Trabajo Fin de Máster

Detección de anomalías en espacios de alta dimensión

Víctor Alonso Macías

Máster en Técnicas Estadísticas

Curso 2016-2017

Propuesta de Trabajo Fin de Máster

Título en galego: Detección de anomalías en espazos de alta dimensión
Título en español: Detección de anomalías en espacios de alta dimensión
English title: Anomaly detection in high-dimensional spaces
Modalidad: Modalidad B
Autor/a: Víctor Alonso Macías, Universidad de Santiago de Compostela
Director/a: Javier Roca Pardiñas, Universidad de Vigo
Tutor/a: Bruno Fernández Castro, Gradiant
Breve resumen del trabajo: En este trabajo se tratará de comprender la problemática de la alta dimensión en los métodos de detección de anomalías con el objetivo de realizar una comparativa entre diferentes algoritmos. Ésta se llevará a cabo investigando sus prestaciones, características, limitaciones y comportamiento sobre diferentes conjuntos de prueba, logrando de esta manera validar un método para su aplicación directa en el análisis de datos generados por máquinas.
Recomendaciones: Conocimientos del lenguaje estadístico de programación *R*.
Otras observaciones: Se realizarán estudios numéricos sobre conjuntos de datos simulados y una aplicación a datos reales.

Don Javier Roca Pardiñas, profesor de la Universidad de Vigo y don Bruno Fernández Castro, Responsable Técnico de *Machine Generated Data Analytics* ($MGDA$) | Área de Sistemas Inteligentes en Red ($INetS$) de Gradiant informan que el Trabajo Fin de Máster titulado

Detección de anomalías en espacios de alta dimensión

fue realizado bajo su dirección por don Víctor Alonso Macías para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En Vigo, a 10 de Julio de 2017.

El director:

Don Javier Roca Pardiñas

El tutor:

Don Bruno Fernández Castro

El autor:

Don Víctor Alonso Macías

Agradecimientos

En primer lugar, agradecer a Gradiant la oportunidad de hacer las prácticas para el trabajo de fin de máster junto a ellos y a todos mis compañeros por la ayuda y el buen trato recibido, especialmente a mi tutor Bruno Fernández Castro.
Por otra parte, me gustaría agradecer a Javier Roca Pardiñas su ayuda a la hora realizar el trabajo y porque sin él no habría sido posible mi experiencia en Gradiant.
También me gustaría dar las gracias a mi familia y amigos por su comprensión y el apoyo recibido.

Índice general

Resumen

Resumen en español

La detección de anomalías en espacios de alta dimensión es un problema que en los últimos años ha interesado a un gran número de empresas de diversos sectores. Por este motivo, se ha realizado en el presente trabajo un estudio de diferentes algoritmos que tratan de dar solución a la búsqueda de atípicos.

Para comenzar, se realizará una exposición de la metodología empleada en el proyecto, formada concretamente por cuatro algoritmos basados en diferentes aproximaciones para la resolución del problema: *Angle-Based Outlier Detection* que se basa en las variaciones de los ángulos entre puntos; *Subspace Outlier Degree* que realiza la búsqueda de atípicos una vez se ha encontrado un subespacio adecuado; *Local Outlier Factor* que compara la densidad de las zonas donde se encuentran los puntos; y *Projection-Indexed Nearest-Neighbour* que es un caso particular del anterior, en el cual se realiza previamente una reducción de dimensión. Seguidamente, utilizando tanto conjuntos de datos simulados como reales se procederá a realizar una comparativa entre los diferentes algoritmos con el objetivo de seleccionar el más adecuado. A continuación, se incluirán algunas técnicas para la reducción de los falsos positivos en los resultados anteriormente obtenidos. Una de ellas se basa en la reducción de la dimensión del conjunto de datos inicial, mientras que la otra es un análisis *a posteriori* de los datos que han sido marcados como atípicos por el algoritmo. Se terminará el proyecto con una exposición de las conclusiones obtenidas tras la aplicación de las diferentes técnicas empleadas.

English abstract

Anomaly detection in high-dimensional spaces is an issue concerning an increasing number of companies from different sectors lately. Due to this fact, in this work it has been conducted an analysis over different algorithms which attempt to solve the search of outliers.

Initially, there will be a description of the methodology employed, which is composed of four different algorithms aimed to solve the problem: the Angle-Based Outlier Detection, which is based on variations on the angles between points; the Subspace Outlier degree, which accounts for atypicals once appropriate subspaces have been found; the Local Outlier Factor, which compares the density of the areas where points are located; and the Projection-Indexed Nearest-Neighbour, which is a particular case of the former that implies a previous dimension reduction. Next, using simulated data sets as well as real ones, a comparison between algorithms will be conducted. Afterwards, two approaches to reduce the number of false positives will be included in the results obtained. One is based on the dimension reduction of the initial data set, whereas the other one performs a posteriori analysis of the atypical piece of data selected by the algorithm. The project will be finished with a summary of the main conclusions of the results.

Prefacio

Hoy en día, la amplia mayoría de empresas generan diariamente grandes cantidades de información que se almacenan en gigantes bases de datos de las cuales esperan poder sacar conclusiones útiles. Algunas tratan de buscar patrones ocultos para poder realizar predicciones sobre ellos, sin embargo, también suscitan gran interés las aplicaciones que logran encontrar eventos raros en campos como la salud, en la detección de fraudes en tarjetas bancarias o en la vigilancia ambiental. La búsqueda de estos casos poco comunes es especialmente complicada cuando nos encontramos con bases que contienen un número muy elevado de datos y variables, es decir, es un problema complicado en espacios de alta dimensión. Este fenómeno se conoce como *curse of dimensionality* (maldición de la dimensionalidad) [11] y surge al tratar de encontrar puntos diferentes al resto trabajando con un número elevado de características. El mencionado problema se debe principalmente a los siguientes factores:

- **Concentración de distancias.** A medida que aumentamos el número de dimensiones, la diferencia entre la mayor distancia y la menor se hace más pequeña, con lo cual se complica la tarea de encontrar puntos alejados del resto.
- **Atributos importantes frente a atributos de ruido.** Cuando el número de dimensiones aumenta, la proporción de ellas para las cuales un punto no es anómalo (ruido) es mayor, por lo que será más complicado encontrar las variables que realmente influyen en que un dato sea atípico.
- **Explosión combinatoria de posibles subespacios.** La cantidad de combinaciones posibles de seleccionar un subespacio donde sea más sencillo encontrar datos atípicos aumenta exponencialmente con el número de dimensiones. Además, a medida que van aumentando las combinaciones, es posible que cualquier punto sea anómalo en alguna de ellas, lo que nos lleva a la conocida expresión de que en alta dimensión 'todo punto es anómalo'.

En los últimos años se han llevado a cabo diferentes ideas para tratar de solventar los mencionados problemas utilizando algoritmos para la detección de atípicos basados en diversos enfoques. Algunos utilizan la reducción de dimensión y la búsqueda en subespacios como por ejemplo el *Subspace Outlier Degree* [8] o el *Anomaly Detection based on Principal Component Classifier* [9] que utiliza la técnica conocida como *Principal Component Analysis* (PCA); otros miden la variación de ángulos como el *Angle-Based Outlier Detection* [7]; mientras que los más clásicos utilizan distancias como el *k-Nearest Neighbors* [2] o densidades como el *Local Outlier Factor* [3] o el *Density-Based Spatial Clustering of Applications with Noise* ($DBSCAN$) [4].

En el primer capítulo de este trabajo se expondrá el problema así como los objetivos que se esperan conseguir gracias a su resolución. En el segundo se incluirá la metodología necesaria para llevar a cabo el objetivo propuesto y se comentarán los diferentes algoritmos que van a ser utilizados en las siguientes secciones. Los estudios numéricos realizados sobre datos simulados y reales se expondrán en el tercer capítulo junto con los resultados pertinentes. En el cuarto se mostrarán diferentes métodos cuyo objetivo es mejorar los resultados obtenidos en la sección anterior. El último capítulo recoge y expone las conclusiones extraídas en el presente trabajo.

Capítulo 1

Introducción

En este primer capítulo trataremos de contextualizar la problemática de la detección de anomalías en espacios de alta dimensión. Para ello comenzaremos por presentar a la empresa que ha planteado este proyecto, Gradiant. A continuación se seguirá con una exposición del problema junto con los motivos que han generado la creación de dicho proyecto. Posteriormente se introducirán los métodos de resolución necesarios para llevar a cabo este cometido. Por último, se definirán los objetivos del proyecto desde el punto de vista de la empresa.

1.1. Gradiant

El Centro Tecnolóxico de Telecomunicacións de Galicia (*Galician Research and Development Center in Advanced Telecomunications*) se conforma a partir de un patronato que agrupa a representantes del sector público y privado. Éste está formado por: Consellería de Economía e Industria, Axencia Galega de Innovación, Universidade de A Coruña, Santiago de Compostela y Vigo, Arteixo Telecom, Egatel, Indra, R, Telefónica, Grupo Televés, y la Asociación empresarial INEO.

Gradiant es una fundación privada, sin ánimo de lucro, establecida en diciembre de 2007 y nacida con el objetivo de alinear la $I + D + i$ universitaria con las demandas empresariales. Desempeña un papel fundamental en la generación y transferencia de conocimiento en tecnologías de la información y de las comunicaciones (TIC) hacia las empresas. El Centro tiene además un compromiso con la calidad que ha propiciado que obtenga la certificación de Gestión de Calidad UNE-EN ISO 9001 : 2008 y la de Sistemas de Gestión de Proyectos de $I + D + i$ UNE 166002. Ambas acreditan y garantizan la máxima calidad en la gestión de todos los procesos y proyectos que se llevan a cabo.

Algunas de sus cifras clave son: 100 empleados (93 ingenieros, 23 *PhD*), más de 100 clientes, más de 60 proyectos activos y la participación en 9 proyectos europeos. Para conseguir estos números cuenta con una sólida estructura organizativa representada en la Figura 1.1.

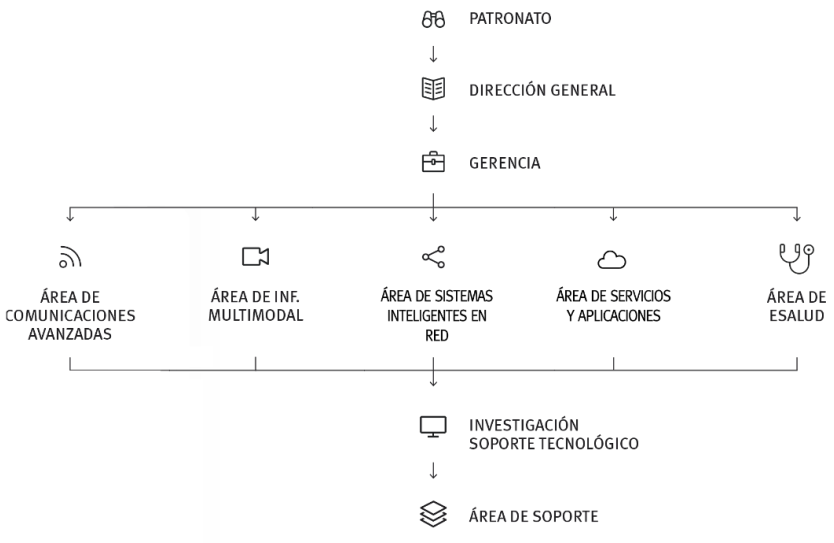


Figura 1.1: Estructura organizativa de Gradiant.

