Universidad Central de Venezuela

Facultad de Ciencias

Escuela de Matemáticas

Técnicas de Aprendizaje no Supervisado para la Clusterización de Transacciones Bitcoin

Trabajo Especial de Grado presentado ante
la ilustre Universidad Central de Venezuela
por el **Br. Alfredo J. Quintana R.** para optar al
título de Licenciado en Matemática.

Tutor: Dr. José B. Hernández.

Co-Tutor: Msc. Jesús Lares.

Caracas, Julio y 2018

2

Nosotros, los abajo firmantes, designados por la Universidad Central de Venezuela como integrantes del Jurado Examinador del Trabajo Especial de Grado titulado "**Técnicas de Aprendizaje no Supervisado para la Clusterización de Transacciones Bitcoin**", presentado por el **Br. Alfredo José Quintana Rodríguez**, titular de la Cédula de Identidad **20365469**, certificamos que este trabajo cumple con los requisitos exigidos por nuestra Magna Casa de Estudios para optar al título de **Licenciado en Matemática**.

José Benito Hernández
Tutor

por: prof. Robinson Rivas

Jesús Lares
Jurado

Mairene Colina
Jurado

Índice general

0.1. Resumen

Un estudio y análisis de cluster para transaccionoes Bitcoin usando el algoritmo K-medias y un modelo Mixto Gaussiano.

0.2. Palabras Claves:

Blockchain, Cluster, Gaussiana.

Dedicado a mi familia, por su siempre constante apoyo, esto es gracias a ustedes.

Primeramente a Dios, y a mi familia, a mi Madre y Padre, mi hermano, mis TÃas. Gracias a la Universidad Central de Venezuela y los profesores que hacen posible nuestra formaciÃ^3n.

El problema del reconocimiento de patrones en conjuntos de datos es un problema fundamental y tiene una larga y exitosa historia. Este importante campo de la ciencia estudia el descubrimiento automático de regularidades en los datos mediante el uso de algoritmos computacionales y el uso de estas regularidades para llevar cabo acciones como clasificar los datos en diferentes categorías.

Por otra parte, vivimos en un mundo interconectado a través de la información. Las primeras cuatro décadas de internet nos han traído el correo electrónico, la red informática global (world wide web), las empresas electrónicas, los medios sociales, la red móvil, el almacenamiento en la nube y los primeros días del internet de las cosas. Internet ha servido para reducir los costos de investigar, colaborar e intercambiar información. Ha permitido la aparición de nuevos medios de comunicación y entretenimiento, de nuevas formas de comerciar y de organizar el trabajo, y de empresas digitales como nunca las ha habido. Gracias a la tecnología de sensores, se ha incorporado inteligencia en nuestras carteras, en nuestra ropa, en nuestros automóviles, en nuestros edificios, en nuestras ciudades y hasta en nuestra biología, es por esto que es tan importante el análisis de todos los datos que generan estas nuevas tecnologías, he ahí el papel fundamental de la ciencia de datos como rama interdisciplinaria capaz de convertir semejante cantidad de información en conocimiento valioso y novedoso.
En general, internet ha posibilitado muchos cambios positivos para los que pueden acceder a la red, pero tiene serias limitaciones para los negocios y la actividad económica. En una fecha tan temprana como 1981 ya había expertos tratando de resolver con criptografía los problemas de privacidad, seguridad e inclusión que internet planteaba. Reformaran como reformasen el proceso, siempre se producían filtraciones, porque había terceras partes implicadas. En el 2008, el sistema financiero global se hundió. Quizá aprovechando el momento, una persona o serie de personas, con el pseudónimo de Satoshi Nakamoto, esbozaron el protocolo de un nuevo sistema de pago electrónico directo y entre iguales (peer-to-peer o P2P) que usaba una criptomoneda llamada "bitcoin". Las criptomonedas (monedas digitales) se diferencian de la moneda tradicional en que no las crean ni las controlan los países. Este protocolo establece una serie de normas en forma de computación distribuida que garantiza la integridad de la información intercambiada entre esos miles de millones de ordenadores sin pasar por terceros.
El comercio en Internet ha llegado a depender casi exclusivamente de instituciones financieras fungiendo como terceros de confianza para procesar pagos electrónicos. Si bien es cierto que el sistema funciona lo suficientemente bien para la mayoría de las transacciones, aún sufre de las debilidades inherentes del modelo basado en la confianza. Transacciones completamente no reversibles no son realmente posibles, debido a que las instituciones financieras no pueden evitar mediar disputas. El costo de la mediación aumenta los costos de las transacciones, limitando así el tamaño mínimo de transacciones prácticas y eliminando la posibilidad

de pequeñas transacciones casuales. Lo que propone Nakamoto es un sistema electrónico de pago basado en la prueba criptográfica en lugar de la confianza, permititendo a dos partes interesadas realizar transacciones directamente el uno con el otro sin la necesidad de un tercero de confianza.

El objeto de este trabajo es exponer e implementar una serie de técnicas provenientes del campo del Aprendizaje de Máquinas para la busqueda de patrones dentro de la cadena de bloques que puedan ser usados como conocimiento valioso en la toma de decisiones, particularmente en la detección de transacciones sospechosas.

En el primer capítulo nos proponemos esbozar los fundamentos de la tecnología Blockchain y el Bitcoin, partiendo desde sus definiciones mas generales y su funcionamiento. Se listarán una serie de casos de uso de la tecnología Blockchain y la historia del primero y mas famoso caso de uso : La red Bitcoin. Luego haremos enfásis en describir la manera en que son almacendas, procesadas y ejecutadas las transacciones Bitcoin.

Si bien es cierto que el diseño de la cadena de bloques permite realizar transacciones entre dos partes sin la necesidad de compartir información personal, existen un conjunto de herramientas desarrolladas por Fergal Reid y Martin Harrigan cuyo objeto es el de analizar el anónimato en la red Bitcoin. En el segundo capítulo se exponen de manera parcial los resultados de dicho trabajo, y que usaremos para construir una tabla de transacciones a partir de la cadena de bloques.

Luego de construir una tabla de transacciones, nos proponemos la tarea de detectar patrones que permitan la clusterización de dichas transacciones. Para llevar esta tarea a cabo nos valdremos de las herramientas y resultados provenientes de la estadística y el aprendizaje no supervisado. Nuestra vista minable, o tabla de transacciones, es un conjunto de datos no etiquetado, es decir que cada observación, en este caso una transacción, no posee un identificador de ningún tipo, es por esta razón que implementaremos técnicas de clusterización sobre la misma, para poder identificar grupos que describan las características de las transacciones.

Para la obtención de los datos y la implementación práctica usaremos los lenguajes de programación predilectos en el campo de la ciencia de datos, *R y Python*.

Capítulo 1

Blockchain y Bitcoin

1.1. Introducción

Blockchain (o Cadena de Bloques) es una tecnología que consiste de una lista de registros en constante crecimiento, a dichos registros se les llama bloques y están conectados y asegurados entre si usando criptografía. Podemos pensar en ella como un registro contable público y distribuido. Cada bloque normalmente contiene información como una función Hash, una marca de tiempo y datos de la transacción. Por su diseño, las cadenas de bloques son inherentemente resistentes a la modificación de sus registros son normalmente administrados por una red P2P (Peer to peer) de manera colectiva apegados a un protocolo para la generación de nuevos bloques.

Una Cadena de Bloques es un registro contable digital y distribuido de transacciones, registradas y replicadas en tiempo real a lo largo de una red de computadores o nodos. Cada transacción debe ser criptográficamente validada a través de un mecanismo de consenso ejecutado por los nodos antes de ser agregada como un nuevo bloque al final de la cadena. No existe la necesidad de una autoridad central para aprobar la transacción, razón por la cual la Cadena de Bloques es en ocasiones catalogada como un mecanismo de confianza par a par (p2p). La tecnología Blockchain permite la administración segura de un registro contable seguro, donde transacciones son verificadas y almacenadas en una red sin un gobierno o autoridad central. Puede presentarse en diferentes configuraciones, variando desde redes open-source públicas hasta Cadenas de Bloques privadas que requieren permisología explícita para leer o escribir. Computación y matemáticas avanzadas (en forma de funciones hash criptográficas) son lo que hacen que la tecnología funcione, no solo haciendo posible las transacciones sino protegiendo la integridad y anonimato de la misma.

1.2. Cómo Funciona Blockchain?

- **Transacciones :** Dos partes intercambian datos, estos podrían representar dinero, contratos, títulos de propiedad, registros médicos, detalles de un cliente, o cualquier otro activo que pueda ser descrito de forma digital.
- **Verificación :** Dependiendo de los parametros de la red, la transacción es verificada intantáneamente o transcrita en un registro asegurado y asignado a una cola de transacciones pendientes. En este caso, los nodos (computadores o servidores que integran la red) determinan si las transacciones son válidas basándose en un conjunto de reglas que la red ha acordado.
- **Estructura :** Cada bloque es identificado por un *hash*, un número 256-bits, creados usando un algoritmo estipulado por la red. Un bloque contiene un encabezado, una referencia al *hash* del bloque anterior y un grupo de transacciones. La secuencia de hashes enlazados crea una cadena segura e interdependiente.
- **Validación :** Los bloques deben ser validados antes de ser agregados a la blockchain. La forma mas aceptada de validación para blockchains open-source es una **prueba de trabajo**, que es una solución a una ecuación derivada del encabezado del bloque.
- **Minería Blockchain :** Mineros tratan de resolver el bloque haciendo cambios incrementales a una variable hasta que la solución satisfaga el objetivo de toda la red. Esto es llamado *prueba de trabajo* porque respuestas correctas no pueden ser falsificadas; soluciones potenciales deben probar que el nivel apropiado de potencia de cómputo fue agotado para hallar la solución.
- **La Cadena :** Cuando un bloque es validado, los mineros que lo resuelven son compensados y el bloque es distribuido a lo largo de la red. Cada nodo agrega el bloque a la cadena de la mayoría, la red inmutable y auditable blockchain.
- **Defensa Incorporada :** Si un minero malicioso tratara de enviar un bloque alterado a la cadena, la función hash de ese bloque y de todos los bloques siguientes, cambiarían. Los nodos restantes detectarían estos cambios y rechazarían el bloque de la red la mayoría, previniendo la corrupción de la misma.

