tiende a expandirlo. La presión hacia fuera depende de la temperatura, que en un caso típico se mantiene con la energía producida en el interior de la estrella. Este equilibrio seguirá esencialmente igual en la medida en que la estrella mantenga el mismo ritmo de producción energética,

ya que una gran variedad de reacciones diferentes de fusión tienen lugar dentro de los núcleos de las estrellas, las cuales dependen de su masa y composición.

- **Galaxia**: Es un conjunto de varias estrellas, nubes de gas, planetas, polvo cósmico, materia oscura y quizá energía oscura, unido gravitatoriamente. La cantidad de estrellas que forman una galaxia es muy variable en tipo y número.

- **Cuásar**: Tienen aparentemente el mismo aspecto que una estrella, de ahí su nombre, que proviene de la contracción inglesa quasi-stellar y son una fuente de energía electromagnética, que incluye radiofrecuencias y luz visible. Son objetos extremadamente lejanos y luminosos, lo que permite su visión a pesar de la distancia. Hoy en día, se piensa que los cuásares son los núcleos de galaxias muy jóvenes.

Los datos semiempíricos se simularon a distintos valores de la magnitud G, la cual es una medida que indica la cantidad de energía emitida por el objeto que recibe el detector. Dichos valores considerados fueron $G = 15, 18{,}5$ y 20.

Para los datos sintéticos las simulaciones no sólo tenían en cuenta los valores de G sino también el de otros parámetros como por ejemplo la T_{eff}. En este caso se consideraron dos metodologías distintas: *Nominal* y *Random*. En la primera de ellas las simulaciones se hacen para valores específicos de los parámetros astrofísicos como por ejemplo, $G = 10, 15, 18$ y $T_{eff} = 5000, 7500, 1000$. En la segunda metodología estructura los valores específicos de las simulaciones como una malla de puntos en el espacio de parámetros astrofísicos y lleva a cabo simulaciones para valores intermedios de dicha malla mediante extrapolación. Tal y como veremos en la sección 3.2, la metodología empleada para la simulación es importante e influye en los resultados finales.

Las características de las estrellas provenientes de los datos sintéticos son:

- Corpus Phoenix: Contiene datos espectrales de estrellas, han sido generados a partir de dos bibliotecas de datos diferentes y con diferentes

valores de magnitud (9, 15 y 20) y (9, 11, 15, 18,5 y 20) respectivamente, por lo tanto, este corpus está formado por dos conjuntos diferentes de datos según su generación y donde encontramos caracterizadas estrellas en unos rangos de temperaturas donde las más calientes coinciden con las estrellas simuladas a partir de los datos de SDSS, y sus estrellas más frías, con temperaturas que llegan casi a las de las estrellas ultra frías. Mantiene una proporción similar para todos los rangos de temperaturas lo cual no es realista y su proporción respecto a las estrellas normales de SDSS lo es.

- Estrellas de tipo A: Conjunto de datos formado por estrellas muy calientes cuyo color convencional es blanco, simuladas con una temperatura efectiva entre 10,000 − 15,000K para magnitudes de $G = 8, 11$ y 15.

- WR o Wolf-Rayet: Estrellas masivas y calientes, con temperaturas superficiales entre 25,000 y 50,000K, elevadas luminosidades y muy azules como queda reflejado en sus espectros. Existen muy pocos tipos de estrellas de este tipo por lo que podemos considerar su proporción adecuada respecto al número de estrellas del catálogo SDSS, 26 frente a 50,000. Han sido simuladas para magnitudes $G = 10, 12, 15, 18{,}5$ y 20.

- Fast rotators: Corpus de datos que contiene estrellas con un rápido movimiento giratorio que produce diversos efectos físicos en la estrella [28], como la reducción de la luminosidad y cambios en la forma de la estrella que distorsionan la forma esférica. El efecto en la evolución de la estrella no es significativo pero cambia el espectro de la estrella. Además estas estrellas aparentan ser más frías de lo que son. La generación de datos se realizó para valores de $G = 10, 12, 14$ y 16. La proporción en cuanto al número de instancias respecto a otros tipos de estrellas no se mantiene.

- Ultra Frías: Corpus de datos que contiene solo este tipo de estrellas y utiliza dos modelos diferentes para el cálculo de los parámetros básicos, dependiendo de la temperatura efectiva (T_{eff}) de la estrella, *Dusty* para espectros con $T_{eff} = 1500 - 2500$K y *Cond* para $T_{eff} = 100 - 1600$K.

Observamos que son temperaturas extremadamente frías y son un tipo de estrellas infrecuentes por lo que su proporción respecto el resto de estrellas no es realista. Se utilizan valores de $G = 8, 11$ y 15 para su generación.

Tanto el corpus de Phoenix como el de Ultra Frías contienen datos que provienen de ambas metodologías, mientras que las simulaciones de las estrellas tipo A, WR y Fast Rotators, únicamente provienen de la metodología *Nominal*. En los trabajos [49, 53, 12] y en la Wiki de DPAC se detalla el origen y se describen las simulaciones.

Tabla 3.1: Número de instancias por corpus de datos

Fuente	Clase	Tipo	# instancias
Semiempíricas	Estrella	–	50000
	Galaxia	–	33670
	Cuásar	–	70556
Sintéticas	Estrella	tipo A	252
		Fast Rotator	288
		Corpus Phoenix	10000
		Ultrafrias	11188
		Wolf-Rayet	43

El número de objetos estelares de cada tipo puede verse en la tabla 3.1. Si atendemos a las tres clases principales de datos disponibles, estrellas, galaxias y cuásares, el número de instancias de estrellas (sumando todos los tipos) es de 71,771.

3.2 Problemas en las muestras

Tras la descripción teórica de los datos y previamente a su análisis encontramos problemas de diferente naturaleza.

- La proporción de los diferentes tipos de objetos, ya sean entre los tipos de estrellas de las diferentes simulaciones sintéticas o el número de galaxias y cuásares, respecto a las estrellas en los datos semiempíricos, dista mucho de ser realista. Por ejemplo en los datos originados de SDDS, la proporción es de 32,4 % de estrellas, 21,8 % de galaxias y 45,7 % de cuásares mientras que el universo observable contiene, según una estimación conservadora [18], aproximadamente 6×10^{22} estrellas organizadas en $8*10^{10}$ galaxias y el número de cuásares reales es mínimo en comparación.

- El bajo número de instancias de algunos tipos de objetos, por ejemplo, estrellas de tipo Wolf-Rayet. El problema de las distribuciones no balanceadas de datos puede conducir, con métodos tradicionales de minería de datos, a aprendizajes sesgados en perjuicio de la clase minoritaria. [55]

- Utilización de diferentes valores de magnitud en las muestras, el valor de la magnitud se refiere a la cantidad de energía captada por el detector, tendrá una relación con el efecto del ruido en los datos, pues será mucho mayor su efecto en objetos donde la energía captada es débil, por lo tanto tendremos simulaciones con diferente afectación de ruido.

- Las estrellas de los datos semiempíricos sólo contienen datos de los tipos estelares más comunes con unas temperaturas entre 4.000 y 7800 grados Kelvin y según la clasificación espectral encontramos sólo ciertos tipos de estrellas. Esto trata de subsanarse con la introducción de los datos de origen sintético, pero aparte del problema de la falta de proporción en instancias de diferentes tipos de estrellas, encontraremos posteriormente, que los espectros se encuentran en magnitudes de valores totalmente distintas.

Una vez comenzó la fase de análisis y preprocesamiento de los datos, encontramos numerosos problemas relativos a los datos, principalmente debido a las instancias procedentes de datos sintéticos que dificultan la tarea posterior de agrupamiento.

- Sólo los datos originados a partir de SDSS, incorporan todos los parámetros astrofísicos que utilizaremos en fases posteriores de la experimentación.

- Sólo las estrellas y galaxias de SDSS, incorporan algún mecanismo que nos puede indicar los valores observados de objetos existentes, a través de la identificación del objeto original, que utilizaremos en fases posteriores del trabajo 7.5 para caracterizar la información agrupada.

- Los datos de estrellas Ultra Frías y del corpus Phoenix necesitan de un proceso de escalado, pues sus valores se encuentran en magnitudes diferentes respecto a los demás corpus de datos y entre sus propios espectros. Se trata de un problema de mezcla por las diferentes fuentes de datos.

Por estos motivos y también por la necesidad de reducir el número de instancias para poder realizar las simulaciones, tras una primera fase de experimentación y ante los malos resultados obtenidos al mezclar los datos provenientes de fuentes sintéticas con los datos provenientes de observaciones en SDSS, realizamos las posteriores fases de experimentación sólo con los datos provenientes de SDSS.

Debemos mencionar, que con el presente trabajo contribuimos también en el proceso de mejora de los datos generados por *DPAC*, dado que encontramos en los datos iniciales provenientes del ciclo 5, instancias con valores inválidos para ciertos atributos que debieron ser corregidos en posteriores generaciones de corpus de datos.

Con los datos del ciclo 6, encontramos durante los procesos de normalización de los datos, que los espectros contenían algunos valores negativos.

Dado que los espectros expresan el flujo de energía en la correspondiente banda, este valor no puede ser negativo, por lo que se concluyó que el motivo de estos valores se debe a la introducción de ruido.

Finalmente, para realizar los experimentos en un entorno más controlado y dado que no conocíamos el modelo de ruido introducido en los datos,

utilizamos un conjunto de datos sin ruido y otro con ruido, comparando los resultados en una fase de de experimentación que llevamos a cabo en 8.

Que el presente trabajo haya convivido con diferentes corpus de datos ha afectado en gran medida a la realización de los experimentos, como se puede entender, debido a que continuamente debían repetirse experimentos muy costosos temporalmente.

3.3 Atributos presentes

Esta sección muestra una descripción de los atributos presentes en los datos, de forma previa al inicio de la experimentación.

Los dos espectros presentes en cada dato se corresponden con los fotómetros de baja resolución equipados por el satélite Gaia, uno de ellos operando en la banda azul, entre $330-680$nm correspondiente a los datos de BP (*Blue Photometer*) y otro operando en la banda roja entre $640-1050$nm, cuyos datos conocemos como RP (*Red Photometer*).

Cada espectro está formado por 60 datos discretos. Nuestra experimentación se ha planteado inicialmente con estos espectros, que nos hablan de las propiedades físicas de los objetos. Además, los datos semiempíricos de SDSS contienen atributos provenientes de la astrometría, que nos habla de las agrupaciones de objetos en el espacio y nos interesan porque las agrupaciones espaciales corresponden con alta probabilidad a objetos que se formaron en las mismas condiciones físicas de abundancia de metales y al mismo tiempo, estos atributos son:

- Paralaje, que podemos entender como la inversa de la distancia del objeto al punto de referencia, es decir al satélite Gaia.

- Ascensión y declinación. Información de la posición del objeto que podemos entender como la latitud/longitud en la esfera celeste.

- Movimiento propio, es el cambio angular de la posición a lo largo del tiempo visto desde el observador, medido en segundos de arco por año.

Está formado por los desplazamientos angulares en ascensión y en declinación.

Cabe destacar respecto al paralaje, que encontramos valores negativos en lo datos de galaxias y cuásares, concluimos que el motivo es que no se encuentra disponible la información de paralaje para estos objetos y existe un ruido introducido en la simulación que sigue una distribución normal, como podemos comprobar por la existencia de, aproximadamente la mitad, de valores negativos en estas muestras. Por lo tanto, únicamente podemos incorporar el paralaje en experimentos donde intervengan únicamente las estrellas.

A lo largo del presente trabajo, trataremos de combinar la información física con la astrométrica en diferentes puntos. 7

Otro atributo importante es la magnitud aparente. Utilizaremos su valor en los datos como criterio de filtrado. Valores con menor magnitud aparente tendrán menor presencia de ruido, en 8.1 estudiaremos cómo varían los resultados si ampliamos el rango de valores de G, es decir, introduciendo en la muestra objetos con mayor ruido.

También estudiaremos los resultados cuando cambiemos el corpus de estudio por la versión con ruido sintético 8.2, que presenta los mismos atributos e información en los datos, aunque veremos durante la experimentación que el conjunto de datos no es el mismo que el de las muestras sin ruido, dificultando la comparación de resultados.

Encontramos otros atributos en los datos como la velocidad radial, pero no presentaba valores, por lo que no se pudo utilizar en el estudio.

3.4 Análisis de los datos

Esta sección trata sobre diferentes aspectos de los atributos y datos, encontrados durante la experimentación y no de forma previa como la sección anterior y del análisis y preparación de los datos para la experimentación.

Sobre los datos de los espectros BP/RP, en el satélite Gaia, los fotones incidirán sobre un CCD, consistente en una matriz de pixels de cierto

Tabla 3.2: Los 125 atributos iniciales.

Tipo de medida	Medida	#datos
Espectrometría	BP	60
	RP	60
Astrometría	Paralaje	1
	Movimiento propio	A.R: 1
		Dec: 1
Coordenadas	A.R.	1
	Dec.	1

tamaño sensibles a la luz. Los datos del CCD corresponderán a las cargas fotoeléctricas leídas.

Con el objetivo de simular la respuesta del CCD se introduce una característica en los datos que debemos tener en cuenta, se trata del *oversampling*.

Si observamos una representación de los datos, vemos que éstos se repiten siguiendo un patrón de desplazamiento entre ellos. Esto es debido a que existen en la simulación tres repeticiones de cada espectro simulando un cierto desplazamiento en el CCD del objeto medido, modelando de esta forma el desplazamiento sub-pixel de la imagen de un objeto en el detector que creará una cierta función de dispersión sobre los datos. Gaia deberá combinar estas informaciones para obtener el espectro final de un objeto. Esta técnica no era conocida aún en la documentación de DPAC disponible sobre el oversampling [48].

La forma de introducción de este oversampling en nuestros datos, se realiza añadiendo un desplazamiento, de forma que los datos del primer muestreo no tienen desplazamiento, los del segundo muestreo lo tienen de un tercio de desplazamiento subpixel para un factor escogido para los datos y el tercer y último muestreo, tiene un desplazamiento de dos tercios del factor escogido.

Para nuestra experimentación utilizaremos directamente los datos del primer oversample, es decir, sin desplazamiento, descartando los datos repetidos con el desplazamiento añadido, dado que con los datos de Gaia tendremos un espectro final por objeto, una vez realizada la fusión y se tengan en cuenta

los posibles desplazamientos, no sólo por la medición, sino incluso por el tamaño y forma de los pixeles del detector y las distorsiones que introduzcan. Si calculamos la gráfica media de los espectros BP y RP de cada una de las clases, podemos observar el gran parecido, especialmente entre las clases semiempíricas de SDSS, e intuir la dificultad que va a suponer el agrupamiento utilizando únicamente esta información.

Presentamos las diferentes clases en la Figura 3.2, con todos los datos disponibles en cada clase, incluyendo desplazamientos de los objetos (*oversamples*) a través de las primeras dos componentes principales.

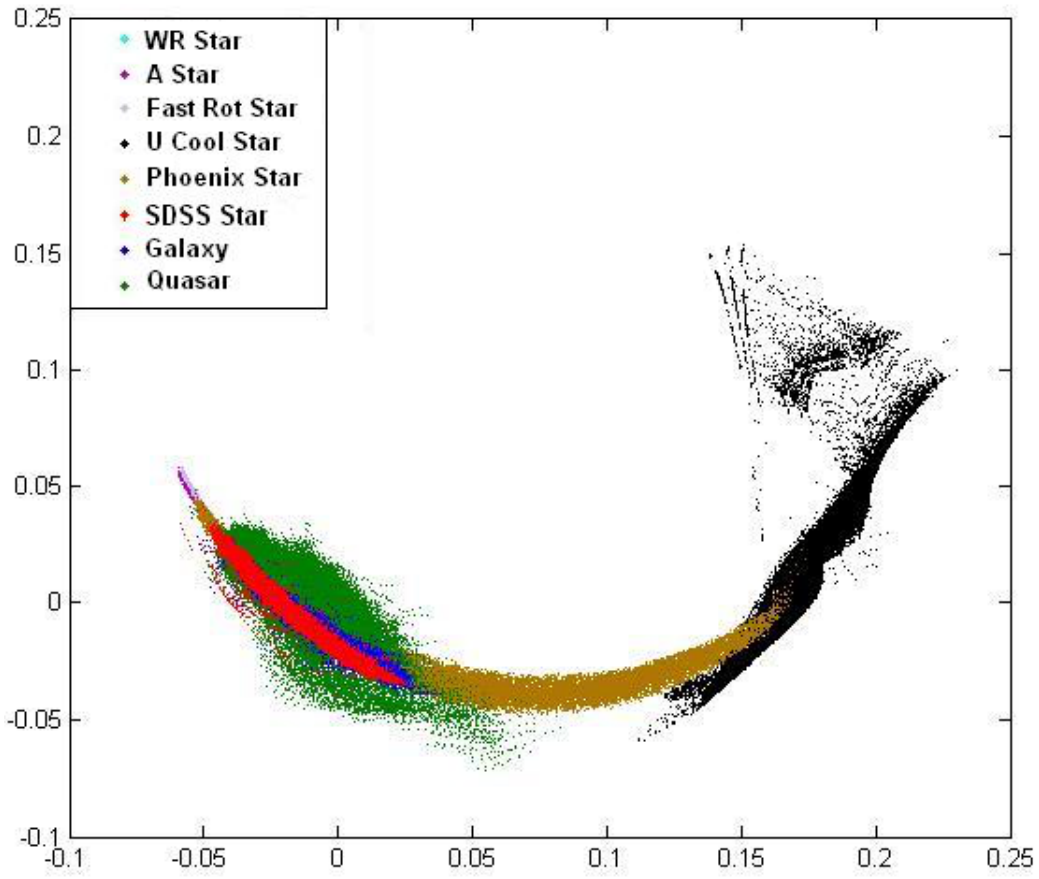


Figura 3.2: Representación de las diferentes clases.

Podemos observar el gran solapamiento que existe entre las clases y la dificultad que tendremos para realizar un clustering correcto de los datos.

Clases como las estrellas tipo A o las Wolf-Rayet, se encuentran a la izquierda de la imagen sin existir solapamiento, sin embargo el problema con estos datos como comentamos anteriormente, es la poca cantidad disponible, lo que provoca que exista un gran desbalanceo entre las clases suponiendo un problema para las tareas de agrupamiento.

Los datos provenientes de *Sloan Digital Sky Survey* se encuentran totalmente solapados, mientras que las estrellas con origen sintético del corpus Phoenix cumplen un amplio espectro que va desde el solapamiento con los datos de SDSS hasta alcanzar el rango de estrellas Ultra Frías, que son las únicas que se encuentran separadas de las demás y sólo solapan parcialmente con el corpus de Phoenix.

Observamos que esta representación a través de los componentes principales tiene una relación directa con la temperatura de las estrellas como parámetro físico y con su composición, pues la temperatura tiene relación directa con el flujo o energía que incide en cada uno de los espectros.

A la izquierda de la imagen encontramos estrellas de tipo Wolf-Rayet y A, siendo los tipos de estrellas más calientes.

Observamos también, como comentamos en la descripción de los datos, que las estrellas procedentes de SDDS están acotadas en un rango de temperaturas que sabemos son las más usuales, sin contener estrellas muy calientes o muy frías.

A la derecha de la gráfica encontramos las estrellas Ultra Frias.

Se puede observar cierta diferencia entre algunos de los datos de Ultra Frías, explicables por la diferente procedencia, ya que como vimos en la descripción de los datos, este corpus de datos en concreto, está formado por dos simulaciones distintas, utilizando diferente metodología en la generación.

Se observa el efecto del oversampling en los puntos de la propia imagen, por ejemplo, observando las tres lineas formadas por los mismos puntos con diferentes desplazamientos en la parte superior derecha de la imagen.

Tener información tanto astrometrica como fotométrica, ya nos indica que podemos y debemos realizar experimentación del agrupamiento desde ambas perspectivas. Un posible nexo de unión para fusionar ambas informaciones puede ser la utilización de la magnitud, presente de forma indirecta en los da-

tos, debido a que encontramos el valor del flujo de energía. Podemos calcular la magnitud G de una fuente con flujo F [42] como:

$$G = -2{,}5 \log_{10} F + zeropoint$$

Donde la constante de la magnitud en el punto cero, es la magnitud de una fuente que tiene un flujo de $1erg/s/cm^2/Angstrom$. En las bibliotecas del proyecto se encuentra bajo la constante $Satellite.MAGNITUDE_{ZEROPOINT}$ y su valor es 25,6632.

La magnitud, G, es una escala exponencial, inversa de la medición del brillo de un objeto en el cielo, es decir, un número de magnitud mayor implica un objeto menos brillante. Objetos muy brillantes tienen incluso un valor negativo y como ya indicamos anteriormente no será útil durante la experimentación para filtrar datos y eliminar aquellos menos brillantes, que coinciden con los datos más afectados por el ruido. En nuestra experimentación podría interesarnos utilizar la magnitud absoluta, que es la magnitud aparente de un objeto si éste se encontrara a una distancia fija del observador; en concreto se utiliza el valor de 10 parsecs o 32.6 años luz, es decir, relacionar la información astrométrica con los espectros a partir del brillo de un objeto normalizado según la distancia, incorporando la magnitud absoluta en la unión de astrometría y espectrometría.

Sin embargo, no es posible la utilización de la magnitud absoluta en todos los experimentos, pues como hemos explicado, la información de la distancia sólo está disponible para las estrellas provenientes de los datos semiempíricos de SDSS, pero no para galaxias o cuásares, por lo tanto siempre que hablemos de magnitud, se tratará de la magnitud aparente en nuestra experimentación.

3.5 Preparación para la experimentación

3.5.1 Extracción y preprocesado de datos

Los ficheros de datos originales se encontraban en un formato propietario, utilizado por el consorcio para el análisis y procesado de datos de GAIA.

El consorcio cuenta con diferentes herramientas desarrolladas en Java, para la visualización de los datos, como la aplicación MDB Explorer y otras [13, 41, 40].

Se realizó un trabajo de análisis de las librerías utilizadas en el repositorio de Gaia y de los códigos fuentes, no siempre disponibles, para poder leer los datos de los ficheros y poder trabajar con ellos, junto con diferentes documentos sobre los datos disponibles, como [51].

Una vez se pudo acceder a los datos, observamos que existe una gran cantidad de información disponible, estructurada por las mismas unidades de trabajo que los utilizan o los producen.

Inicialmente nos interesa acceder al espectro BP y RP para cada uno de los tres *oversamples* y a los atributos astrométricos, paralaje, ascensión recta, declinación y movimiento propio, descritos anteriormente.

También debemos calcular el valor de la magnitud a partir del flujo de energía como explicamos en la sección anterior.

3.5.2 Normalización

Es necesaria la normalización de las instancias de los datos para expresar los atributos de diferentes instancias en la misma magnitud.

La normalización sobre área unidad es una de las normalizaciones más utilizadas por aportar mayor inmunidad al ruido frente otros tipos de normalizaciones; es más robusta que la realizada entre 0 y 1 por ejemplo, o que la estandarización.

La normalización aplicada consiste en dividir cada valor del espectro por el área del propio espectro, de esta forma, los espectros de un mismo objeto visto a diferentes distancias serían idénticos, que es lo que nos interesa.

El área del espectro se obtiene mediante una integral, por ejemplo, utilizando el método de Simpson, o del Trapezoide y posteriormente se divide cada punto del espectro por el valor de dicha integral.

Como aproximación al cálculo integral y pensando en la velocidad de preprocesamiento de los datos obtenidos por GAIA, aproximaremos la función suponiendo que todas las columnas expresadas por los 60 datos que forman

cada uno de los espectros, tienen la misma anchura y normalizaremos cada componente del espectro por su valor dividido por la suma de los valores del espectro, es decir,

$$x_i = \frac{x_i}{\sum_{j=0}^{j=n} x_j}$$

Cuando existan más datos sobre la calibración de Gaia, quizá sea necesaria alguna modificación en esta etapa de preprocesamiento pero con los datos que tenemos, es una buena elección.

Durante le proceso de normalización encontramos los siguientes problemas:

- Presencia de valores negativos en los componentes normalizados.

- Problema de escala en los datos.

El primer problema se debe a la existencia de valores negativos en los componentes antes de normalizar, por la presencia de ruido.

Experimentaremos con otras dos formas de normalizar los datos. En primer lugar dividiendo cada componente por la suma del módulo de los valores y en segundo lugar convertiremos los valores negativos en cero. Comprobamos que realizando esto, estamos distorsionando los valores del espectro, por lo que mantendremos la normalización por área unidad del espectro como nuestra elección.