Las cinco áreas técnicas en las que se divide la capacidad $I + D + i$ de Gradiant trabajan de manera transversal, siendo el apoyo y trabajo entre las áreas uno de los valores añadidos del Centro. Las fortalezas de Gradiant, acreditadas en proyectos de $I + D + i$ a nivel regional, nacional e internacional se desarrollan sobre las siguientes líneas de investigación:

Área de Sistemas Inteligentes en Red

Línea Tecnológica	Sub-Línea Tecnológica
Data Analytics	Big Data
	Procesado natural del lenguaje
	Machine Learning
Sistemas integrados	Dispositivos Middleware
Internet of things	Protocolos (IPv6,CoAP,6LoWPAN)
	Discovery technologies
	Aplicaciones (SmartCities, SmartHome, Factories)
Redes	Análisis y caracterización
	Calidad de Experiencia (QoE)
	Redes inalámbricas (4G, 5G, LTE, Wifi, Zigbee)

Área de Información Multimodal

Línea Tecnológica	Sub-Línea Tecnológica
Biometría	Cara, firma manuscrita y habla
Análisis inteligente de vídeo	Seguridad, análisis de vídeo aéreo, protección ambiental mediante análisis de vídeo
	Procesado de datos 3D
Seguridad multimedia	Trazabilidad documental, watermarking y verificación de integridad
HCI	Computación afectiva
	Análisis de habla

Área de eSalud

Línea Tecnológica	Sub-Línea Tecnológica
Procesado de señal	Procesado de señales ECG
	Detección de caídas y de agitación en pacientes encamados
Bioinformática	Análisis de datos NGS
Privacidad y seguridad	Procesado de datos médicos manteniendo la privacidad
Dispositivos	Gateway para comunicación de dispositivos médicos
Sistemas biométricos	Aplicaciones médicas
Algoritmos	Análisis prospectivo de procesos sanitarios
	Predicción en tiempo real de urgencias hospitalarias

Área de Comunicaciones Avanzadas

Línea Tecnológica	Sub-Línea Tecnológica
Subsistemas de comuniación	Procesado digital de señales
	Implementación SW/HW (FPGA/SDR)
	Prototipado y diseño RF
	Ingeniería del espectro
Comunicaciones móviles y satélite	Nuevas formas de onda
	Estimación de parámetros de calidad
	LTE, LTE-A y 5G
Sistemas embarcados	Control y Data links
	Sistemas embarcados para payloads
Sistemas de posiciona-miento y localización	Posicionamiento y tracking de alta precisión
	Técnicas múltiples (UWB, ultrasonidos)
Sistemas basados en sensores	Captado y procesado de señales

Área de Servicios y Aplicaciones

Línea Tecnológica	Sub-Línea Tecnológica
Seguridad y privacidad Cloud	Procesado y almacenamiento de datos sensibles
Gestión Cloud	Procesado y almacenamiento de datos sensibles
Gestión Cloud	Infraestructuras y servicios autoconfigurables
	Soluciones de brokering e intercompatibilidad
Backup avanzado y monitorización de sistemas	
Realidad aumentada y geolocalización	
Learning analytics y Adaptative learning	

1.2. Exposición del problema

El término datos generados por máquinas (*Machine-generated Data* o simplemente *Machine Data* en inglés) se refiere a los datos formados por eventos discretos que se han generado automáticamente a partir de un proceso informático, aplicación u otra máquina sin la intervención de un ser humano. Los tipos más comunes de datos generados por máquinas incluyen los generados por ordenadores, redes u otros registros de equipos de tecnologías de la información; lecturas de sensores de cualquier tipo dentro del ámbito conocido normalmente como *Internet of Things* (IoT) u otros datos como información de localización de dispositivos, *etc.* En términos generales, estos datos pueden ser generados únicamente por procesos que se están ejecutando de forma autónoma en diferentes máquinas y que son conocidos como procesos Máquina a Máquina ($M2M$) o por observaciones de las interacciones de las actividades humanas realizadas con las máquinas, pero nunca incluyen los datos proporcionados directamente por humanos.

Los datos generados por las máquinas se crean de forma continua $24 \times 7 \times 365$ por cualquier aplicación TI o dispositivo electrónico que proporciona información acerca de su situación y de las actividades que realiza. Estos datos, además, se pueden presentar como ficheros de *logs* de sistemas TI, información de uso de los equipos de red mediante $SNMP$, registros de llamadas de operadores de telecomunicaciones ($CDRs$), registros de uso de aplicaciones informáticas por parte de los usuarios, registros de los procesos de negocio que se están ejecutando entre diferentes máquinas, ficheros de configuración de sistemas TI, alertas sobre mal funcionamiento, *etc.* Se trata, por tanto, de datos que además de estar en continuo crecimiento contienen información muy valiosa una vez procesada y filtrada adecuadamente.

La detección de valores anómalos en datos generados por máquinas es un campo de investigación que ha suscitado gran interés en los últimos años debido su aplicación directa en una gran variedad de ámbitos como la gestión avanzada de infraestructuras software/hardware en el ámbito de las empresas TI (por ejemplo centros de datos o grandes redes corporativas). La detección temprana de patrones o valores que se desvían de su comportamiento habitual permite la automatización de ciertos procesos de control y soporte y/o ayuda en la toma de decisiones disponiendo de información más completa. Sin embargo, no es una tarea trivial tenido en cuenta el gran volumen de datos generados por este tipo de infraestructuras y los reducidos tiempos de cómputo requeridos para la detección del evento extraño.

1.3. Métodos de resolución

La detección de anomalías en espacios de alta dimensión es uno de los métodos propuestos en la literatura científica para resolver esta problemática y superar las limitaciones expuestas, además de una de las líneas de investigación del centro tecnológico Gradiant para su posible aplicación en el sector TI e industria.

Los principales métodos consultados se centran en construir espacios vectoriales en los que cada dimensión se corresponde con una variable que contenga información acerca del sistema a monitorizar (por ejemplo, una máquina con un número N suficientemente grande de sensores que miden diferentes características del equipo susceptibles de producir errores en su funcionamiento). A partir de estos espacios, se genera un modelo que representa el comportamiento habitual del sistema sobre el que evaluar nuevas muestras o vectores mediante diferentes aproximaciones según el algoritmo (densidad de puntos vecinos, distancia al grupo más cercano, ángulos formados con el resto de vectores, *etc.*).

Sin embargo, estos espacios presentan ciertos inconvenientes o problemas en la detección de puntos anómalos, como el *enmascaramiento de variables*, efecto por el que una variable que se desvía de su valor habitual queda oculta por el resto de variables normales cuando crece la dimensión del vector que

las contiene, produciendo como consecuencia un aumento de la tasa de falsos negativos descubiertos. Algunos de estos algoritmos, y debido no únicamente a estos efectos sino también para reducir los tiempos de cómputo, aplican además diferentes técnicas para la reducción de la dimensión del espacio vectorial y la complejidad del problema como *PCA* (*Principal Component Analysis*) o *Random Projections*.

1.4. Objetivos

Los objetivos de este trabajo se centran en tres aspectos fundamentales:

1. Comprender la problemática de la alta dimensión en los métodos de detección de anomalías, identificando y valorando las limitaciones inherentes impuestas por este tipo de espacios, principalmente el enmascaramiento de variables y su relación con los falsos positivos y/o negativos.

2. Comparar y validar diferentes algoritmos de detección de anomalías aplicados a espacios de alta dimensión, investigando y comparando sus prestaciones, características, limitaciones y comportamiento partiendo de diferentes conjuntos de información; datos de prueba sintéticos generados en el laboratorio y datos reales obtenidos del ámbito de las redes e infraestructuras TI.

3. Validar la idoneidad de esta aproximación, la detección de anomalías en espacios de alta dimensión, para su aplicación directa en el análisis de datos generados por máquinas en otros campos.

Capítulo 2

Metodología

Como ya hemos comentado en el prefacio, existe una extensa literatura sobre algoritmos para la detección de anomalías en espacios de alta dimensión debido a los múltiples enfoques que se pueden tomar a la hora de denotar un punto como atípico. Algunos como el *Angle-Based Outlier Detection* ($ABOD$) [7] utilizan la varianza de los ángulos entre los distintos puntos de un conjunto y sus vecinos más cercanos para localizar *outliers*. Otros tratan evitar los problemas derivados del elevado número de dimensiones mediante la reducción de las mismas, como pueden ser el *Subspace Outlier Degree* (SOD) [8] o el *Anomaly Detection based on Principal Component Classifier* [9]. Enfoques más clásicos pueden verse en los algoritmos como el *k-nearest neighbors* [2] que está basada en la distancia a los vecinos más cercanos, o en los basados en comparar la densidad de cada punto con la de su vecindad como el *Local Outlier Factor* (LOF) [3] o el *Density-Based Spatial Clustering of Applications with Noise* ($DBSCAN$) [4].

En este capítulo explicaremos los cuatro diferentes algoritmos que vamos a utilizar para la detección de anomalías a lo largo de este trabajo: *Angle-Based Outlier Detection*, *Subspace Outlier Degree*, *Local Outlier Factor* y *Projection-Indexed Nearest-Neighbour* ($PINN$) [10], los cuales han sido elegidos debido a sus diferentes enfoques en la búsqueda de atípicos.

2.1. *Angle-Based Outlier Detection*

En esta sección veremos un algoritmo para la detección de atípicos que en lugar de tomar las distancias como medida utiliza variaciones de ángulos llamado $ABOD$, el cual puede consultarse en Kriegel et al. (2008) [7]. Este algoritmo evita el problema de la maldición de la dimensionalidad basándose en la variación del ángulo que forma cada punto (expresado como vector) con respecto al resto del conjunto o bien con parte de él (para reducir los tiempos de computación). De esta forma, un punto será considerado más atípico cuanto menor sea la amplitud del ángulo necesario para abarcar a sus k vecinos más cercanos, es decir, el punto en cuestión no está rodeado, sino que probablemente se encuentre alejado de cualquier nube de puntos (zona de alta densidad). Los puntos situados en zonas más densamente pobladas tendrán variaciones de ángulo mayores, puesto que sus vecinos se situarán a su alrededor sin una dirección claramente marcada. Trataremos de ayudar a la formación de una idea intuitiva de cómo funciona este algoritmo con la Figura 2.1 donde se puede apreciar cómo la variación de ángulo necesaria para abarcar a los vecinos más próximos de los tres puntos que se estudian difiere mucho según su posición dentro del conjunto de datos, es decir, el punto que está más alejado del conjunto presenta poca amplitud, el que está en el borde presenta una amplitud media y el que se encuentra en medio presenta la mayor amplitud.