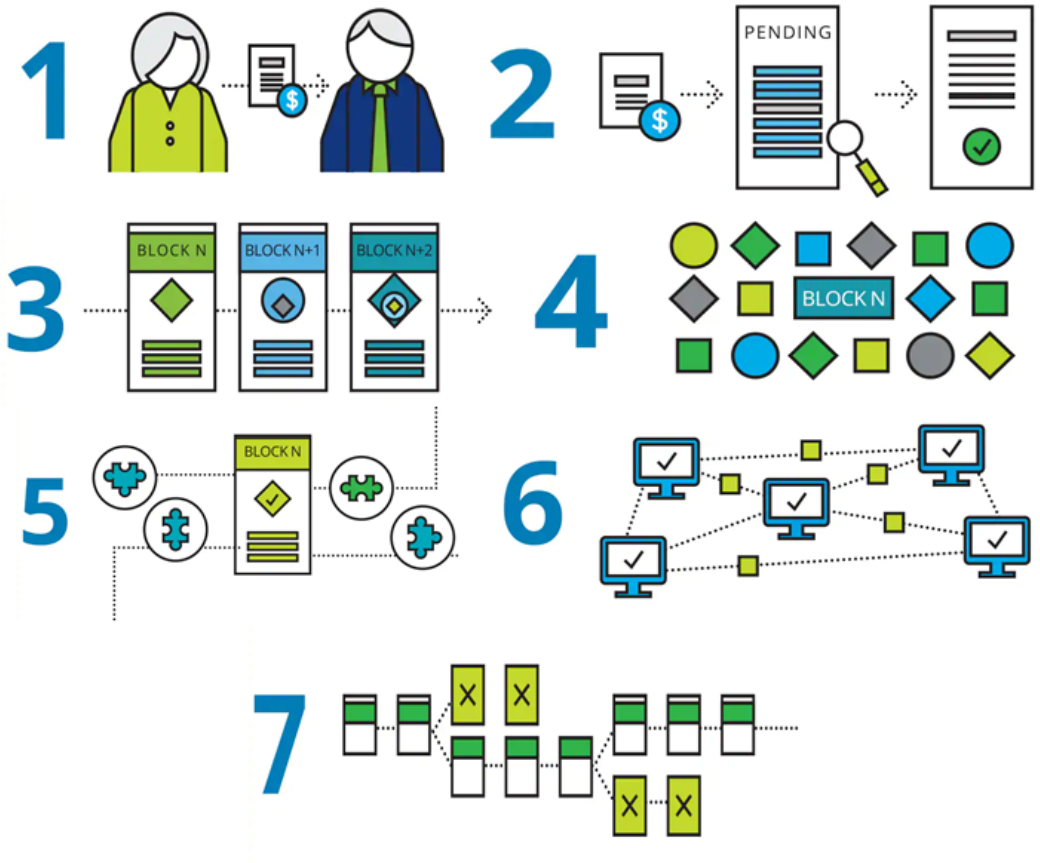

Figura 1.1: Esquema gráfico del funcionamiento de la tecnología Blockchain.

1.3. Blockchain y Casos de Uso

En la Cadena de Bloques pueden identificarse tres niveles de utilización: Almacenamiento de registros digitales, intercambio de bienes digitales y registro y ejecución de contratos inteligentes. La Cadena de Bloques puede ser usada para almacenar identidades digitales de individuos, organizaciones, bienes, títulos, derechos de voto, y esencialmente cualquier cosa que pueda ser representada de forma digital. Puede ejecutar transacciones persona a persona sin la presencia de intermediarios de confianza, reduciendo el costo y el tiempo de compensación y liquidación. Los contratos inteligentes son códigos digitales que habilitan la ejecución automática de acciones específicas basadas en condiciones estipuladas y validadas por todas las partes implicadas. Por ejemplo contratos inteligentes de arrendamiento, los cuales habilitan los pagos de renta automáticos, o en contratos inteligentes de préstamos, los cuales transfieren la propiedad de los bienes en caso de incumplimien-

to. Contratos inteligentes almacenados en la cadena de bloques son inmutables, por lo que es ncesario código de alta calidad para evitar errores y fraudes. Entre las aplicaciones a lo largo de diversas industrias podemos citar:

- Servicios Financieros: pagos internacionales en una forma mas rápida, barata y segura.
- Sector Público: Registro para administrar la identidad digital de las personas y la información de propiedad y transacciones de diferentes bienes como inmuebles y vehículos para aumentar la eficiencia y reducir el fraude.
- Energía y recursos: Contratos inteligentes para una ejecución mas eficiente y rápida de comercialización y pagos de nergía.

1.4. Historia de la Red Bitcoin

Bitcoin es un sistema de moneda electrónica sobre una red par a par (p2p) el cual fue descrito por primera vez por Satoshi Nakamoto en 2008. Dicho sistema se basa en firmas digitales para probar la propiedad y un registro público de las transacciones para prevenir el doble gasto de la moneda. Las primeras transacciones Bitcoin ocurrieron en Enero de 2009. En Octubre del mismo año New Liberty Standard publica una tasa de cambio fijada en

$$1US\$ = 1309{,}03BTC,$$

usando una ecuación que incluía el costo de la electricidad consumida por una computadora para generar Bitcoin . Para Junio de 2011 habían 6.5 millones de Bitcoins en circulación entre un estimado de 10000 usuarios. Desde entonces la moneda ha sufrido un rápido crecimiento tanto en la atención mediática como en su precio relativo a otras monedas ya existentes.

Muchos usuarios han adoptado Bitcoin tanto como por razones políticas y filosóficas como por razones prágmaticas. Los ususarios mas técnicos consideran que la anonimidad del sistema no es un objetivo principal del diseño del mismo; no obstante las opiniones varian ampliamente acerca de cuan anónimo es el sistema en la práctica.

1.5. Transacciones Bitcoin

S. Nakamoto define una moneda electrónica como una cadena de firmas digitales. Cada portador transfiere la moneda al siguiente firmando digitalmente un hash de la transacción anterior y la clave pública del

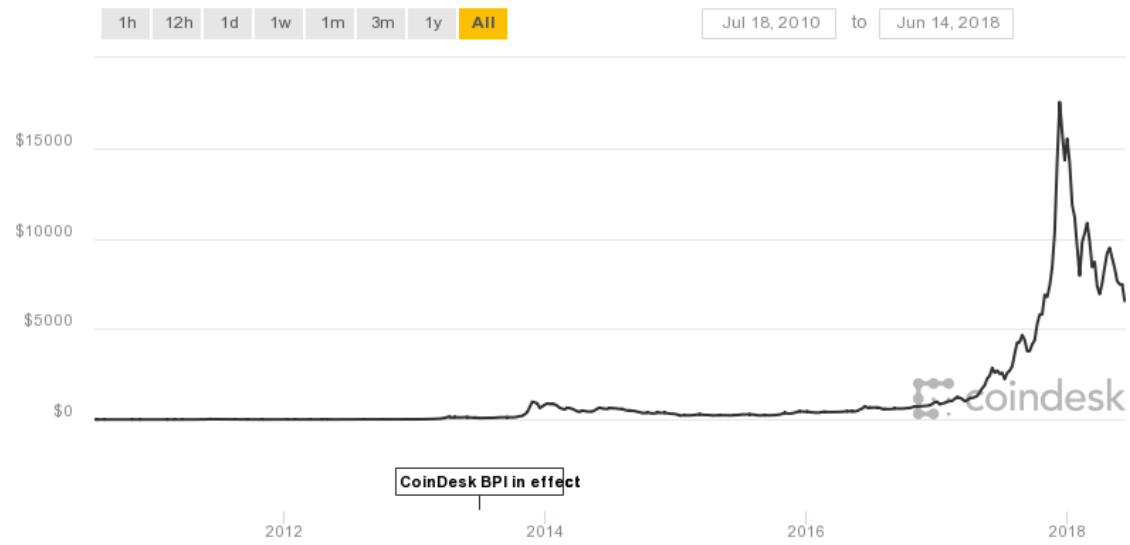

Figura 1.2: Histórico de precios en Dólares americanos del bitcoin.

siguiente dueño, añadiendo estas al final de la moneda. En los sistemas financieros tradicionales, se introduce la figura de autoridad central o casa de moneda, con el objetivo de solucionar el problema de que un portador gaste la misma moneda dos veces. El problema con esta solución radica en que el destino de todo el sistema monetario depende de la compañía a cargo de la casa de moneda, con cada transacción pasando a través de ella, como en un banco.

Una transacción es un intercambio de datos con una firma digital, cada una de ellas es transmitida a la red y almacenada en los bloques. El conjunto de datos dentro de una transacción incluye una referencia a una transacción anterior e indica una cantidad de Bitcoins que pasan a estar disponibles para una dirección Bitcoin de destino. Esta información no se encuentra cifrada, de la misma manera que no lo está el resto de la información que se almacena en la cadena de bloques.

Siendo el objetivo de este trabajo desarrollar técnicas de aprendizaje no supervisado para la clusterización de transacciones, es de sumo interés detallar la forma en la que se encuentran almacenadas las mismas dentro de la cadena de bloques. En la Figura 1.3 podemos observar un ejemplo de una transacción.