Observamos a continuación el espectro BP de una instancia de estrella perteneciente al corpus de Phoenix, como ejemplo, sin normalizar y normalizando por la suma del módulo de valores y por la suma del área unidad, para observar el efecto comentado de distorsión que produce una normalización que trate de evitar los valores negativos con un mal criterio.

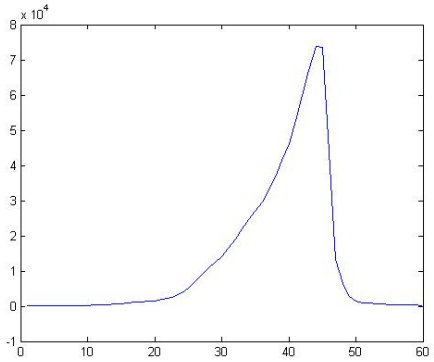


Figura 3.3: Datos sin normalizar, observamos que por la presencia de pequeños valores negativos no empieza en el valor 0 la gráfica y el valor máximo se encuentra cerca de 8×10^4 unidades.

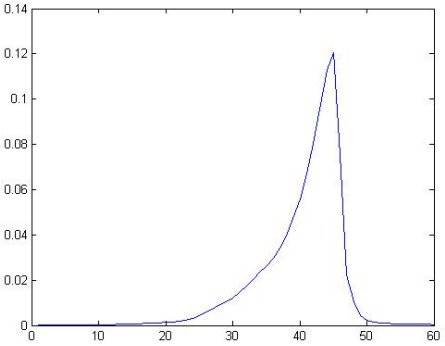


Figura 3.4: Normalización por la suma del módulo, desaparecen los valores negativos pero amplificamos el ruído en la señal, al no contabilizar los valores negativos, por ejemplo en el primer tercio de la señal, estamos incrementando artificialmente el valor del área en esa parte de la función por lo que el resultado en los puntos con mayor flujo es menor, distorsionamos la gráfica e incrementamos el efecto del ruido.

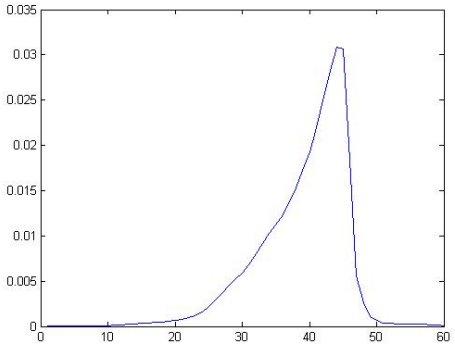
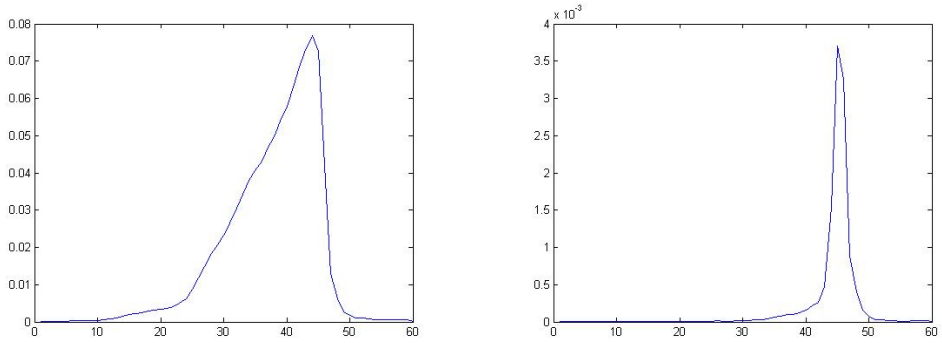


Figura 3.5: La normalización sobre área unidad no distorsiona el espectro. Es más robusta frente a la introducción de ruido, algo necesario para el funcionamiento de los experimentos conforme aumentemos el nivel de ruido. 8

Respecto al segundo problema, observamos que al calcular los componentes principales, las estrellas de tipo Ultra Frías y las estrellas pertenecientes a las simulaciones de Phoenix, se encuentran en una escala totalmente distinta al resto, el problema se debe a la naturaleza de los objetos simulados para los espectros BP y RP por separado. Las simulaciones de estrellas Phoenix, son estrellas relativamente frías y si observamos el flujo que tienen, sus valores en el espectro BP son pequeños y tienen poco flujo y el caso es aún mas grave para las estrellas Ultra Frías, con valores extremadamente pequeños en BP.

Las componentes principales de estas clases no tienen casi flujo en el azul (BP), los objetos fríos no emiten o emiten muy poco, por lo que además están especialmente influenciados por el ruido. Si normalizamos los espectros por separado, esta diferencia entre los flujos de un mismo objeto se perdería. Por este motivo se hace necesario normalizar los atributos teniendo en cuenta el espectro BP y RP en su conjunto y no por separado.

Podemos comparar las imágenes y valores de dos espectros en BP normalizados, de una estrella proveniente de Sloan, con un valor máximo cercano a 0,08 y de una estrella del conjunto de ultras frías, con un valor máximo cercano a 0,004, observando también el total de flujo en el espectro, es decir el área total de la imagen, que es mucho menor.



(a) Estrella SDSS.

(b) Estrella ultrafría.

Figura 3.6: Comparación de espectros BP normalizados de estrellas SDSS y Ultrafrías.

Concluimos por lo tanto, que el procedimiento originalmente planteado de normalizar por separado BP y RP para posteriormente calcular las componentes principales por separado debe cambiar, normalizando ambos, BP junto a RP para cada objeto.

Posteriormente, el análisis de componentes principales lo haremos, por lo tanto, con los datos de ambos espectros unidos, no por separado.

Aunque durante la experimentación con el algoritmo de agrupación no trabajemos con datos provenientes de los corpus de Phoenix ni Ultra Frías, mantendremos esta decisión, pues en los datos provenientes de Gaia obtendremos mediciones de objetos de todo tipo.

3.5.3 Disminución de la dimensionalidad

El problema fundamental al que se enfrentarán los algoritmos de agrupamiento en nuestro trabajo, es la ingente cantidad de datos tomados por Gaia.

La reducción de dimensionalidad, especialmente con los datos de espectrometría, que son 120 atributos (60 del espectro BP y 60 de RP), a un número mucho menor, hará más manejable el problema, mientras perdemos un mínimo de la cantidad de información disponible.

Una técnica ampliamente utilizada es el análisis de componentes princi-

pales, siendo PCA sus siglas en inglés.

El análisis de componentes principales es una técnica estadística de síntesis de la información que tiene sentido si existen altas correlaciones entre las variables, ya que esto es indicativo de que existe información redundante y, por tanto, pocos factores explicarán gran parte de la variabilidad total.

PCA construye por lo tanto, una transformación lineal, donde la elección de los factores, que denominaremos Componentes Principales o CP, se realiza de tal forma que el primero recoja la mayor proporción posible de la variabilidad original, el segundo CP debe recoger la máxima variabilidad posible no recogida por el primero, y así sucesivamente, hasta llegar al porcentaje de variabilidad deseado.

Para construir esta transformación lineal debe construirse primero la matriz de covarianza o matriz de coeficientes de correlación. Debido a la simetría de esta matriz existe una base completa de vectores propios de la misma. La transformación que lleva de las antiguas coordenadas a las coordenadas de la nueva base es precisamente la transformación lineal necesaria para reducir la dimensionalidad de datos.

Una ventaja adicional de la técnica PCA es que permite transformar las variables originales, en general correladas, en nuevas variables no correladas, facilitando la interpretación de los datos.

Además, en el presente trabajo, utilizamos la proyección de las primeras dos y tres CP de los datos, es decir, las que mayor información retienen, para representar gráficamente los datos en dos y tres dimensiones, cuando creamos necesario representar los datos y agrupaciones obtenidas.

No es objetivo del presente trabajo profundizar en las bases matemáticas de la técnica del análisis de componentes principales, podemos encontrar mucho material al respecto, como por ejemplo en [35, 45].

Dada la cantidad de datos para calcular PCA, se utilizó una implementación de PCA en versión *stream* o incremental, es decir, a la que podemos incorporar posteriormente nuevos datos, con unos resultados lo suficientemente buenos como para poder utilizar esta técnica en vez de PCA con todos los datos para construir el modelo, algo que simplemente no será posible con los datos reales de Gaia.

PCA se puede realizar de diferentes formas. Las principales técnicas utilizadas son:

- Descomposición EVD. Obteniendo sobre la matriz de covarianza, los autovalores y autovectores a partir de la llamada descomposición EVD: Sea C la matriz de covarianza, entonces se busca la descomposición $C = EDE^T$ donde E es la matriz de autovectores y D la matriz diagonal con los autovalores.

- Mediante la descomposición SVD. En este caso se busca la descomposición de la covarianza a partir de la matriz de datos X directamente $X = UDV^T$. Donde D es la matriz diagonal con lo autovalores, y U ó V será la matriz de autovectores dependiendo de si el número de objetos es mayor que el número de variables o no.

En el punto 5.4 podemos encontrar pruebas de rendimiento del algoritmo PCA incremental, versión con descomposición EVD.

Debemos encontrar el número adecuado de componentes principales a utilizar en nuestra experimentación.

Pruebas realizadas con PCA versión SVD y EVD, con la versión no incremental del algoritmo implementada y comprobando los resultados con la implementación de *Weka*, nos indican la siguiente cantidad de varianza sobre la información, retenida según el número de componentes principales.

Pct de inf. retenida (%)	#CPs (SVD)	#CPs (EVD)
95	14	2
98	21	3
99	25	4

Llama inmediatamente la atención el pequeñísimo número de componentes principales utilizando la versión con descomposición EVD frente la descomposición SVD para obtener el mismo resultado, las diferencias, especialmente entre el primer y segundo autovalor son mucho mayores en el caso

EVD que en SVD, estos resultados nos expresarían que dado que la mayor parte de la información está contenida en las dos primeras componentes principales, los datos al proyectarse estarían dispuestos en el plano formado por ambas componentes principales, sin embargo la diferencia entre ambos métodos no puede ser tan acusada.

La explicación, es que la varianza no es una buena medida para decidir el número de componentes principales a utilizar, salvo que los datos sigan una distribución gaussiana multivariante.

Por lo tanto se realiza el siguiente experimento, se calculan las diferencias entre el espectro original y el reconstruido como combinación lineal del diferente número de componentes principales.

La Figura 3.7 muestra para el número de componentes utilizado (eje X) el error cuadrático total (SSE) de la diferencia entre el espectro original y el calculado a través de ese número de componentes.

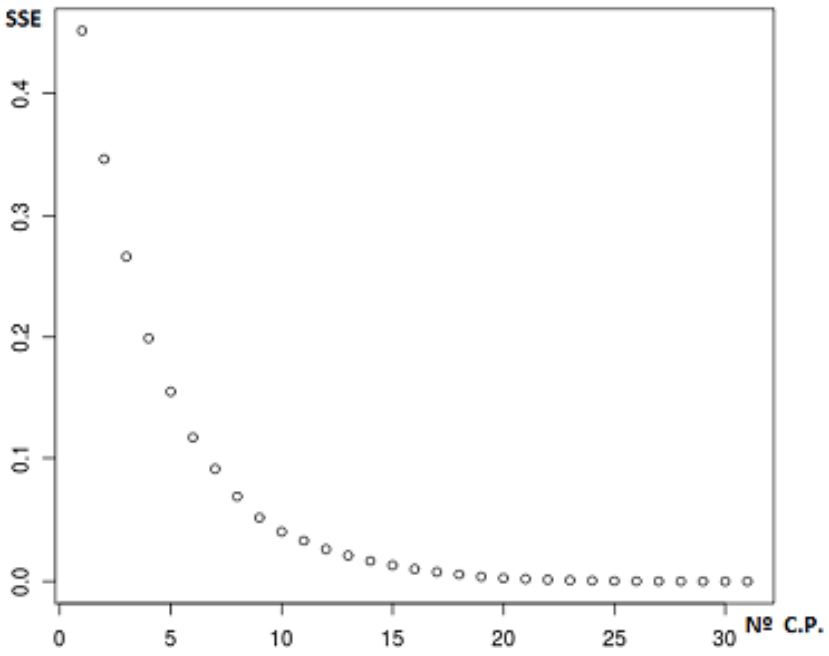


Figura 3.7: SSE de la diferencia entre el espectro original y el calculado por las componentes principales.

A partir de cierto número de componentes se aprecia que el error es muy

pequeño, además cuando los espectros contengan errores habrá que tener cuidado con utilizar más componentes principales de los necesarios, porque los errores de la medida serán mayores que el efecto de despreciar componentes principales.

Esto también significa que en los experimentos, cuanto mayor sea la magnitud y por lo tanto, menos brillante sea un objeto y mayor peso tengan los errores en los datos, necesitaremos, quizá, adaptarnos y utilizar un menor número de componentes principales.

A raíz de estos resultados, parece indicado que utilizar un número entre 14 y 16 componentes principales es adecuado, sin embargo durante los primeros experimentos de agrupamiento, encontramos que utilizando tan sólo 8 componentes principales con la descomposición EVD, obteníamos unos resultados muy similares a la utilización de 15 CPs, con cualquiera de las dos descomposiciones.

Las diferencias, prácticamente no eran perceptibles y dado que los experimentos sufren una gran penalización por la cantidad de datos y su dimensión, nos pareció una buena metodología trabajar en general con 8 componentes principales y en aquellos experimentos que considerásemos importante repetir, utilizar 15 componentes para mejorar los resultados, aunque siempre mejoran de una forma mínima.

Durante los experimentos, se utilizó el algoritmo de PCA en versión incremental o *stream*, con descomposición EVD, dado que fué el primero en encontrarse implementado y comprobado su correcto funcionamiento con pruebas unitarias.

Capítulo 4

Técnicas empleadas en la experimentación

La estructura que engloba el conjunto de tecnologías y metodologías usadas para agrupar los datos se denomina *pipeline*. Dicho pipeline hace uso del paradigma divide y vencerás y de técnicas de muestreo como el K-means++ y K-means# para poder agrupar con el algoritmo HMAC.

A lo largo de este capítulo describiremos las diferentes técnicas y algoritmos utilizados en este trabajo y que se integrarán en el *pipeline*. Con el objetivo de mejorar el rendimiento del agrupamiento, se ha propuesto incorporar el uso de una estructura de vecinos. Aunque dicha estructura aún sólo está propuesta de forma teórica, procederemos a explicar el árbol KD por haber realizado pruebas preliminares para analizar su impacto en el agrupamiento, tanto en la eficiencia como en los grupos resultantes.

4.1 K-medias

K-medias [31] es seguramente el algoritmo de agrupamiento más conocido y utilizado, por lo que no es necesario una descripción en profundidad. Lo presentamos brevemente pues en este trabajo se usan dos técnicas de muestreo (K-means++ y K-means#) relacionadas con el algoritmo.

Dado un número K de grupos, el algoritmo agrupa los datos. Para ello

selecciona, inicialmente, K ejemplos aleatoriamente como centroides iniciales. A partir de aquí el procedimiento sigue iterativamente los siguientes pasos hasta que se alcanza un criterio de parada como la convergencia o un número máximo de iteraciones:

- Cálculo de la distancia de los diferentes objetos o datos a cada uno de los centroides, asignando cada objeto al grupo cuya distancia es mínima respecto a los centroides.
- Actualización de las posiciones de los centros de los grupos basándose en los valores medios de los objetos que pertenecen al conglomerado.

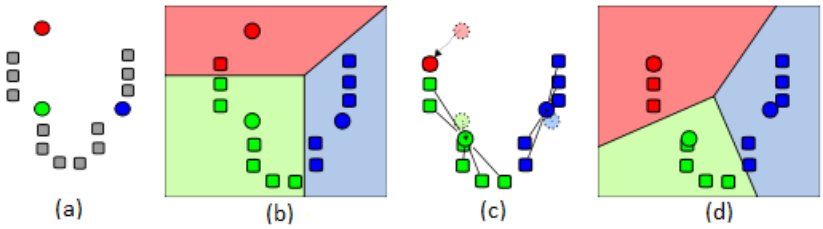


Figura 4.1: Ejemplo de ejecución del K-medias. En (a) se realiza la selección aleatoria de k=3 centroides iniciales (colores rojo, verde y azul) los objetos a agrupar aparecen en color gris. En (b) se generan k clusters asociando cada observación al vecino más cercano, en (c) se calcula el nuevo centroide de cada cluster y se repiten los pasos (b) y (c) hasta que se alcanza la convergencia en (d).

La Figura 4.1 muestra un ejemplo de ejecución del K-medias. Los principales inconvenientes de este algoritmo son:

- Elección de los K grupos iniciales. El número de grupos naturales presente en el problema no es un dato que siempre se conozca a priori.
- La elección de los centroides iniciales. Los resultados del algoritmo y el número de iteraciones hasta la convergencia están influidos por su elección.

- Los grupos encontrados son de forma aproximadamente esférica, no puede manejar datos que formen agrupaciones de formas complejas, tampoco funciona bien con grupos de diferentes tamaños y densidades.

Existen numerosas variaciones del algoritmo, que buscando su mejora en cuanto a eficiencia, rendimiento o ambos aspectos modifican uno o varios de los siguientes puntos del algoritmo:

- Mejorar las condiciones iniciales, la elección de los centroides.
- Método para determinar la distancia de los objetos al centroide, distinto al cálculo de la distancia euclídea o la distancia de Manhattan, utilizadas más habitualmente.
- Cálculo de la pertenencia de variables a un grupo.
- Criterio de parada del algoritmo.

Respecto a la complejidad temporal de K-Medias, vemos en [33] que la mayoría del tiempo se gasta calculando las distancias y existen algoritmos que pueden responder en tiempo lineal $\mathcal{O}(M)$, con un paso de asignación a grupos que calcula $K \times N$ distancias durante I iteraciones, con lo cual el coste asintótico es de $\mathcal{O}(IKNM)$, donde I son las iteraciones, K el número de conglomerados, N el número de datos y M la dimensionalidad del espacio.

4.1.1 K-means++

El K-means++ [3] es una variante del K-medias que busca mejorar su precisión y eficiencia mediante una selección probabilística de los centroides inicales. Por ello a lo largo de este trabajo hacemos referencia al K-means++ como técnica de muestreo más que como variante del K-medias.

Comienza seleccionando aleatoriamente el primer centroide y escoge los siguientes entre los puntos restantes con una probabilidad proporcional a la distancia al cuadrado al centroide más cercano al punto. El pseudocódigo del algoritmo puede verse en la Figura 4.2. El único parámetro de entrada es el número de grupos que buscamos.

Algoritmo *K-means++*(k:entero)
begin
1: $C \leftarrow \emptyset$;
2: $C \leftarrow Escogeraleatoriamenteuncentroide$;
3: $seleccionados \leftarrow 1$;
4: **repetir**;
5: Seleccionar un nuevo centroide $c_i = x' \in X$, con una probabilidad $\frac{D(x',C)^2}{\sum_{x \in X} D(x,C)^2}$;
6: $seleccionados \leftarrow seleccionados + 1$;
7: **hasta** $(k = seleccionados)$;
end

Figura 4.2: Pseudocódigo del K-means++.

$D(x, C)$ Denota la distancia más corta desde un dato x al centroide más cercano que hemos escogido.

Es interesante conocer que K-means++ proporciona una solución competitiva respecto a K-medias, con un coste de tan sólo $\mathcal{O}(\log(k)) - competitivo$ con respecto al agrupamiento óptimo del K-medias como podemos ver demostrado en [3]. La definición de competitivo se utiliza para algoritmos dinámicos u online (se trabaja con datos masivos o en *streaming*) y la idea es comparar los resultados que el algoritmo online tiene con respecto a la versión offline de modo que tengamos una noción de cuánto se distancia la versión online de la offline. En [3] se muestra la medida de competitividad del agrupamiento de los datos de K-medias con K-means++ como inicialización y compara dicho resultado con la del agrupamiento óptimo del algoritmo K-medias.

Definición. ***Competitividad***: Considerando que K-medias busca optimizar el error cuadrático medio (f), definimos la relación de competitividad (R_A) como la fracción máxima entre el valor f obtenido con el algoritmo dinámico A y el valor óptimo de la versión offline; $R_A = max\frac{f_A}{f_{OPT}}$ donde f_{OPT} es el óptimo de la versión offline. Un algoritmo se dice que es competitivo si dicha fracción está acotada. Por ejemplo, si un algoritmo dinámico A es C-competitivo, entonces estamos diciendo que $R_A \leq C$.

4.1.2 K-means#

Este algoritmo, como podemos ver en [2], se trata de una mejora basada en la anterior técnica y por lo tanto se centra de nuevo en la mejora de la inicialización de K-medias.

Es similar a K-means++ excepto que en cada ronda para escoger los centroides, escogemos $k \times \log(k)$ centroides en vez de un único centroide. El pseudocódigo del algoritmo se muestra en la Figura 4.3.

Algoritmo *K-means#*(k:entero)
begin
1: $C \leftarrow \emptyset$;
2: $C \leftarrow$ Escoger aleatoriamente $k \times \log(k)$ centroides;
3: $i \leftarrow 1$;
4: **repetir**;
5: Seleccionar $k \times \log(k)$ centroides, independientemente, con una probabilidad $\frac{D(x',C)^2}{\sum_{x \in X} D(x,C)^2}$;
6: $i \leftarrow i + 1$;
7: **hasta** $(i = k)$;
end

Figura 4.3: Pseudocódigo del K-means#.

$D(x, C)$ Denota las distancias más cortas al subconjunto de puntos escogidos en rondas previas.

En [2] muestran que es un algoritmo O(1)-competitivo; de modo que mejora la anterior cota de K-means++ a un valor constante, seleccionando un mayor número de instancias que el número de centroides K que queremos.

4.2 Algoritmo HMAC/MAC

4.2.1 Algoritmo de tipo EM

EM (*Expectation-Maximization*) pertenece a una familia de modelos conocidos como *finite mixtures* o mezcla de distribuciones, dentro del llamado

clustering de tipo probabilístico, donde en vez de utilizar conceptos como la distancia a centroides o puntos representativos, densidades de datos, etc, los objetos tienen una cierta probabilidad de pertenecer a un grupo o cluster y donde buscamos, para un conjunto de datos, el grupo de conglomerados más probable.

La mezcla de distribuciones, es un conjunto de K distribuciones que representarían los K clusters, estas distribuciones pueden ser de cualquier tipo, aunque para datos continuos se utiliza de forma común la distribución normal, en este caso dado un conjunto de datos, debemos determinar las K distribuciones normales (medias y varianzas) y las probabilidades particulares de cada distribución

Dadas dos distribuciones A y B por ejemplo, con sus medias y varianzas, la probabilidad de que un objeto x pertenezca a un cluster, por ejemplo A es:

$$P(A|x) = \frac{P(x|A)P(A)}{P(x)} = \frac{f(x;\mu_A,\sigma_A)P(A)}{P(x)}$$

donde $f(x;\mu_A,\sigma_A) = \frac{1}{\sqrt{(2\pi\sigma^2)}}e^{-\frac{(x-\mu)^2}{3\sigma^2}}$ es la distribución normal

El problema es que no sabemos de qué distribución viene cada dato y no conocemos los parámetros de las distribuciones. Para obtener la función de densidad de probabilidad o fdp desconocida a la que pertenecen los datos utilizaremos un algoritmo de tipo EM.

Éste comienza adivinando los parámetros de las distribuciones y los utiliza para calcular las probabilidades de que cada objeto pertenezca a un cluster y usa estas probabilidades para volver a estimar los parámetros de las probabilidades, de forma iterativa. El cálculo de las probabilidades de las clases o los valores esperados de las clases es la parte llamada *Expectation*, mientras que el paso de calcular los valores de los parámetros de las distribuciones es llamado *Maximization*, pues se maximiza la verosimilitud de las distribuciones dados los datos.

El algoritmo tiende a converger, sin llegar a un punto fijo, por lo que debemos fijar un umbral como condición de parada y aunque la convergencia del algoritmo está garantizada de forma teórica, esta convergencia puede ser

hacia un máximo local.

Con una explicación un poco más formal, podemos decir que la función de densidad de probabilidad se puede aproximar mediante una combinación lineal de NC componentes definidas a falta de una serie de parámetros $\{\Theta\} = \cup\{\Theta_j, \forall\ j = 1, \cdots, NC\}$ que son los que hay que averiguar, $P(x) = \sum_{j=1}^{NC} \pi_j p(x; \Theta_j)$ con $\sum_{j=1}^{NC} \pi_j = 1$, donde π_j son las probabilidades a priori de cada cluster cuya suma debe ser 1, que también forman parte de la solución buscada, $P(x)$ denota la función de densidad de probabilidad y $p(x; \Theta_j)$ la función de densidad del componente j.