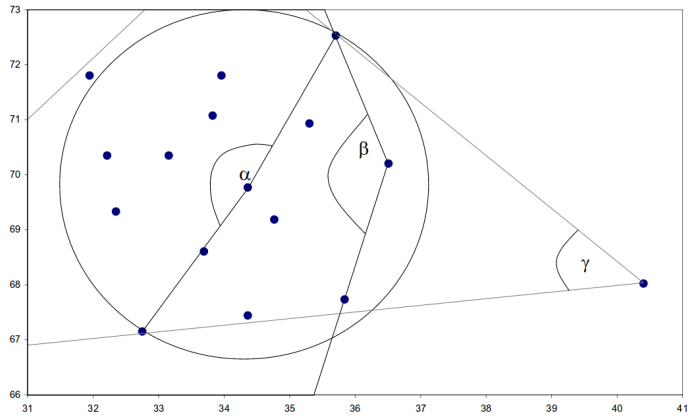


Figura 2.1: Gráfico de $ABOD$.

Por último, expondremos de manera formal los cálculos necesarios para ejecutar el algoritmo. Sea un conjunto de datos $\mathcal{D} \subseteq \mathbb{R}^d$, un punto $\vec{A} \in \mathcal{D}$, y una norma $\|.\| : \mathbb{R}^d \to \mathbb{R}_0^+$. El producto escalar denotado por $<,>: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Para dos puntos $\vec{B}, \vec{C} \in \mathcal{D}$, $\overline{BC}$ la diferencia de vectores denotada por $\vec{C} - \vec{B}$.

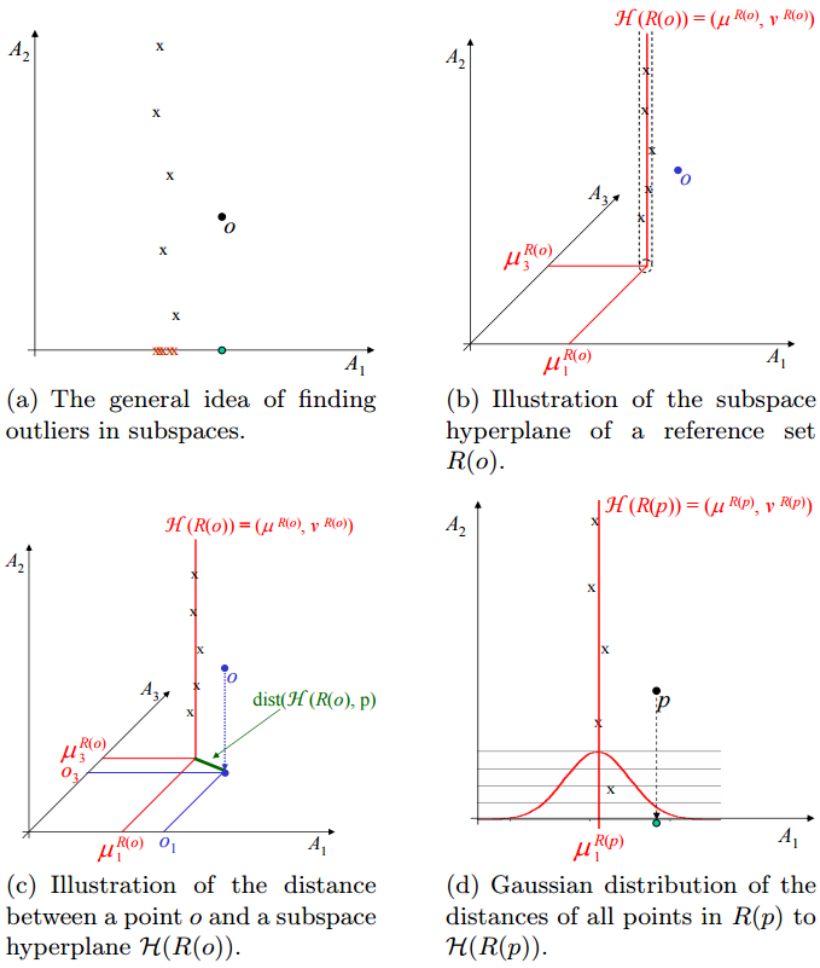
El factor atípico basado en ángulo $ABOF(\vec{A})$ es la varianza de los ángulos entre los vectores formados desde el punto $\vec{A}$ a todos los puntos del conjunto $\mathcal{D}$ ponderado por la distancia de los puntos:

$$ABOF(\vec{A}) = VAR_{\vec{B},\vec{C}\in\mathcal{D}} \left(\frac{\langle \overline{AB}, \overline{AC} \rangle}{\|\overline{AB}\|^2 \cdot \|\overline{AC}\|^2} \right) \tag{2.1}$$

2.2. *Subspace Outlier Degree*

Ya hemos comentado que cuando el número de dimensiones es elevado puede haber problemas para encontrar datos atípicos, por tanto, el siguiente algoritmo que expondremos llamado SOD tratará de solventarlos mediante la reducción de dimensionalidad. Puede verse con más detalle en Kriegel et al. (2009) [8]. Este método tratará de buscar subespacios en los cuales la mayor parte de los datos hayan sido generados por un mismo mecanismo, pero donde también se puedan encontrar puntos que hayan sido generados de forma distinta. De este modo, se evita la condición de que un dato deba tomar valores extraños en todas sus dimensiones, puesto que es suficiente con que los tome en un conjunto reducido de ellas. Es decir, la idea general es evaluar para cada punto cómo de bien se ajusta al subespacio generado por un conjunto de puntos (puntos de referencia), el cual estará formado en su mayoría por los vecinos más próximos al punto en cuestión.

Para entender un poco mejor cómo funciona el SOD utilizaremos la Figura 2.2 que ha sido extraída de [8], donde podemos observar como se ha escogido un subespacio en el cual los puntos parecen presentar una menor dispersión, ya que todos tienen un valor muy similar en el eje A_1 salvo uno, el punto o (el atípico), que será más fácilmente descubierto utilizando medidas de distancia en este subespacio que en el conjunto original. Esto es, parece que en el subespacio escogido todos los puntos han sido generados por una misma distribución excepto el atípico.



(a) The general idea of finding outliers in subspaces.

(b) Illustration of the subspace hyperplane of a reference set $R(o)$.

(c) Illustration of the distance between a point o and a subspace hyperplane $\mathcal{H}(R(o))$.

(d) Gaussian distribution of the distances of all points in $R(p)$ to $\mathcal{H}(R(p))$.

Figura 2.2: Gráfico de SOD.

De ahora en adelante consideraremos que $\mathcal{D} \subseteq \mathbb{R}^d$ es un conjunto de n puntos en un espacio d-dimensional y *dist* es una métrica para los puntos de $\mathcal{D}$. Para cualquier punto $p \in \mathbb{R}^d$, denotaremos la proyección de p en la dimensión *i-ésima* por p_i.

Intuitivamente, se busca un subespacio (hiperplano) de un conjunto de puntos S en el cual la varianza de los puntos es alta, mientras que en el hiperplano perpendicular la varianza es baja. La varianza $VAR^S \in \mathbb{R}$ del conjunto S, es la distancia media al cuadrado de los puntos de S a la media del conjunto μ^S, es decir, $VAR^S = \dfrac{\sum_{p \in S} dist(p, \mu^S)^2}{Card(S)}$, donde $Card(S)$ denota el cardinal del conjunto S. De forma análoga, la varianza en la dimensión i, denotada por $var_i^S \in \mathbb{R}$ de S se define como $var_i^S = \dfrac{\sum_{p \in S} dist(p_i, \mu_i^S)^2}{Card(S)}$.

Sea $R(p) \subseteq \mathcal{D}$ un conjunto de puntos de referencia para $p \in \mathcal{D}$, llamado *conjunto de referencia*, debemos encontrar un punto y un vector adecuados para definir el subespacio hiperplano de $R(p)$. El *vector generador del subespacio* $v^{R(p)} \in \mathbb{R}^d$, especifica los atributos (dimensiones) del conjunto que presentan baja varianza. Para decidir cuáles de los atributos debemos escoger, tendremos en cuenta que si tenemos d dimensiones y una varianza total $VAR^{R(p)}$, la varianza esperada para cada atributo será de este valor entre el número de dimensiones (d). Por tanto, diremos que un atributo tiene baja varianza

si es menor que el valor esperado multiplicado por un coeficiente α (valor entre 0 y 1). Formalmente:

$$v_i^{R(p)} = \begin{cases} 1 & si \quad var_i^{R(p)} < \alpha \dfrac{VAR^{R(p)}}{d}, \\ 0 & \text{en otro caso.} \end{cases}$$

Por tanto el subespacio hiperplano $\mathcal{H}(R(p))$ de $R(p)$ quedará definido por la tupla de valores medios $\mu^{R(p)}$ y por $v^{R(p)}$. Entonces, será necesario calcular la distancia de un punto a un hiperplano para medir el grado de anomalía de cada punto $o \in \mathcal{D}$, lo cual se hará utilizando la distancia euclídea:

$$dist(o, \mathcal{H}(S)) = \sqrt{\sum_{i=1}^{d} v_i^S \cdot (o_i, \mu_i^S)^2}.$$

Es inmediato ver que los valores próximos a 0 indican que el punto o se ajusta bien al subespacio, mientras que valores grandes aportan información sobre cómo de atípico es el punto. Procedemos entonces a definir el *grado de atípico en un subespacio*:

Sea $R(p)$ un conjunto de puntos de referencia para un punto $p \in \mathcal{D}$. El *subspace outlier degree* de p con respecto a $R(p)$ es:

$$SOD_{R(p)}(p) := \frac{dist(o, \mathcal{H}(R(p)))}{\|v^{R(p)}\|_1},$$

es decir, es la distancia del punto en cuestión al hiperplano, ponderado por el número de atributos relevantes del conjunto de referencia $R(p)$.

Por último, debemos definir una metodología para escoger el conjunto de referencia. Puesto que ya hemos mencionado anteriormente que los k vecinos más próximos en alta dimensión puede no ser un método fiable, mediremos la similitud teniendo en cuenta el número de vecinos compartidos. Sean $N_k(p) \subseteq \mathcal{D}$ los k vecinos más próximos de $p \in \mathcal{D}$. Se define el grado de similitud de los vecinos más próximos compartidos entre $p, q \in \mathcal{D}$ como $sim_{SNN}(p, q) = Card\,(N_k(p) \cap N_k(q))$. De este modo, se define el conjunto de referencia $R(p)$ de p es el conjunto de los l vecinos más próximos de p usando como medida el sim_{SNN}.

2.3. *Local Outlier Factor*

En esta sección veremos un algoritmo que trata de buscar datos atípicos de forma local y que proporciona un valor de cuánto de atípico es un punto basado en la diferencia de densidad que tiene con respecto a sus vecinos más cercanos. Este algoritmo puede estudiarse con detalle en Breunig et al. (2000) [3]. Debemos entender que intuitivamente un punto presenta una densidad alta si sus vecinos se encuentran muy próximos a él, mientras que diremos que tiene una densidad baja cuando sus vecinos estén muy próximos entre sí, pero no se encuentren cerca de él.

Comenzaremos por ver en la Figura 2.3 como el punto A tiene una densidad mucho menor que los demás puntos, por lo que el algoritmo debería devolver un valor elevado de anomalía.