La **entrada** (input) de esta transacción transafiere 50 BTC desde la salida (output) #0 de la transacción f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6. La salida de esta transacción asigna esos 50 BTC a una dirección Bitcoin. Una **entrada** (input) es una referencia a una salida de otra transacción existente. Cada transacción puede tener múltiples entradas, sumándose los valores de cada salida. El valor

```
Input:
Previous tx: f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6
Index: 0
scriptSig: 304502206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446618c4571d10
90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6cc8d25c6b241501

Output:
Value: 5000000000
scriptPubKey: OP_DUP OP_HASH160 404371705fa9bd789a2fcd52d2c580b65d35549d
OP_EQUALVERIFY OP_CHECKSIG
```

Figura 1.3: Ejemplo de la estructura de una transacción Bitcoin.

total de las entradas es la suma máxima disponible para las salidas. El valor Previous tx es un hash que representa una transacción anterior. **Index** es el índice de la salida concreta en la transacción de origen. **ScriptSig** es la primera parte de un script. El script consta de dos componentes : una firma y una clave pública. La clave pública pertenece al propietario de la transacción de origen cuya salida se utiliza como entrada y demuestra que el creador de la transacción está autorizado para gastar la suma de esas salidas de la transacción previa. El otro componente es una firma digital ECDSA (algoritmo de firma digital con curvas elípticas) aplicada a un hash de una versión simplificada de la transacción. Esa firma digital basada en criptografía asimétrica de curvas elípticas, junto a la clave pública permite verificar que la transacción ha sido generada por el auténtico propietario de la dirección en cuestión. Hay además varios parámetros que definen cómo simplificar la transacción y que permiten crear diferentes tipos de pago.

Una **salida** contiene las instrucciones para enviar la suma de bitcoins. **Value** es el número de satoshis, donde

$$1 \text{ BTC} = 100000000 \text{ satoshis},$$

que la salida pone a disposición del destinatario. **ScriptPubKey** es la segunda mitad de un script que se usa para la verificación. Pueden existir mas de una salida, de modo que todas ellas se reparten el valor combinado de las entradas. Debido a que una salida solamente puede conectarse a una entrada única, todo el valor neto de las entradas tendrá que enviarse a través de alguna salida si no se desea perderlo. Tomemos como ejemplo un caso en el que la entrada es 50 BTC, pero solamente se desean enviar 25 BTC, se tendrán que crear dos salidas con valor de 25 BTC: una para el destinatario y otro que vuelva a una dirección del pagador. La diferencia que pueda haber entre la suma de bitcoins de las entradas y las salidas se considerará una comisión de transacción y se asignará a la dirección en la que se genere el bloque que registra la transacción.

Para verificar que las entradas tienen autorización para disponer de los valores de las salidas a las que se hace

referencia, Bitcoin utiliza un lenguaje de script similar a Forth. El valor scriptSig de la entrada y la referencia scriptPubKey de la salida se evalúan en ese orden, de modo que scriptPubKey utiliza los valores que deja scriptSig en la pila. La entrada se autoriza si scriptPubKey retorna valor *true*. Mediante este sistema de script, el remitente puede crear condiciones sumamente complejas para que se pueda acceder a las cantidades de salida.

Existen 3 tipos de transferencias Bitcoin:

- **Transacción a una dirección IP :** El remitente obtiene la clave pública al comunicarse con el receptor por IP. Al gastar monedas enviadas a una dirección IP, el receptor solamente presenta una firma, que se coyeja con la clave pública de *scriptPubKey*.
- **Transferencia a una dirección Bitcoin :** Una dirección Bitcoin es simplemente un hash, por lo que el remitente no puede proporcionar una clave pública en *scriptPubKey*. Cuando se gastan los bitcoins que han sido recibidos previamente en una dirección Bitcoin, el remitente proporciona tanto la firma como la clave pública. El script verifica que la clave pública proporcionada tiene el mismo valor hash que el que aparece en *scriptPubKey*, y comprueba también la firma con la clave pública.
- **Generación :** En la generación de nuevos bitcoins como resultado del proceso de minado de un nuevo bloque se crea un tipo de transacción especial que consta de una sola entrada y esta entrada tiene un parámetro *coinbase* en lugar de un *scriptSig*. Los datos de *coinbase* pueden ser cualquier cosa, ya que no se usa. El programa cliente original de Bitcoin pone ahí el objetivo actual en formato compacto y el número de precisión arbitraria.

Capítulo 2

Anonimato en el sistema Bitcoin

Como se ha mencionado en el capítulo anterior, existe una amplia gama de opiniones acerca del anonimato en la red Bitcoin, no obstante este tema ha sido ampliamente estudiado y se han establecido herramientas y técnicas que permiten analizar la red e identificar usuarios, el objeto de este capítulo es exponer de manera parcial los resultados del trabajo de Ried y Harrigan, además de citar la metodología utilizada por Brugere para construir una tabla compacta de transacciones.

Para este trabajo es de particular interés considerar tres características del sistema Bitcoin. Primero, el historial de transacciones Bitcoin está disponible de manera pública. Esto es necesario en orden de validar las transacciones y prevenir el doble gasto de la moneda en ausencia de una autoridad central. La única manera de confirmar una transacción es estar al tanto de todas las transacciones previas. La segunda característica que nos interesa es que una transacción puede tener múltiples entradas y múltiples salidas. Una entrada de una transacción es una salida de una transacción previa o es la suma de Bitcoins o comisiones por transacciones recién generados. Una transacción frecuentemente tiene una sola entrada proveniente de una transacción previa mas grande, o múltiples entradas provenientes de transacciones previas mas pequeñas. Además, una transacción frecuentemente tiene dos salidas: una enviando el pago y otra retornando el cambio. La tercera característica se refiere a que pagadores y beneficiarios ción son identificados mediante claves públicas. De hecho, es considerada una buena práctica que un beneficiario genere un par clave pública-privada nuevo para cada transacción. Adicionalmente, un usuario puede tomar medidas para mejorar la protección a su identidad: puede evitar la revelación de cualquier información de identificación relacionada con las claves públicas; puede enviar fracciones de sus bitcoins a él mismo usando nuevas claves públicas, entre otras. Sin embargo estas medidas no son universalmente aplicadas.

Las tres caracaterísticas anteriormente expuestas constituyen el fundamento para la construcción de dos estructuras de red: la red de transacciones y la red de usuarios. La red de transacciones representa el flujo de Bitcoin entre transacciones en el tiempo. Análogamente la red de usuarios representa el flujo de Bitcoin entre usuarios en el tiempo.

2.1. La Red de Transacciones

Para la construcción de estas estructuras a partir de la cadena de bloques, Ried y Harrigan se valen del concepto matemático de grafo:

Definición 2.1.1 *Un grafo es un par ordenado (V, E), donde V es un conjunto no vacío de vértices y E un conjunto de aristas. Una arista es un par no ordenado de vértices.*

Cada vértice del grafo representa una transacción y cada arista direccionada entre un origen y destino representa la salida de una trasacción correspondiente al origen y que es una entrada de la transacción correspondiente al destino. A cada arista dirigida se le asocia un par de escalares que corresponden al valor en Bitcoins y una marca de tiempo. Es una tarea bastante sencilla construir dicho grafo a partir del historial de transacciones de la cadena de bloques.

En la Figura 2.1 se muestra un ejemplo de un subconjunto de la red de transacciones. t_1 es una transacción con una entrada y dos salidas. La misma fue añadida a la cadena de bloques el 1 de Mayo de 2011. Una de sus salidas asignó 1.2 BTC a un usuario identificado por la clave pública pk_1 (las claves públicas no se muestran en la figura). Similarmente t_2 es una transacción con dos entradas y dos salidas. Esta fue aceptada el 5 de Mayo de 2011. Una de sus salidas envía 0.12 BTC a un usuario identificado por una clave pública distinta pk_2. t_3 es una transacción con dos entradas y una salida. Esta fue aceptada el 5 de Mayo de 2011. Ambas entradas de dicha transacción estan conectadas con las salidas antes mencionadas de las transacciones t_1 y t_2.

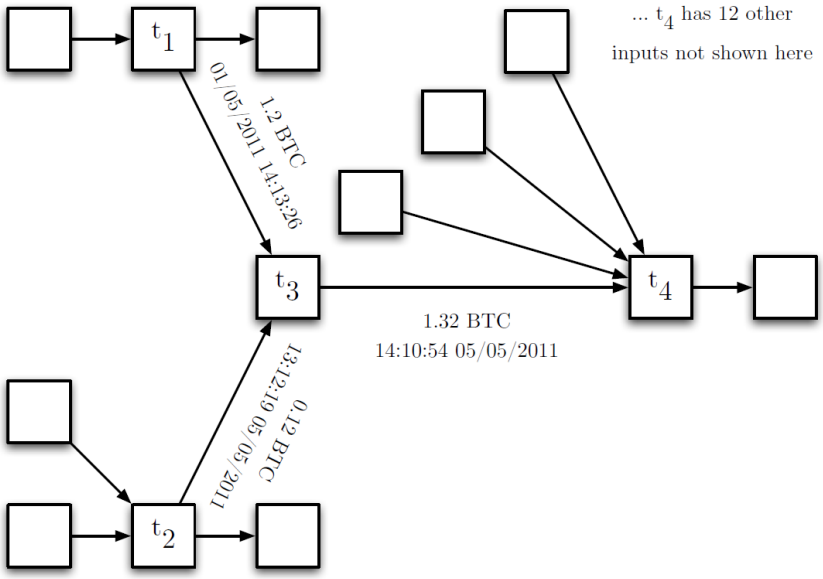

Figura 2.1: Ejemplo de un subconjunto de la red. Cada vértice cuadrado representa una transacción y cada arista dirigida representa el flujo de bitcoins desde la salida de una transacción hasta la entrada de otra.