La fdp estimada usualmente, corresponde con una forma de distribución normal y utilizando combinaciones lineales de estas distribuciones y con un número suficiente de éstas, ajustando sus medias, covarianzas y los coeficientes de la combinación lineal, se puede aproximar cualquier fdp hasta una precisión arbitraria.

El ajuste de los parámetros del modelo requiere de una medida de bondad, para conocer cómo encajan los datos sobre la distribución que los representa, este valor de bondad se conoce como *likelihood* o verosimilitud, de los datos, por lo tanto se trataría de estimar los parámetros buscados Θ maximizando la verosimilitud, calculándose normalmente su logaritmo, conocido como *log-likelihood*, debido a que es más sencillo su cálculo de forma analítica, siendo la solución obtenida la misma debido a la propiedad de monotonicidad del logaritmo.

La forma de esta función *log-likelihood* es

$$L(\Theta, \pi) = \log \Pi_{n=1}^{NI} P(x_n)$$

donde NI es el número de instancias que suponemos independientes entre si.

El algoritmo EM, procede en dos pasos que se repetirán de forma iterativa, dando nombre al proceso:

- *Expectation*: Se calculan las probabilidades a posteriori para cada punto de haber sido generado por cada una de las distribuciones existentes, usando la estimación actual de los parámetros, que es proporcionada

por el paso *Maximization* de la iteración anterior o los valores iniciales de los parámetros la primera vez.

- *Maximization*: Proporcionalmente al peso de cada dato perteneciente a una distribución se modifican los parámetros del paso anterior, maximizando la verosimilitud.

Después de una serie de iteraciones el algoritmo tiende a converger a un máximo local de la función L, esta convergencia será rápida en las iteraciones iniciales y cada vez más lenta y utilizaremos un valor umbral de mejora o cambio de las soluciones entre iteraciones, como criterio de parada del algoritmo. El algoritmo EM requiere muchas más iteraciones para alcanzar una convergencia aproximada que otros algoritmos como K-Medias y los cálculos de cada iteración son significativamente mas costosos en tiempo de cálculo.

Finalmente, obtendremos un conjunto de clusters cada no de ellos definido por los parámetros de una distribución de la forma de la fdp utilizada, por ejemplo una gaussiana o normal.

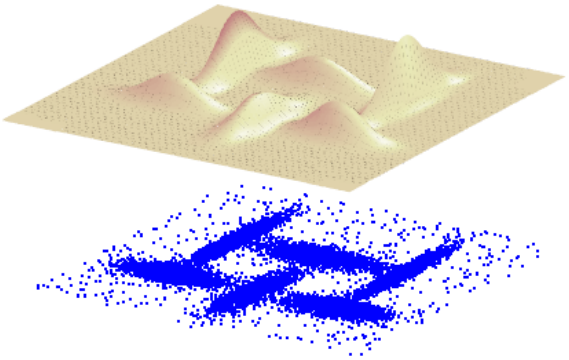


Figura 4.4: Modelo de distribuciones para 6 grupos.

Vemos en 4.5 un ejemplo para ilustrar el comportamiento del algoritmo EM, extraído del libro de Bishop [4].

- a) Los datos iniciales en verde y la configuración inicial del modelo de mezclas, en dos puntos aleatorios.

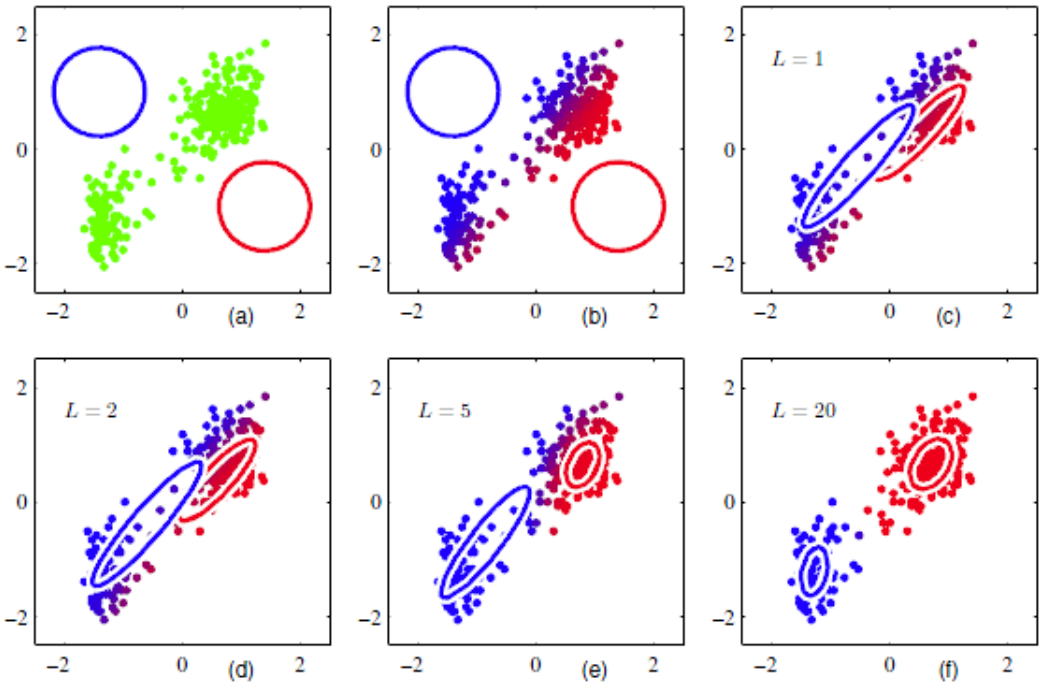


Figura 4.5: Ejemplo de una mezcla de dos distribuciones normales.

- (b) Resultado del paso E inicial. Se calculan las probabilidades a posteriori para cada punto de haber sido generado por cada una de las dos distribuciones azul o roja, el color de cada punto representa la proporción de pertenecer a para cada distribución. Los puntos morados por ejemplo indican similar probabilidad de haber sido generados por cualquiera de los dos clusters.

- (c) Resultado del primer paso M, la media de cada distribución se ha movido al punto central proporcionalmente al peso de cada uno de los datos que pertenecen a su distribución y de forma similar la covarianza.

- (d), (e) y (f) muestran los resultados tras 2, 5 u 20 ciclos completos respectivamente de EM.

- En (f) el algoritmo está próximo a converger.

4.2.2 Mode Association Clustering MAC

Como comentamos en el punto anterior, MAC [29] pertenece al clustering probabilístico, en concreto cada cluster estará caracterizado por una distribución multivariada Gaussiana, por tratarse de datos continuos (con datos discretos se utilizan distribuciones de Poisson), y la función de densidad de probabilidad está formada por una mezcla de distribuciones paramétricas.

El procedimiento del algoritmo sigue los pasos de un algoritmo EM, explicado anteriormente, para obtener las diferentes distribuciones y se asigna cada punto a un cluster o componente en base al cálculo de las probabilidades a posteriori. El razonamiento detrás del método EM parece indicar que cada componente posee una colina, la moda de esa distribución, separada de las demás colinas de otras componentes pero esto no se cumple para Gaussianas lo suficientemente cercanas, que resultan en una única componente.

En este caso igualar una componente con un sólo cluster es incorrecto, MAC sigue una aproximación no paramétrica que trata de evitar estos problemas.

Para evitar estas restricciones, se modelan los datos utilizando un kernel, usando un nuevo algoritmo llamado Modal EM (MEM) que permite encontrar un camino creciente desde cualquier punto a un máximo local de la densidad, es decir, a una colina. El nuevo algoritmo agrupará los puntos en un cluster si están asociados con la misma colina, llamando a esta aproximación agrupamiento modal.

Profundizamos a continuación en estas dos partes del algoritmo, Modal EM y la propia agrupación.

Modal EM

El algoritmo *Modal EM* (MEM), resuelve un máximo local de una mezcla de densidades a través de iteraciones ascendentes empezando desde cualquier punto inicial, lo cual supone frente algunos métodos una ventaja, pues se vuelve irrelevante la inicialización de los datos.

Este algoritmo se compone de los conocidos pasos EM pero su objetivo es diferente, ya que EM trata de estimar los parámetros maximizando la medida

de bondad, likelihood, o verosimilitud, mientras que el objetivo de MEM es encontrar los máximos locales, las modas de una distribución dada.

Por lo tanto dada la anterior fórmula, vista con los algoritmos de tipo EM, sea una mezcla de densidades $f(x) = \sum_{j=1}^{J} \pi_j f_j(x)$ donde π_j son las probabilidades a priori del componente j y $f_j(x)$ es la densidad del componente j.

Dado un valor inicial x^0 MEM encuentra un máximo local de la mezcla alternando los siguientes pasos hasta que se cumple un criterio de parada del algoritmo.

1. Paso 1. *Expectation*, donde la probabilidad a posteriori de cada componente de la mezcla en el punto actual x^r es calculada.

2. Paso 2. *Maximization*, asumiendo que $\sum_{j=1}^{J} p_j \log(f_j(x))$ tiene un único máximo que es cierto cuando las densidades de $f_j(x)$ siguen una distribución normal, como en los algoritmos EM, se cumple que la resolución analítica de $\sum_{j=1}^{J} p_j \log(f_j(x))$ es más sencilla que $\sum_{j=1}^{J} \pi_j f_j(x)$ por eso la utilizamos en su lugar.

Agrupación por identificación de modas

Si las funciones de densidades condicionales de una clase son desconocidas, pueden ser estimadas usando la muestra tomada y aplicando métodos no paramétricos de estimación, como vemos en el texto de Edgar Acuña [1], uno de ellos es el método de *kernels* que es el utilizado por MAC.

El *kernel* o núcleo es una función simétrica que hace una representación real de los datos para luego hacer una ponderación, de esta manera, se tiene un estimador para $f(x)$.

Existen diferentes diseños de núcleo, que caracterizan la forma que tomaría, sobre un determinado punto x, MAC utiliza un *kernel* de tipo Gaussiano $\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$ donde $u = \frac{(x-x_i)}{h}$ siendo x el punto donde se trata de estimar la densidad, x_i el valor de la variable en el caso $i = 1, \cdots, n$ y h la anchura de la ventana, parámetro de suavizado o *bandwidth*.

El objetivo de MAC, es estimar, dado un conjunto de datos, las funciones de densidad de probabilidad de los diferentes conglomerados, de forma no paramétrica utilizando núcleos Gaussianos.

La aproximación no paramétrica se origina para evitar la asunciones restrictivas impuestas por la aproximación paramétrica sobre la forma de la función de densidad de probabilidad, estimándola en nuestro caso, directamente a través de los datos, utilizando los kernels como estimadores.

Dado que le estimación de densidad por núcleo está en la forma de una mezcla de distribuciones, se aplica MEM para encontrar una moda usando cada punto como el valor inicial para la iteración.

Formalmente, dado un conjunto de datos $S = \{x_1, x_2, \ldots, x_n\}, x_j \in \mathbb{R}^d$ la estimación de densidad por kernel Gaussiano es $\{(x) = \sum_{i=1}^{n} \frac{1}{n} \Phi(x|x_i, \Sigma)$, donde la función de densidad Gaussiana es $\Phi(x|x_i, \Sigma)$.

Un aspecto clave en el algoritmo de agrupación es la identificación de una moda empezando desde cualquier punto x_i ya que MEM se mueve desde x_i a través de un camino ascendente a una moda. Los puntos que ascienden a la misma moda, se localizan en la misma cuesta y por lo tanto se agrupan en un conglomerado. Éste proceso es llamado MAC (*Mode Association Clustering*), obteniéndose una función de densidad de probabilidad para cada conglomerado, de forma que los puntos quedan agrupados en las diferentes modas y nuevos puntos no pertenecientes a las muestras pueden ser asignados a un *cluster*.

La elección del ancho de banda o bandwidth, es de crucial importancia en la estimación de densidad. Éste indica el grado de suavizamiento precisando a qué distancia se le permite a las observaciones estar del punto donde se estima la densidad y contribuir a la estimación de la función de densidad. En este sentido determina el grado de dependencia del estimador sobre la información cercana al punto de estimación. Cuando se tienen anchos de banda grandes se obtienen mayores sesgos pero menor varianza. Ocurre lo contrario cuando se tienen anchos de banda pequeños [11].

La elección apropiada del parámetro de suavizamiento estará siempre influenciada por el uso que se hará del estimador de densidad como afirma Silverman en [47]. De esta manera, si se desea realizar una exploración de

datos para sugerir posibles modelos e hipótesis, el parámetro se elige subjetivamente. El problema de la elección objetiva del parámetro de suavizamiento es un asunto de interés en diferentes investigaciones y no existe un método universalmente aceptado [46].

De esta forma, en MAC, cuando el ancho de banda se incrementa la estimación de densidad por el *kernel* se suaviza y más puntos tienden a ascender a la misma moda mientras que al disminuir el bandwidth, puntos que antes convergían a la misma moda ahora lo hacen a modas distintas, obteniendo un mayor número de conglomerados.

Pseudocódigo

Algoritmo *MAC*
begin
1: $probabilidad\_a\_priori \leftarrow \frac{1}{\#datos}$
2: $modas\_iniciales \leftarrow datos$
3: $lista\_modas \leftarrow ObtenerListaModas(datos, probabilidad\_a\_priori, bandwidth, modas\_iniciales)$
4: $resultados \leftarrow AgruparModas(lista\_modas)$
end

Figura 4.6: Pseudocódigo de MAC.

En 4.6 vemos el pseudocódigo de MAC, la función *ObtenerListaModas* calcula las modas del conglomerado a partir del algoritmo EM, es decir realiza el paso MEM explicado anteriormente 4.2.2, mientras que *AgruparModas* realiza el paso de la agrupación por la identificación de las modas visto en 4.2.2.

Podemos observar el pseudocódigo de todos los métodos principales en el anexo B.

HMAC

HMAC (*Hierarchical* MAC) surge de forma natural por el efecto del ancho de banda del kernel explicado anteriormente y no es más que una repetición

de la ejecución de MAC con diferentes valores ordenados de bandwidth, de forma que obtendremos de forma natural una agrupación jerárquica, dado que con valores pequeños del ancho de banda obtendremos un mayor número de conglomerados y al repetir la ejecución de MAC con mayores valores de *bandwidth*, puntos que convergían a diferentes modas convergerán a la misma moda.

HMAC se diferencia de la estrategia utilizada por los algoritmos jerárquicos que agrupan los grupos cercanos al ascender en la jerarquía, dado que en HMAC el efecto sobre los grupos de cada punto de los datos es tenido en cuenta en cualquier nivel de la jerarquía. A cambio de la mayor precisión que presenta esta aproximación, el rendimiento de ejecutar sucesivamente MAC para los diferentes anchos de banda se ve perjudicado frente a la simple unión de grupos basándose en distancias entre sus centroides o entre sus componentes aplicado por otros algoritmos de agrupación de tipo jerárquico.

En el anexo D, podemos observar de forma gráfica los resultados de HMAC ejecutado sobre un conjunto de datos sintéticos con 3 distribuciones normales parcialmente solapadas y cómo varían los resultados según el valor de *bandwidth*.

4.3 Árbol KD

Introducimos brevemente esta estructura de datos, también conocida como árbol k-dimensional, o KD-Tree puesto que una posible mejora teórica propuesta sobre el algoritmo MAC, que estudiaremos de forma experimental, se basa en la utilización de esta estructura para acelerar el cálculo de las probabilidades a posteriori calculadas por MAC.

Se trata de una estructura de datos de particionado del espacio que organiza los puntos en un espacio euclídeo de k dimensiones, empleando planos perpendiculares a uno de los ejes del sistema de coordenadas y en donde todos los nodos almacenan un punto, no sólo las hojas. Es un árbol binario en el que cada nodo es un punto k-dimensional, donde cada nodo en el árbol está asociado a una de las k dimensiones con el hiperplano perpendicular

a los ejes de esa dimensión y el punto seleccionado para crear el plano de corte será la mediana de los puntos en esa dimensión, de forma que todos los puntos situados en el subárbol izquierdo tendrán un valor menor en la dimensión de ese nivel del árbol y los puntos del subárbol derecho, mayor.

Algoritmo *KDtree*(puntos:lista,profundida:entero)
begin
1: eje ← profundidad mod k;
2: selecciona mediana del eje de lista de puntos;
3: nodo.localizacion ← mediana;
4: nodo.hijoIzquierdo ← kdtree(puntos anteriores a mediana, profundidad + 1);
5: nodo.hijoDerecho ← kdtree(puntos posteriores a mediana, profundidad + 1);
6: devuelve nodo;
end

Figura 4.7: Pseudocódigo del árbol KD.

La Figura 4.8 muestra un ejemplo de árbol KD Aplicado sobre una pequeña lista de puntos.

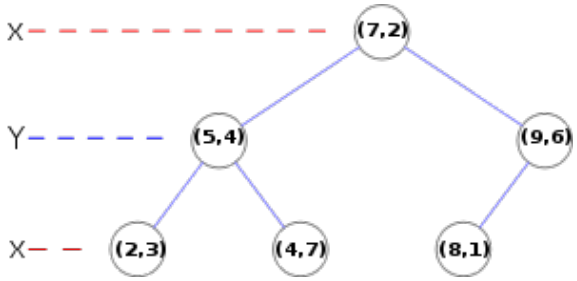


Figura 4.8: Ejemplo de árbol KD.

Nos interesan conocer los costes de diferentes operaciones sobre los datos del árbol:

- Construcción del árbol tiene un coste de $\mathcal{O}(n \log^2(n))$ si se utiliza para el cálculo de la mediana en cada nivel un algoritmo de ordenación con coste $\mathcal{O}(n \log(n))$ como *heapsort*.

- Inserción de un nuevo punto: $\mathcal{O}(\log(n))$.
- Borrado de un punto: $\mathcal{O}(\log(n))$.
- Petición de un rango de datos en el mismo nivel del árbol $\mathcal{O}(n^{1-1/k}+m)$ donde m es el número de puntos afectados, y k la dimensión del árbol kd.

La búsqueda de los vecinos más cercanos en el árbol, próximos a un punto dado se puede realizar de forma muy eficiente dada la forma de organizar la información que tiene esta estructura de datos, debido a que se eliminan rápidamente grandes porciones del espacio de búsqueda, propiedad que nos interesa para nuestro propósito.

El objetivo del uso del árbol kd, es que el algoritmo MAC no tenga que calcular la contribución de todos los puntos para el cálculo de la probabilidad a posteriori, sino que únicamente accederá a los puntos distribuidos sobre los nodos cercanos en el árbol kd, ya que contendrán los puntos próximos que contribuyen al cómputo de la probabilidad.

Capítulo 5

Escalando el algoritmo

Una de los principales problemas de las técnicas de agrupamiento es la escalabilidad del algoritmo para poder manejar una gran cantidad de datos. Para que un algoritmo sea escalable, dados unos recursos limitados de memoria y espacio en disco, su tiempo de ejecución debería crecer linealmente respecto al tamaño de los datos [16, 5]. De forma menos formal Provost [39] entiende la escalabilidad como la capacidad de procesar grandes volúmenes de datos de forma eficiente encontrando los mejores modelos posibles.

Gaia proporcionará del orden de 10^9 datos; de modo que es necesario desarrollar una estrategia para escalar HMAC tanto espacial como temporalmente.

La escalabilidad espacial hace referencia a que los datos deben de poder estar accesibles en memoria para el algoritmo independientemente de su coste temporal. Si los datos no están disponibles en memoria el problema no escalará. Para lograrlo, la estructura propuesta usa el paradigma divide y vencerás en combinación con procesar sólo un subconjunto de puntos denominados centroides. Cada punto de dicho subconjunto representa un conjunto de puntos del conjunto original. A partir de un centroide podremos acceder a los datos que lo formaron, transformando los conglomerados resultantes compuestos por centroides, por los puntos que representan.

En cuanto a la escalabilidad temporal, la utilización de los centroides re-

ducirá la magnitud del problema, con lo que un problema que no era tratable en principio con HMAC, podrá ser resuelto con este algoritmo. Con el objetivo de mejorar la selección de los centroides, se presenta un modelo multinivel de selección de los centroides.

5.1 *Pipeline* para el procesado de datos

La complejidad temporal de HMAC así como la limitación de recursos espaciales impiden que apliquemos directamente el algoritmo. Considerando que se conoce el orden de magnitud del número de objetos que conformarán la muestra, pasamos a introducir el *pipeline* que llevará a cabo el paquete de trabajo OCA para proceder con el agrupamiento.

Dicho *pipeline* fue presentado al consorcio DPAC por el grupo OCA y consta de tres fases principales:

- Fase inicial de preprocesamiento.
- Fase de muestreo.
- Fase de agrupación.
  - Obtención de centroides. (K-means++ / K-means#)
  - HMAC

Fase inicial de preprocesamiento

Se basará en el trabajo realizado de estudio de los datos simulados, realizados en el capítulo 3, que aporta los siguientes puntos para la fase de preprocesamiento de los datos en el *pipeline*:

- Elección del proceso de normalización de los datos.
- Utilización del algoritmo PCA en versión incremental para poder ser aplicado a todos los datos.
- Estudio de los métodos de descomposición para proceder con el PCA.

- Número de componentes principales que debemos utilizar.
- Elección de los atributos a utilizar de los datos.

Fase de muestreo.

Esta fase hace referencia al modo en que se proporciona el flujo de datos a nuestro algoritmo.

Los datos de Gaia, no son proporcionados al *pipeline* en una única partición sino que se nos proporcionará como un flujo de datos divididos en grupos o particiones distintas. El sistema de archivos distribuido utiliza *Hadoop* [38]

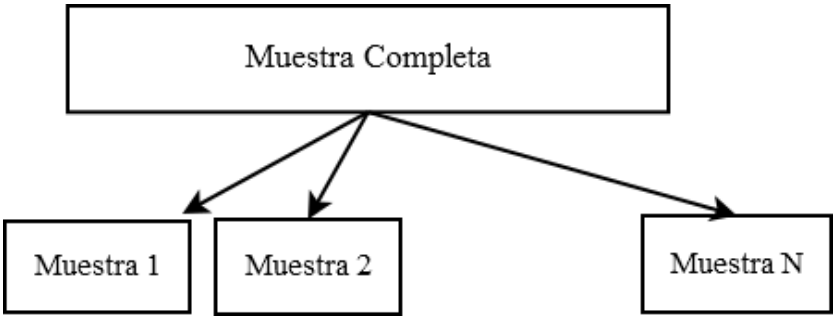


Figura 5.1: Fase de muestreo del pipeline.

mientras que la base de datos se estructura alrededor de HEALPix [54] que es un esquema aceptado y utilizado por la comunidad astronómica para muestrear los objetos.

El motivo de la utilización de este esquema en vez de un acceso clásico a través de una base de datos de propósito general es que aunque existen bases de datos con capacidades de indexación espacial para conjuntos de datos multidimensionales, el uso de estas características no es sencillo, no está optimizado para el uso astronómico y su eficiencia es inadecuada para el comportamiento exigido. HEALPix *(Hierarchical Equal Area isoLatitude Pixelization)* como vemos en [21], fue desarrollado originalmente para solucionar las necesidades de procesamiento de datos y análisis en experimentos de astronomía, distribuyendo o proyectando los datos, sobre una esfera.

Se trata de un marco de trabajo que permite una indexación jerárquica de la información, a diferentes niveles, que trata de facilitar el trabajo con tareas

como el análisis global de los problemas, favorecer el análisis morfológico de la información, detección identificación y caracterización de objetos, e incluso simplificar la correlación espacial o espectral con conjuntos de datos externos.

En definitiva, HEALPix acapara muchas capacidades:

- Es una metodología para la discretización, análisis y síntesis de datos distribuidos en la esfera, trabajaremos con particiones de la esfera celeste que contendrán objetos y por lo tanto debido a que la densidad de los objetos no es igual en todas las particiones de la esfera, el número de datos en las muestras será variable.

- Es una estructura de datos intermedia, una capa funcional entre los datos astronómicos y el dominio de la aplicación.

- Es un formato de mapa celeste, con herramientas asociadas de análisis y visualización.

- Se ha convertido en un estándar de facto, en una interfaz entre las tecnologías de la información y la ciencia espacial.