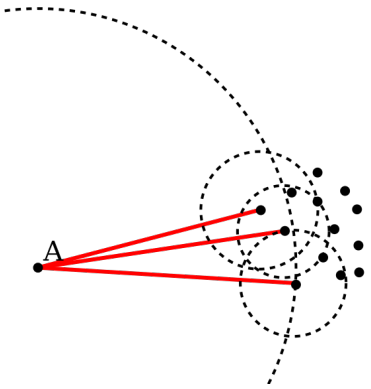


Figura 2.3: Gráfico de LOF.

Para ejecutar dicho algoritmo, debemos definir antes algunos términos que se emplearán. Primeramente, definiremos la distancia k de un punto $p \in \mathcal{D}$, denotada por k-distancia(p). Para cualquier entero positivo k, la k-distancia de un punto p, se define como la distancia $d(p, o)$ entre p y un punto $o \in \mathcal{D}$ tal que:

(i) para al menos k puntos $o' \in \mathcal{D} \setminus \{p\}$ se tiene que $d(p, o') \leq d(p, o)$, y

(ii) para al menos $k - 1$ puntos $o' \in \mathcal{D} \setminus \{p\}$ se tiene que $d(p, o') < d(p, o)$.

Otro concepto que es necesario definir es el de vecindad de distancia k de un punto p. Dada la k-distancia de p, la vecindad de distancia k de p contiene a todo punto cuya distancia a p sea menor o igual que la k-distancia, es decir, $N_{k\text{-}distancia(p)}(p) = \{q \in \mathcal{D} \setminus \{p\} \mid d(p, q) \leq k\text{-}distancia\}$. Por simplicidad y cuando no haya lugar a confusión emplearemos $N_k(p)$ en lugar de $N_{k\text{-}distancia(p)}$.

Es necesario también definir la *reach-distance* o distancia de accesibilidad de un punto p con respecto a otro o, de la siguiente forma: $reach\text{-}dist_k(p, o) = max\{k\text{-}distancia(o), d(p, o)\}$, siendo k un número natural.

Ahora tenemos definidos estos conceptos para cualquier k, sin embargo a la hora de ejecutar el algoritmo debemos fijar este valor, denotado por $MinPts$ (mínimo de puntos), para mantener fija la condición de densidad. Definimos entonces la *local reach density* de un punto p, denotada por $lrd(p)$, como:

$$lrd_{MinPts}(p) = \left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)}{|N_{MinPts}(p)|} \right)^{-1}.$$

Finalmente, definimos el *local outlier factor* de un punto p como:

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

Atendiendo a la expresión de la fórmula, observamos que los valores altos (los atípicos) se obtendrán cuando la $lrd(p)$ sea pequeña, mientras que las de sus vecinos más próximos ($MinPts$-vecinos) sean grandes.

2.4. *Projection-Indexed Nearest-Neighbour*

Cuando el número de dimensiones y la cantidad de datos en el conjunto es muy grande, utilizar directamente el LOF puede llevar a tiempos de computación elevados. Para solucionar este problema

expondremos el $PINN$, que es un algoritmo basado en el LOF pero que agiliza el cálculo de los k vecinos más próximos apoyándose en la reducción de dimensionalidad mediante una técnica llamada *Random Projections* (RP). Pueden estudiarse con mayor detalle sus características en T. de Vries et al. (2010) [10]. De dicha referencia hemos extraído la Figura 2.4, en la cual se muestran las etapas que se producen en el cálculo del algoritmo para tener una primera idea de cómo se combinan la reducción de dimensionalidad realizada mediante RP y el cálculo del grado de anomalía de un punto utilizando el LOF.

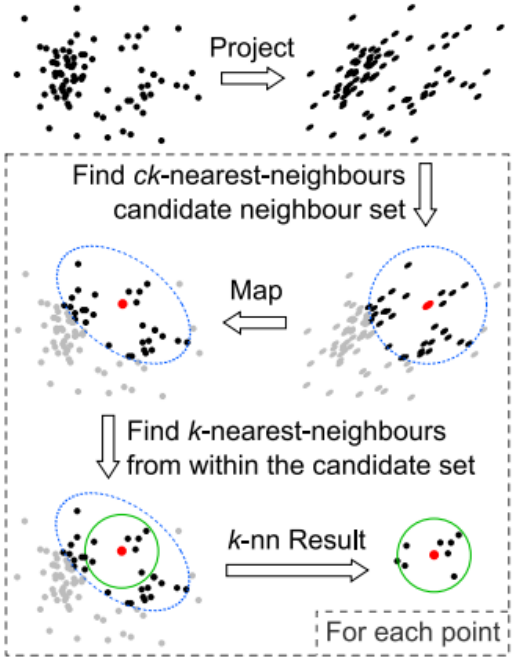


Figura 2.4: Gráfico de $PINN$.

Comenzaremos entonces por definir los aspectos necesarios para la ejecución del algoritmo. En primer lugar, para realizar una reducción de dimensión (de m a t), dada una matriz de datos $X \in \mathcal{M}_{n\times m}$, debemos calcular otra matriz $R \in \mathcal{M}_{m\times t}$, obteniendo con el producto de ambas la matriz $Y \in \mathcal{M}_{n\times t}$ de t dimensiones. De ahora en adelante para cada $x \in X$ denotaremos a la imagen de la proyección por $x' = XR$. Como hemos mencionado anteriormente, la matriz R quedará definida por el método RP, formalmente:

$$r_{ij} = \sqrt{s} \begin{cases} +1 & \text{con probabilidad} \quad \dfrac{1}{2s}, \\ 0 & \text{con probabilidad} \quad 1 - \dfrac{1}{s}, \\ -1 & \text{con probabilidad} \quad \dfrac{1}{2s}. \end{cases}$$

Una vez obtenido el espacio de dimensiones reducidas, para cada $p \in \mathcal{D}$ buscamos sus h vecinos más cercanos $N_h(p')$, siendo $p' \in \mathcal{D}'$ la proyección de p, y $\mathcal{D}'$ la imagen por la proyección de $\mathcal{D}$. Para llevar de nuevo estos puntos al espacio original definimos el siguiente conjunto:

$$RP^{-1}(N_h(p')) = \{x \in \mathcal{D} \mid RP(x) \in N_h(p')\}.$$

Si escogemos un h lo suficientemente grande (aproximadamente $2k$ o $3k$), puede comprobarse en T. de Vries et al. (2010), que los k vecinos más cercanos de p en el espacio original estarán dentro de este

conjunto con una probabilidad alta, es decir, $N_k(p) \subseteq RP^{-1}(N_h(p'))$. Puesto que cabe la posibilidad de que no todos lo vecinos más cercanos estén dentro de este conjunto, cogeremos los k más cercanos que sí lo estén y llamaremos a este conjunto la estimación de $N_k(p)$, denotándolo por $\overline{N}_k(p)$.

Por tanto, una vez calculado este conjunto podemos proceder para cada $p \in \mathcal{D}$ con una estimación del $LOF(p)$, denotada por $\overline{LOF}(p)$, que se puede expresar como:

$$\overline{LOF}(p) = \frac{\frac{1}{k}\sum_{q \in \overline{N}_k(p)} lrd(q)}{lrd(p)}.$$

Capítulo 3

Estudios numéricos

En este capítulo probaremos los algoritmos previamente explicados en diferentes conjuntos de datos, tanto simulados como reales. En la primera sección trataremos de dictaminar si es posible, en alguno de los casos, fijar un umbral para distinguir entre atípicos e *inliers*. Por otra parte, también vamos a estudiar si al variar los parámetros a partir de los cuales se han generado los puntos atípicos del conjunto de datos logramos apreciar algún patrón en los valores obtenidos por los algoritmos. En la segunda sección, realizaremos pruebas similares sobre conjuntos de datos reales para ver si se mantienen las conclusiones cuando se aumenta el número de *outliers*.

Para llevar a cabo esta comparación hemos escogido un método gráfico conocido como *Receiver Operating Charasteristic* (ROC) que explicaremos como paso previo a la exposición de los resultados. Una gráfica (ROC) es una técnica muy útil para visualizar los resultados obtenidos mediante diferentes clasificadores. Puede verse con mayor profundidad en Fawcett 2003 [5]. Aunque existen varias posibilidades, en nuestro caso nos centraremos en la que trata de enfrentar el porcentaje de falsos positivos (FP) frente al porcentaje de verdaderos positivos (TP), los cuales se calculan de la siguiente manera:

$$\text{FP rate} = \frac{\text{negativos incorrectamente clasificados}}{\text{total de negativos}}$$

$$\text{TP rate} = \frac{\text{positivos correctamente clasificados}}{\text{total de positivos}}$$

A continuación, añadiremos un ejemplo de gráfica ROC extraído de Fawcett 2003 [5] en el cual se han dibujado cinco puntos representando a cinco clasificadores distintos (véase Figura 3.1).

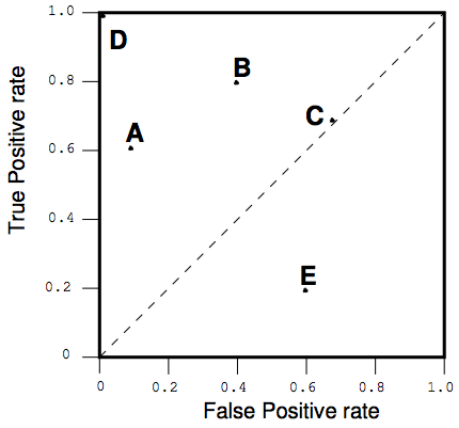


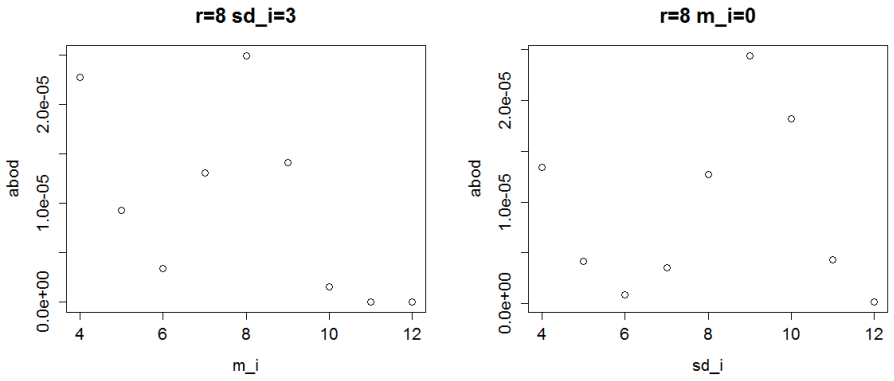
Figura 3.1: Ejemplo de curva ROC.

Atendiendo a la figura podemos concluir que nuestro objetivo será buscar algoritmos cuyo resultado sea lo más similar posible a D. En caso de no conseguirlo, buscaremos clasificadores cuya salida sea próxima a A o en su defecto a B, decantándonos por una u otra según el marco del problema. Descartaremos los clasificadores cuyos resultados queden por debajo de la línea de puntos, como es el caso de E. Los puntos que aparezcan sobre la misma línea, como en este caso C, tampoco aportan una gran solución, por lo que también los evitaremos.