2.2. La Red de Usuarios

Cada vértice representa un usuario y cada arista dirigida entre origen y destino representa un par entrada-salida de una transacción única, donde la clave pública de la entrada pertenece al usuario correspondiente al origen y la clave pública de la salida pertenece al usuario correspondiente al destino. Cada arista dirigida incluye un par de escalares que representan el valor en Bitcoin y la marca de tiempo de la transacción.
Es necesario realizar un paso de preprocesamiento antes de construir el grafo a partir del historial de transacciones de la cadena de bloques. Supongamos que $\mathcal{U}$ está en principio incompleto en el sentido en que cada vértice representa un única clave pública en lugar de un usuario. Para perfeccionar el grafo debemos contraer cada subconjunto de vértices correspondientes a aquellas claves públicas que pertenezcan a un mismo ususario. El problema radica en que las claves públicas son un mecanismo de la red bitcoin para asegurar el anonimato, es decir, en la cadena de bloques se almacena un registro donde puede verse que alguien identificado por una clave pública está enviando bitcoins a alguien más identificado por una clave pública también, pero sin información que pueda relacionar la transacción a nadie. De hecho se recomienda a los usuarios crear un nuevo par clave privada-pública para cada transacción. Por lo tanto, es imposible perfeccionar el grafo

usando únicamente el registro de transacciones. Vale la pena citar a Nakamoto:

> "Como un cortafuegos adicional, un par nuevo de claves debe ser utilizado para cada transacción de modo que puedan ser asociadas a un dueño en común. Algún tipo de asociación es inevitable con transacciones de múltiples entradas, las cuales pueden revelar que sus entradas fueron apropiadas por el mismo dueño. El riesgo está en que si el dueño de una clave es revelado, el enlazado podría revelar otras transacciones que pertenecieron al mismo dueño"

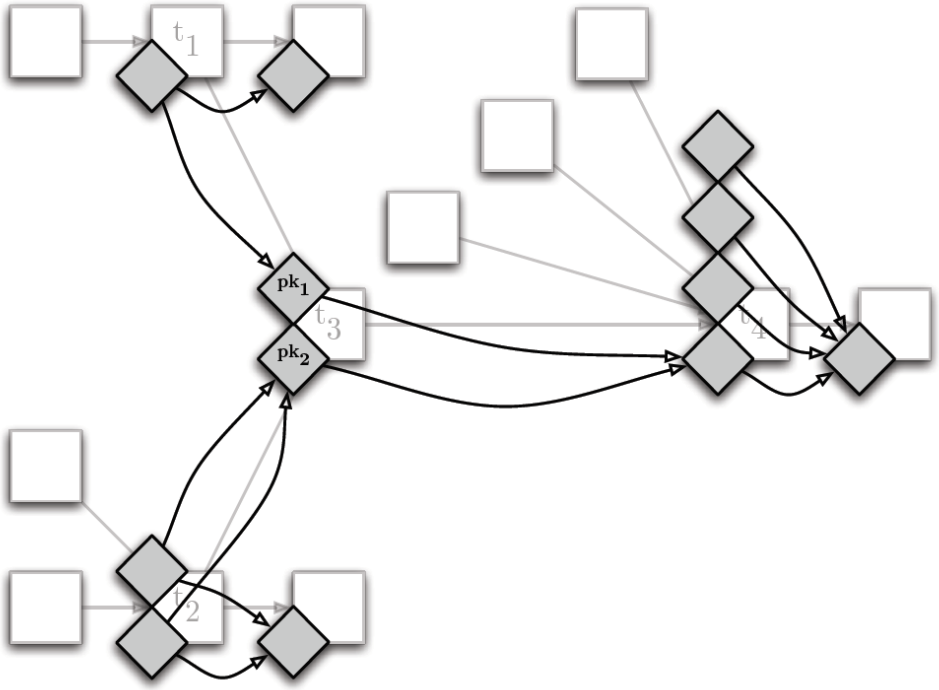

Figura 2.2: Ejemplo de un subconjunto de la red. Cada vértice diamante representa una clave pública, y cada arista dirigida el flujo de bitcoins desde una clave pública hasta otra.

La Figura 2.2 muestra un ejemplo de un subconjunto de la red de transacciones, superpuesta en el ejemplo anterior. Las salidas de t_1 y t_2 que fueron eventualmente procesadas como entradas de t_3, fueron enviadas a un usuario cuya clave pública era pk_1 y un usuario cuya clave pública era pk_2 respectivamente.

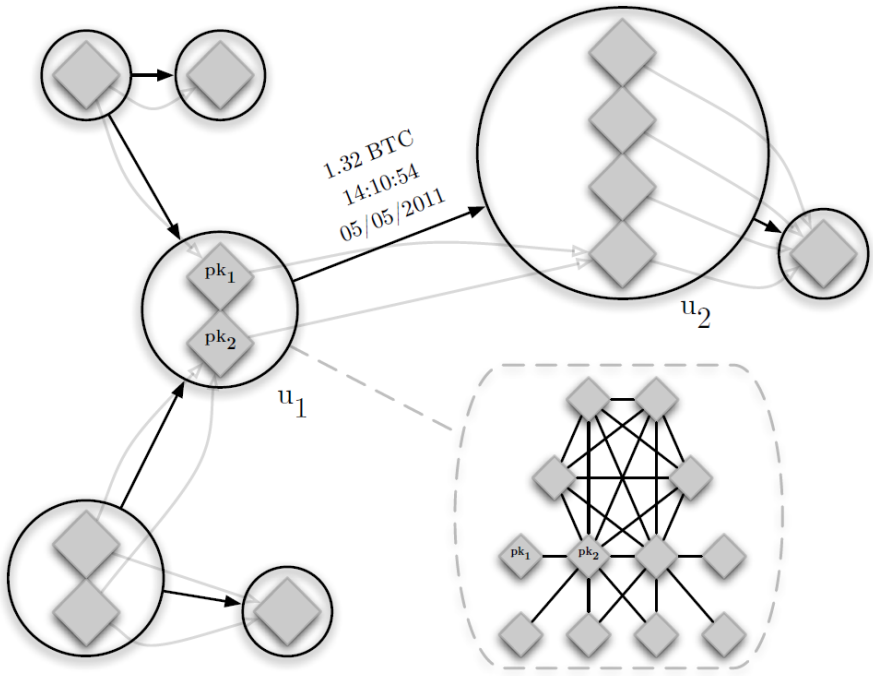

Figura 2.3: Ejemplo de un subconjunto de la red. Cada vértice circular representa un usuario y cada arista dirigida representa el flujo de bitcoins entre usuarios.

Capítulo 3

Aprendizaje no supervisado y Clusterización

3.1. Fases de la minería de datos

La minería de datos puede ser definida como un campo dentro de la ciencia cuyo interés se centra en coleccionar, limpiar, procesar, analizar y generar conocimiento valioso a partir de los datos.
En la actualidad, todos los sistemas automatizados generan de manera virtual algún tipo de datos, bien sea para diagnosticos o con propósitos analíticos. La mayoría de los dispositivos usados por las personas, gobiernos, organizaciones y empresas generan una vasta cantidad de datos. La consecuencia natural de este hecho es buscar maneras de revelar el conocimiento útil y concreto a partir de las diversas fuentes de datos disponibles. He aquí la importante tarea de la minería de datos. Los datos 'crudos' pueden ser impredecibles, no estructurados o inclusive estar presentados en un formato no apropiado de manera inmediata para el procesamiento automático.
Para lidiar con este problema, la minería de datos usa una secuencia lineal de procesamiento, donde se obtienen los datos, se limpian, y se transforman en un formato estándar. Los datos pueden estar almacenados en un sistema de base de datos comercial y ser finalmente procesada para obtener conocimiento mediante el uso de métodos analíticos. Las tareas principales de la minería de datos pueden resumirse en : *Agrupación, Reglas de Ascociación, Clasificación y detección de anomalías.*
El proceso de la minería de datos puede ser descrito como una secuencia lineal consistente de varias fases tales

como limpieza de los datos, extracción de características y diseño de los algoritmos. Se identifican tres fases principales:

- **Recolección** : La recolección de los datos puede requerir el uso de hardware especializado tal como sensores, labores manuales como encuestas, o herramientas de software tales como programas de web scrapping. Leugo de la recolección, por lo general los datos son almacenados en una base de datos o data warehouse.
- **Extracción de Características y Limpieza** : Cuando los datos son recolectados usualmente no están en un formato adecuado para el análisis. Un claro ejemplo son los archivos logs o registros de los servidores. Para adecuar los datos al proceso de análisis, es esencial aplicar transformaciones a los mismos para obtener un formato amigable para los algoritmos, tales como arreglos multidimensionales, series de tiempo, o formato semiestructurado. El arreglo multidimensional es usualmente el mas frecuente, en donde las filas representan observaciones y las columnas variables o métricas de los datos. Es crucial extraer características relevantes para el proceso de minería. La fase de extracción de características es usualmente llevada a cabo en paralelo con la fase de limpieza, fase en la cual los valores faltantes o erróneos son estimados o corregidos. El resultado final de este procedimiento es un conjunto de datos estrucuturado, que en muchas ocasiones es llamado vista minable.
- **Procesamiento analitico y algoritmos :** La parte final del proceso se encarga de diseñar modelos analíticos a partir de los datos procesados. En muchos casos, puede no ser posible abordar el problema usando una de las cuatro principales treas de la minería de datos. No obstante estas cuatro tareas cubren la mayoría de los problemas, y en el caso de no ser así, el problema puede ser dividido en partes y cada una de estas partes puede ser tratada usando una de estas tareas.

3.2. Analisis de Clusters

Una gran cantidad de problemas y aplicaciones requieren la partición de conjuntos de datos en subconjuntos intuitivamente similares. La partición de grandes conjuntos de datos en grupos mas pequeños ayudan a sumarizar las características de los datos en una amplia variedad de aplicaciones de la minería de datos. Intuitivamente el proceso de clusterización puede definirse como sigue :

Dado un conjunto de puntos, se particiona el mismo en grupos de puntos similares.

Esta es una definición algo rudimentaria pues no nos dice mucho acerca de las diferentes maneras en las que el problema puede ser formulado, tales como el número de grupos, o el criterio de similitud. Sin embargo esta simple descripción sirve como base a un número de modelos que están especificamente diseñados para diferentes aplicaciones. Algunos ejemplos que pueden ser citados:

- Sumarización de datos : en el nivel mas amplio, el problema de clusterización puede ser considerado una forma de sumarización. Como la minería de datos se trata de resumir la información y obtener información precisa a partir de los datos, el proceso de clusterización es a menudo el primer paso de muchos algoritmos de la minería.
- Segmentación de clientes : A menudo se desea analizar patrones comunes de conducta dentro de grupos de clientes similares. Un ejemplo de la segmentación de clientes lo constituyen los sistemas de recomendación de las tiendas online como Amazon.
- Análisis de Redes Sociales : En el caso de datos provenientes de redes, los nodos que son estrechamente clusterizados juntos debido al enalzamiento y relaciones son a menudo grupos de amigos o comunidades.