Fase de agrupación.

Una vez tenemos un particionado de los datos, realizado durante la fase previa, procederemos a usar una estrategia o paradigma de divide y vencerás, aplicando a cada muestra de datos el algoritmo K-means# y obteniendo una muestra de centroides para cada partición.

Recordemos que con K-means#, obtendremos un grupo de centroides, con una estrategia más conservadora que K-means++, ya que escogeremos un número mayor de objetos como centroides.

Estos objetos escogidos, serán los que tengan una mayor probabilidad de pertenecer a grupos distintos dentro de cada muestra y se fusionan formando una nueva muestra de datos, que contiene sólo estos centroides, por lo que reduciremos el tamaño del problema.

Esta reducción dependerá de los resultados que obtengamos, una vez finalizada la estructura del *pipeline* en las experimentaciones que llevemos a cabo. Esta unión de centroides al provenir de las muestras particionadas de

los datos globales, van a contener objetos que pertenecen a regiones que se solapan, por lo tanto aplicaremos esta vez el algoritmo K-means++, para obtener el número de centroides globales requeridos, ya que el número de grupos deseados k es un parámetro de entrada del problema.

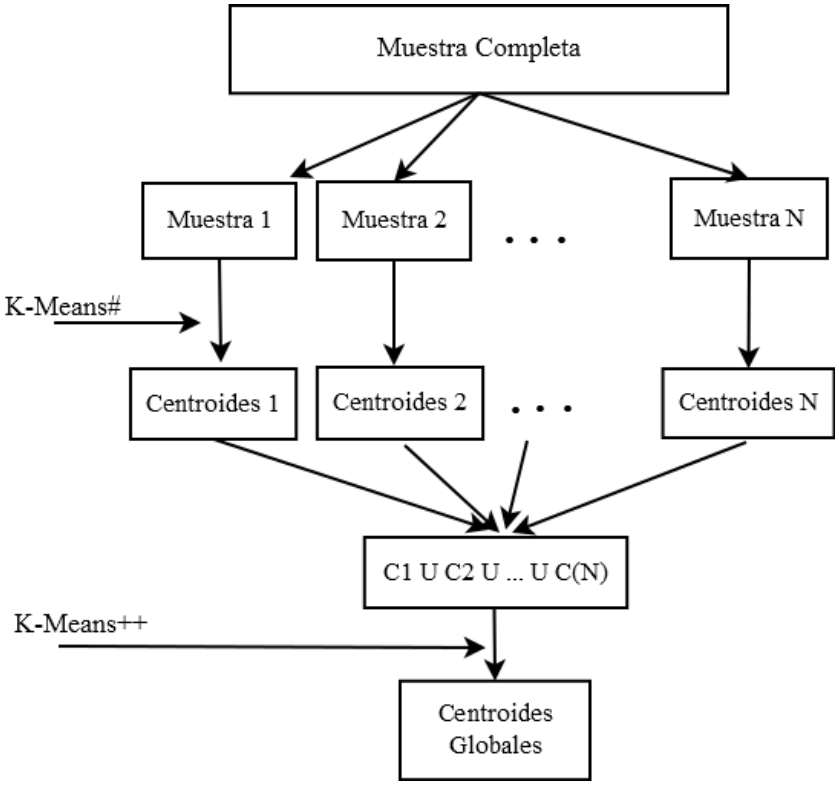


Figura 5.2: Fase de agrupación del pipeline.

El número de centroides globales que buscamos será un número muy elevado, ya que la aplicación de K-means# y K-means++, tiene como objetivo resumir los puntos muy cercanos por el centroide del grupo que les contiene, por lo que de forma jerárquica reduciremos el número de instancias del problema.La figura 5.3 muestra un esquema de este proceso.

A partir de los centroides se podrá volver a los objetos originales, los puntos agrupados por ese centroide, actualmente se encuentra el desarrollo del *pipeline* en esta fase, en la búsqueda de una estructura de datos óptima

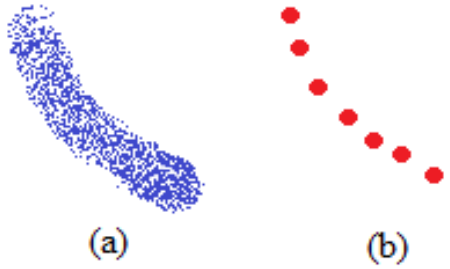


Figura 5.3: Representación del proceso de resumen de información. La idea es reducir el número de puntos de (a) por los centroides encontrados en (b) reduciendo el número de datos en un factor muy importante.

que guarde la referencia a los objetos representados por un centroide y a su vez sea fácil encontrar para un objeto a que centroide pertenece.

Finalmente sobre estos centroides globales que resumen la información de la muestra, se ejecutará el algoritmo de agrupación HMAC y con los resultados obtenidos, podremos descender desde los centroides agrupados en los clusters a los datos que pertenecen a esos centroides.

Tendríamos una agrupación con una doble jerarquía, por un lado la introducida por el algoritmo HMAC, que con la variación del ancho de banda, encontraría diferentes niveles de agrupación de objetos y por otro lado, estos objetos pertenecientes a los conglomerados serían los centroides, desde los cuales se podría llegar a las instancias o puntos que lo generaron.

El número de centroides para realizar la agrupación y disminuir el tamaño del problema se valorará experimentalmente cuando el *pipeline* se encuentre en una fase más avanzada de desarrollo, pero se espera una reducción del problema original de 10^9 datos a un número entre 10^5 y 10^7 datos para la ejecución del algoritmo HMAC.

También es posible aplicar la misma estrategia con varios niveles, dando lugar a la alternativa multinivel, que es la generalización de la anterior técnica para n niveles. La variación presentada trata de aplicar en repetidos niveles K-means# en primer lugar con las muestras de datos para obtener los centroides. Posteriormente repetimos la ejecución del algoritmo, con la

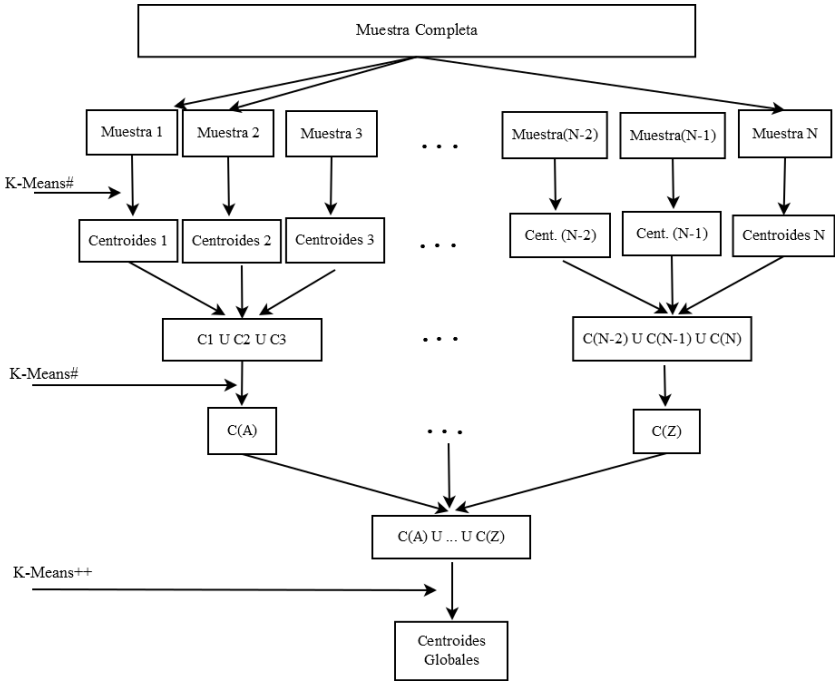


Figura 5.4: Agrupación multinivel para n=2

fusión de varios conjuntos de centroides y este esquema se puede repetir varios niveles, siempre con la fusión de los centroides del nivel anterior, salvo en el primer paso que se ejecuta con las particiones de la muestra completa.

Finalmente, siempre en el último nivel, se ejecutaría K-means++ para obtener el conjunto de centroides final con el número de instancias k introducidas como parámetro de entrada del algoritmo.

La estructura de datos utilizada para mantener la referencia de las instancias que pertenecen a un centroide, ahora debe mantener múltiples niveles, pues un dato puede contener referencias a centroides y éstos a su vez a conjuntos de centroides que contiene y así hasta llegar al nivel donde las referencias son de puntos de las muestras, por lo que la implementación de la estructura multinivel presenta una mayor complejidad.

El objetivo de esta variedad frente a la utilización con tan sólo un nivel, es su utilización cuando la cantidad de datos sea tan grande, que la fusión de los centroides con tan sólo un nivel de profundidad, contenga demasiados

datos para utilizar HMAC.

Encontramos respecto a este proceso de reducción en el número de datos, utilizando técnicas de agrupamiento, trabajos con planteamientos similares como el realizado por Cervantes en [7], en este caso aplicado a algoritmos de tipo SVM o Máquinas de Vectores de Soporte que son un conjunto de algoritmos de aprendizaje supervisado, sin que afectara significativamente la precisión de la clasificación.

Otra posible posible mejora propuesta para su introducción, trata de incorporar la estructura de árbol kd para guardar los datos, su objetivo es que el algoritmo MAC utilice para la fase de cálculo de la probabilidad a posteriori, únicamente los datos situados en los nodos más cercanos dado que son los que más contribuyen en una distribución normal a la estimación de la verosimilitud, para ello seguiremos la llamada regla tres sigma, por lo que los puntos que contribuyen a una distribución normal están en un 99,7 % de los casos a una distancia menor de tres veces la desviación estándar de la media. Se comprueba experimentalmente los resultados de utilizar el algoritmo MAC con y sin el árbol kd al final del presente capítulo y se extraen las debidas conclusiones.

5.2 Medición del coste espacial y temporal

En el presente apartado, mostraremos diferentes experimentos, llevados a cabo para estudiar el comportamiento temporal y espacial de diferentes algoritmos utilizados en el trabajo de forma empírica y cuyos resultados servirán para tomar ciertas decisiones que afectarán a muestra estructura para resolver el problema.

5.2.1 Pruebas de rendimiento de K-means++ y K-means#

Se han realizado dos experimentos distintos:

- En primer lugar con el objetivo de estudiar el comportamiento con un

problema de un tamaño similar al esperado en las muestras de Gaia y el comportamiento no sólo como técnica de inicialización sino cómo afecta a K-medias su utilización respecto a la velocidad de ejecución y calidad de los resultados.

- Un segundo experimento con el objetivo de estudiar el crecimiento del coste temporal respecto al número de instancias y estudiar cómo escalan los algoritmos.

Para la realización de la primera prueba, se ha utilizado una variación de una base de datos de *UCI Machine Learning Repository*, en concreto una variante con datos sintéticos procedentes de la base de datos *Forest Covertype*, con 7 clases y 581,012 puntos con 54 dimensiones o atributos cada uno, con un número de datos dentro del rango esperado por las muestras de Gaia aunque con una dimensionalidad superior.

Se ha realizado el agrupamiento con objetivo de comparar los tiempos de ejecución de las inicializaciones K-means++ y K-means#, estudiando cómo afectan al algoritmo K-medias frente a la inicialización aleatoria original en cuanto a tiempo de ejecución del algoritmo.

Los resultados en cuanto a la eficiencia de K-medias, no son objetivo de nuestro estudio, dado que nuestro interés se centra únicamente en la obtención de un buen subconjunto de centroides, a los que aplicaremos HMAC. Simplemente mencionaremos que como era de esperar una mejor inicialización de K-medias frente a la inicialización aleatoria supone también una mejora de los resultados tal y como puede verse en la Tabla 5.1.

Tabla 5.1: Porcentaje de acierto del K-medias con varias inicializaciones.

Inicialización	% Aciertos
Aleatoria	67.29 %
K-means++	73.33 %
K-means#	77.25 %

En cuanto al tiempo de ejecución, se usó un ordenador con procesador *Intel Pentium Dual CPU E2140* a *1.60GHz* de velocidad, con *3Gb* de memoria.

Los resultados se muestran en la Tabla 5.2. K-medias mejora su eficiencia si usa, como método de inicialización, K-means++ o K-means#. Los tiempos de ejecución de ambas inicializaciones indican la viabilidad en muestras de aproximadamente 6×10^5 instancias; tamaño similar al de las proporcionadas en Gaia.

Tabla 5.2: Tiempo de ejecución del K-medias con varias inicializaciones. El valor *na* indica no aplicable y se debe a que el tiempo es inferior a la precisión de la medida.

Inicialización	T.Inicialización	T. Ejecución
Aleatoria	*na*	41.8s
K-means++	1.3s	25.7s
K-means#	4.7s	24.1s

Para nuestro segundo experimento, estudiar la variación de forma experimental del coste temporal respecto al tamaño de las instancias, preparamos muestras con datos sintéticos, generadas con diferentes tamaños conteniendo 10 conglomerados de datos, con 25 atributos o dimensiones, siguiendo cada uno de ellos una distribución normal con diferentes medias y desviaciones estándar, que se solapan parcialmente entre sí.

El tamaño de las muestras generadas fue de 5×10^3, 10^4, 10^5, $2{,}5 \times 10^5$, 5×10^5 y 10^6 instancias y las pruebas se realizaron sobre un ordenador con procesador *Intel i5-2500K* a *3.70GHz* de velocidad, con *8Gb* de memoria.

Tabla 5.3: Escalado de K-means# frente K-means++

	5×10^3	10^4	10^5	$2{,}5 \times 10^5$	5×10^5	10^6
K-means++(ms.)	8	16	78	275	344	640
K-means#(ms.)	16	24	182	500	880	1859

Aunque el crecimiento de K-means# asintóticamente es mayor, es lineal y sabemos por la estructura de particionado de datos que se va a aplicar, que los conjuntos de muestras sobre los que calcularemos los centroides serán de un tamaño entre 10^4 y 10^6 puntos, por lo que conocido el tamaño del

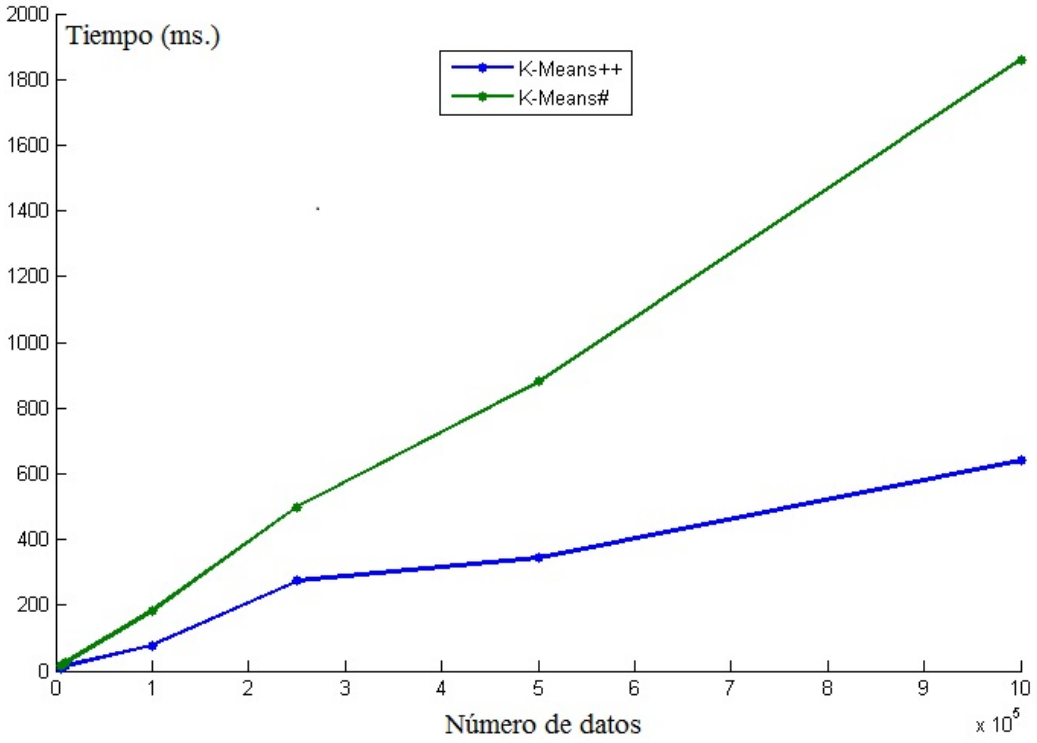


Figura 5.5: Coste temporal

problema, podemos concluir que el algoritmo podrá responder en un tiempo prudencial.

Siguiendo la estructura del pipeline, para las observaciones previstas por Gaia del orden de 10^9 datos, tendremos suponiendo particiones de 10^5 datos, como ejemplo, un total de 10^4 muestras. Por lo tanto podemos calcular una aproximación del tiempo de cálculo como $10^4 \times 182ms$ que supone aproximadamente 30.3 minutos, en un ordenador similar al utilizado para las pruebas, para la ejecución de la primera fase de cálculo de centroides en todas las particiones de los datos de Gaia. Este cálculo aproximado se refiere únicamente al tiempo de ejecución de la suma de todos los algoritmos K-means# ejecutados en una estructura con un sólo nivel de resumen con K-means# antes de aplicar K-means++ al resultado.

La reducción del tamaño del problema que tratará en la fase final *K-means++* dependerá de pruebas realizadas en fases posteriores de desarrollo

del pipeline, pero con los datos experimentales anteriores concluimos que no supondrá ningún problema.

5.2.2 Pruebas de rendimiento del algoritmo PCA incremental

Como vimos en el punto dedicado a la reducción de la dimensionalidad, utilizamos la técnica de análisis de componentes principales con una implementación de tipo incremental, dado que no es posible construir el modelo ni proyectar todos los datos del problema, por lo que utilizaremos esta técnica que construye un modelo que vamos actualizando con nuevas muestras y que sigue la misma estrategia para la proyección de los datos.

El objetivo del siguiente experimento es conocer el tamaño óptimo para dividir el problema en muestras, por ejemplo si dividimos los 10^9 puntos en un millón de bloques, cada bloque contendrá mil puntos, y deberemos construir el modelo con el primer bloque y posteriormente actualizarlo $10^6 - 1$ veces, con el resto de bloques. Para proyectar los datos con el modelo construido y obtener los componentes principales de los datos, necesitaremos de nuevo proyectar los datos con el primer bloque y realizar $10^6 - 1$ proyecciones, hasta obtener el modelo final con todos los datos. Para el experimento se generaron datos sintéticos de forma aleatoria con una distribución $N[0, 1]$ y 10 dimensiones por punto.

Tabla 5.4: PC incremental, coste en *Intel Nehalem x7550,* 32 cores, 64GB de memoria

#points	#points/block	#blocks	$t(s)$		memory (GB)
			update	project	
10^9	10^2	10^7	3283	n/a	n/a
	10^3	10^6	3125	32000	0,001
	10^4	**10^5**	**3093**	**20400**	**0,001**
	10^5	10^4	7532	55190	0,147
	10^6	10^3	9327	68014	1,440
	10^7	10^2	10118	72751	14,132

Podemos observar como el consumo de memoria depende únicamente del número de puntos por bloque, debido a que el algoritmo de tipo *stream* o incremental, sólo trabaja con los datos de un bloque realizando la operación de actualización o de proyección, el resto de bloques no es necesario que permanezca en memoria. Nunca deberemos utilizar un tamaño de bloque que sea superior a la memoria libre del sistema.

Respecto al coste temporal de la actualización observamos que no sufre una variación significativa para bloques de tamaño entre 10^2 y 10^4 datos y a partir de este tamaño sufre un ligero incremento el coste temporal. La explicación a este incremento se debe a los tiempos de acceso a disco y al sistema de ficheros del sistema operativo. Si cada bloque de datos sobrepasa el tamaño del bloque de disco duro, el acceso a la información requerirá acceder en más de una ocasión al disco duro. Además en este caso puede que el bloque de datos sea almacenado en regiones distintas del disco duro.

5.2.3 Velocidad de ejecución de MAC y K-medias

El objetivo del siguiente experimento es comprobar la necesidad de la reducción del tamaño del problema para la ejecución de MAC y su viabilidad con el nuevo tamaño reducido.

Para ello, la siguiente prueba compara el tiempo de ejecución de un algoritmo conocido como K-medias, en su implementación básica, sin mejoras ni uso de paralelización u otras técnicas para acelerar su ejecución, con el algoritmo MAC, de forma empírica.

Debemos tener en cuenta dos aspectos fundamentales, por una parte la velocidad de MAC depende del valor de *bandwidth*, dado que lo utilizará para decidir sobre la convergencia y final de las iteraciones del algoritmo EM subyacente, y en segundo lugar HMAC no es mas que la ejecución para diferentes valores de este parámetro de MAC como vimos anteriormente, por lo que el coste de HMAC es el coste de repetir tantas veces como diferentes valores usemos de bandwidth, un algoritmo MAC.

De nuevo, se generaron para el experimentos datos sintéticos de forma aleatoria pertenecientes a varias agrupaciones distintas que se caracterizaban

por seguir una distribución normal, con diferentes medias y desviaciones.

Tabla 5.5: Comparativa coste temporal y espacial MAC y K-medias

$\mathcal{A}$	#var.	#points	K	$t(s)$	memory (GB)
K-medias	10	10^4	10^3	4	0,221
		10^5		291	0,692
		10^6		11835	3,043
		10^5	10^4	530	2,188
		10^6		44092	17,387
		10^6	10^5	89480	9,992
	120	10^4	10^3	22	0,055
		10^5		805	0,170
		10^6		83827	3,110
		10^5	10^4	3062	0,404
		10^6		117530	3,729
		10^6	10^5	1391128	3,699
MAC	10	10^3	–	8	0,036
		10^4	–	800	0,078
		$2 \cdot 10^4$	–	3139	0,590
		$3 \cdot 10^4$	–	7147	0,396
		$4 \cdot 10^4$	–	12574	0,978
		$5 \cdot 10^4$	–	20647	2,232
	120	10^3	–	44	0,010
		10^4	–	4635	0,145
		$2 \cdot 10^4$	–	20425	0,189
		$3 \cdot 10^4$	–	49365	0,291

Los experimentos se realizaron en un centro de cálculo que reparte la cola de trabajos a diferentes máquinas y no ha sido posible averiguar las características concretas de la máquina que ejecutó las pruebas, pero nuestro interés radica en los costes asintóticos.

El coste temporal de MAC mostrado experimentalmente, podemos obser-

var que es aproximado a $\mathcal{O}(n^2)$ respecto al número de puntos, pero también crece respecto al número de atributos, no de forma cuadrática pero si de forma ligeramente superior a la lineal, aproximándose a un coste $\mathcal{O}(m \times log(m))$ de forma que el coste asintótico que se aproxima mejor a la experimentación para el algoritmo MAC con n puntos y m atributos es de $\mathcal{O}(m \times log(m) \times n^2)$.

Otro dato relevante es la correcta elección del parámetro *bandwidth*, un valor demasiado pequeño, tendrá un grave efecto sobre el coste temporal, debido a que el paso *Expectation-Maximization* de MAC, continuará iterando mientras no llegue a un criterio de parada que depende de un número máximo de iteraciones muy elevado si no alcanza antes un umbral que se calcula respecto al valor del ancho de banda.

Respecto los resultados empíricos en K-medias, debido a la inicialización aleatoria, el número de iteraciones necesarias para la convergencia de los datos hasta los centroides cambia y produce una variación del coste asintótico, entre $\mathcal{O}(\frac{n^2}{3})$ y $\mathcal{O}(n^2)$. Su coste ha variado dependiendo del número de atributos también, pero con estos resultados, no es posible estimar claramente en este experimento si la influencia del número de atributos añade un coste lineal o menor.

Queda demostrado experimentalmente que la utilización del algoritmo MAC y por lo tanto HMAC, que no es sino una sucesión de ejecuciones de MAC, es absolutamente inviable con 10^9 datos y que la reducción de datos por centroides de grupos de datos, permitirá la utilización de la técnica de agrupamiento, dado que por ejemplo la reducción del tamaño del problema a 10^5 puntos, aunque costoso temporalmente, es factible.

5.2.4 Algoritmo MAC frente a variación utilizando árboles kd

El objetivo del siguiente experimento, era estudiar el comportamiento de MAC, introduciendo como mejora el uso de estructuras *KD-Tree* o árboles kd, para que el algoritmo MAC no tenga que computar la contribución de todos los puntos para el cálculo de la probabilidad a posteriori, sino que únicamente accederá a los puntos distribuidos sobre los nodos cercanos del

árbol kd, tomando por cercanos los que cumplen encontrarse a una distancia menor de tres veces la desviación estándar de la media.