3.1. Aplicación a datos simulados

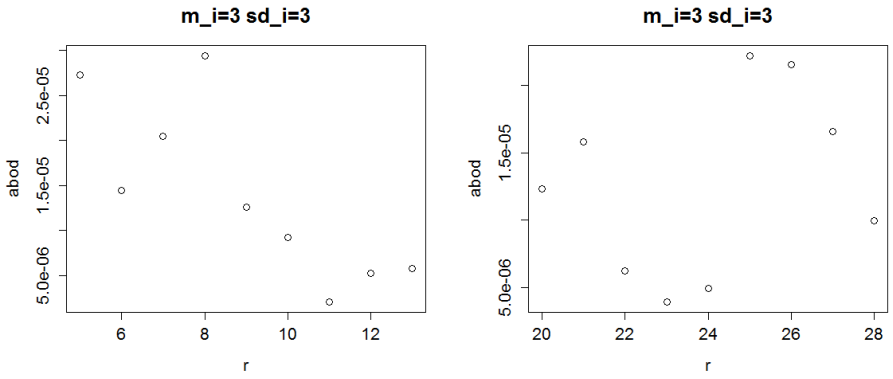
En esta sección crearemos un conjunto de $n = 500$ datos normales en un espacio de dimensiones $d = 30$ generados con la ayuda de los paquetes: `clusterGeneration` y `mvtnorm`. El primero nos permitirá crear una matriz de varianzas-covarianzas Σ con d dimensiones y en un rango de valores controlado, en este caso hemos decidido que sus autovalores estén entre 1 y 10. Luego, con la función *sample* generamos un vector de medias aleatorio μ con valores entre 1 y 10. Finalmente, utilizando la función *rmvnorm* del segundo paquete junto con la matriz Σ y el vector μ previamente calculados, obtendremos el conjunto sintético de 500 datos. A continuación, le añadiremos un punto atípico generado utilizando diferentes parámetros (como su media o su desviación típica) para comparar la capacidad que los algoritmos anteriores presentan para descubrirlo. El punto nuevo seguirá la misma distribución que el resto en la mayoría de sus dimensiones, pero variará en un número r de ellas debido a que le añadiremos un error de media μ_i y desviación típica σ_i. Las pruebas consistirán en dejar fijos dos de estos parámetros y variando el otro observar la capacidad de los diferentes algoritmos de detectar que se ha introducido un dato atípico en el conjunto de datos. El objetivo será tratar de escoger un valor como umbral entre atípicos y normales para posteriores pruebas. Además, es también interesante comparar la variación de qué parámetro es más sencillamente detectable para los algoritmos.

En la Figura 3.2 y la Figura 3.3 se muestran los resultados obtenidos para el algoritmo *ABOD* utilizando la función *Func.ABOD* del paquete `HighDimOut` con $perc = 0.1$, es decir, utilizando un 10 % del total de datos para calcular la variación del ángulo. En el primer gráfico (arriba izq.) se varía la media que se ha introducido como ruido al generar el atípico, mientras que en el segundo (arriba der.) variamos la desviación típica. Las dos siguientes son las salidas obtenidas al dejar fijos en 3 los dos parámetros anteriormente mencionados pero variando el número de dimensiones que se ven afectadas por estos cambios en el punto a descubrir. En la primera (abajo izq.) se realiza la prueba modificando el valor de pocas dimensiones, mientras que en la segunda (abajo der.) el número de dimensiones modificadas será elevado. El tiempo de computación de estas pruebas es bastante alto (supera las dos horas) en especial si se aumenta el parámetro *perc*.



(a) *ABOD* variando la media (b) *ABOD* variando la *sd*

Figura 3.2: Resultados para el algoritmo *ABOD* variando μ_i (izq.) y σ_i (der.).



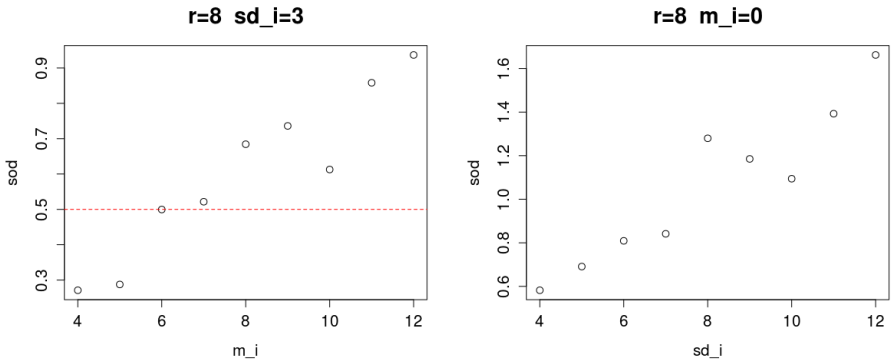
(a) *ABOD* con pocas dimensiones afectadas en el punto anómalo

(b) *ABOD* con muchas dimensiones afectadas en el punto anómalo

Figura 3.3: Resultados para el algoritmo *ABOD* variando las dimensiones afectadas.

Se puede comprobar que no existe una relación lineal clara de decrecimiento de los valores obtenidos por el *ABOD* a medida que la anomalía difiere más. Es decir, cabría esperar que al aumentar el número de dimensiones en las que se desvía el atípico o la media con la que ha sido generado, el valor resultante sea cada vez más bajo, sin embargo esto no es lo que sucede si atendemos a los gráficos. Parece que no es posible establecer un umbral para discriminar atípicos de puntos normales ya que los valores son más o menos estables aunque variemos los parámetros en gran medida.

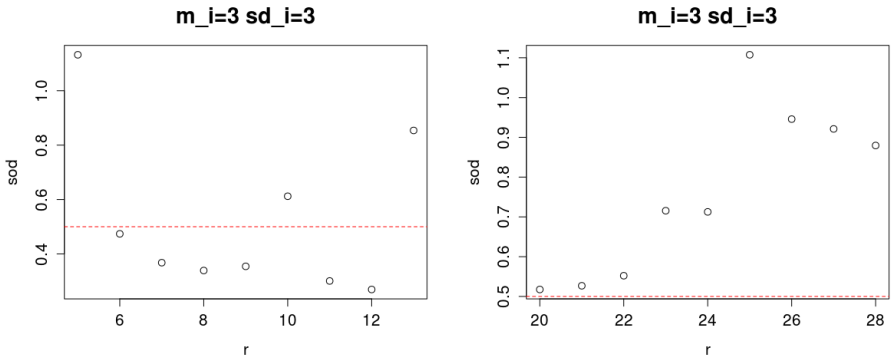
Observemos ahora en la Figura 3.4 y Figura 3.5 los resultados que se obtuvieron al utilizar el algoritmo *SOD* ejecutando la función *Func.SOD* del paquete `HighDimOut` con los siguientes parámetros: $k.nn = 40$ número de vecinos compartidos; $k.sel = 10$ tamaño del conjunto de referencia; y $alpha = 0.8$ valor de α antes explicado. Se ha procedido de manera análoga al caso anterior variando en primer lugar la media y la desviación típica del punto atípico y, posteriormente, se ha modificado el número de dimensiones afectadas por ruido en el dato a descubrir. En este caso, se reduce en gran medida el tiempo de computación empleado para ejecutar el algoritmo en comparación con el que era necesario para el *ABOD*, lo cual permitirá trabajar con conjuntos de datos más grandes de manera cómoda.



(a) *SOD* variando la media

(b) *SOD* variando la *sd*

Figura 3.4: Resultados para el algoritmo *SOD* variando μ_i (izq.) y σ_i (der.).



(a) *SOD* con pocas dimensiones afectadas en el punto anómalo

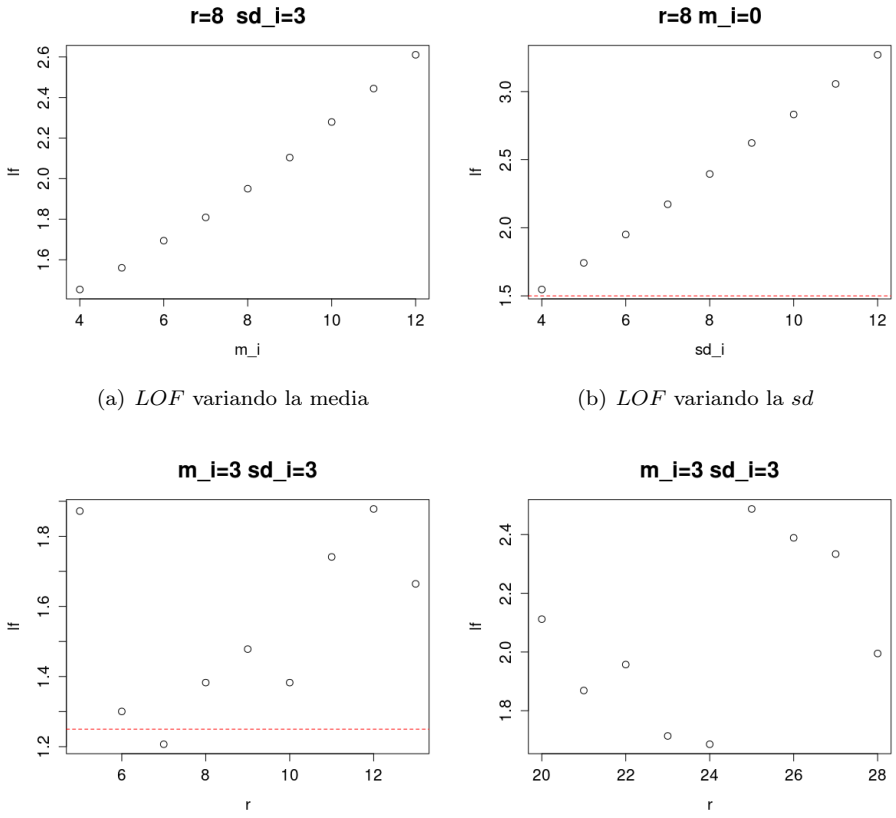
(b) *SOD* con muchas dimensiones afectadas en el punto anómalo

Figura 3.5: Resultados para el algoritmo *SOD* variando las dimensiones afectadas.

En este caso parece que hay una marcada tendencia lineal ascendente, es decir, a medida que aumentamos alguno de los parámetros que generan el dato atípico, también aumenta su grado de anomalía, con lo cual es más sencilla la búsqueda de un umbral. Esto es, cuanto mayor es el valor que devuelve el *SOD* para un punto, mayor ha sido la variación en los parámetros que lo han generado, o lo que es lo mismo, nos encontramos frente a un dato más extraño.

Para diferentes escenarios, nos encontraremos con que tomar como umbral un valor de 0.5 puede ser lo más adecuado, puesto que nos permitirá descubrir gran cantidad de atípicos, mientras que en otros casos serán más conveniente elevar este umbral hasta 1 o incluso 1.5 pudiendo así reducir la tasa de falsos positivos que estaríamos asumiendo en el caso anterior, ya que de este modo exigiríamos que el atípico se desviase más del conjunto de datos normales.