Además, desde una perspectiva general, pueden identificarse dos tipos de clusterización:

- Clusterización Fuerte : aquí cada punto del conjunto de datos o pertenece a un grupo o no.
- Clusterización Suave : en este tipo cada punto del conjunto de datos pertenece a uno o mas grupos con una probabilidad asociada.

3.3. Algoritmos Representativos

Los algoritmos representativos son los mas simples de todos los algoritmos de clusterización debido a que se basan directamente en las nociones intuitivas de distancia (o similitud) para agrupar puntos del conjunto de datos. En los algoritmos representativos, los grupos son creados en un solo intento, y no existen relaciones jerárquicas entre los distintos grupos. Esto se hace tipicamente usando un conjunto de representantes de partición. Los representantes de partición pueden ser creados como una función de los datos (e.g media aritmética) o pueden ser seleccionados de los datos en cada grupo. La idea principal de estos métodos es que el descubrimiento de grupos de alta calidad en los datos es equivalente al descubrimiento de un conjunto de representantes de alta calidad. Una vez que los representantes han sido determinados, se puede usar una función

de distancia para asignar los puntos a su representante mas cercano. Normalmente se asume que el número de grupos, denotado k, es especificado por el usuario. Consideremos un conjunto de datos D que contiene n puntos, denotados por $X_1, \cdots, X_n$ en un espacio d-dimensional. El objetivo es determinar k representantes $Y_1, \cdots, Y_k$ que minimizen la siguiente función objetiva O :

$$O = \sum_{i=1}^{n} [\min_j Dist(X_i, Y_j)]. \tag{3.1}$$

En otras palabras, la suma de las distancias de los diferentes puntos a su representante mas cercano debe ser minimizada.

Una observación acerca de la formulación de la ecuación (3.1) es que los representantes Y_k y la asignación óptima de los puntos a los representantes son desconocidas en principio, pero dependen la una de la otra de manera recurrente. Por ejemplo si los representantes óptimos son conocidos, entonces la asignación óptima es fácil de determinar, y viceversa. Tal problema de optimización es resuelto usando un enfoque iterativo, donde unos candidatos a representantes y asignaciones son usados para mejorarse el uno al otro. Por lo tanto el enfoque genérico comienza inicializando los k representantes S con el uso de una heurística bastante sencilla (tal como muestreo aleatorio a partir de los datos originales), y luego se refinan los representantes y las asignaciones de grupos, iterativamente, como sigue:

- Asignar cada punto a su representante mas cercano en S usando la función de distancia $Dist(\cdot, \cdot)$, y los grupos correspondientes $C_1, \cdots, C_k$.
- Determinar el representante óptimo Y_j para cada grupo C_j que minimiza su función objetiva local $\sum_{X_i \in C_j} [Dist(X_i, Y_j)]$.

3.4. ALgoritmo K-medias

En el algoritmo k-medias, la suma de los cuadrados de las distancia Euclídea entre un punto y su representante mas cercano es usada para cuantificar la función objetiva de clusterización. Es decir

$$Dist(X_i, Y_j) = ||X_i - Y_j||_2^2. \tag{3.2}$$

Donde $||\cdot||_p$, representa la norma L_p. La expresión para la distancia puede ser vista como el error cuadrático aproximando un punto a su representante mas cercano. Así, la función objetivo minimiza la suma de los errores cuadrados sobre los diferentes puntos. En ese caso se cumple que *el representante óptimo Y_j en cada paso iterativo de optimización es igual a la media de los puntos en el grupo C_j*

Algoritmo 1 k-medias

Entrada: Conjunto de Datos $\mathcal{D}$, número de representantes k.

Salida: Familia de subconjuntos de $\mathcal{D}$, $\{\mathcal{C}_1, \cdots, \mathcal{C}_k\}$.

1: Inicializar el conjunto de representantes $\mathcal{S}$

2: **Repetir**

Crear grupos $(\mathcal{C}_1, \cdots, \mathcal{C}_k)$ mediante la asignación de cada punto en $\mathcal{D}$ a su representante mas cercano usando la función de distancia euclídea.Recreaer el conjunto $\mathcal{S}$ determinando un representante Y_j usando la media aritmética de cada grupo $\mathcal{C}_k$

3: **Hasta que** Convergencia.

4: **Devolver** $(\mathcal{C}_1, \cdots, \mathcal{C}_k)$

3.5. Modelos Mixtos Gaussianos y Maximizacion de la Esperanza

Si bien es cierto que la distribución Gaussiana posee algunas propiedades analíticas importantes, padece de limitaciones significativas a la hora de modelar conjuntos de datos reales. En la práctica se ha comprobado que algunas veces una sola distribución Gaussiana no es capaz de capturar la estructura de un conjunto de datos, mientras que la superposición lineal de varias distribuciones Gaussianas ofrece una mejor caracterización de los datos.

Tales superposiciones, formadas al tomar conbinaciones lineales de distribuciones mas simples como la Gaussiana, pueden ser formulados como modelos probabilísticos llamados distribuciones mixtas (Mclachlan and Bashford, 1988; Maclachlan and Peel, 2000).

En la Figura 3.1 se puede observar que la combinación lineal de Gaussianas puede dar resultado a una densidad bastante compleja. Usando un número suficiente de Gaussianas, y ajustando sus medias y covarianzas tanto como los coeficientes en su combinación lineal, casi cualquier función de densidad continua puede ser aproximada con precisión arbitraria.

Consideremos una superposición de K densidades Gaussianas de la forma:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k), \tag{3.3}$$

que es llamada una mezcla de Gaussianas. Cada función de densidad Gaussiana es llamada una componente de la mezcla y tiene su propia media μ_k y covarianza Σ_k.

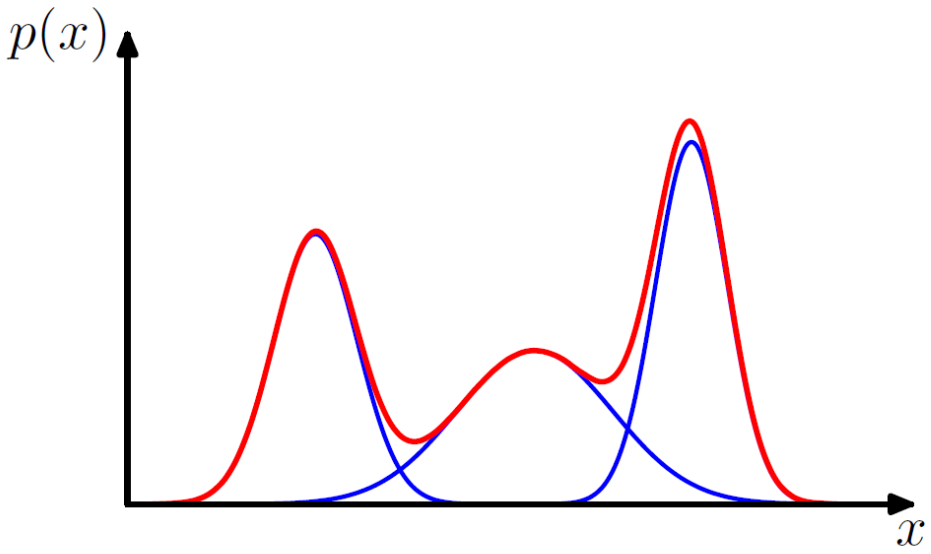

Figura 3.1: Ejemplo de una distribución mixta Gaussiana en una dimensión mostrando tres Gaussianas en azul y su suma en rojo.

3.5.1. Algoritmo EM

El principio básico de un modelo mixto Gaussiano se basa en asumir que los datos son generados a partir de una mezcla de K distribuciones $\mathcal{N}_1, \cdots, \mathcal{N}_K$. Cada una de estas distribuciones representa un cluster. Cada punto X_i, donde $1 \leq i \leq n$, es generado por un modelo mixto como sigue:

1. Seleccionar una componente con probabilidad inicial $\pi_i = P(\mathcal{N}_i)$, donde $1 \leq i \leq k$. Asumir que es seleccionada la r-ésima componente.

2. Generar un punto a partir de $\mathcal{N}_r$.

Denotemos este modelo generador por $\mathcal{M}$. Las diferentes probabilidades π_k y los parametros de la distribución son hasta ahora desconocidos. Para estimar dicho parámetros se usa el algoritmo *expectation-maximization* (EM). Los parámetros de las distintas componentes pueden ser usados para describir cada grupo. Por ejemplo, la estimación de de la media μ_k para cada $\mathcal{N}_k$ es análogo a determinar la media de cada grupo en el algoritmo k-medias. Luego de que los parametros de las componentes han sido estimados, la probabilidad de generación (o asignación) de cada punto con respecto a las componentes (grupos) puede ser determinada.