La contribución de la estructura sólo se realiza en la fase del cálculo de las probabilidades a posteriori. Por ello, se esperaba una aceleración significativa en el tiempo de ejecución. Se utilizaron como muestras los datos de estrellas,galaxias y cuásares del corpus de datos de SDSS filtrados para $G \leq 17$ utilizado en muchos de los experimentos que realizaremos a partir del capítulo sexto, adaptado en este caso a diferentes tamaños de muestras pero manteniendo en el número de datos siempre la misma proporción de cada clase de objetos. El número de dimensiones es de 16.

Tabla 5.6: Tiempos de ejecución en segundos de *MAC* y *KD-Tree+MAC* sobre procesador *Intel i5-2500K* a *3.70GHz*, *8Gb* de memoria

# Problema	MAC	KD-Tree+MAC
100	0.3	0.3
500	7.6	5.6
1000	26.4	22.3
2500	173.8	140.7
5000	647	587.1
7500	1504.5	1516.8
10000	2651.5	2798.5
12500	4117.6	4655.8
15000	5769.1	6973.1
17500	7902.4	9528.9
20000	10272.7	12642.2

Observamos empíricamente como claramente el coste temporal de ambas versiones es de $\mathcal{O}(n^2)$ sin embargo el crecimiento de la versión que utiliza los árboles k-dimensionales para manejar los datos con el algoritmo MAC crece de forma ligeramente superior respecto al número de datos.

Estos resultados empíricos nos sorprendieron, dado que esperábamos que la versión que utiliza árboles kd, presentase un mejor comportamiento asintótico respecto a no utilizar esta estructura, ya que debería accederse en los

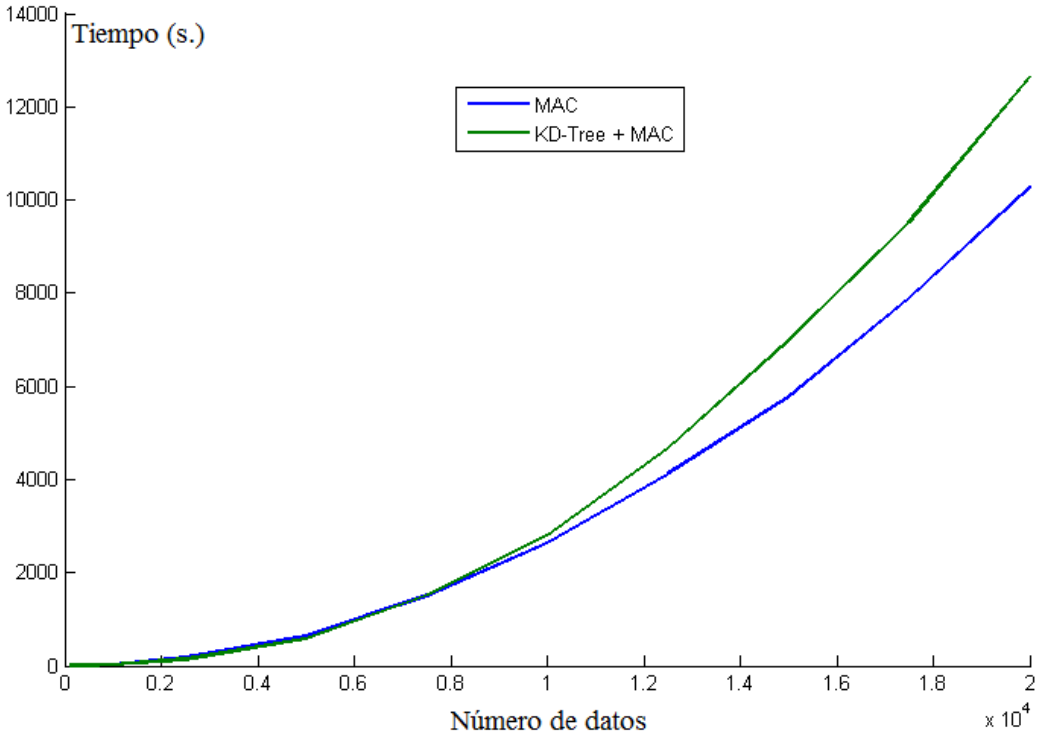


Figura 5.6: Coste temporal

cálculos de la probabilidad a un número menor de datos.

Una explicación para este comportamiento, radica en la forma y distribución de los datos que hemos estudiado en el capítulo tercero. La alta densidad de los datos y su distribución en el espacio, hace que MAC necesite llamar a muchas ramas para tener en cuenta la contribución de todos los puntos.

Además, en el libro de Jacob E. Goodman et al. [23] encontramos que los árboles kd, no son adecuados para encontrar los vecinos más próximos en datos con una alta dimensionalidad. Como regla general, si la dimensión es k , el número de puntos n debería ser mucho mayor a 2^k ya que de otra forma la mayoría de puntos en el árbol serán evaluados y la eficiencia sufrirá las consecuencias, como hemos visto.

El número de atributos utilizado en nuestra experimentación fue de 16, por lo que según [23] para no tener que evaluarse la mayoría de puntos del árbol, el número de datos debería ser muy superior a $2^{16} = 65{,}536$.

Esto supone un problema dado que es necesario la reducción del tamaño del problema para poder utilizar el algoritmo MAC con un coste temporal realista y nuestro objetivo es tratar de llegar con el uso de los centroides a resumir la información de toda la muestra en un número de datos con el que sea posible trabajar, por ejemplo, del orden de 10^5 datos, que no cumple la condición de ser un número mucho mayor a 2^{16}.

Por lo tanto concluimos experimentalmente que la utilización de *KD-Tree* no proporciona ninguna ventaja en el coste temporal sino más bien una pequeña penalización y en [23] encontramos otra argumentación que nos lleva a la misma conclusión.

Otro de los aspectos a tener en cuenta es la variación de los resultados utilizando KD-MAC, respecto a MAC, por lo que repetimos el experimento llevado a cabo en el apartado 7.4 con el mismo número de instancias, que tan buenos resultados nos proporciona en MAC, y nos encontramos unos resultados muy distintos.

Cabe resaltar que frente a los experimentos realizados con MAC que encontraron 3 agrupaciones que contenían prácticamente todos los datos y donde predominaba claramente en cada uno de los grupos una de las tres clases originales del problema, en los experimentos utilizando *KD-Trees*, encontramos numerosos grupos, 23 en concreto, con un número significativo de puntos (entre 210 y 1797 puntos) y un cluster con una gran cantidad de puntos, aproximadamente dos terceras partes del total de los datos y que presenta una gran confusión en los datos, con la proporción de clases que vemos en 5.7.

Tabla 5.7: Resultados incorporando la estructura Árbol kd al algoritmo MAC

Estrellas	Galaxias	Cuásares
60.41 %	22.64 %	16.95 %

Los grupos pequeños que representan el tercio restante de los datos, presentan en todos los casos un porcentaje de aciertos superior al 95 %.

Por lo tanto nos encontramos ante un grupo con gran parte de los datos y muy malos resultados respecto a no utilizar *KD-Trees* como estructura de datos y por otra parte una gran cantidad de grupos pequeños, que aunque

tienen buenos resultados sólo representan un tercio de los datos y dificultan la comprensión y estudio del modelo.

Si no existe error en la implementación y tanto KD-Tree como MAC han pasado diferentes pruebas unitarias, que nos inducen a estimar que son correctos, los resultados parecen indicar que la contribución de los puntos que se deja MAC para el cálculo de probabilidades son importantes y por ello los resultados son mucho peores que sin la utilización de los árboles KD.

Capítulo 6

Experimentos iniciales

6.1 Experimentación con componentes principales

El objetivo de esta primera experimentación es, utilizando los datos semi-empíricos de SDSS sin presencia de ruido, intentar discriminar dos clases a través de los espectros y si el experimento proporciona resultados satisfactorios, repetirlo introduciendo un mayor número de clases de forma iterativa y encontrar si existen dificultades en la clasificación con alguna clase en concreto.

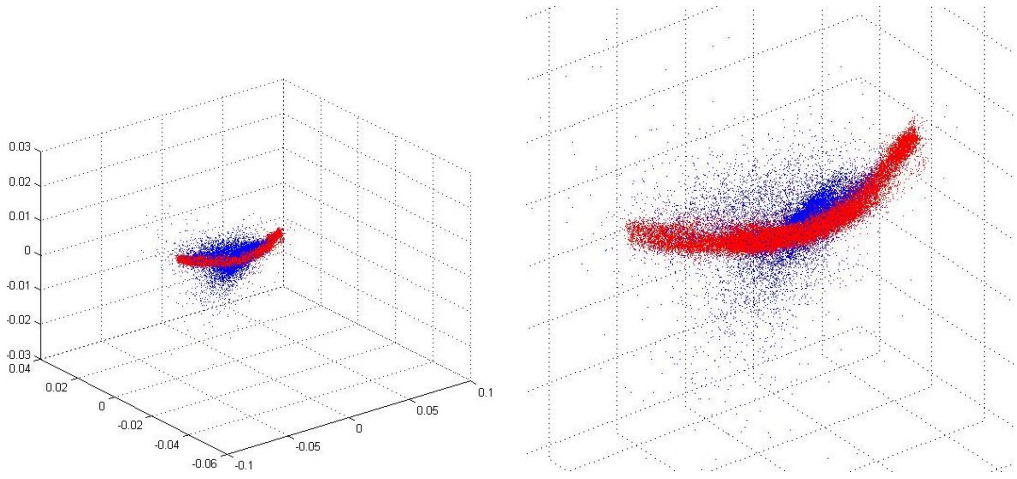
Utilizaremos por lo tanto, los datos provenientes las estrellas y galaxias de SDSS y únicamente trabajaremos con los datos de las componentes principales de los espectros. Queremos averiguar si la utilización de los componentes principales de los espectros es suficiente para realizar un agrupamiento con calidad, algo que por el estudio previo de los datos dudamos, dada la similitud entre los espectros de los diferentes objetos y el solapamiento que mostraban las diferentes clases 3.2.

Para facilitar la tarea al algoritmo de agrupamiento MAC y reducir el tamaño y por lo tanto el tiempo de ejecución del algoritmo de semanas a días, realizamos el experimento filtrando y eliminando los objetos cuya magnitud

es superior a 17, permitiéndonos eliminar aproximadamente el 40 % de los datos, además y como vimos anteriormente 3.2 se trata de datos con mayor afectación de ruido, con lo que facilitamos más el trabajo de MAC.

El número de instancias utilizadas en el experimento es de 51.921, de los cuales 38.566 pertenecen a las Estrellas SDSS y 13.355 a las Galaxias SDSS.

Podemos obtener resultados similares con valores menores de la magnitud, y por lo tanto con una experimentación más rápida al disminuir aun más el tamaño del problema, pero como es lógico, las galaxias son cuerpos mucho menos brillantes, por la distancia a la que se encuentran, que las estrellas de nuestra propia galaxia, por lo tanto la disminución del valor de G no afecta proporcionalmente a todas las clases, a partir de valores de G menores a 15.5 la proporción de datos es de 3.301 galaxias frente a 32.920 estrellas y para un valor de $G \leq 15$ ya no encontramos ninguna galaxia.



(a) Gráfica original. (b) Estrella ultrafría.

Figura 6.1: Solapamiento clases

Como podemos observar, ambas clases están muy solapadas y la labor de clusterización se presenta complicada a priori.

Para el agrupamiento se utilizó el algoritmo MAC, con las primeras 15 componentes principales y se aproximó primero de forma automática el valor del parámetro *bandwidth* con un método desarrollado por nosotros mismos,

que podemos ver en el anexo dedicado a las utilidades desarrolladas durante el presente trabajo y posteriormente se afinó su valor manualmente, para intentar mejorar ligeramente el experimento. La evaluación de la calidad de las agrupaciones obtenidas, es un campo usualmente complejo, sin embargo al utilizar datos etiquetados, podemos estimar de forma sencilla los resultados obtenidos, pues nos interesa encontrar grupos donde predomine una de las clases de datos y ésta es la medida de bondad que utilizaremos en el resto del trabajo, para evaluar la calidad de las agrupaciones.

Los resultados obtenidos en la mejor de las pruebas realizadas, con diferentes anchos de banda o *bandwidth*, fueron los siguientes: Encontramos un grupo principal, dos con un porcentaje mucho menor de datos y finalmente muchos grupos pequeños con unos pocos datos sueltos, o directamente datos sueltos. En los tres grupos con una cantidad representativa de datos, encontramos la siguiente distribución.

Cluster/Clase	Estrella	Galaxia	Total
C0	33467	12726	46193
C1	2663	533	3196
C2	2412	0	2412

Si analizamos los datos como porcentajes obtenemos que cada cluster está formado por:

Cluster/Clase	Estrella	Galaxia
C0	74.45 %	27.55 %
C1	83.32 %	16.8 %
C2	100.0 %	0.0 %

Podemos observar que en la primera agrupación, se encuentran la mayor parte de los datos tanto de estrellas como galaxias, mezclados, en cuanto a número de muestras en cada grupo, el segundo cluster tiene un número muy

inferior de datos pero el porcentaje de objetos es similar, la confusión en ambos grupos es total y esto es debido a la densidad y solapamiento de puntos hace que MAC no pueda encontrar agrupaciones correctas con tan sólo los componentes principales. Tan sólo el tercer cluster encuentra un número de estrellas que corresponderán a agrupaciones de puntos alejados de la zona de densidad que acumula la mayoría de puntos, por tener características espectrales que las hacen diferentes y que forman un grupo aparte. Variando ligeramente el parámetro de *bandwidth*, se ajustan ligeramente los elementos cercanos a las fronteras de decisión entre diferentes grupos, cambiando unos pocos elementos de grupo pero los resultados y conclusiones no cambian en absoluto y al disminuir el valor del parámetro, incrementaremos los datos que pasan a formar parte de grupos muy reducidos, nuevos pequeños grupos o incluso a encontrarse en solitario, como esperamos del algoritmo utilizado.

6.2 Análisis de resultados y líneas de investigación para su mejora

Los resultados, muestran que MAC no puede distinguir ambas clases en unas condiciones ideales:

- Sin presencia de ruido.
- Datos filtrados para una magnitud inferior o igual a 17, descartando los datos que pueden contener una mayor componente de ruido
- Datos de tan sólo dos clases diferentes, provenientes de las adaptaciones a Gaia de observaciones reales del repositorio de SDSS.

Posibles motivos son que o bien el método de reducción de dimensionalidad escogido, PCA, se muestra ineficaz en nuestro caso o bien es necesaria la introducción de más datos aparte de los procedentes de los espectros BP y RP para distinguir los objetos. Vamos intentar mejorar la situación a partir de numerosos puntos que atacan el problema desde diferentes frentes, estudiando la viabilidad y mejora en los resultados de su aplicación y combinar los más prometedores.

Diferentes posibilidades que estudiaremos en el próximo capítulo, son:

- Incrementar los atributos de cada instancia. Añadir al espectro BP/RP, o incluso sustituirlo, por otros atributos, en concreto:
  - Datos estadísticos: A partir de las mismas medidas del espectro BP/RP, podemos añadir los momentos estadísticos principales de cada espectro como son, la media, mediana, varianza, asimetría y curtosis.
  - Datos astrométricos: El objetivo original del trabajo, partía de la experimentación con los espectros, pero los resultados obtenidos nos hacen buscar la posible ayuda en la utilización e incorporación de los datos astrométricos descritos inicialmente, en concreto los relativos a la posición como son la ascensión y declinación, los movimientos propios y el paralaje cuando sea posible.
- Extracción de los datos pertenecientes al cluster curvilíneo de gran densidad. Si observamos la proyección de los primeros dos componentes principales, en todos los objetos, teniendo en cuenta también la densidad, es decir el número de puntos de cada cluster, existe un cluster muy dominante, el cluster curvilíneo que corresponde con las estrellas SDSS. La idea es utilizar una metodología para clusters curvilíneos y detectar los puntos con alta probabilidad de pertenecer a este grupo, extrayendo estos datos del conjunto global. Este cluster curvilíneo extraído, contendrá datos de estrellas SDSS pero también de cuásares y galaxias provenientes de *Sloan Digital Sky Survey*, y será analizado aparte, pero en el conjunto original nos quedarán unos datos más sencillos de analizar, por los algoritmos MAC y HMAC, que buscan los puntos de mayor densidad y están demasiado influidos por la presencia de este cluster curvilíneo de gran densidad en los datos. Para extraer los datos de este cluster, podemos estudiar si es viable la aplicación de una técnica de clustering curvilíneo para nuestro problema y en caso positivo, la integración dentro del proyecto.

Capítulo 7

Segunda fase de experimentación

7.1 Datos astrométricos

Vamos a analizar los resultados de utilizar únicamente la información astrométrica, utilizando los datos relativos a la posición de los objetos, es decir la ascensión y declinación y la información relativa al movimiento de los objetos o movimiento propio en ascensión y declinación. No es posible utilizar la distancia, es decir la inversa del paralaje, debido a que como vimos en el estudio de los datos, sólo está disponible para las instancias de las estrellas.

Los datos astrométricos son interesantes porque tratan de las agrupaciones de los objetos en el espacio y objetos agrupados en el espacio corresponden con una alta probabilidad a objetos que se originaron en las mismas condiciones físicas y de forma cercana en el tiempo. Los experimentos se realizaron filtrando los los datos provenientes de SDSS, estrellas, galaxias y cuásares por un valor de magnitud, en total 55.469 instancias y se normalizaron los datos entre 0 y 1 para cada uno de los valores, el valor utilizado de *bandwidth* para el algoritmo fue de 0.01. Con este valor encontramos varios grupos formados por centenares o miles de objetos, enumeramos los grupos con mayor número

de objetos a continuación, el resto de grupos mantiene una proporción entre clases similar.

Cluster / Clase	Estrella	Galaxia	Cuásar	Total
C0	9.360	2.362	3.207	14.929
C1	8.938	2.118	3.072	14.128
C2	6.366	1.502	2.291	10.159
C3	4.556	1.145	1.651	7.352
C4	3.946	978	1.348	6.272

Estas agrupaciones de objetos, señalan grupos de objetos cercanos y con características de movimientos comunes, sin embargo la utilización de este tipo de datos no demuestra su utilidad con los datos simulados de SDSS para agrupar correctamente en las diferentes clases. Además hemos de afinar manualmente el valor de *bandwidth* con cuidado, pues disminuyendo ligeramente su valor, los grupos se dividen en muchos grupos con tan solo unos pocos objetos cada uno de ellos. Una utilidad de utilizar un valor pequeño para el parámetro *bandwidth* en este caso, obteniendo grupos muy pequeños, la encontramos en ayudar en la identificación de algún tipo de objeto estelar curioso, como sistemas binarios. Trataríamos de identificar parejas de objetos con movimientos propios similares, que sería uno de los criterios a tener en cuenta, para estudiar posteriormente si cumple con otros requisitos como similar velocidad tangencial, igual distancia, etc, como en [20]. Con los datos de Gaia, no los datos semiempíricos que parecen provenir de una región común, podría ser interesante la creación de un atributo que indicara si la región observada del espacio se trata de nuestra propia galaxia. Si tenemos en cuenta la posición de nuestro planeta y Gaia respecto a nuestra Galaxia, nos encontramos en un brazo de la espiral en el exterior de la Vía Láctea, por lo tanto cierta región del espacio observado desde este punto corresponderá a la observación dirigida al centro de nuestra galaxia, por lo que los datos observados corresponderán con una alta probabilidad a estrellas y una baja probabilidad a galaxias. Este conocimiento previo del problema nos puede servir bien para filtrar los datos, bien para añadir un atributo indicando la

probabilidad de pertenecer a nuestra propia galaxia, para intentar mejorar los resultados del algoritmo de clustering.

Si nos resultarán, los parámetros astrofísicos, interesantes, para analizar las características de los distintos objetos que pertenecen a cada uno de los grupos, informaciones relativas a la temperatura, composición, etc. Estos parámetros no están disponibles en los datos simulados, pero como veremos en 7.5, se ha realizado una herramienta para buscar a partir de los parámetros disponibles en las simulaciones el correspondiente objeto original en el repositorio de SDSS y extraer los parámetros buscados del objeto original.

Concluimos que aunque encontramos interesante la utilización de esta información astrofísica por diferentes motivos, para el objetivo perseguido en este punto, no parece resultar de utilidad la utilización directa de estos datos para el algoritmo MAC para identificar la clase/subclase a la que pertenecen los datos, por lo tanto descartaremos su uso en este sentido.

7.2 Momentos estadísticos

Como una aportación original del trabajo, se introducen los momentos estadísticos de los espectros, en lugar de analizar los datos de los espectros directamente. Los momentos estadísticos son indicadores genéricos de una distribución. Se basan en una generalización de la idea de media, concretamente de la media aritmética de la enésima potencia de los valores de la variable. Podemos definir el momento ordinario de orden k como: $a_k = \sum \frac{n}{N} x^k$
Y el momento central de orden k como: $m_k = \sum \frac{n}{N} (x - \overline{x})^k$
Se calculan, para cada espectro en BP y RP, su media, mediana, varianza, asimetría y curtosis. En los experimentos sin ruido, se utiliza la media aritmética de los datos, mientras que en presencia de ruido utilizaremos la mediana, más robusto frente al ruido. Por lo tanto utilizaremos 8 valores, para modelar los espectros BP y RP, y utilizaremos estos valores primero por separado y más adelante junto a los componentes principales.
El cálculo de los valores se realiza mediante sus fórmulas.
Media: $\overline{x} = \frac{1}{N} \sum_{i=1}^{i=n} x_i$ que coincide con el momento ordinario de primer or-

den.
Mediana: El valor de la variable, que ordenados los datos deja a derecha y a izquierda el mismo número de observaciones.
Varianza: Mide la dispersión de los datos respecto a la media aritmética y al igual que la desviación típica, es el mejor indicador de la variabilidad global de una distribución,$S^2 = \frac{1}{N}\sum_{i=1}^{i=n}(x_i - \overline{x})^2$, respecto a los momentos ordinarios puede expresarse como $S^2 = a_2 - a_1^2$
A través de los anteriores valores, estudiamos la situación de los valores de un espectro y cuánto se dispersan en términos globales, añadimos los siguientes parámetros para conocer la forma en que se distribuye los valores de un espectro.
Asimetría: Este indicador nos informa de si los valores de la distribución se disponen simétricamente alrededor de la media, o bien si se decantan en mayor medida hacia la derecha (asimetría a derechas, o positiva) o hacia la izquierda (asimetría a izquierdas, o negativa). Se define respecto al orden central como:$As = \frac{m_3}{(\sqrt{m_2})^3} = \frac{\frac{1}{N}\sum(x_i-\overline{x})^3}{(\sqrt{\frac{1}{N}\sum(x_i-\overline{x})^2})^3}$
Curtosis: Finalmente, nos interesa comparar ponderadamente, el número de observaciones cercanas a la media con el número de observaciones lejanas. Informalmente, decimos que mide la mayor o menor forma de "punta" de la distribución, por ello en algunos textos se llama a la curtosis apuntamiento. La curtosis está relacionada con el momento central de cuarto orden de la siguiente forma: $K = \frac{m_4}{m_2^2} - 3 = \frac{\frac{1}{N}\sum(x_i-\overline{x})^4}{(\frac{1}{N}\sum(x_i-\overline{x})^2)^2} - 3$
El valor restado a la curtosis, 3, se realiza para permitir la comparación del apuntamiento de la distribución con el apuntamiento "patrón"que es el que tiene la distribución normal de probabilidad, cuyo momento de cuarto orden tipificado es tres. Por ello se define el coeficiente de curtosis como el momento central de cuarto orden de la variable tipificada menos tres unidades. Para la normalización de los datos, los valores estadísticos, tras calcularse para cada espectro se normalizaron en el rango de los componentes principales, es decir, en lugar de normalizarse entre cero y uno por ejemplo, se utilizó el primer componente principal que es el que mayor varianza en los datos presenta por definición, y se calculó el menor y mayor valor de esta componente,

utilizándose este valor mínimo y máximo como rango de normalización de los datos, siendo una normalización equivalente a realizarla entre cero y uno pero en la escala de valores de los componentes principales. Este ajuste se realiza en cada espectro por separado, es decir los datos derivados de los momentos estadísticos de BP se normalizan respecto a los componentes principales de BP y lo mismo sucede con RP. De esta forma cuando realicemos posteriormente experimentos junto a los componentes principales, los datos se encontrarán escalados en el mismo rango de valores. Los experimentos se realizaron filtrando los objetos por el mismo valor de magnitud que con los experimentos iniciales, $G \leq 17$ con todos los datos provenientes de SDSS, es decir, estrellas galaxias y cuásares. Se utilizó nuestra herramienta para calcular un valor inicial de *bandwidth* que redondeamos a 0.0005, utilizándose el algoritmo de MAC para realizar el agrupamiento.
Los resultados obtenidos fueron:

Cluster / Clase	Estrella	Galaxia	Cuásar	Total
C0	37772	964	1255	39991
C1	782	899	26556	28237
C2	10	11488	413	11911
C3	0	0	2	2
C4	0	0	3	3

Aparte de los dos pequeños grupos con tan sólo dos y tres datos, encontramos otros siete objetos que quedaron sueltos como un grupo por sí mismos, que podemos considerar como *outliers* o valores atípicos junto a las pequeñas agrupaciones separadas de los demás conglomerados. Estos objetos atípicos representan un porcentaje del total despreciable, dado que no tenemos ruido en los datos (aparte del ruido original introducido en las mediciones en SDSS de cuyas simulaciones provienen los datos utilizados), podrían tratarse de objetos con características que los hacen interesantes por diferir del resto y cuyo estudio puede resultar interesante para ciertas unidades del consorcio. Sobre los primeros tres grupos, observamos cómo ha resultado un éxito la utilización de las medidas estadísticas sobre los espectros, encontrando que el primer grupo está formado en un gran porcentaje por estrellas, el segundo

por los cuásares y el último las galaxias.