Por último, tras ejecutar el algoritmo *LOF*, expondremos en la Figura 3.6 los resultados obtenidos con la función *lof* del paquete `Rlof` fijando $k = 15$ como número de vecinos cercanos. Los pasos seguidos en este estudio son análogos a los dos anteriores. Quedará para más adelante la variación del número de vecinos cercanos, ya que en este momento tratamos solamente de comprobar la capacidad que tiene el *LOF* de descubrir un dato atípico en función de cuanto varían su media, su desviación típica y el número de dimensiones afectadas por estas modificaciones. Este algoritmo ha sido el más rápido, con mucha diferencia respecto al *ABOD* y con una diferencia menor pero clara con respecto al *SOD*, lo cual es un punto a favor que lo hace preferible a medida que aumentan las dimensiones del conjunto de datos de estudio.



(a) *LOF* variando la media

(b) *LOF* variando la *sd*

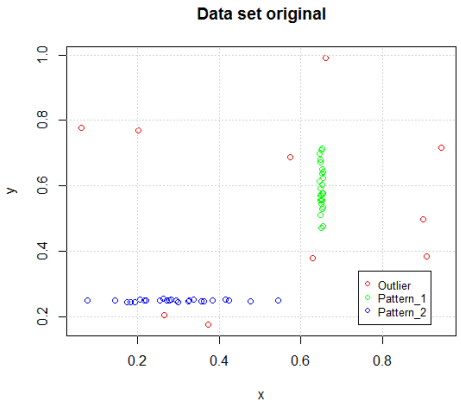
(c) *LOF* con pocas dimensiones afectadas en el punto anómalo

(d) *LOF* con muchas dimensiones afectadas en el punto anómalo

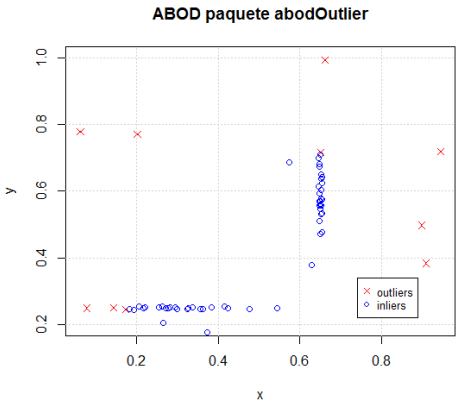
Figura 3.6: Resultados para el algoritmo *LOF*.

De manera análoga al caso anterior, podemos observar que al aumentar los valores de los parámetros, se incrementa también el grado de anomalía del punto, por lo que en este caso también sería posible fijar un umbral. Además, es sencillo apreciar que la relación lineal es muy clara, con lo que cuanto más bajo fijemos el valor del umbral, mayor será el número de atípicos encontrados aunque también aumentaremos en gran medida los falsos positivos. Sin embargo, también es sencillo buscar un umbral más alto, que resulte interesante en otros contextos donde reducir los falsos positivos tenga mayor importancia que detectar un gran número de puntos anómalos.

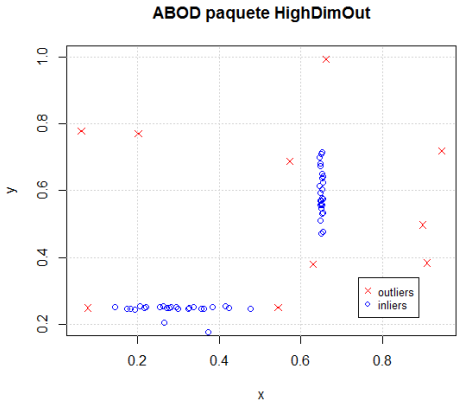
En una segunda parte de esta sección realizaremos un estudio comparativo entre el *ABOD* y el *LOF* sobre un conjunto de datos llamado *TestData* del paquete `HighDimOut`. Por su estructura, este conjunto es especialmente favorable al *ABOD*, ya que los puntos normales se encuentran distribuidos de forma muy clara siguiendo dos líneas rectas, mientras que los atípicos se presentan en posiciones aleatorias. En la Figura 3.7 expondremos el conjunto de datos etiquetado (arriba izq.) junto a los resultados obtenidos por el algoritmo *ABOD* utilizando tanto la función *Func.ABOD* del paquete `HighDimOut` (arriba der.) como la función *abod* del paquete `abodOutlier` (abajo izq.) y la clasificación resultante de emplear el *LOF* con la función *lof* del paquete `Rlof` (abajo der.). Puesto que el conjunto de datos está etiquetado y presenta diez atípicos, hemos marcado como *outliers* los diez puntos que presenten mayor grado de anomalía para cada algoritmo.



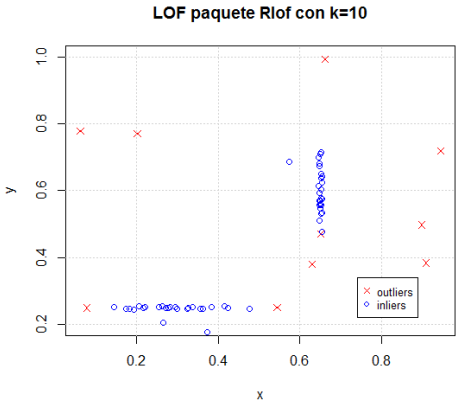
(a) *TestData* etiquetado.



(b) Clasificación *ABOD* paquete *HighDimOut*.



(c) Clasificación *ABOD* paquete *abodOutlier*.



(d) Clasificación *LOF*.

Figura 3.7: Comparativa de resultados entre *ABOD* y *LOF*.

Podemos observar que de los tres resultados el peor es claramente el del *ABOD* del paquete `HighDimOut` ya que identifica menos *outliers* de manera correcta y además es el más lento en tiempo de ejecución. El *ABOD* de `abodOutlier` es quien encuentra más atípicos, seguido de cerca por el *LOF*, siendo este último el más rápido de todos. El resultado obtenido por el *LOF* es realmente bueno, ya que debemos tener en cuenta que la construcción del conjunto de datos favorece en gran medida la forma en la que descubre atípicos el *ABOD*. Esto es así porque los puntos que forman parte de las rectas, tendrán variaciones de ángulo muy altas para abarcar a sus vecinos más próximos, mientras que los *outliers* de este ejemplo claramente necesitarán una variación menor. Sin embargo, la densidad de alguno de los puntos etiquetado como atípico (ej. el dato que está pegado a la izquierda del grupo verde) no parece menor que alguno de los etiquetados como dato normal (los datos de la izquierda de todo del grupo azul).

3.2. Aplicación a datos reales

En esta sección utilizaremos un conjunto de datos reales obtenidos de una competición de análisis de datos, concretamente *KDD Cup 1999*, el cual consta de 178810 observaciones y 42 variables y fue descargado de *http://www.kdd.org/kdd-cup/view/kdd-cup-1999*. Este conjunto de datos era especial-

mente interesante para *Gradiant* puesto que trata sobre intrusiones en sistemas de redes, es decir, el objetivo era crear un modelo capaz de clasificar correctamente las distintas conexiones en normales y peligrosas. Utilizaremos este conjunto etiquetado de datos para valorar y comparar la eficiencia del LOF y el SOD en un escenario real, variando el umbral y el número de vecinos cercanos para escoger los resultados que aporten más información sobre la detección de atípicos. Debido a problemas en el tiempo de computación, utilizaremos un subconjunto de 10000 observaciones para ilustrar los resultados obtenidos mediante el LOF que se presentan en la Figura 3.8.

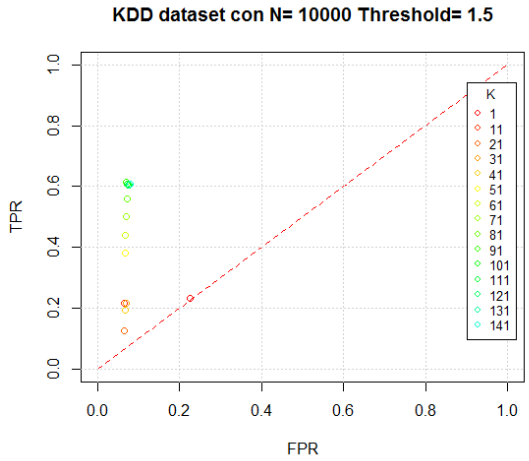


Figura 3.8: Gráfico de los resultados obtenidos para el LOF utilizando 10000 puntos y variando el número de vecinos cercanos.

Podemos observar como al aumentar el número de vecinos con un umbral fijado en 1.5 se aumenta la cantidad de datos atípicos bien señalizados, sin sufrir un aumento tan marcado en la tasa de falsos positivos. El principal problema de aumentar el valor de K recae sobre el tiempo de computación.

A continuación, presentaremos en la Figura 3.9 los resultados obtenidos al variar los valores que toma el umbral, utilizando un número de vecinos $K = 100$ fijado.

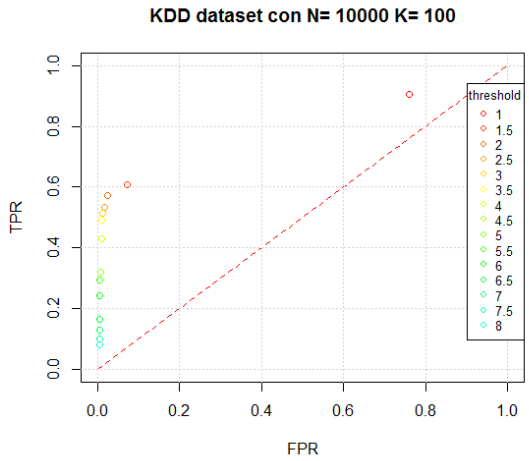


Figura 3.9: Gráfico de los resultados obtenidos para el LOF utilizando 10000 puntos y variando el valor del umbral.

Es sencillo ver como al aumentar el umbral van disminuyendo tanto el número de falsos positivos como el de verdaderos atípicos encontrados, por lo que debemos encontrar el punto de equilibrio que más nos interese, el cual puede variar de un problema a otro. En este caso parece claro que el óptimo se alcanza para el valor del umbral 1.5.

Por otra parte, realizaremos un estudio análogo sobre el mismo conjunto de datos utilizando esta vez el algoritmo *SOD*. Fijado el umbral en 1.5 y los vecinos compartidos en 30 expondremos los resultados obtenidos en la Figura 3.10 empleando esta vez 1000 puntos por problemas con el tiempo de computación.

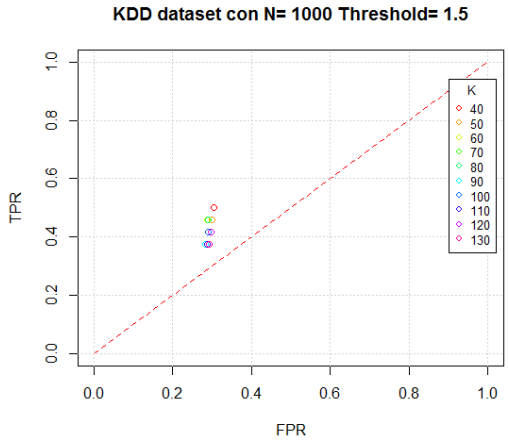


Figura 3.10: Gráfico de los resultados obtenidos para el *SOD* utilizando 1000 puntos y variando el número de vecinos $k.nn$ con $k.sel = 30$.