Asumamos que la función de densidad de probabilidad de cada componente es denotada por $f^i(\cdot)$. La probabilidad de que un punto X_j sea generado por el modelo viene dada por la suma ponderada de las funciones de densidad sobre las distintas componentes, donde la función de peso es la probabilidad inicial $\pi_i = P(\mathcal{N}_i)$ de las componentes:

$$f^{point}(X_j|\mathcal{M}) = \sum_{i=1}^{k} \pi_k(X_j). \tag{3.4}$$

Entonces, para un conjunto de datos $\mathcal{D}$ que contiene n puntos, denotados por $X_1, \cdots, X_n$, la función de densidad generada por el modelo $\mathcal{M}$ es el producto de todas las densidades específicas :

$$f^{data}(\mathcal{D}|\mathcal{M}) = \prod_{j=1}^{n} f^{point}(X_j|\mathcal{M}). \tag{3.5}$$

La función de verosimilitud $\mathcal{L}(\mathcal{D}|\mathcal{M})$ del conjunto de datos $\mathcal{D}$ con respecto al modelo $\mathcal{M}$ es el logaritmo de la expresión anterior, se toma el logaritmo por razones computacionales.

$$\mathcal{L}(\mathcal{D}|\mathcal{M}) = \log(\prod_{j=1}^{n} f^{point}(X_j|\mathcal{M})) = \sum_{j=1}^{n} \log(\sum_{i=1}^{k} \pi_i f^i(X_j)). \tag{3.6}$$

Esta función debe ser maximizada para determinar los parámetros del modelo. Una observación relevante es que si las probabilidades de los puntos de ser genreados en diferentes grupos son conocidas, entonces se vuelve relativamente sencillo determinar los parámetros óptimos separadamente para cada componente.
Sea Θ un vector, consistente de todos los parámetros de las componentes. Entonces el algoritmo EM comienza con un conjunto inicial de valores de Θ y procede como sigue:

1. (Paso E) dado el valor actual de parámetros en Θ, estimar la probabilidad posterior $P(\mathcal{N}_i|X_j, \Theta)$ de la componente $\mathcal{N}_i$ habiendo sido seleccionada en el proceso de generación, dado que se ha observado el punto X_j. La cantidad $P(\mathcal{N}_i|X_j, \Theta)$ es también la probabilidad suave de asignación a un grupo que estamos tratando de estimar. Este paso es ejecutado para cada punto X_j y componente $\mathcal{N}_i$.

2. (Paso M) Dadas las probabilidades actuales de asignación de los puntos a los grupos, usar la estimación por máxima verosimilitud para calcular todos los parámetros en Θ que maximizen la función en la ecuación (3.6) basado en las asignaciones actuales.

Los dos pasos son ejecutados repetidamente en aras de mejorar el criterio de máxima verosimilitud. Decimos que el algoritmo converje cuando la función objetiva no mejora significativamente en cierto número de iteraciones.

Capítulo 4

Resultados y Conclusiones

Hoy en día existen un sin fin de herramientas de software que hacen sencilla la implementación de análisis estadísticos sobre grandes volumenes de datos. Según encuestas realizadas por Kaggle, una plataforma de competencias, búsqueda de empleo y noticias acerca de la ciencia de datos, las herramientas mas usadas son los lenguajes R y Python. En este trabajo usaremos Ptyhon y algunas librerias ya existentes para la obtención de los datos, y los análisis, y R para el preprocesamiento y limpieza de los mismos.

4.1. R

R es un entorno y lenguaje de programación con un enfoque al análisis estadístico. R es una implementación de software libre del lenguaje S pero con soporte de alcance estático. Se trata de uno de los lenguajes más utilizados en investigación por la comunidad estadística, siendo además muy popular en el campo de la minería de datos, la investigación biomédica, la bioinformática y las matemáticas financieras. A esto contribuye la posibilidad de cargar diferentes bibliotecas o paquetes con funcionalidades de cálculo y gráficas. R es parte del sistema GNU y se distribuye bajo la licencia GNU GPL. Está disponible para los sistemas operativos Windows, Macintosh, Unix y GNU/Linux.

4.2. Python

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación

a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma. Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, e incompatible en ciertas versiones anteriores.

4.3. Obtención de los Datos

4.3.1. Interfaz de Programación de Aplicaciones

Una interfaz de programación de aplicaciones, abreviada como API del inglés Application Programming Interface, es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
Una API representa la capacidad de comunicación entre componentes de software. Se trata del conjunto de llamadas a ciertas bibliotecas que ofrecen acceso a ciertos servicios desde los procesos y representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla. De esta forma, los programadores se benefician de las ventajas del API haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio. Las API asimismo son abstractas: el software que proporciona una cierta API generalmente es llamado la implementación de esa API.

Para acceder a la cadena de bloques de la red Bitcoin hicimos uso de la API de *blockchain.info*, a través de la biblioteca *blockchain* de Python. La línea de procesamiento consiste en los siguientes pasos :

1. Mediante la función get_block_height se descargó toda la información de doce (12) bloques selccionados aleatoriamente. Estos bloques son almacenados en una lista.

2. Se crea una función capaz de iterar sobre cada bloque y obtener las transacciones almacenadas en cada bloque. Las variables asociadas a cada transacción que son de interés para el trabajo son :

   - **Double Spend** : Dato del tipo lógico, que indica si los bitcoins asociados a dicha transacción han sido gastados mas de una vez.
   - **Block Height** : Dato del tipo integer, que indica el bloque al que pertenece la transacción.

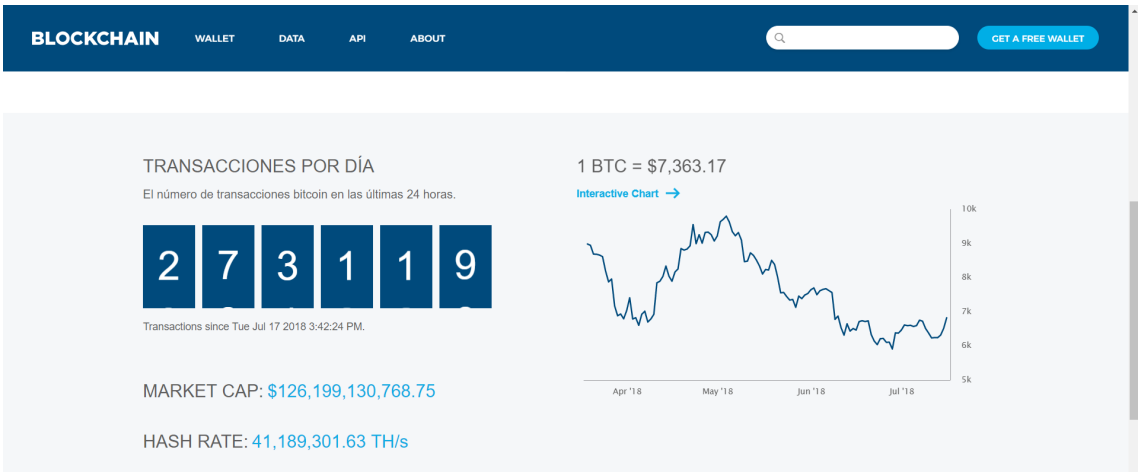

Figura 4.1: Visualización de la interfaz web de la API de blockchain.info .

- **Time**: Dato del tipo integer, que indica la fecha exacta en la que fue realizada la transacción en formato unix.
- **Relayed By** : Dato del tipo string, indica la dirección IP desde la cual fue realizada la transacción.
- **Hash** : Dato del tipo string, identificador único de la transacción.
- **Size** : Dato del tipo integer, indica el espacio físico que ocupa la transacción.
- **Inputs** : Dato del tipo array, contiene un arreglo con los datos de la entrada o entradas que realizan la transacción.
- **Outputs** : Dato del tipo array, contiene un arreglo con los datos de la salida o salidas a donde llega la transacción.

3. Se crea una segunda función, esta vez capaz de iterar sobre los arreglos de origen y destinos de cada transacción. Las variables asociadas a cada origen y destino de interés para el trabajo son :

   - **Input Value** : Valor del tipo integer, indica el monto a ser enviado expresado en satoshis.
   - **Input Address** : Dato del tipo string, representa la clave pública asociada al usuario que está enviando bitcoin.
   - **Output Value** : Dato del tipo integer, indica el monto recibido en la dirección asociada.
   - **Output Address** : Dato del tipo string, representa la clave pública donde se recibe bitcoin.

time	hash	relayed_by	size	input value	input address	output address	output value
1485382280	2f06fd88f7c867c43734ef2a93671dd1cf9db14df77ca07ead454a20686ffaca	127.0.0.1	225	10742278	128nRaMd4PSNHhAHGCyJExWqfBJ9mXETZH	1GMPKRfaXaSXv5JHR2dPx5ehXUjfSMsksy	142510
1485382280	2f06fd88f7c867c43734ef2a93671dd1cf9db14df77ca07ead454a20686ffaca	127.0.0.1	225	10742278	128nRaMd4PSNHhAHGCyJExWqfBJ9mXETZH	1HgJ4d8wxCosR7oyBNSQeaptkwDcwog1wh	10299768
1485382253	2c14fbd463a6a1f37ef2507b71f2bd2a5b78391602c4bd20c6b86a5537bedd95	149.202.206.118	225	134531	1PjrSVBRZuiWdddmccfCa1XvaunyXxEuvh	1J4m13hgTqNcxbBb5sNi8VKkdzabWVi4fx	14531
1485382253	2c14fbd463a6a1f37ef2507b71f2bd2a5b78391602c4bd20c6b86a5537bedd95	149.202.206.118	225	134531	1PjrSVBRZuiWdddmccfCa1XvaunyXxEuvh	1KphzHczLDJnJnpHFCeSsAJNJsbSU6wd6V	30000
1485382285	5d5126ef0db38de3051ac79d01b654719305fd32d7224066afb0d275992644a1	108.61.199.182	223	2255584	16v8twRPHMCLCHrEMqCJi9x9oJSEwQmHJi	3QL47GNx6gtyp91bg4KRjWs8rs2qxdXLrK	1080000

Figura 4.2: Ejemplo de la tabla de transacciones resultante del proceso de obtención de datos.

4.3.2. Preprocesamiento y Transformación

Luego de obtener una tabla como la mostrada en la Figura 4.2, usaremos la heurística propuesta por Ried y Harrigan para concatenar las claves públicas que pertenecen a un mismo usuario dentro de un mismo identificador único. Este proceso es llevado a cabo creando una función en R que itera sobre la tabla de transacciones y crea una nueva tabla creando una nueva variable : User ID. Para cada hash de transacción la función verifica si la clave pública asociada se repite en alguna otra transacción y además verifica cuantas entradas posee cada transacción asociada a esta clave pública, es decir verifica si cada transacción posee múltiples claves públicas. Luego la función asigna el mismo identificador de usuario a todas las transacciones que cumplan con el criterio. Aunque los datos no poseen valores faltantes, se realizaron un conjunto de modificaciones a los datos originales, podemos enumerarlos como sigue :

1. Usando una tabla de tasas de conversión Bitcoin/Dólar se convirtieron las variables que estaban expresadas en bitcoin a Dólar.