Si calculamos respecto al total de datos agrupados correctamente, tomando

Tabla 7.1: Resultados de clustering con momentos estadísticos

Cluster / Clase	Estrella	Galaxia	Cuásar
C0	**94.45** %	2.41 %	3.14 %
C1	2.77 %	3.18 %	**94.05** %
C2	0.08 %	**96.45** %	3.47 %

los tres clusters obtenidos como las clases estrellas, galaxias y cuásares, el porcentaje de acierto es de 94.61 % lo que podemos considerar un éxito para la ejecución del algoritmo de agrupamiento.

7.3 Eliminación de agrupación curvilínea de gran densidad

Uno de lo posible puntos de mejora que observamos en el análisis de los datos anteriormente, partía de la idea de la extracción de los datos pertenecientes a un cluster curvilíneo de gran densidad, el grupo de las estrellas SDSS en concreto con una gran densidad de objetos que provocan que el algoritmo MAC (y por lo tanto HMAC), esté demasiado influido por la densidad de los datos y no realice un buen agrupamiento. No es objetivo del presente trabajo la implementación del algoritmo de agrupación curvilíneo, pero si estudiar la viabilidad de su aplicación, para aislar los datos de un grupo determinado, analizarlo por separado para estudiar si el algoritmo MAC obtiene mejores resultados, analizando por separado los datos eliminando el grupo curvilíneo de mayor densidad y este grupo por separado. Para realizar este estudio se utilizó como base un trabajo previo [34]. Este algoritmo de agrupamiento, en líneas generales, utiliza el término de Curva Principal, que es un resumen de unos datos n-dimensionales en un curva. El algoritmo trata de adaptar una curva a un conjunto de puntos minimizando la distancia entre cada punto del subconjunto y la curva, utilizando unos grados de libertad, que permiten

adaptar la curva a los datos. Sin embargo es necesario un cierto conocimiento sobre los datos analizados, por ejemplo el número de grupos esperados y forma que puede tener la curva para definir ciertos valores de pesos utilizados en el algoritmo, lo cual no supone un problema para los datos conocidos que utilizamos pero si para los datos reales tomados por Gaia. Pero el aspecto clave del algoritmo, que finalmente hará que resulte inviable su uso para Gaia, es su complejidad temporal, que depende directamente de la complejidad del algoritmo utilizado para encajar la curva en los datos. Las pruebas iniciales con el algoritmo nos mostraron que no escalaba linealmente con el número de datos, su crecimiento es mucho más rápido que el algoritmo MAC y no sólo en tiempo de CPU para el cálculo sino también en uso de memoria, resultados que concuerdan con los originales en [34]. Durante el presente trabajo hemos tenido que reducir el conjunto de datos sobre el que aplicamos la agrupación con MAC, e incluso así muchos experimentos han tardado semanas en obtener resultados, con el clustering curvilíneo el coste temporal y también espacial resultaba mucho mayor y hacía totalmente inviable los experimentos. Además, tenemos una estrategia como se ha explicado en el apartado relativo a la escalabilidad de los datos para poder trabajar con MAC cuando tengamos los datos reales, sin embargo esta estrategia no es aplicable a la agrupación curvilínea o al menos no sin profundas transformaciones del algoritmo que no son objetivo del presente trabajo. No podemos saber a priori si una muestra entregada al pipeline, de una determinada zona de la esfera celeste, contendrá una gran densidad de estrellas mezcladas con otros objetos o si se tratará casi exclusivamente de estrellas o su número no será tan alto en proporción con los demás objetos. Por este motivo y principalmente a causa del coste temporal y espacial del algoritmo, abandonamos esta vía de investigación para centrarnos en otras más interesantes.

7.4 Incorporación a los experimentos

Como hemos analizado en los experimentos realizados con datos astrométricos por separado, no resultará interesante mezclarlos junto a los componentes

principales. Si podrían utilizarse de forma previa para introducir un mayor conocimiento por ejemplo con un nuevo atributo que indicara para la región del espacio observable que apuntara hacia la Vía Láctea, una alta probabilidad de que el objeto observado fuera una estrella y no una galaxia, pero con los datos simulados disponibles no podemos introducir este conocimiento previo, por lo tanto descartaremos realizar experimentos mezclando la información de espectrometría con la astrometría. Respecto a los resultados obtenidos con los datos estadísticos derivados de los espectros, nos interesa conocer si mejoran unidos a los componentes principales. Utilizando exactamente los mismos datos, corpus de estrellas, galaxias y cuásares filtrados para un valor de $G \leq 17$, concatenando sus correspondientes valores estadísticos calculados en el apartado anterior, es decir, media, varianza, asimetría y curtosis del espectro BP y RP, para obtener un total de 23 atributos por instancia de objeto, 15 de las componentes y 8 estadísticos. Recordamos, que los atributos derivados de los momentos estadísticos, se encuentran normalizados en el mismo rango de valores que los componentes principales de sus correspondientes espectros BP o RP, paso necesario para obtener resultados satisfactorios ya que en otro caso concatenaríamos atributos en un rango de valores con una magnitud distinta. Comentar que los resultados con tan sólo 8 componentes fueron casi idénticos, por los motivos expuestos en el apartado del trabajo que trata la reducción de la dimensionalidad. Para la elección del parámetro *bandwidth* utilizamos el mismo valor del experimento anterior, 0.0005, debido a la normalización realizada, el valor del parámetro calculado es similar y redondeamos a la misma cifra. Utilizamos el algoritmo MAC para realizar el clustering de nuevo obteniendo los siguientes resultados:

Cluster / Clase	Estrella	Galaxia	Cuásar	Total
C0	38.337	401	834	39.572
C1	217	946	26.457	27.620
C2	10	12.303	632	12.945

Igual que en el experimento con datos estadísticos únicamente, se obtienen muy pocos datos dispersos, dos grupos con dos instancias, candidatos a representar algún caso particular y extremo del objeto estelar en cuestión (en

ambos casos de trata de cuásares igual que en experimento anterior) y unas pocas instancias de objetos dispersos. Observemos el porcentaje de instancias de cada clase dentro de un cluster. Si comparamos los resultados con

Tabla 7.2: Resultados de clustering con CP y momentos estadísticos.

Cluster / Clase	Estrella	Galaxia	Cuásar
C0	**96.88** %	1.01 %	2.11 %
C1	0.79 %	3.42 %	**95.79** %
C2	0.08 %	**95.04** %	4.88 %

el experimento utilizando únicamente datos estadísticos, los resultados de la agrupación de las estrellas ha mejorado 2.43 %, bajando la confusión con las otras clases a casi la mitad, la identificación de los cuásares ha mejorado un 1.74 % disminuyendo notablemente la confusión con las estrellas y empeorando ligeramente con las galaxias, sin embargo la identificación de las galaxias ha empeorado ligeramente un 1.41 %, por el aumento de la confusión con los cuásares. Respecto al total de datos agrupados correctamente tomando los tres clusters obtenidos como las clases estrellas, galaxias y cuásares, el valor es de 96.19 % que es un 1.58 % superior.

La conclusión es que los resultados son muy similares, ligeramente superiores, pero por un margen tan pequeño que el incremento de tiempo en la ejecución por la utilización de más atributos nos debe hacer valorar a la hora de lanzar un experimento, si nos interesa o no incluir los componentes principales.

7.5 Análisis de los clusters obtenidos

Un aspecto que hemos considerado interesante, es el estudio o caracterización de los grupos obtenidos por la variación del parámetro de bandwidth realizada durante HMAC. Como vimos, HMAC no es más que una repetición de la ejecución de MAC con diferentes valores ordenados de bandwidth, de forma que obtendremos de forma natural una agrupación jerárquica, como podemos ver gráficamente en el anexo 2. Nuestra experimentación en el

presente capítulo se ha centrado en MAC, pero si utilizamos HMAC encontraremos cómo los conglomerados agrupados utilizando un ancho de banda inferior, se dividen en varios clusters, respecto a los mismos puntos en niveles de la jerarquía de HMAC con un bandwidth mayor, por lo tanto, con el objetivo del análisis de los resultados en el domino del problema en mente, nos interesa estudiar las características de los diferentes clusters formados. En nuestro caso, dado que como veremos en el siguiente punto sólo es posible caracterizar actualmente las estrellas, veremos cómo las estrellas se dividen en diferentes conglomerados y trataremos de caracterizarlos. Para ello, podemos utilizar parámetros físicos, sin embargo, las simulaciones disponibles no cuentan con estos datos, por lo tanto hemos intentado encontrar un vínculo entre los datos simulados y el repositorio original en SDSS para poder obtener los datos originales a partir del cual se simuló para Gaia cada instancia. Este estudio es una idea novedosa dentro de los diferentes experimentos que se han realizado en las diferentes unidades de DPAC.

7.5.1 Problemas encontrados

El presente experimento, ha presentado numerosos problemas. En primer lugar, durante la generación de los datos simulados, no se pensó en utilizar un campo para identificar el origen del dato, por lo que cada dato cuenta con un identificador dentro del repositorio de Gaia, pero no es posible de forma directa obtener el identificador del objeto original en el repositorio de SDSS. Por lo tanto realizamos un estudio de los atributos disponibles dentro de las simulaciones intentando encontrar algún atributo que nos ayude a identificar el objeto original. Otro problema que encontramos es que cada corpus de datos ha sido generado de forma diferente. El corpus de cuásares no contiene prácticamente ninguna información adicional y no es posible la identificación del objeto original en el repositorio de SDSS. Puestos en contacto con los encargados de su simulación, no han podido proporcionarnos esta información pero si han encontrado suficientemente interesante nuestra idea, para repetir la simulación de los datos, aunque estas nuevas simulaciones de cuásares no estarán disponibles hasta como mínimo, aproximadamente las fechas de

entrega del presente trabajo, pero si podrán ser utilizados con posterioridad para la escritura de un artículo sobre la caracterización de los grupos encontrados en las simulaciones, realizando clustering con el algoritmo MAC. Respecto a las galaxias, en su simulación si pensaron en mantener la correspondencia con el repositorio SDSS, por lo que encontramos un atributo llamado *objID* y un atributo *specID*. En el repositorio de SDSS, encontramos tablas con información de fotometría y *objID* corresponde con la clave principal de una tupla de información y *specID* corresponde con la clave principal de una tupla cuya tabla contiene la información respecto los espectros (en el formato de SDSS, recordemos que nuestros datos están simulados a partir de estas informaciones para resultar como los observaría Gaia). La tabla de espectrometría contiene una clave ajena a la mejor observación del objeto estelar del que proviene, pues un mismo objeto estelar tiene múltiples observaciones y a su vez la tabla de información fotométrica contiene una clave ajena a la tabla con información del espectro, si hay un espectro disponible para esa observación ya que en muchos casos puede no existir información espectrométrica relacionada. Hemos encontrado un grave problema de coherencia en los datos simulados de las galaxias, si fueran correctos, la clave ajena de la tupla de tabla de fotometría apuntaría a una tupla de espectometría que correspondería a la clave que también tenemos. A su vez, la tupla de información espectrométrica tendría una clave que la relacionaría con la fotometría para la mejor observación del objeto estelar correspondiente o al menos, para una observación cualquiera de ese objeto estelar. Sin embargo las tuplas de información fotométrica que obtenemos para los *objID* disponibles, no disponen de clave ajena a información espectrométrica, es decir el identificador nos lleva a objetos estelares sin información de espectros asociados y a su vez, la clave *specID* que nos lleva a una tupla de información espectrométrica, contiene una clave ajena a una observación de objeto estelar que no se corresponde con ninguna observación del mismo objeto que identificamos con *objID*. Por lo tanto el objeto estelar y la información del espectro no se corresponden si utilizamos las claves proporcionadas. El personal encargado de la generación de estos datos, está revisando también este problema y estamos esperando una respuesta al respecto o un nuevo corpus

de datos con el problema corregido. En el caso de las estrellas, aunque no se incluyó el identificador del objeto correspondiente al repositorio SDSS, encontramos tres parámetros que definen de forma única una tupla en el repositorio original. Se trata de los valores *plate*, *mjd* y *fiber*. El primero identifica la placa que realizó la observación, cada placa tiene un conjunto de fibras ópticas *(fiber)* para dirigir la luz de objetos individuales a la correspondiente hendidura en la placa formando el conjunto un espectógrafo. Finalmente, *mjd* es la fecha *(Modified Julian Date)* de la última observación. Los tres valores, *plate*, *mjd* y *fiber*, están identificados con una única tupla en la tabla de información espectrométrica y a partir de ésta, podemos extraer información de la tabla *sppParams*, en el repositorio de *Sloan Digital Sky Survey*, a través de la web.

7.5.2 Parámetros estelares estudiados

La anterior tabla, contiene la información procesada por el Spectro Parameter Pipeline, o spp, que calcula más de 200 parámetros estelares atmosféricos comunes para estrellas a través de varios medios. Un estudio de los datos, realizado sobre la propia web de SDSS y en diferentes artículos como $[41_X XX]$ y $[42_X XX]$, nos indica que los datos más interesantes para caracterizar las estrellas que formen parte de un grupo son:

- *Sptypea*: Es el tipo de clasificación estelar asignado, tipo de estrella, los posibles valores son A, F, G, K, M y L.

- *Feha*: Mide la metalicidad del objeto, que describe la abundancia relativa de elementos más pesados que el helio en una estrella. Dado que la fracción de elementos más pesados que el helio aumenta en función del tiempo cósmico, la metalicidad de una estrella esta relacionada con el momento en que se formó, con la edad de la estrella. Se expresa utilizando como patrón la abundancia de elementos metálicos del Sol como referencia, comparando las líneas de absorción del hidrógeno con las del hierro a través de la espectroscopia. El índice de metalicidad se expresa como $[Fe/H]$ que representa el logaritmo del cociente entre la

abundancia de metales en la estrella y la abundancia solar, siendo el índice de metalicidad del Sol 0 y usando una escala logarítmica, una metalicidad de -1 equivaldrá a una abundancia diez veces menor a la del Sol y un valor de 1 a una abundancia diez veces mayor.

- *Teffa*: La temperatura de la estrella en grados Kelvin, este valor es el resultado de la media ponderada con hasta diez diferentes métodos utilizados para calcular la temperatura de la superficie de la estrella. El valor de la temperatura, nos ayuda en la clasificación del tipo estelar, podemos observar la clasificación de las estrellas según su temperatura incrementa etiquetadas con las letras O, B, A, F, G, K y M pertenecientes a la clasificación estelar. El diagrama muestra el resultado

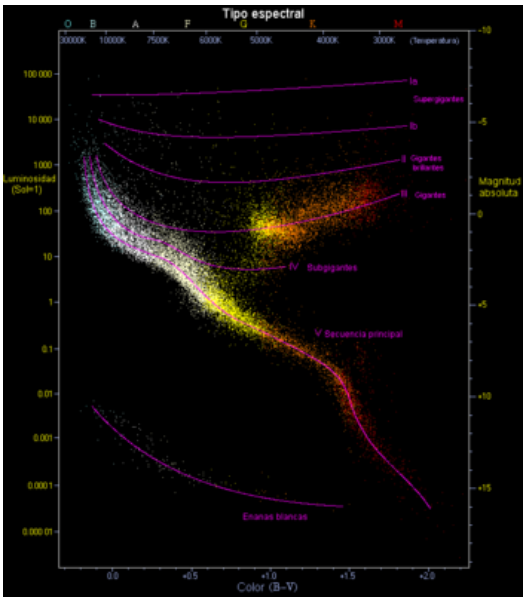


Figura 7.1: Diagrama de Hertzsprung-Russel.

de numerosas observaciones sobre la relación existente entre la magnitud absoluta de una estrella y su tipo espectral, información usada para diferenciar tipos de estrellas y estudiar la evolución estelar. Con la información de la magnitud aparente de los objetos, junto a la distancia, podemos calcular la magnitud absoluta. Un diagrama HR, como

el mostrado, puede contener en sus ejes también información de luminosidad y temperatura, aunque la relación exacta con la magnitud y tipo espectral debe ser ajustada dependiendo del modelo de atmósfera estelar utilizado.

- *Logga*: Valor adoptado de la magnitud, recordamos que utiliza una escala logarítmica e indica de forma inversa el brillo del objeto, cuanto mayor brillo, menor magnitud. Calculado de igual forma que el anterior atributo, como la media ponderada de hasta 8 diferentes métodos.

- *Ebv*: Exceso de color. El espacio interestelar no está completamente vacío y parte de la radiación emitida se dispersa y absorbe por el polvo interestelar antes de llegar a los detectores, atenuando el brillo y desplazándolo hacia la banda del rojo, debido a una mayor absorción de la banda azul que la roja. Esta situación genera un exceso de color. El exceso de color se puede determinar por la diferencia entre el índice de color observado $(B - V)$ y el índice de color intrínseco $(B - V)_0$, este último se puede deducir por el tipo espectral y la clase de luminosidad. Una vez conocida la extinción visual, podemos obtener la distancia correcta a la estrella. El índice de color B-V está relacionado con la temperatura superficial de las estrellas, las estrellas más calientes tienen los valores más bajos mientras las estrellas frías los más altos.

7.5.3 Herramienta desarrollada

Por lo tanto se desarrolló una aplicación que extrae los datos que nos permiten identificar, a través de la utilización de un servicio web implementado contra el repositorio de SDSS, el objeto original que se utilizó para generar el dato en Gaia. Continúa la ejecución obteniendo los parámetros que hemos considerado más interesantes en nuestro estudio para caracterizar las estrellas, detallados en el punto anterior. Nuestra herramienta puede extenderse fácilmente para descargar otros parámetros del repositorio, si se consideran de interés. La aplicación se encuentra preparada para el acceso a información sobre las tablas relativas a Galaxias cuando obtengamos un identificador co-

rrecto y también podrá utilizarse para extraer la información de los cuásares. El acceso a cada uno de los datos a través de la web es un proceso tedioso temporalmente, con velocidades que varían según el estado de la red y carga del repositorio SDDS, pero con una media de aproximadamente 1800 consultas resueltas de objetos por hora, teniendo en cuenta que caracterizamos decenas de miles de objetos y que incluso puede fallar temporalmente la conexión y perderse algunos datos, la herramienta implementada utiliza una conexión a base de datos PostgresSQL que puede ser local o en una máquina remota, que contiene tablas para guardar la información de estrellas y otros objetos celestes. La herramienta consulta los identificadores extraídos contra la base de datos y en caso de no encontrar la información, se conecta al repositorio de SDSS para descargar esta información y la introduce en las tablas correspondientes, de forma que sólo la primera vez que se consulta información sobre un objeto éste se descarga. Por lo tanto una vez utilizada la herramienta con un conjunto de datos por primera vez, su funcionamiento en sucesivas ejecuciones es inmediato, al extraer toda la información de la base de datos. El problema de perder algún dato por una desconexión puntual en la red se soluciona ya que en la siguiente ejecución se volverá a pedir de nuevo la información al repositorio ya que el dato no está presente en la base de datos. Dado que existen más trabajos realizándose sobre las muestras generadas a partir de las simulaciones de SDSS dentro del consorcio de DPAC, pondremos esta herramienta en manos de la comunidad, ya que permite realizar un análisis de la información útil en el dominio del problema.

7.5.4 Resultados

Se realizó un experimento con HMAC con tres valores de ancho de banda $\sigma_1 = 5 \times 10^{-4}$, $\sigma_2 = 2 \times 10^{-4}$ y $\sigma_3 = 5 \times 10^{-5}$, con las muestras utilizadas anteriormente, es decir, estrellas, galaxias y cuásares de las muestras provenientes de SDSS filtradas para valor de $G \leq 17$.
Los resultados con σ_1, son los mismos que en la sección anterior del presente capítulo, con σ_2 obtenemos unos resultados similares, menos de una decena de objetos, un porcentaje ínfimo, cambia de conglomerado, sin embargo con

σ_3 encontramos que el conglomerado de las estrellas se ha dividido en tres clusters diferentes, como vemos en su representación gráfica:

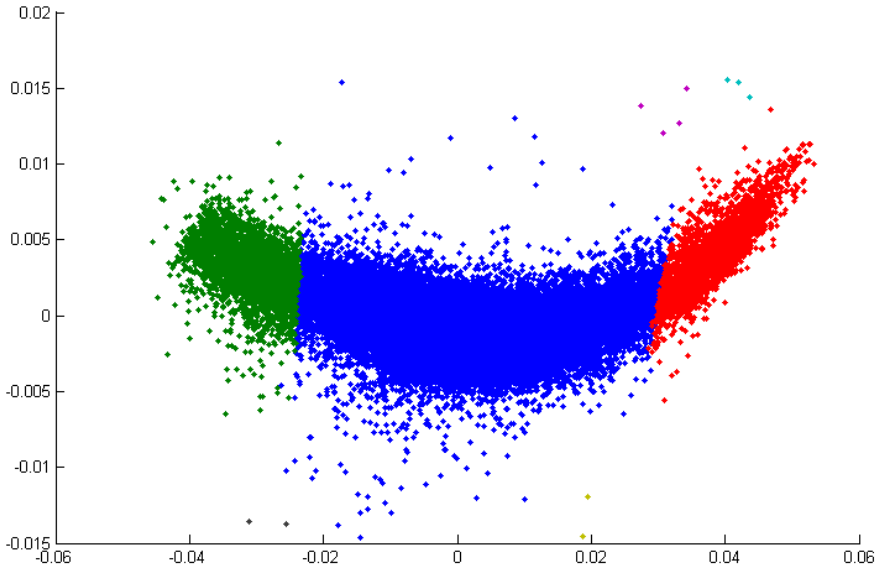


Figura 7.2: Resultados MAC con bandwidth $\sigma_3 = 5 \times 10^{-5}$

Vemos claramente que aparte de unos pequeños datos que forman sus propias agrupaciones (parte superior derecha de la imagen e inferior izquierda y derecha), la gran mayoría de los puntos se encuentran divididos en tres grupos que parecen corresponder a diferentes temperaturas, recordemos que averiguamos una relación entre la representación de las primeras componentes principales y la temperatura de los objetos durante el análisis de los datos en el tercer capítulo. Sin la herramienta desarrollada nuestras posibilidades de analizar la información en el dominio del problema se reducirían a este aspecto, sin embargo gracias a nuestra herramienta podemos extraer la siguiente información de cada uno de los clusters principales que de izquierda (verde) a derecha (rojo) denominaremos cluster 0, 1 y 2, para los diferentes parámetros mostramos la media de los valores del conglomerado y la desviación típica.

Tabla 7.3: Características extraídas.