Atendiendo a los resultados se puede concluir que este parámetro no es tan importante como en el caso del *LOF* (los puntos están muy concentrados), pero no se consigue mejorar la clasificación anteriormente obtenida para ningún valor.

Por último, se expondrán en la Figura 3.11 los resultados de variar el umbral tratando de encontrar el mejor valor para dicho parámetro al utilizar el *SOD*.

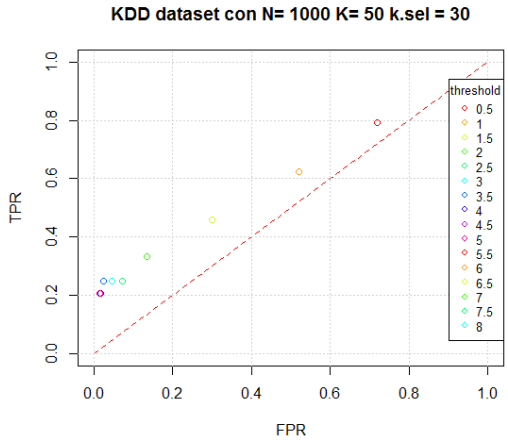


Figura 3.11: Gráfico de los resultados obtenidos para el *SOD* utilizando 1000 puntos y variando el umbral, fijado $k.nn = 50$ y $k.sel = 30$.

Podemos observar una relación lineal en los valores, es decir, a medida que aumentamos el umbral disminuyen de la misma manera los falsos positivos y los verdaderos atípicos, con lo cual se escogerá como óptimo aquel que más interese en función del problema que se nos plantea.

Capítulo 4

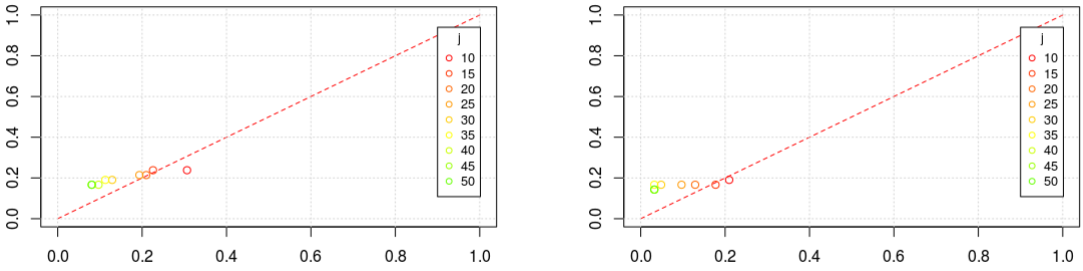
Reducción de falsos positivos

En este capítulo presentaremos dos técnicas para tratar de reducir el número de falsos positivos: la primera utiliza el algoritmo $PINN$ anteriormente explicado, y se basa en reducir las dimensiones del conjunto inicial antes de ejecutar el LOF; la segunda consiste en realizar un análisis individual de los datos marcados como atípicos por el algoritmo LOF. Para ello, utilizaremos varios conjuntos de datos del paquete `datamicroarray`: *Chowdary*, *Gordon* y *Gravier*.

El primero de ellos es un conjunto del año 2006 que contiene 104 muestras de pacientes con cáncer representados mediante 22283 variables de expresiones genéticas, de los cuales 62 casos son de mama y el resto de pulmón. El conjunto de *Gordon* está formado por los datos de 181 pacientes que se dividen en dos grupos según la variedad de cáncer de pulmón que representan. Cuenta con 150 casos de adenocarcinoma y 31 de mesotelioma, estando cada uno de ellos expresado por 12533 variables. El restante de los conjuntos es el más reciente, data de 2010, y es el más pequeño, pues cada uno de sus 168 casos de cáncer de mama han sido representados por 2905 características. En este conjunto nos encontramos una división en dos grupos diferenciados según si después de 5 años de tratamiento se ha producido metástasis (57 casos) o no (111 casos).

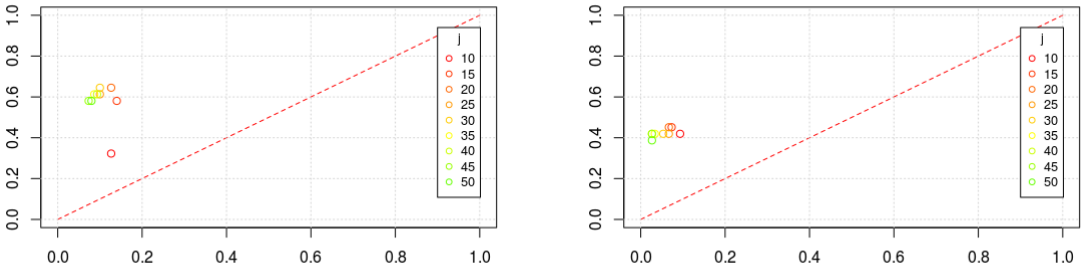
4.1. Reducción de dimensionalidad

En esta sección trataremos de comparar los resultados obtenidos al ejecutar directamente LOF sobre los conjuntos anteriormente enunciados frente a los que se obtienen cuando aplicamos previamente una reducción de dimensionalidad (recordemos que este segundo método es más eficiente en cuanto al tiempo de computación). En las Figuras 4.1, 4.2 y 4.3 se mostrarán el porcentaje de verdaderos atípicos que se han detectado en cada ejecución denotado por TP (*True Positives*), frente al porcentaje de datos que han sido marcados como anomalías cuando en realidad no lo eran, denotados por FP (*False Positives*), variando en cada iteración el número de vecinos cercanos que se consideran pero dejando fijo el umbral en 1.5 para todos los casos.



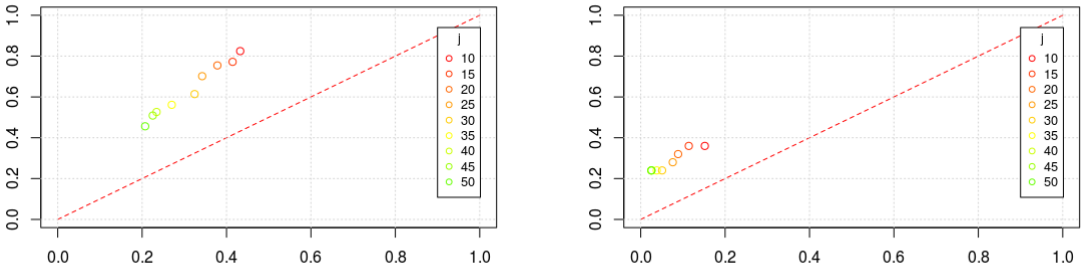
(a) *LOF* directo (b) *RP* + *LOF*

Figura 4.1: Resultados para el conjunto *Chowdary*.



(a) *LOF* directo (b) *RP* + *LOF*

Figura 4.2: Resultados para el conjunto *Gordon*.



(a) *LOF* directo (b) *RP* + *LOF*

Figura 4.3: Resultados para el conjunto *Gravier*.

A la vista de los resultados, podemos apreciar como en todos los casos se reduce el número de falsos positivos, lo cual en determinadas situaciones puede ser muy relevante. Como contrapartida, tenemos que el número de anomalías bien detectadas también se ve reducido, es decir, parece que reducir la dimensión del conjunto de datos da lugar a un resultado más conservador.

4.2. Análisis individual de atípicos

Con el objetivo de reducir los falsos positivos sin que el número de anomalías encontradas se vea afectado, presentaremos en esta sección una nueva técnica que tratará de analizar cada punto individualmente para determinar si es un verdadero atípico o un falso positivo. Comenzaremos ejecutando el LOF, para posteriormente realizar un estudio de cada dato marcado como atípico. Una vez obtenido este conjunto formado por los posibles datos atípicos utilizaremos la siguiente regla: si un dato se encuentra a una distancia superior a 3 desviaciones típicas de la media de los q vecinos más cercanos etiquetados como datos normales en un porcentaje alto de dimensiones, comparado con el número de dimensiones para las cuales sus valores se encuentran a una distancia de 2 desviaciones típicas, entonces esta es una verdadera anomalía (tiene muchas dimensiones con valores extremos), mientras que en el caso contrario nos encontraríamos con un falso positivo (un valor alejado de la media, pero no extremo en un elevado número de dimensiones). El valor crítico que tomaremos para diferenciar los verdaderos atípicos de los falsos positivos será 4, es decir, si el porcentaje de dimensiones para las cuales el dato se encuentra a más de 3 desviaciones típicas multiplicado por 4 es menor que el porcentaje de dimensiones que se alejan 2 desviaciones típicas, es un falso positivo, y en caso contrario es un verdadero atípico. Es importante destacar que para la aplicación de esta técnica es necesario contar con un conjunto de datos etiquetados como no atípicos, ya que éstos formarán la matriz de la cual se extraerán las medias y desviaciones típicas por columnas imprescindibles para el cálculo del valor crítico . En el Cuadro 4.1 expondremos los resultados obtenidos para algunos de los datos marcados como atípicos por el LOF para el conjunto de datos *Chowdary*. Dicha tabla estará formada en cada fila por los resultados obtenidos para un mismo punto, y tendrá cinco columnas que respectivamente expresan: su valor obtenido mediante el LOF; porcentaje de dimensiones que se separan más de 2σ de la media; porcentaje de dimensiones que se separan más de 3σ de la media; cociente entre la segunda y tercera columna; y si es un verdadero atípico o un falso positivo. En la última columna escribiremos en color azul las buenas correcciones y en rojo las que sean incorrectas.

LOF	$> 2\sigma$	$> 3\sigma$	$\frac{> 2\sigma}{> 3\sigma}$	Resultado
1.48	45	10	4,5	FP
1.42	10	0,6	16.67	FP
7.72	86	81	$\simeq 1$	TP
8.62	88	84	$\simeq 1$	TP
10.37	94	90	$\simeq 1$	TP
8.94	90	86	$\simeq 1$	TP
9.53	89	84	$\simeq 1$	TP

Cuadro 4.1: Resultados *Chowdary* con $k = 25$, umbral $= 1.4$ y $q = 15$.

En este ejemplo el LOF devolvía 8 atípicos de los cuales hemos escogido aleatoriamente 7 para probar la técnica mencionada, obteniendo un resultado perfecto.

A continuación, en el Cuadro 4.2 se exponen los resultados para el conjunto *Gordon*.

LOF	$> 2\sigma$	$> 3\sigma$	$\frac{> 2\sigma}{> 3\sigma}$	Resultado
3.48	30	2	15	FP
2.63	48	9	> 5	FP
1.5	29	15	$\simeq 2$	TP
1.78	34	20	$\simeq 2$	TP
1.90	51	36	$\simeq 2$	TP
1.66	24	11	$\simeq 2.5$	TP
1.82	39	24	$\simeq 1.5$	TP

Cuadro 4.2: Resultados *Gordon* con $k = 25$, umbral $= 1.5$ y $q = 12$.