2. Se le asignó un formato de fecha estándar a partir del formato Unix a la variable Time.

3. Fueron calculadas métricas adicionales a partir de los datos que ya se tenían : monto total, máximo y

promedio enviado por una dirección.

4.4. Análisis Exploratorio

Para el momento en que fue realizado este análisis se habían realizado mas de *num* transacciones en ese día. Con respecto al número de bloques podemos decir que eran mas de quinientos treinta mil (530000).

Fecha	18 de Julio de 2018
Precio de Mercado en USD	7483.23
Hash Rate	4.30E+12
Número de BTC Minados	2087,5
Número de Transacciones	241218
Número de Bloques minados	167
Número Total de Bloques	532502
Volumen de Transacciones en USD	6.67E+09
Volumen de Transacciones en BTC	88969.34

Figura 4.3: Tabla resumen con las estadísticas generales del 18 de Julio de 2018

El conjunto de datos obtenido en la sección anterior contiene un total de 229452 observaciones, por motivos prácticos se propuso tomar una muestra aleatoria de este conjunto de datos de tamano 10000. Las 10000 transacciones en nuestro conjunto de datos están distribuidas en un total de 5481 usuarios.

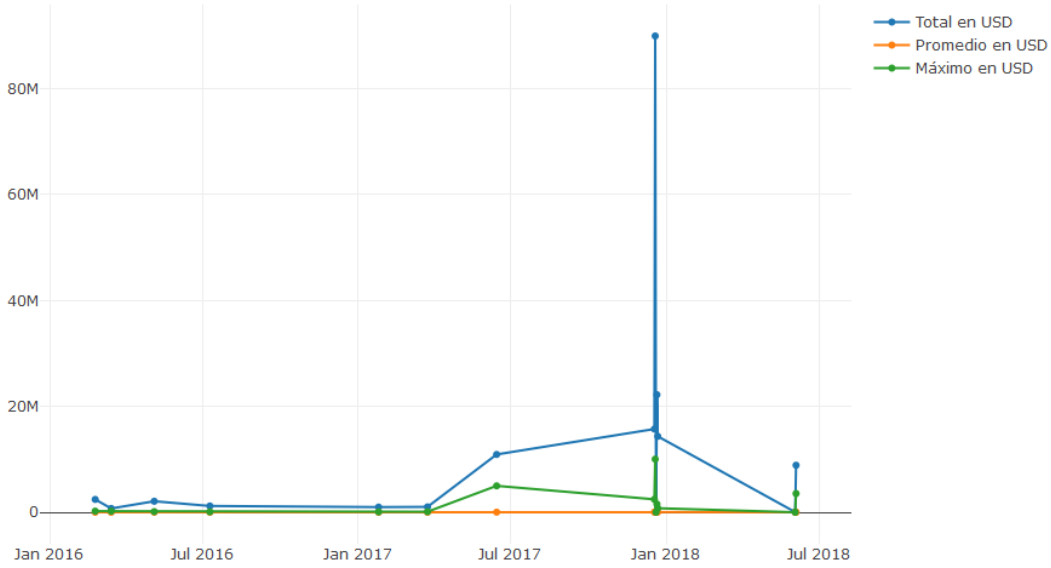

Figura 4.4: Gráfica del histórico de valores máximos, totales y promedio. Puede observarse que existe una correlación entre los valores máximos y los valores totales, esto explica en cierta forma el pico que se presenta entre Diciembre de 2017 y Enero de 2018.

4.5. Análisis de Clusters

4.5.1. K-medias

Para la implementación del algoritmo *k*-medias se usó la biblioteca Scikitlearn de Python. Una de las desventajas de usar este algoritmo radica en la no existencia de un método formal para la asignación del número de grupos que se forman, en este trabajo se usó el criterio de Inercia o distancias intra-cluster, que es una medida de la compacidad de cada grupo o lo dispersos que se distribuyen los puntos dentro de cada grupo.
En nuestro caso se elije $k = 5$ debido a que a partir de ese valor se puede observar que no hay cambios siginificativos en el valor de la Inercia. Luego se procede a ajustar el modelo a nuestro conjunto de transacciones. Los resultados de los centroides (valores medios de cada variable), y la cantidad de puntos en cada grupo pueden ser leídos en la Figura 4.7. Esto nos da una descripción de las características de cada grupo, por ejemplo, en el cluster etiquetado como 0 puede apreciarse que si un punto está en ese grupo entonces tendrá valores medios en cada una de sus variables cercanos a los siguientes valores:

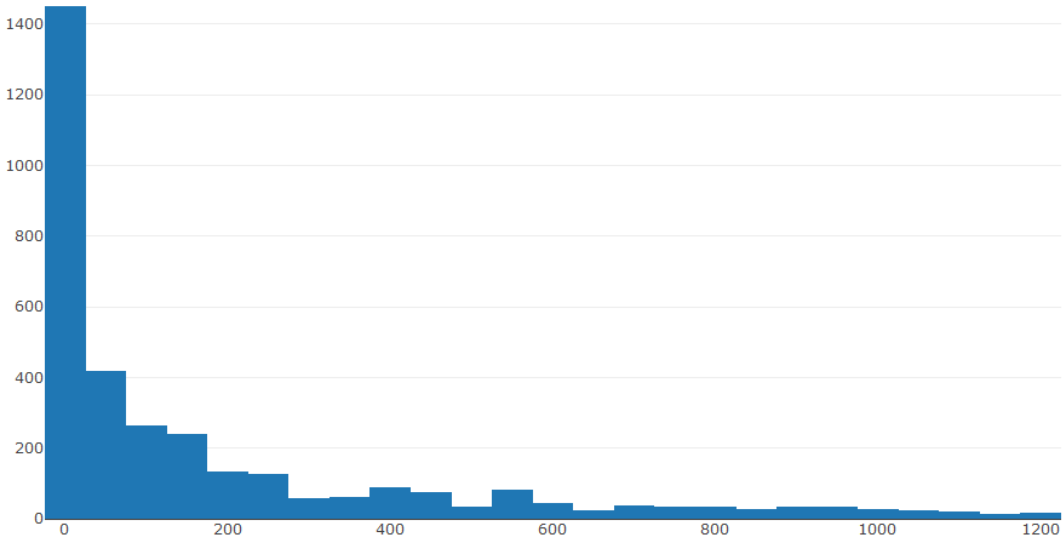

Figura 4.5: Histograma de frecuencias de la variable USD Amount, que representa el monto en Dólares americanos de cada transacción. Si bien es cierto que se observa que no es una distribución ideal, puede observarse que la mayoría de las transacciones poseen un monto associado menor a los 50 Dólares, lo cual describe como es la muestra con la que se está trabajando.

- Monto USD : 305,257158.
- Monto Máximo : 423,89776.
- Monto Total : 548,791269.
- Monto Promedio : 319,826574.
- Volumen : 1,033588.

4.5.2. Modelo Mixto Gaaussiano

Como fue descrito en el capítulo anterior, un modelo mixto Gaussiano es un modelo probabilístico que asume que los puntos del conjunto de datos son generados a partir de una mezcla finita de distribuciones Gaussianas con parámetros desconocidos.

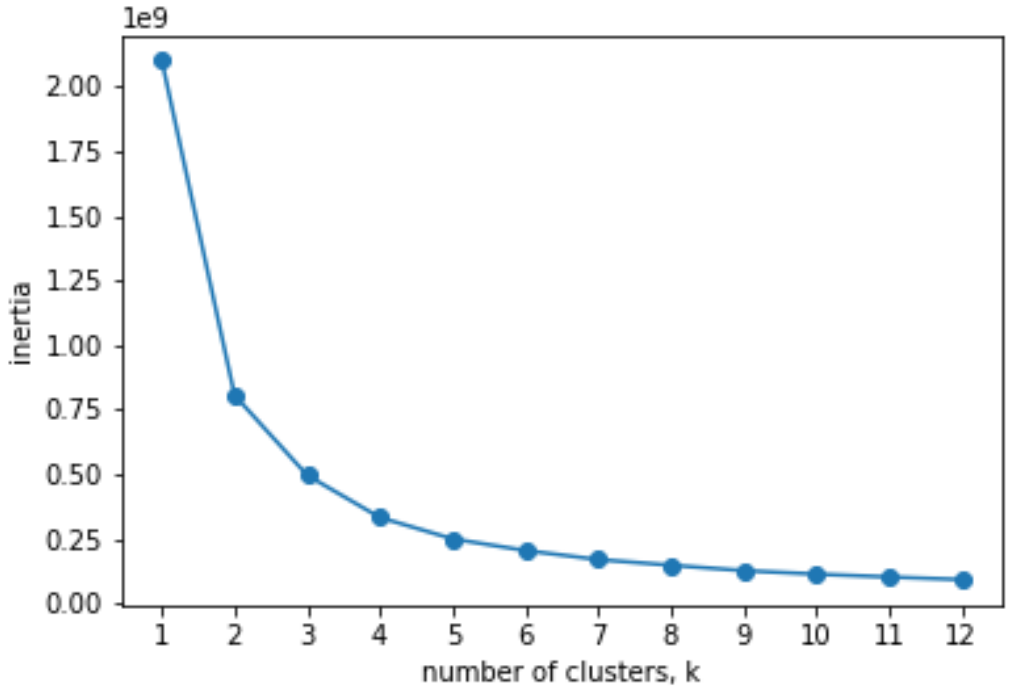

Figura 4.6: Gráfica de la inercia para distintos números de grupos variando en un rango de 1 a 7. Puede observarse que a partir de $k = 5$ la variación es casi nula.