	Cluster 0	Cluster 1	Cluster 2
Clasificación Estelar	A(96.5 %), B (0.16 %) F(3.3 %), G(0.04 %)	A(9.22 %), F(65.47 %) G(13.39 %),K(11.92 %)	F(0.27 %), G(0.09 %) K(99.64 %)
Metalicidad (log.)	$\overline{X} = -0{,}98; \sigma = -0{,}66$	$\overline{X} = -0{,}91; \sigma = -0{,}65$	$\overline{X} = -0{,}56; \sigma = -0{,}39$
Temperatura (K^o)	$\overline{X} = -6809; \sigma = 2194$	$\overline{X} = 5299; \sigma = 1817$	$\overline{X} = 3960; \sigma = 1327$
Magnitud (log.)	$\overline{X} = 3{,}53; \sigma = 1{,}21$	$\overline{X} = 3{,}67; \sigma = 1{,}3$	$\overline{X} = 3{,}96; \sigma = 1{,}37$
Exceso de color	$\overline{X} = 0{,}035; \sigma = 0{,}026$	$\overline{X} = 0{,}047; \sigma = 0{,}052$	$\overline{X} = 0{,}048; \sigma = 0{,}043$

Sin ser expertos en el dominio del problema, podemos deducir de los conglomerados resultantes mucha información, el cluster 0 prácticamente está compuesto por estrellas de tipo A, las más calientes de la muestra, con la temperatura media más elevada como esperábamos. En el cluster 1, encontramos muchas estrellas de tipo F aunque no predomina un sólo tipo estelar tan claramente como en las demás agrupaciones, hemos de tener en cuenta que es el conglomerado que mayor número de puntos contiene, su temperatura se mantiene en un grado intermedio entre los otros conglomerados como la gráfica de los componente principales nos indica también. El último cluster, está formado por estrellas de tipo K casi exclusivamente, las más frías de la muestra y presenta una menor dispersión en los datos respecto a este parámetro. Al igual que observamos cómo decrece la temperatura de los conglomerados de izquierda a derecha, observamos cómo crece la magnitud, que recordemos es una medida inversa, es decir, que a mayor brillo menor magnitud, por lo tanto los objetos más calientes son más brillantes, como la teoría nos dice y como podemos observar en el diagrama de *Hertzsprung-Russel*. El exceso de color, relacionado con la temperatura superficial de las estrellas, de forma que las estrellas más calientes tienen valores menores que estrellas más frías, nos da el resultado esperado. Todas las estrellas de la muestra, presentan un índice de metalicidad negativo, por lo que todos los datos presentan menor abundancia de metales que el Sol, estos datos parecen sugerir que las muestras pertenecen probablemente a una misma región de la esfera celeste, con estrellas mas jóvenes que la de nuestro sistema solar. Dada la situación de nuestro sistema solar respecto a la Vía Láctea, donde

nos encontramos en un brazo de la espiral, en el exterior del disco que forma nuestra galaxia y dada la metalicidad de las muestras, lo más probable, es que la muestra pertenezca a una porción de la esfera celeste que mira hacia el centro de la Vía Láctea, ya que la metalicidad decrece dentro del disco a medida que nos alejamos del centro. Al igual que los datos sobre tipo estelar, temperatura, magnitud y exceso de color no hacen sino corroborar la teoría, este resultado nos da una información, que a priori es desconocida en los datos, suponiendo que no se ha realizado un estudio con las posiciones y valores procedentes de la astrometría o que no se encuentren presentes estos parámetros. Observamos que somos capaces de dividir, con un valor correcto de *bandwidth*, no sólo la muestra con un alto porcentaje de aciertos en las clases originales, sino que también podemos dividir los objetos estelares en sus diferentes tipos y clasificación. Contamos con una herramienta, que fácilmente podemos extender en cuanto el número de parámetros extraídos del repositorio, para la realización de estudios más complejos sobre el análisis de los conglomerados resultantes en el área del problema. Con esta herramienta, un profesional del dominio del problema, sin necesidad de tener un gran conocimiento sobre las técnicas empleadas, puede centrarse en el análisis de la información.

Capítulo 8

Introducción de ruido

Los experimentos con el algoritmo MAC, han sido realizados en los capítulos 6 y 7 en condiciones ideales, es decir:

- Con un corpus de datos sin ruido sintético.
- Datos generados mediante la misma técnica, a través de simulaciones de observaciones existentes de datos reales en el repositorio de SDSS.
- Filtrados para un valor de magnitud determinado $G \leq 17$ o $G \leq 16$ según el experimento, con el doble objetivo de reducir el tamaño de la muestra para facilitar la experimentación y utilizar los objetos más brillantes, que a priori son los objetos cuyos datos originales fueron menos sensibles al ruido.

Los experimentos se realizaron en estas condiciones porque su objetivo era el estudio de la respuesta del algoritmo de agrupamiento frente a los datos, análisis y proposición de soluciones para mejorar los resultados 6 y el estudio ante la deficiente respuesta a la utilización de tan sólo los componentes principales de los espectros, con nuevos conjuntos de atributos, relativos a la astrometría y a la utilización de información derivada de los momentos estadísticos de los espectros 7.

Una vez obtenido un conjunto de técnicas de preprocesamiento y preparación de los datos 3 y unos excelentes resultados del algoritmo de agrupamiento

en condiciones ideales y para cierto conjunto de atributos (7.1 y 7.2), llega el momento de estudiar la respuesta del algoritmo frente a una situación menos ideal y más real respecto al corpus de datos.

8.1 Efecto de incrementar G

Aunque los datos semiempíricos utilizados en la experimentación no tienen ruido sintético introducido, hemos de recordar que estos datos provienen de simulaciones realizadas para adaptar al modelo de Gaia datos procedentes de un repositorio con datos reales. Estos datos tienen un valor, la magnitud aparente o G que es una medida del flujo o cantidad de energía recibida por unidad de área y tiempo por un detector respecto a un objeto celeste, concepto que podemos aproximar al brillo del objeto y utiliza una escala logarítmica de forma inversa, es decir, a mayor brillo menor valor de magnitud tendrá. Los objetos más brillantes por lo tanto, son los objetos de los que hemos recibido mayor flujo de energía.

El efecto del ruido en las señales con un flujo de energía débil es mayor, podemos deducir que a mayor valor de G, mayor será el efecto del ruido en los datos del repositorio utilizados para originar la muestra del experimento.

Vamos por lo tanto a estudiar los resultados de aplicar el algoritmo de clustering MAC sobre los datos de estrellas, galaxias y cuásares originados en las simulaciones semiempíricas de SDSS, con valores de $G \leq 17$,$G \leq 19$ y $G \leq 21$, para un *bandwidth* de 0.0005, presentando los resultados de los tres conglomerados principales ya que la mayoría de los datos para los tres valores de G, recaen sobre uno de estos conglomerados.

Tabla 8.1: Efecto de incrementar valor de filtrado G en resultados agrupación, por clases.

		$G \leq 17$			$G \leq 19$			$G \leq 21$	
Cluster/Clase	Est.	Gal.	Cuás.	Est.	Gal.	Cuás.	Est.	Gal.	Cuás.
C0	96.88 %	1.01 %	2.11 %	97.16 %	<0.01 %	2.83 %	96.84 %	<0.01 %	3.15 %
C1	0.79 %	3.42 %	95.79 %	0.26 %	2.46 %	97.8 %	0.68 %	2.45 %	96.87 %
C2	0.08 %	95.04 %	4.88 %	<0.01 %	91.42 %	8.57 %	<0.01 %	91.44 %	8.55 %

El número de objetos que pasan el filtro y por lo tanto son utilizados en

el experimento son:

Valor filtrado	#instancias
$G \leq 17$	80.151
$G \leq 19$	129.564
$G \leq 21$	154.226

Los resultados en cuanto a los grupos que identifican las clases muestran cómo la variación es muy pequeña respecto el parámetro G, no podemos formular conclusiones sin tener en cuenta el número de instancias en cada cluster para calcular una medida sobre la totalidad de los datos, tomando como agrupaciones naturales que C0 corresponde a las estrellas, C1 a los cuásares y C2 a las galaxias. En este caso los resultados son los siguientes:

	$G \leq 17$	$G \leq 19$	$G \leq 21$
% Aciertos Totales:	96.19 %	95.96 %	95.61 %

Tabla 8.2 Efecto de incrementar valor de filtrado G en resultados globales.

Podemos observar como esperábamos, que al aumentar G y por lo tanto aumentar el corpus de datos con objetos con mayor probabilidad de contener ruido en sus espectros, disminuye el porcentaje de datos correctamente agrupados, aunque esta disminución es muy pequeña, ésto es debido, a que los datos procedentes de SDSS han sufrido numerosas revisiones y para un mismo objeto estelar en el catálogo de SDSS, pueden existir numerosas observaciones y se han seguido de forma rigurosa criterios para la elección de la mejor observación de un objeto estelar y la eliminación de componentes de ruido en los datos durante la creación del repositorio.

Con los resultados obtenidos y suponiendo que en Gaia existirán unos mecanismos de eliminación de ruido de efectividad similar a los tratados en el repositorio SDSS, teniendo en cuenta que Gaia va a observar objetos con valores de magnitud mayores que los contenidos en SDSS, podemos concluir que para los mecanismos de preprocesamiento de datos y elección de atributos del presente trabajo, encontramos unos resultados de agrupación muy eficientes.

8.2 Efecto de introducir ruido sintético

Para el estudio de datos con ruido sintético, utilizamos un corpus de datos preparado por otra unidad de coordinación del consorcio, con los datos semi-empíricos procedentes de SDSS, con una componente de ruido desconocida y la única variación introducida fue la utilización del atributo de la mediana en vez de la media, más robusto frente al ruido en los datos, aunque la diferencia en los resultados por este cambio fue mínima al repetir la experimentación. Podemos observar el ruido con tan sólo visualizar los datos y encontrar, por ejemplo, valores en los atributos de los espectros negativos, esto es debido a que ha sido modificado el valor del atributo, al igual que el resto de los valores del espectro, con la adición de ruido siguiendo algún tipo de modelo desconocido. Concretamente, en los casos que el ruido resta valor al componente y éste tiene un valor muy pequeño o cero, este puede convertirse en un valor negativo que claramente nos indica que el espectro ha sufrido introducción de ruido sintético pues no tiene sentido físico tener una medición de flujo de energía negativa en los detectores. Como explicamos en el punto dedicado al preprocesamiento de los datos, la normalización a área unidad aplicada, no distorsiona el espectro frente a la introducción de ruido, en este punto vamos a evaluar especialmente la robustez de la normalización y preprocesamiento aplicado. Lamentablemente, la muestra proporcionada para la experimentación, que contiene 119.110 datos no se corresponde con los datos de la muestra proporcionada para la experimentación sin ruido, tiene la misma procedencia, la simulación a partir de los datos de SDSS, pero el diferente tamaño indica que contienen al menos un porcentaje de datos distintos y estas muestras no presentan identificadores que nos permitan extraer un subconjunto idéntico de datos sin ruido y con ruido para comparar con idéntica muestra el experimento. Por lo tanto deberemos conformarnos en este caso con la evaluación de la bondad de nuestro preprocesado de los datos, comparando los porcentajes que en caso de no existir ruido en ambas muestras deberían ser similares.

Respecto al conjunto de datos correctamente agrupados si tomamos la clase C0 como las estrellas, C1 cuásares y C2 galaxias, obtenemos:

Tabla 8.2: Comparación resultados insertando ruido sintético

		Sin ruido			Con ruido	
Cluster/Clase	Est.	Gal.	Cuás.	Est.	Gal.	Cuás.
C0	96.88 %	1.01 %	2.11 %	96.43 %	0.41 %	3.16 %
C1	0.79 %	3.42 %	95.79 %	0.6 %	2.45 %	96.95 %
C2	0.08 %	95.04 %	4.88 %	0.06 %	91.27 %	8.68 %

	Sin ruido	Con ruido
% Aciertos Totales:	96.19 %	95.47 %

Tabla 8.4 Comparación resultados datos sin ruido/con ruido globales.

Estos resultados parecen indicar que se realiza un tratamiento correcto de los datos que incorporan una pequeña componente de ruido dado que la calidad de los resultados global disminuye muy poco, sin embargo, no poder realizar las pruebas con el mismo conjunto de muestras limita nuestras conclusiones, por ejemplo observamos que aumenta especialmente la confusión con los cuásares mientras disminuye en menor medida entre las demás clases, también observamos que la calidad del cluster que identificamos con las galaxias disminuye sensiblemente, pero otros parecen aumentar ligeramente el número de aciertos y esto es debido únicamente a que las muestras utilizadas fueron otras, no podemos por tanto extraer conclusiones claras y válidas a partir de la matriz de confusión de clases, únicamente se deduce que la metodología utilizada se comporta frente a datos con ruido sintético, de forma robusta, por el porcentaje de aciertos total.

Capítulo 9

Conclusiones y trabajos futuros

9.1 Conclusiones

Este trabajo presenta un estudio y análisis de los distintos algoritmos que se usarán en OCA para poder escalar el algoritmo de agrupamiento MAC. Esto permite no sólo estudiar la viabilidad de las disintas técnicas empleadas sino también su idoneidad. Para ello se diseñaron varios experimentos que se aplicaron sobre muestras de datos simulados de Gaia.

También describe el diseño del *pipeline* de agrupamiento del paquete de trabajo OCA, el cual considera el paradigma divide y vencerás para poder escalar el agrupamiento.

Diferentes conclusiones que extraemos en nuestro trabajo son:

- La normalización de área unidad de la suma de espectros BP y RP, se presenta como la más robusta frente a los datos.
- Encontramos experimentalmente el número óptimo de componentes principales sobre los espectros que debemos utilizar.
- La descomposición con PCA mantiene la información de los espectros reduciendo el número de atributos, pero los componentes principales no son suficientes para obtener buenos resultados con la agrupación.
- La combinación de información estadística derivada de los espectros

junto a los componentes principales de éstos, produce los mejores resultados.

- La información astrométrica no mejora la agrupación, probablemente, debido a la limitación de las muestras a una pequeña región de la esfera celeste. Se necesitan más experimentos con nuevas muestras para extraer conclusiones.

- La complejidad temporal del algoritmo para la eliminación de agrupaciones curvilíneas de gran densidad hace que no podamos utilizarlo en nuestro problema.

- Debemos tener cuidado durante la experimentación con los datos simulados que utilizamos por los problemas que introducen los corpus de datos generados de forma sintética.

- K-means# y K-means++ se presentan como una buena técnica de muestreo para la obtención de representantes de grupo, además su escalado en ambos casos, es lineal con el tamaño del problema.

- Encontramos de forma experimental el tamaño adecuado de las particiones a utilizar con los datos de Gaia para el algoritmo de PCA incremental, con un tamaño de bloque entre 10^2 y 10^4 datos, pero deberemos realizar nuevos experimentos en los medios donde se use el *pipeline*, dado que el tamaño del bloque ideal puede variar respecto el sistema de ficheros utilizado.

- MAC debido a su complejidad espacial y temporal, necesita una estrategia de tipo divide y vencerás para afrontar la agrupación frente los datos de Gaia.

- La utilización de árboles kd en el algoritmo MAC no presenta las mejoras esperadas en coste temporal y presenta una disminución en la calidad de los resultados, un posible motivo es que la estructura de datos no es adecuada con una alta dimensionalidad.

- MAC se presenta como una buena propuesta para realizar el agrupamiento, ya que logró encontrar agrupaciones de gran calidad con muy poca confusión en los datos.

9.2 Trabajos futuros

Nuestro trabajo en OCA continuará la experimentación con los datos de nuevos ciclos que se reciban.

Mejoraremos la herramienta desarrollada para la caracterización de los grupos, cuando estén disponibles los atributos que faltan para caracterizar ciertas clases de objetos estelares y analizaremos en mayor profundidad los datos semiempíricos.

Estudiaremos la introducción de optimizaciones en los algoritmos, especialmente en aquellos con un mayor coste computacional, como HMAC, con mejoras como:

- Paralelización del código. La paralelización de HMAC es trivial, dado que podemos entenderlo como diferentes ejecuciones de MAC con diferentes valores de *bandwidth* independientes. También será interesante el estudio de la paralelización de los subalgoritmos de MAC. Esta paralelización junto la aplicación de la estrategia divide y vencerás permitirá acelerar la ejecución de MAC y plantear su uso para un número de datos para el que no es viable su utilización actualmente.

- Optimizaciones de los algoritmos utilizando los paradigmas de acceso a datos implementados, en concreto HEALPix.

- Estudiar en mayor profundidad el parámetro *bandwidth* e implementar mejoras en el cálculo automático de su valor y su ampliación para la generación de diferentes valores, correspondientes a diferentes jerarquías de HMAC.

- Estudiar la optimización del árbol KD, mejorando el acceso a disco respecto al tamaño de los bloques del sistema operativo, realizando

estudios sobre implementaciones que presentan mejoras como el el árbol KDB, una mezcla entre el árbol KD como estructura de datos y los B-Trees como estructura para almacenar de forma eficiente la información.

Otras posibles líneas de investigación, se refieren a la búsqueda de nuevos atributos sobre la información, la incorporación de conocimiento en los datos a partir de conclusiones extraídas sobre los datos observados, o el estudio de métodos alternativos para la reducción de la dimensionalidad como la utilización de *Diffusion Maps*, para intentar encontrar una proyección de los datos de los espectros donde no exista tanto solapamiento y que se muestre más eficaz que el análisis de componentes principales.

Apéndice A

Glosario

- Ascensión recta: Junto a la declinación, una medida de la posición de un objeto, especifica un lugar en el cielo respecto a un observador, de forma similar a la longitud, se expresa en horas, minutos y segundos, dado el movimiento estelar se debe especificar el momento temporal, el tiempo o época, actualmente se utiliza normalmente la referencia respecto al año 2000 o J2000.
- Bandwidth: Parámetro del algoritmo MAC, que controla la distancia y por lo tanto el número de vecinos que contribuyen en el cálculo de un punto hacia la moda.
- BP: Blue Photometer. Fotómetro de baja resolución incorporado en Gaia que opera en la banda azul, entre 330-680nm, hace referencia a los 60 datos discretos que forman un espectro para cada instancia de objeto observado o simulado. Junto a RP, forman los datos de espectrometría.
- CU (CU1-CU8) Coordination Unit. Ver la definición de DPAC, las unidades de coordinación son las diferentes unidades que forman DPAC, cada una de ellas es responsable de un aspecto clave en el procesamiento de los datos. En líneas generales las unidades de coordinación se ocupan de:
  - CU1: Definición del sistema general de procesamiento, filosofía, arquitectura y estrategia.

  - CU2: Responsable de cubrir las necesidades de simulación para el trabajo en otras unidades.
  - CU3: Cubre diferentes aspectos, desde la creación de los algoritmos centrales usados en los cálculos intermedios de los datos, su actualización, validación y test, hasta el proceso iterativo realizado con datos imprecisos sobre posiciones, parámetros de calibración de los instrumentos y globales para la obtención de datos astrométricos fiables.
  - CU4: Procesamiento de objetos, se encargará de analizar ciertos tipos de objetos particulares, provenientes de las unidades CU3, CU5, CU6 y CU7 como las estrellas no solitarias, estrellas binarias u objetos del sistema solar.
  - CU5: Se encarga del procesamiento de los datos fotométricos.
  - CU6: Encargada del procesamiento de los datos de espectrometría. El objetivo es monitorizar el estado del espectógrafo y calibrarlo, para obtener unos espectros limpios y calibrados al resto de unidades.
  - CU7: Unidad para el procesamiento de la variabilidad, a cargo de caracterizar la variabilidad de las características fotométricas y espectrales y modelizarlas.
  - CU8: Encargada de la estimación de parámetros astrofísicos y clasificación.
  - CU9: Crear y publicar los catálogos para la comunidad científica.

- Declinacion: Junto a la ascensión recta, una medida de la posición de un objeto, especifica un lugar en el cielo respecto a un observador, de forma similar a la latitud, expresada en horas, minutos y segundos, respecto a un determinado momento en el tiempo o época.

- DPAC: Data Processing and Analysis Consortium. Consorcio creado para el procesamiento y análisis de la información recogida. Es una comunidad de más de 400 científicos e ingenieros de software a lo largo

de más de 20 paises, está formada alrededor de un conjunto de unidades de coordinación, (CU), cada una de ellas responsable de un aspecto clave del procesamiento de los datos. Es responsable de la preparación de los algoritmos de análisis para los datos astrométricos y fotométricos y formar un marco de trabajo coherente común a todas las unidades, se encarga también de la generación y suministro de datos simulados para soportar el diseño previo a la misión, el diseño, desarrollo y soporte de los entornos de software y hardware necesarios y finalmente del diseño y desarrollo de la base de datos de Gaia que contendrá los puntos de interés para la comunidad científica desarrollados durante la misión.

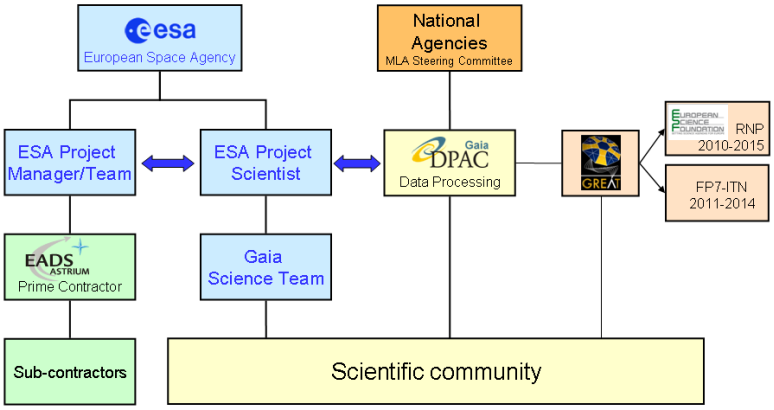


Figura A.1: DPAC dentro de la organización de la misión Gaia

- ESA: Agencia Espacial Europea.
- G: Ver *magnitud aparente*.
- Gaia: *Global Astrometric Interferometer for Astrophysics*. Satélite de la ESA cuyo lanzamiento proporcionará los datos para cuyo análisis se

ha creado DPAC. En el presente trabajo se utiliza este término para referirse a los datos procedentes de las observaciones del satélite, en numerosas ocasiones.

- Paralaje: Ángulo formado por la dirección de dos líneas visuales relativas a la observación de un mismo objeto desde dos puntos distintos, suficientemente alejados entre sí y no alineados con él. Los puntos utilizados de referencia son una estrella situada a una unidad astronómica de distancia (Sol) y Gaia. Utilizamos su inversa como aproximación a la distancia, expresada en parsecs.

- Magnitud absoluta: Magnitud aparente de un objeto si se encontrara a una distancia fija de 10 parsecs o 32.6 años luz del observador.

- Magnitud aparente: Conocida también como G. Es una medida del brillo del objeto de forma inversa y con una escala logarítmica, a mayor brillo menor magnitud. Es una medida de los flujos o cantidad de energía recibida por unidad de área y tiempo de los objetos celestes.

- Movimiento propio: El cambio angular de la posición a lo largo del tiempo visto desde el observador, medido en segundos de arco por año.

- RP: Red Photometer. Fotómetro de baja resolución incorporado en Gaia que opera en la banda roja, entre 640-1050nm, hace referencia a los 60 datos discretos que forman un espectro para cada instancia de objeto observado o simulado. Junto a BP, forman los datos de espectrometría.

- SDSS: *Sloan Digital Sky Survey*, es un proyecto de inspección del espacio mediante imágenes en el espectro visible, utilizamos este término en el presente trabajo, para hacer referencia a los corpus de datos utilizados, originados con simulaciones sobre observaciones de este repositorio.

Apéndice B

Pseudocódigo de MAC

Métodos utilizados que no codificamos en el anexo:

- *CalcularCriterioParada* simplemente calcula si se cumple las condiciones para terminar el bucle, bien por llegar a un número máximo de iteraciones o porque la máxima distancia entre una moda y el valor anterior de la moda no superan un umbral que depende de una constante y el valor de *bandwidth*. Ésta es la razón por la cual en las pruebas de rendimiento comentamos que la velocidad de ejecución también depende del valor del ancho de banda escogido, ya que con un número muy pequeño el algoritmo repetirá el bucle por cada punto hasta el máximo número de iteraciones, mientras que con un valor no demasiado pequeño, encontrará más rápidamente que la diferencia entre las modas no ha superado el umbral y parará antes de llegar al máximo número de iteraciones.

- *GaussianPDFLoglikelihood* es una función que calcula el logaritmo de la verosimilitud en un punto dado, usando la función de densidad de probabilidad Gaussiana.

SubAlgoritmo $ObtenerListaModas(kernel\_centers, probabilidad\_a\_priori, bandwidth, modas_ante$
begin
1: **Para** $i \leftarrow 0 Hasta \# kernel_centers$
2: $moda[i] \leftarrow EstimaModa(modas_anteriores)$
3: **Fin Para**
4: **devolver** moda
end

Figura B.1: Pseudocódigo ObtenerListaModas.