En este segundo ejemplo, escogiendo de nuevo un subconjunto aleatorio de los datos etiquetados como atípicos, la técnica garantizaría un acierto total en la reducción de falsos positivos.

Los resultados del último ejemplo se mostrarán en el Cuadro 4.3.

LOF	$> 2\sigma$	$> 3\sigma$	$\frac{> 2\sigma}{> 3\sigma}$	Resultado
2.06	36	4	9	FP
1.82	31	4	$\simeq 8$	FP
1.63	31	7	> 4	FP
2.52	46	31	$\simeq 1.5$	TP
1.57	44	26	$\simeq 1.5$	TP
1.61	35	18	$\simeq 2$	TP
1.94	47	30	$\simeq 1.5$	TP

Cuadro 4.3: Resultados *Gravier* con $k = 25$, umbral $= 1.5$ y $q = 15$.

Después de realizar un análisis similar al de los casos anteriores nos encontramos con un pleno de aciertos a la hora de discriminar entre falsos positivos y verdaderos atípicos.

Capítulo 5

Conclusiones

En este trabajo se han expuesto diversas técnicas para la detección de anomalías en espacios de alta dimensión con diferentes resultados para cada una de ellas. A continuación expondremos algunos de sus factores más importantes.

Comenzaremos por el *ABOD*, el algoritmo basado en las varianzas de ángulos para asignar el grado de anomalía a cada punto, el cual sufre de elevados tiempos de computación cuando el conjunto de datos es grande. Además es complicado fijar un umbral puesto que retorna valores que difieren mucho de un conjunto de datos a otro. Como parte positiva nos quedamos con su enfoque distinto que no se ve tan penalizado por la *curse of dimensionality* como sí que les sucede a aquellos algoritmos que se basan en distancias.

Seguidamente analizaremos el *SOD*, que trataba de buscar subespacios de baja varianza donde los atípicos sean más fácilmente detectables. Éste suponía una mejora en cuanto al tiempo de computación además de permitir la estimación de un umbral, ya que sus valores eran bastante similares aunque los conjuntos de datos sobre los que se aplicase fuesen diferentes. Sin embargo, a medida que el tamaño del *dataset* de estudio aumentaba su tiempo de computación también lo hacía, no pudiendo competir en estos términos con el *LOF*.

A continuación comentaremos los resultados del *LOF*, uno de los algoritmos más utilizados para la detección de atípicos, que se basa en las diferencias de densidad entre un punto y sus vecinos más cercanos. Este algoritmo era el más ágil computacionalmente, por lo que permitía la posibilidad de utilizar amplias combinaciones de parámetros (como variar el número de vecinos cercanos o el umbral) para poder escoger el resultado más adecuado. Además, si contamos con un conjunto de datos no atípicos etiquetado, podemos poner en práctica la técnica antes expuesta para reducir los falsos positivos sin que el porcentaje de anomalías detectado descienda.

Por último, debemos mencionar que utilizar una técnica de reducción de la dimensionalidad como *Random Projections* ayuda a que el *LOF* sea más robusto, es decir, se encuentren menos falsos positivos, pero esto se consigue con un descenso en la tasa de verdaderos positivos. Por lo tanto, emplearemos la reducción de dimensión cuando sea muy importante para el problema evitar los falsos positivos en la medida de lo posible.

Llegados a este punto, es importante destacar que aún quedan muchas cosas por hacer en la detección de anomalías en alta dimensión, ya que no hay una definición universal de lo que debe ser considerado anomalía, sino que es algo que puede variar de un escenario a otro, e incluso en el mismo escenario puede cambiar según qué busquemos en un momento determinado. Por otro lado, todavía hay margen de mejora en los algoritmos a la hora de reducir los tiempos de computación, lo cual es de vital importancia hoy en día para realizar procesado de datos en *streaming*.

Apéndice A

Desarrollo de software

En este apéndice se incluirán dos funciones que se han programado para agilizar las pruebas realizadas en el trabajo. La primera se ha utilizado para calcular el grado de atípico de un punto mediante el algoritmo $PINN$, incluyendo como paso previo el cálculo de la matriz necesaria para realizar RP, mientras que la segunda se ha empleado para la generación de gráficos ROC.

Para la primera función contamos con cuatro parámetros de entrada: A la matriz formada por el conjunto de datos, *query_point* el punto sobre el cual se desea calcular el $PINN$, r porcentaje de dimensiones al que se quiere recudir el conjunto de datos, y nn número de vecinos cercanos que se buscarán en el espacio de origen.

```
PINN<-function(A, query_point,r, nn){

    library(FNN)
    library(Rlof)

    #Dimensiones del espacio original
    N=nrow(A)
    d=ncol(A)
    #Dimensiones del espacio reducido
    D=d*r
    #Número de vecinos cercanos
    P=nn
    A2=A3=NULL

    #Creamos la matriz R para ejecutar Random Projections
    R=matrix(sample(x=c(sqrt(3)*1,0,-1*sqrt(3)),prob=c(1/6,2/3,1/6),
    size=D*N, replace=TRUE),nrow=d,ncol=D)
    E= 1/sqrt(D)*(A%*%R)
    #Calculamos las coordenadas del punto en el espacio reducido
    query_projected_point = 1/sqrt(D)*(query_point%*%R)
    #Calculamos los 2*P vecinos cercanos en el espacio reducido
    index<-knnx.index(E,query_projected_point, k=2*P)
    A2 <- A[index,]
    #Calculamos los P vecinos cercanos en el espacio original
    index<-knnx.index(A2,t(as.matrix(query_point,ncol=ncol(A2))), k=P)
    A3 <- A2[index,]
```

```
    #Realizamos la estimación del LOF con P/4 vecinos cercanos en el espacio original
    z<-lof(rbind(A3,query_point),round(P*0.25),cores= detectCores() - 2)
    z[which(z=="NaN")]=1
    #Devolvemos el valor del LOF para el punto en cuestión
    return(z[P+1])
}
```

A continuación se muestra la función con la cual se obtienen las gráficas de las curvas *ROC*, es decir, se representarán los falsos positivos frente a los verdaderos atípicos detectados para cada combinación de parámetros (se incluirá como ejemplo el caso del *LOF* con umbral fijo y variando el número de vecinos). Los parámetros de entrada serán: *data* el conjunto de datos etiquetados y *threshold* el umbral fijado para la prueba.

```
roc <- function(data,threshold){
    test_data<-data
    #Creamos la matriz formada por los datos y sus componentes
    E<-test_data$x
    DATA<-scale(E)
    N=1000
    threshold_lof=threshold
    string=paste("KDD dataset con N=",as.character(N),
    "Umbral=",as.character(threshold_lof))
    plot(0:1,0:1,type="l",col=2,xlab="FPR",ylab="TPR",main=string,lty=2)
    grid()
    cl= rainbow(7)
    point_color=0
    texto = NULL

    #Realizamos un bucle variando el número de vecinos cercanos
    for (j in seq(10,30,3)){
      TOTAL=nrow(DATA)
      NUM_OUTLIERS=length(which(test_data$y!="n"))
      NUM_INLIERS=TOTAL- NUM_OUTLIERS
      z<-lof(DATA,j,cores= detectCores() - 2)
      outliers<- which(z >= threshold_lof)
      inliers<- which(z < threshold_lof)
      TP= sum(as.numeric(outliers %in% which(test_data$y!="n")))
      TN= sum(as.numeric(inliers %in% which(test_data$y=="n")))
      FP= sum(as.numeric(outliers %in% which(test_data$y=="n")))
      FN= sum(as.numeric(inliers %in% which(test_data$y!="n")))
      TPR=TP/(TP+FN)
      FPR= FP/(FP+TN)
      point_color=point_color+1
      color= cl[point_color]
      points(FPR,TPR,col=color)
      texto=append(texto, as.character(j),length(texto))
    }
    legend(0.9, 0.94, legend=texto, col=cl, pch=1, cex=0.8, title="K")
    #Devolvemos un vector con los valores del LOF
    zl<-return(z)
}
```

Bibliografía

[1] C.C. Aggarwal (2013). An Introduction to Outlier Analysis. Springer Science+Business Media New York.

[2] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician. 46 (3): 175–185

[3] M. Breunig, H.-P. Kriegel, R. Ng and J. Sander (2008). LOF: Identifying Density-based Local Outliers, ACM SIGMOD Conference, 2000.

[4] M. Ester, H.-P. Kriegel, J Sander and X. Xu (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Institute for Computer Science, University of Munich. Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).

[5] T. Fawcett (2003). ROC Graphs: Notes and Practical Considerations for Data Mining Researchers, HP Laboratories Palo Alto, Palo Alto, CA 94306, USA.

[6] F. Keller, E. Müller and K. Böhm (2012). HiCS: high contrast subspaces for density-based outlier ranking, In Proceedings of the 28th International Conference on Data Engineering (ICDE), Washington, DC.

[7] H.-P. Kriegel, M. Schubert, and A. Zimek (2008). Angle-based Outlier Detection in High-Dimensional Data, ACM KDD Conference.

[8] H.-P. Kriegel, P. Kroger, E. Schubert and A. Zimek (2009). Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data. PAKDD Conference.

[9] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn and L. Chang (2003). A novel anomaly detection scheme-based on principal component classifier. In Proceedings of the 3rd IEEE International Conference on Data Mining. 353–365.

[10] T. de Vries, S. Chawla and M. E. Houle (2010). Finding local anomalies in very high dimensional space, In Proceedings of the 10th IEEE International Conference on Data Mining (ICDM), Sydney, Australia, 128–137.

[11] A. Zimek, E. Schubert and H.-P. Kriegel (2012). A Survey on Unsupervised Outlier Detection in High-Dimensional Numerical Data. Wiley Online Library (*wileyonlinelibrary.com*).

[12] A. Beygelzimer, S. Kakadet, J. Langford, S. Arya, D. Mount and S. Li (2013). FNN: Fast Nearest Neighbor Search Algorithms and Applications. R package version 1.1. https://CRAN.R-project.org/package=FNN

[13] D. Chung, H. Chun and S. Keles (2013). spls: Sparse Partial Least Squares (SPLS) Regression and Classification. R package version 2.2-1. https://CRAN.R-project.org/package=spls

[14] C. Fan (2015). HighDimOut: Outlier Detection Algorithms for High-Dimensional Data. R package version 1.0.0. https://CRAN.R-project.org/package=HighDimOut

[15] A. Genz, F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl and T. Hothorn (2017). mvtnorm: Multivariate Normal and t Distributions. R package version 1.0-6. URL http://CRAN.R-project.org/package=mvtnorm

[16] Y. Hu, W. Murray and Y. Shan (2015). Rlof: R Parallel Implementation of Local Outlier Factor(LOF). R package version 1.1.1. http://CRAN.R-project.org/package=Rlof

[17] W. Qiu and H. Joe. (2015). clusterGeneration: Random Cluster Generation (with Specified Degree of Separation). R package version 1.3.4. https://CRAN.R-project.org/package=clusterGeneration

[18] R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

[19] J. A. Ramey (2016). datamicroarray: Collection of Data Sets for Classification. https://github.com/ramhiser/datamicroarray, http://ramhiser.com.

[20] W. N. Venables and B.D. Ripley (2002). Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0