	0	1	2	3	4
0	305.257158	423.897760	548.791269	319.826574	1.033588
1	531.049300	1127.045675	3998.379886	531.049300	2.800000
2	42.061947	46.729651	56.667371	42.336266	1.131047
3	851.756492	857.507604	929.635371	772.064086	1.015038
4	476.340213	933.250303	1705.378408	531.497265	1.220000

Figura 4.7: Tabla resumen con los valores para los centroides de cada grupo.

Para la elaboración de este modelo se usa el paquete *mixture* de la biblioteca Scikitlearn de Python, en particular, la función *GaussianMixture*, que realiza una implementación del algoritmo EM para calcular los parámetros a través de el método de máxima verosimilitud. Analogamente al caso del algoritmo *k*-medias no existe un método formal para determinar el número de componentes, razón por la cual se usa el criterio de información Bayesiana (BIC) para determinar los parámetros que hacen óptimo al modelo . El BIC se define formalmente como :

$$BIC = \ln(n)k - 2\ln(\bar{L}), \tag{4.1}$$

donde x representa a los datos, n el número de observaciones, k el número de parámetros a ser estimados, $\bar{L}$ el valor máximo de la función de verosimilitud. En la Figura 4.8 pueden observarse los resultados de la aplicación de dicho criterio a nuestro conjunto de datos. Es importante destacar que además de estimar el número de componentes, a través de este criterio es posible estimar otro parámetro que es el tipo de covarianza a usar. El tipo de covarianza está relacionado con la forma geométrica que pueden tener los grupos, que en el caso de el algortimo *k*-medias era esférico por defecto.

Luego se selecciona el modelo con parámetros número de componentes 6 y tipo de covarianza $full$. Una matriz con las probabilidades asociadas a cada punto de pertenecer a cada grupo es mostrada en la Figura 4.9.

En el caso del modelo mixto Gaussiano, y como se expuso en el capítulo anterior, la caracterización de los grupos obtenidos viene dada en términos de las componentes, es decir en términos de las medias y covarinzas asociadas a cada una de ellas.

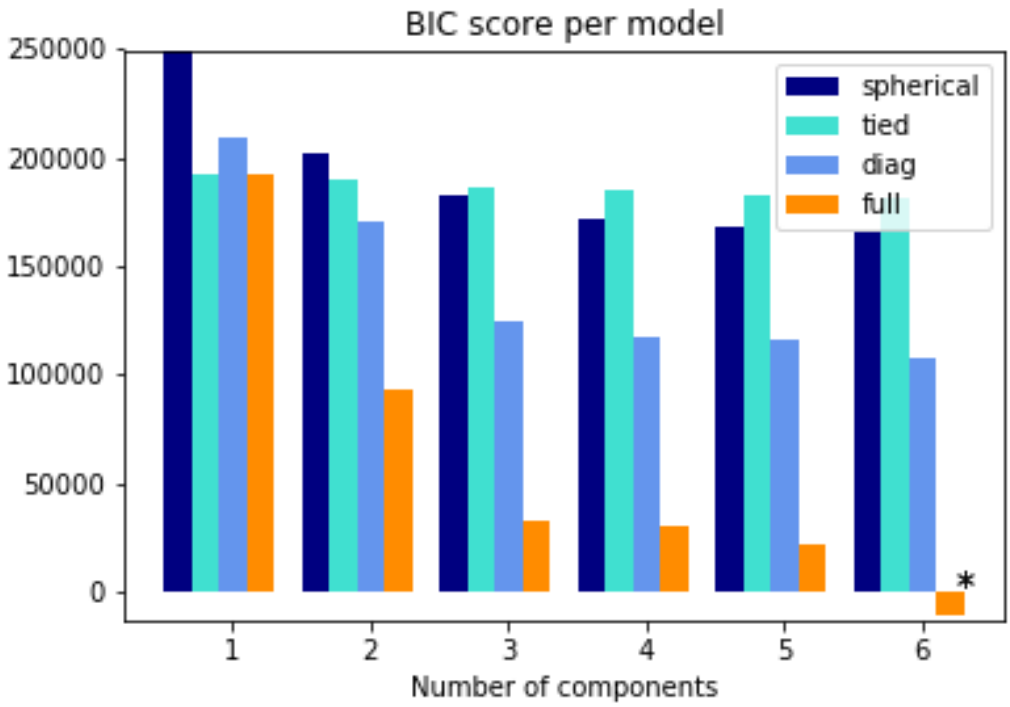

Figura 4.8: Gráfica de los puntajes BIC para modelos con número de componentes variando desde 1 hasta 6. La leyenda indica que cada color representa un tipo de matriz de covarianza distinta. Se selecciona el modelo con 6 componentes porque es el que muestra el menor puntaje con el tipo de covarianza $full$.

4.6. Conclusiones

El uso de los modelos mixtos ayudan a flexibilizar los resultados obtenidos con el algoritmo k-medias ya que ofrecen una caracterización mas completa de los grupos encontrados. Las características intrínsecas de los datos influyen en el muestreo de los mismos, razón por la cual la distribución de nuestro conjunto de datos no fue la esperada, esto fue un punto decisivo en la construcción de los modelos. Si bien es cierto que los resultados no permiten esbozar patrones muy diferenciados para cada grupo, basándonos en el trabajo de investigación y los fundamentos existentes, podemos decir con toda seguridad que las técnicas expuestas sirven de marco referencial para el estudio y descubrimiento de patrones de conducta de los usuarios de la red Bitcoin, incluso de cualquier cadena de bloques bien sea pública como Ethereum o privada como Hyperledger de IBM. Los resultados aquí expuestos pueden ser usados para la detección de anomalías bajo la creación de

	0	1	2	3	4	5
0	0.0	1.000	0.0	0.000	0.0	0.0
1	0.0	1.000	0.0	0.000	0.0	0.0
2	0.0	1.000	0.0	0.000	0.0	0.0
3	0.0	1.000	0.0	0.000	0.0	0.0
4	0.0	1.000	0.0	0.000	0.0	0.0
5	0.0	1.000	0.0	0.000	0.0	0.0
6	0.0	1.000	0.0	0.000	0.0	0.0
7	0.0	0.545	0.0	0.455	0.0	0.0
8	0.0	1.000	0.0	0.000	0.0	0.0
9	0.0	1.000	0.0	0.000	0.0	0.0

Figura 4.9: Tabla donde se mustran las probabilidades asociadas a cada punto con respecto a cada grupo. Esta tabla es una herramienta útil a la hora de reforzar y darle flexibilidad al modelo construido con el algoritmo *k*-medias.

	0	1	2	3	4
0	212.438959	212.438960	212.439006	212.438955	1.000000
1	225.740293	523.831165	1028.996351	280.351046	1.753809
2	391.951351	391.951352	638.941599	252.240119	1.000000
3	120.494036	412.559027	646.945659	218.409697	1.000000
4	0.077529	0.310957	1.360894	0.101835	1.000000
5	460.588325	1072.890483	3606.139261	463.788966	3.272385

Figura 4.10: Tabla donde se muestran las medias asociadas a cada variable en cada componente conseguida con la implementación del modelo mixto. Podemos observar que no existe una diferenciación clara entre cada grupo excepto en el caso de las componentes 5 y 6. Esto puede ser el resultado de la distribución original de los datos al ser una muestra de todo el conjunto de datos original.

un sistema antifraude similar al que ya existe en las entidades bancaria para el uso de tarjetas de crédito.
Concluimos que el proceso de conexión con la cadena de bloques, la recolección de los datos y la preparación y transformación debe prestarse mucha atención, pues de eso dependerán en gran medida la calidad de los

resultados. Además la importancia de considerar varios algoritmos de clusterización y de diferente especie permite reforzar y darle robustes a los análisis.
Finalmente consideramos que la implementación de estás técnicas puede llegar a ser mucho más eficiente si se lleva a cabo en un å'mbito de Big Data, usando herramientas que permitan crear una línea de procesamiento en tiempo real de tales cantidades de datos.

4.6.1. Trabajos Futuros

Partiendo del hecho de que cada diía se crean nuevas criptomonedas basadas en cadenas de bloques y la adopción de esta tecnología en muchos sectores, no sólo el financiero se propone lo siguiente como trabajos futuros para continuar con el desarrollo de este trabajo :

- Adopción de una herramienta como Apache Hadoop para la implementación de estos análisis en gran escala, es decir, tomando en cuenta todas las transacciones almacenadas en la cadena de bloques.
- Estudio de otros algoritmos de clusterización que puedan ser probados para medir su rendimiento.
- Replicar las técnicas usadas en este trabajo para la realizaci]'on de análisis en otras cadenas de bloques.

Bibliografía

[1] Satoshi Nakamoto (2008) *Bitcoin: A Peer-to-Peer Electronic Cash System*

[2] Reid, F., Harrigan, M. (2012). *An Analysis of Anonymity in the Bitcoin System*, Clique Research Cluster, Complex and Adaptive Systems Laboratory, University College Dublin, Ireland

[3] Pham, T. and Lee, S. (2017). *Anomaly Detection in Bitcoin Network Using Unsupervised Learning Methods*. arXiv:1611.03941v2 [cs.LG] 25 Feb 2017

[4] Harrigan, M. and Fretter, C. (2016).*The Unreasonable Effectiveness of Address Clustering*. arXiv:1605.06369v2 [cs.CR] 19 Sep 2016

[5] Fleder, M. , Kester, M. , Pillai, S. (2014) *Bitcoin Transaction Graph Analysis*. Arxiv, 2015.

[6] Aggarwal, C. (2015). *Data Mining, The Text Book*, Springer.

[7] Bishop ,C. *Pattern Recognition and Machine Learning*, Springer.

[8] Poikonen, S. *Unsupervised Learning of Bitcoin Transaction Data*.

[9] Arriojas, M. *Teoría de las Probabilidades*. Universidad Central de Venezuela.

[10] Hérnandez, J. *Estadística*. Universidad Central de Venezuela.