SubAlgoritmo $EstimaModa(datos)$
begin
1: $moda \leftarrow datos$
2: $anterior\_moda \leftarrow datos$
3: $iteracion \leftarrow 0$
4: **Repetir**
5: $verosimilitud, probabilidadPosterior \leftarrow ProbabilidadAPosteriori(moda)$;
6: $criterioDeParada \leftarrow CalcularCriterioParada(iteracion, bandwidth, moda, anterior\_moda,$
7: **Si**(!criterioDeParada)
8: $anterior\_verosimilitud \leftarrow verosimilitud$
9: $anterior\_moda \leftarrow moda$
10: $moda \leftarrow ActualizaModa(probabilidadPosterior)$
11: $iteracion \leftarrow iteracion + 1$
12: **Fin Si**
13: **Hasta** (criterioDeParada)
14: **devolver** moda
end

Figura B.2: EstimaModa realiza los pasos EM para encontrar la moda de un punto dado resolviendo un máximo local de una mezcla de densidades a través de iteraciones ascendentes, empezando por el punto dado

SubAlgoritmo *ProbabilidadAPosteriori*(*datos*)
begin
1: $max\_log\_posteriori \leftarrow 0$
2: **Para** $k \leftarrow 0 Hasta \# kernel\_centers$
3: $probabilidadPosterior[k] \leftarrow GaussianPDFLoglikelihood(datos, kernel_c enters[k], bandwidth)$
4: **Si** $k = 0$
5: $max\_log\_posteriori \leftarrow probabilidadPosterior[k]$
6: **Si no**
7: **Si**$(max\_log\_posteriori < probabilidadPosterior[k])$
8: $max\_log\_posterior \leftarrow probabilidadPosterior[k]$
9: **Fin Si**
10: **Fin Si**
11: **Fin Para**
12: {Calcula la densidad del kernel k en el punto x y la suma de las densidades en el punto x}
13: $sumaDensidades \leftarrow 0$
14: **Para** $k \leftarrow 0 Hasta \# kernel\_centers$
15: $probabilidadPosterior[k] \leftarrow probabilidad\_a\_priori[k] \times e^{max\_log\_posteriori}$
16: $sumaDensidades \leftarrow sumaDensidades + probabilidadPosterior[k]$
17: **Fin Para**
18: {Probabilidades a posteriori = densidad del kernel K / suma de densidades de kernels}
19: **Si** $sumaDensidades \leq 0$;
20: **Para** $k \leftarrow 0 Hasta \# kernel\_centers$;
21: $probabilidadPosterior[k] \leftarrow \frac{1}{\#kernel\_centers}$;
22: **Fin Para**
23: **Si no**
24: **Para** $k \leftarrow 0 Hasta \# kernel\_centers$
25: $probabilidadPosterior[k] \leftarrow \frac{probabilidadPosterior[k]}{sumaDensidades}$
26: **Fin Para**
27: **Fin Si**
28: **devolver** $max\_log\_posteriori + logaritmo(sumaDensidades)$
end

Figura B.3: ProbabilidadAPosteriori calcula el paso Expectation del algoritmo EM, calcula las probabilidades a posteriori en cualquier punto y devolverá el logaritmo de la verosimilitud del modelo de mezclas y el logaritmo de la suma de las densidades de los kernels.

SubAlgoritmo $ActualizaModa(probabilidadPosterior)$
begin
1: **Para** $i \leftarrow 0 Hasta \#kernel\_centers$
2: **Para** $j \leftarrow 0 Hastamoda$
3: $moda[j] \leftarrow moda[j] + probabilidadPosterior[i] \times kernel_c enters[i][j]$
4: **Fin Para**
5: **Fin Para**
6: **devolver** moda
end

Figura B.4: Paso Maximization, calcula las modas actualizadas con la contribución de cada kernel.

SubAlgoritmo $AgruparModas(modas)$
begin
1: $umbral \leftarrow umbral\_agrupacion \times bandwidth$
2: $resultados \leftarrow CrearResultados()$
3: $numero\_de\_clusters \leftarrow 1000$
4: **Para** $i \leftarrow 0 Hasta modas$
5: $terminado \leftarrow falso$
6: **Para** $j \leftarrow 0 Hasta modas$
7: $c\_moda \leftarrow resultados.ObtenerModaCluster(j)$
8: **Si** $MaximaDistancia(modas[i], c_moda)) < umbral$
9: $terminado \leftarrow falso$
10: resultados.AñadirPunto(j,i)
11: **Fin Si**
12: **Fin Para**
13: **Si** (!terminado)
14: resultados.AñadirCluster(CrearNuevoCluster(i,modas[i]))
15: **Fin Si**
16: **Fin Para**
17: **devolver** resultados
end

Figura B.5: Agrupa por la distancia entre modas asociadas. El valor del umbral para decidir si dos conglomerados se consideran un único cluster depende de un valor que podemos fijar en el algoritmo y que en el trabajo original de [29] era de 0.01 y en el ancho de banda utilizado. CrearResultados inicializa el objeto que contendrá los resultados, inicialmente se crean 1000 grupos vacíos, por eso el bucle interno, aunque no hay en el inicio clusters insertados, tiene un tamaño de 1000 elementos para recorrer. ObtenerModaCluster, AñadirPunto y AñadirCluster realizan la función que su nombre indica, el primero devuelve la moda de un determinado cluster y los demás añaden un punto a un conglomerado (AñadirPunto) o crean un nuevo cluster (AñadirCluster) cuando la máxima distancia entre su moda y la de los demás conglomerados supera un umbral en todos los conglomerados existentes, por lo que debemos crear un nuevo cluster para añadir a los resultados que tendrá una determinada moda y comenzará conteniendo ese punto.

Apéndice C

Aplicaciones complementarias desarrolladas

Durante el desarrollo del trabajo se han desarrollado numerosas utilidades, para diferentes tareas, como la automatización de la extracción y preparación de los datos, preprocesamiento, generación automática de los experimentos a partir de los datos originales, etc. Destacamos algunas de las implementaciones más interesantes desarrolladas, aparte de las utilidades desarrolladas con el objetivo de preparar y lanzar la experimentación.

- BWaleatorio: Peuqeña utilidad basada en una idea propia y buenos resultados empíricos. El artículo y trabajo original de Jia Li en [29], presenta una forma de calcular automáticamente un rango de valores para los *bandwidth*, si no se proporcionan valores al algoritmo. Esta forma de cálculo se implementó como una pequeña utilidad obteniendo los mismos resultados que el algoritmo original, sin embargo, no encuentran un buen valor de *bandwidth* para nuestras muestras, ni sobre diferentes conjuntos de datos que utilizamos como prueba, con magnitudes diferentes a los datos proporcionados en el algoritmo original de ejemplo. Por este motivo se realizó una aplicación que calculaba para un pequeño porcentaje de los datos sobre los que se realizará el clustering, la menor distancia con el siguiente elemento y la media de los siguientes 10 y 100 elementos, teniendo en cuenta el resto de datos, es

decir las distancias a las que se encontraba del siguiente y la media con los puntos cercanos, para todos los datos, se repite el proceso un pequeño número de veces, por si los puntos escogidos al azar se encontraran alejados de los demás o se deja escoger a la aplicación, que al obtener varios resultados similares o al llegar a un porcentaje de datos utilizados, para el proceso para dar la respuesta. El valor obtenido, demostró ser bastante útil, para tomarlo como punto inicial para el valor de *bandwidth*, aunque debe ser afinado posteriormente con los resultados obtenidos por el clustering de forma manual, pero obtenemos un buen valor de partida. El único argumento que necesita la utilidad, es el fichero de datos que vamos a utilizar para agrupar posteriormente con MAC. Aparte de aportar una estimación útil del parámetro bandwidth, tiene otra ventaja respecto al original y es que es mucho más rápido en su ejecución, ya que el coste temporal del algoritmo es de $\mathcal{O}(nlog(n))$, el coste de un algoritmo de ordenación.

- consultaSloan: Para el estudio de las propiedades físicas y caracterización de los grupos, dado que no se encuentran los parámetros que necesitamos en los datos simulados, hemos realizado la siguiente herramienta. Una vez obtenidos los clusters, extraemos los identificadores del objeto Gaia al que corresponde cada objeto. Volvemos a los datos sin tratar de las simulaciones y aunque no encontramos ahí los parámetros físicos que necesitamos, ni tampoco un identificador que relacione directamente el objeto del repositorio de simulaciones Gaia con el repositorio de SDSS, si encontramos una serie de parámetros que nos permiten identificar en el repositorio de SDSS una estrella, no es posible con cuásares pues no recogen actualmente más datos aparte de los espectros en las simulaciones ni con galaxias, pues en su caso existen dos datos uno que referencia al espectro en SDSS y otro la información astrofísica en SDSS, pero estos identificadores en SDSS pertenezcan a dos objetos distintos, no al mismo objeto. Con los identificadores de las estrellas, SDSS facilita la consulta por web de datos, de forma que una vez identificados los parámetros que necesitamos y la consulta contra

el repositorio de SDSS que debemos realizar, la aplicación conecta con una base de datos PostgreSQL local para obtener los parámetros. En caso de no encontrarse en la base de datos, consulta automáticamente la página web de SDSS, y procesa la página de respuesta para obtenerlos y guardalos en la base de datos para no repetir la consulta para el mismo objeto de nuevo en análisis posteriores. Finalmente, obtenemos tantos ficheros como grupos existen, pero en vez de contener el índice del objeto como los ficheros de grupos originales, contienen los parámetros físicos que nos interesan.

- K-means++ Implementación del algoritmo, integrada en el proyecto global del cual forma parte nuestro trabajo.

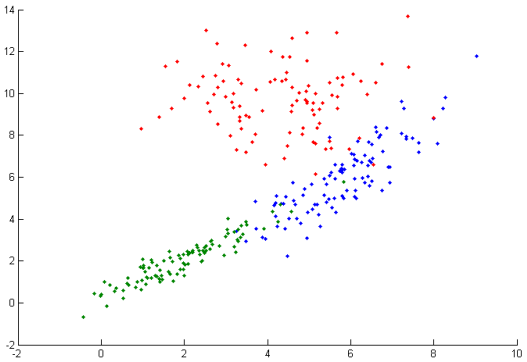
- K-means# Implementación del algoritmo, integrada en el proyecto global.

- MAC. El algoritmo MAC fué implementado, pero al encontrar una versión completa en el proyecto global del cual formamos parte, dejamos este código para pasar a trabajar con el ya desarrollado. Se llegó a implementar el código de MAC, pero no de otras técnicas como el algoritmo *ridgelines* presente en el mismo artículo de [29], que no hemos tratado en el presente trabajo.

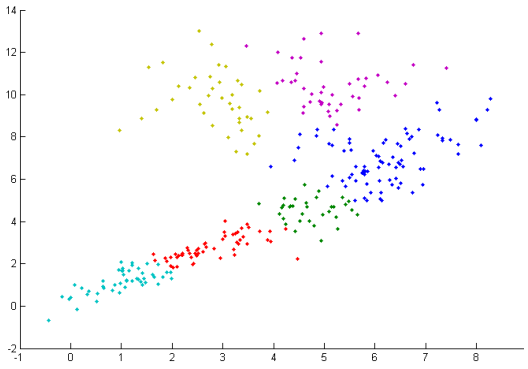
Apéndice D

HMAC a través del *bandwidht*

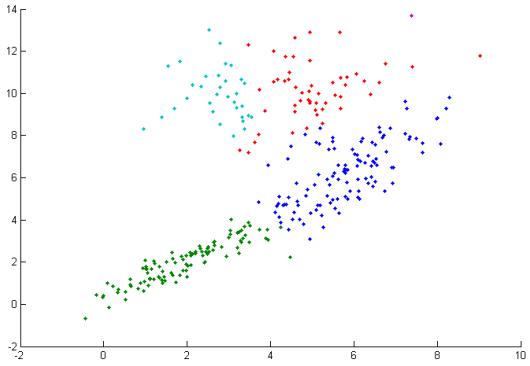
Presentamos unas gráficas con el propósito de mostrar el agrupamiento jerárquico de HMAC, a través del valor del parámetro *bandwidth*. Los experimentos se realizaron sobre 300 puntos pertenecientes a tres clases generadas sintéticamente a partir de tres distribuciones normales que se solapan. Como se explica en el punto 4.2.3, cuando el ancho de banda se incrementa la estimación de densidad por el kernel se suaviza y más puntos tienden a ascender a la misma moda mientras que al disminuir el bandwidth, puntos que antes convergían a la misma moda ahora lo hacen a modas distintas, por lo tanto a menor ancho de banda obtendremos un mayor número de conglomerados y HMAC aprovecha esta característica para formar un modelo jerárquico de agrupaciones.
La imagen con la distribución original de los datos es:



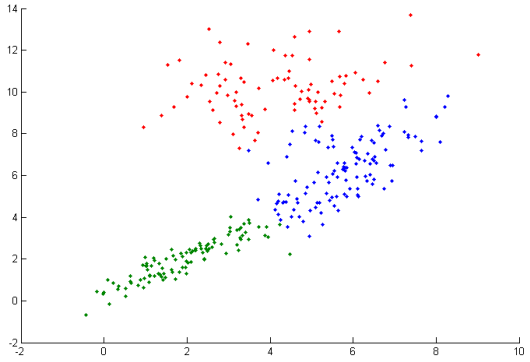
Se ejecutará HMAC sobre los valores de ancho de banda siguientes: $\sigma_1 = 0{,}25 < \sigma_2 = 0{,}5 < \sigma_3 = 0{,}8 < \sigma_4 = 1{,}5$
Un aspecto característico que podremos observar con los diferentes valores es cómo este algoritmo jerárquico no utiliza las distancias para la unión de conglomerados en un grupo mayor, sino que la ejecución con todos los puntos del algoritmo MAC de nuevo para otro valor de ancho de banda produce unos resultados, en los cuales puntos que pertenecían a un grupo que converge a una nueva moda fusionándose con otro conglomerado, pasa a ascender a una moda de otro conglomerado distinto. Para observar este efecto, debemos observar por ejemplo que ocurre con los puntos que se encuentran en la frontera de decisión de un conglomerado, cuando utilizamos un valor distinto para el ancho de banda. Los resultados para $\sigma_1 = 0{,}25$ encuentran 6 agrupaciones distintas, si agrupamos los cluster en parejas de forma adecuada, encontraremos conglomerados similares a los originales, aunque un elevado número de clusters dificulta la identificación de los grupos naturales.



Para $\sigma_2 = 0{,}5$ encontramos 4 grupos, si observamos algunos grupos se han fusionado pero esta fusión no es exactamente la fusión de todos sus puntos, al volver a ejecutarse MAC de nuevo para el nuevo valor de ancho de banda la fusión de grupos no es la simple unión de grupos a menor distancia, sino que mucho puntos han pasado a converger a otras modas y por lo tanto a pertenecer a otros conglomerados.



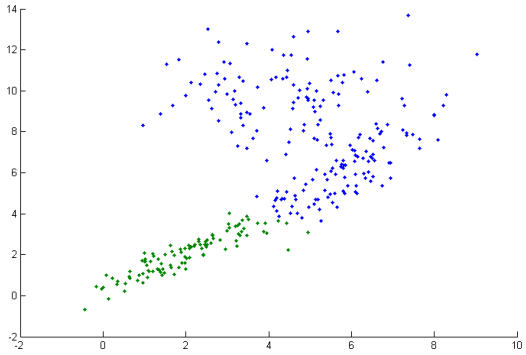
Con $\sigma_3 = 0{,}8$ encontramos tres grupos, el mismo número de conglomerados que los grupos naturales o clases de los datos del experimento



La matriz de confusión nos indica los buenos resultados obtenidos:

	Clase 0	Clase 1	Clase 2
Cluster 1	91	5	19
Cluster 2	8	94	0
Cluster 3	1	1	81

Para $\sigma_4 = 1{,}5$ encontramos dos conglomerados, para este valor de *bandwidth*, los puntos convergen tan sólo a dos modas diferentes, de nuevo podemos observar cómo en la frontera de decisión, algunos puntos han pasado de un grupo a otro, dado que la ejecución de MAC con un valor distinto de bandwidth vuelve a calcular la contribución de todos los puntos, no es la simple unión de grupos anteriores.


14
12
10
8
6
4
2
0
-2
-2
0
2
4
6
8
10

Bibliografía

[1] E. Acuña. Notas de análisis discriminante. Technical report, Departamento de Matemáticas. Universidad de Puerto Rico., 2000. 4.2.2

[2] N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. In *Advances in Neural Information Processing Systems*, NIPS, 2009. 4.1.2, 4.1.2

[3] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007. 4.1.1, 4.1.1

[4] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 1st ed. 2006. corr. 2nd printing edition, 2006. 4.2.1

[5] P. Bradley, U. Fayyad, and R. Reina. Clustering very large databases using em mixture models. In *Proceedings of 15th International Conference on Pattern Recognition. Vol 2*, pages 76–80, 2000. 5

[6] S. Brecheisen, H. Kriegel, and M. Pfeifle. Parallel density-based clustering of complex objects. In *Lecture Notes in Computer Science*, volume 3918, pages 179–188, 2006. 2.3, 2.1

[7] J. C. Canales. *Clasificación de grandes conjuntos de datos vía Máquinas de Vectores Soporte y aplicaciones en sistemas biológicos*. PhD thesis, Instituto Politécnico Nacional de Méjico, 2009. 5.1

[8] H. Choi and E. J. Yoon. *Apache HAMA: An Introduction to Bulk Synchronization Parallel on Hadoop*. Apache Hama Team, 2012. 2.4

[9] S. Chu, J. F. Roddick, and J. Pan. A incremental multi-centroid, multi-run sampling scheme for k-medoids-based algorithms. In *Proceedings of the Third International Conference on Data Mining Methods and Databases, Data Mining III*, 2002. 2.2, 2.1

[10] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008. 2.4

[11] M. Delgado and P. Robinson. Nonparametric and semi-parametric methods for economics research. *Economic Surveys*, 6:201–249, 1992. 4.2.2

[12] Cu8 training data: Cycle 6 requirements, 2009. 3.1

[13] DPAC. ESA. *DPCC: How-To: use the MDB explorer*, 2012. 3.5.1

[14] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *In Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996. 2.2, 2.1

[15] K. N. A. et al. The seventh data release of the sloan digital sky survey. *The Astrophysical Journal Supplement Series*, 182:543–558, 2009. 1.1

[16] V. Ganti, J. Gehrke, and R. Ramakrishnan. Mining very large databases. *COMPUTER*, 32(8):38–45, 1999. 5

[17] A. Garg, A. Mangla, N. Gupta, and V. Bhatnagar. Pbirch: A scalable parallel clustering algorithm for incremental data. In *10th International Database Engineering and Applications Symposium*, pages 315–316, 2006. 2.3, 2.1

[18] J. R. Gott, M. Juric, D. Schlegel, F. Hoyle, M. Vogeley, M. Tegmark, N. Bahcall, and J. Brinkmann. A map of the universe. *The Astrophysics Journal*, 624(2):463–484, 2005. 3.2

[19] M. J. Graham. How to analyze a petabyte. In *Gaia Research for European Astronomy Training*, 2011. 2.4

[20] J. Greaves. Sdss data release 6 proper motion information. Technical report, The Webb Society, 2008. 7.1

[21] K. M. Górski. Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622:759–771, 2005. 5.1

[22] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In *In Proceedings of ACM SIGMOD*, pages 73–84, 1998. 2.1, 2.1

[23] J. O. Jacob E. Goodman and P. Indyk. *Handbook of Discrete and Computational Geometry*, chapter Chapter 39:Nearest neighbours in high-dimensional spaces. CRC Press, 2004. 5.2.4

[24] M. N. Joshi. Parallel k-means algorithm on distributed memory multiprocessors, 2003. 2.3, 2.1

[25] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990. 2.2, 2.1

[26] M. Khalilian, N. Mustapha, N. Suliman, and A. Mamat. A novel k-means based clustering algorithm for high dimensional data sets. In *Proceedings of the International MultiConference of Engineers and Computer Scientist*, volume I, 2010. 2.1, 2.1

[27] W. Kim. Parallel clustering algorithms: Survey, 2009. 2.3

[28] H. Kirbiyik. Rotation in stars. *Turkish Journal of Physics*, 22:83–106, 1998. 3.1

[29] J. Li, S. Ray, and B. G. Lindsay. A nonparametric statistical approach to clustering via mode identification. *Journal of Machine Learning Research*, 8(8):1687–1723, 2007. 1.2, 4.2.2, B.5, C

[30] S. LLoyd. Least square quantization in pcm. *IEEE Transactions on Information Theory*, pages 129–137, 1957. 2.1

[31] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967. 4.1

[32] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, SIGMOD '10, pages 135–146. ACM, 2010. 2.4

[33] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. 4.1

[34] M. A. F. Montecelo. *Implementation of a Distributed Clustering System for Gaia Space Mission*. PhD thesis, UNED, 2010. 7.3

[35] D. P. na. *Análisis de datos multivariantes*, chapter Componentes Principales. McGraw-Hill Interamericana, 2003. 3.5.3

[36] M. F. Pace. Bsp vs mapreduce. *Procedia Computer Science*, 9:246–265, 2012. 2.4

[37] D. Peng and F. Dabek. Large-scale incremental processing using distributed transactions and notifications. In *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation*, 2010. 2.4

[38] A. M. y. B. F. Pierre-Marie Brunet. Big data challenges, an insight into the gaia hadoop solution. Technical report, CNES, 2012. 5.1

[39] F. Provost and V. Kolluri. A survey of methods for scaling up inductive algorithms. *Data Mining and Knowledge Discovery*, 3(2):131–169, 1999. 5

[40] Research and Scientific Support Department. ESA. *CU1:DalToolsExample*, 2012. 3.5.1

[41] Research and Scientific Support Department. ESA. *CU8: DictionaryTool*, 2012. 3.5.1

[42] Measures of flux and magnitude. sloan digital sky survey iii documentation. http://www.sdss3.org/dr8/algorithms/magnitudes.php. 3.4

[43] S. Seo, E. J. Yoon, J. Kim, S. Jin, J. Kim, and S. Maeng. Hama: An efficient matrix computation with the mapreduce framework. In *2nd IEEE International Conference on Cloud Computing Technology and Science*, pages 721–726, 2010. 2.4

[44] F. Shahnaz. Clustering: Large databases. *Lecture Notes In Data Mining*, pages 133–141, 2006. 2

[45] J. Shlens. *A Tutorial on Principal Component Analysis*. Center for Neural Science, New York University, 2009. 3.5.3

[46] B. W. Silverman. *Interpreting multivariate data*, chapter Density estimation for univariate and bivariate Data. Number 37–53. Wiley, 1981. 4.2.2

[47] B. W. Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, 1986. 4.2.2

[48] K. W. Smith. How to use oversampled simulated bprp data. dpac technical note. Technical report, DPAC, 2008. 3.4

[49] R. Sordo and A. Vallenari. Description of the cu8 cycle 6 simulated data, 2010. 3.1

[50] D. Talia. Parallelism in knowledge discovery techniques. In *Proceedings of the 6th International Conference on Applied Parallel Computing Advanced Scientific Computing*, pages 127–138, 2002. 2.3

[51] F. Thevenin. *Interface Control Document for the Gaia Spectral libraries*. 3.5.1

[52] P. Tsalmantza, M. Kontizas, C. A. L. Bailer-Jones, I. Bellas-Velidis, E. Kontizas, E. Livanou, B. Rocca-Volmerange, R. Korakitis, A. Dapergolas, A. Karampelas, M. Fioc, and A. Vallenari. Semi-empirical library of galaxy spectra from sdss, 2008. 3.1

[53] A. Vallenari. Cu8 simulation specifications for algorithm development cycle 2b, 2008. 3.1

[54] F. van Leeuwen. Organizing and accessing the multiple data streams. Technical report, Institute of Astronomy.Cambridge, 2012. 5.1

[55] G. J. Vicente. *Distribuciones de clases no balanceadas: Métricas, análisis de complejidad y algoritmos de aprendizaje.* PhD thesis, Universidad Jaume I, 2011. 3.2

[56] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, volume 25, pages 103–114, 1996. 2.2, 2.1

[57] Y. Zhang, Z. Xiong, J. Mao, and L. Ou. The study of parallel k-means algorithm. In *The Sixth World Congress on Intelligent Control and Automation*, pages 5868–5871, 2006. 2.3, 2.1