Universidad Zaragoza
1542

Trabajo Fin de Máster

UTILIZACIÓN DE TÉCNICAS DE *MACHINE LEARNING* PARA ANÁLISIS DE GRAN CANTIDAD DE DATOS.

Alberto Javier Morales Ramos

Director académico: Dr Carlos Cajal Hernando

Director militar: N / A

Centro Universitario de la Defensa-Academia General Militar

2022

Agradecimientos

Deseo dar las gracias a mi director de Trabajo Final de Máster por su ayuda en el enfoque de los conceptos plasmados en el mismo y su comprensión a la hora de aceptar los cambios de última hora reflejados en el texto.

Mi agradecimiento también va hacia el personal de la Oficina del Programa A400M, que me ha ayudado en las siempre inciertas cuestiones de edición de textos, y el del Ala 31 del Ejército del Aire, que me ha proporcionado valiosa información que procesar.

Madrid, 11 de junio de 2022

RESUMEN

El *Machine Learning* o Aprendizaje Automático explora la capacidad que los modernos sistemas computerizados pueden tener para autoaprender. Este autoaprendizaje se entiende como una mejora de los algoritmos que los controlan, aprovechando la retroalimentación de información cuando se comparan los resultados estimados de sus modelos con la información de grandes Conjuntos de Datos de la realidad física.

Las herramientas de consolidación de esta disciplina se mueven en el terreno científico de la Matemática y la Estadística, y se concretan en términos técnicos y de ingeniería mediante entornos de desarrollo y programación enfocados a aquéllas, y que están dotados de funciones de manejo de variables aleatorias y que puedan enfrentarse estadísticamente a cantidades apreciables de información numérica o cuantificable.

En este Trabajo se ha descrito en términos básicos lo más imprescindible para entender las diferentes herramientas que permiten la mejora de algoritmos software. Posteriormente se ha implementado una aplicación básica en un entorno comercial de programación de entre los más populares, para interpretar conjuntos de datos, lo que se ha llevado a cabo contra un conjunto de datos simulados y también contra una serie masiva de datos de la experiencia real.

Por último, se han explotado los datos analizados para extraer algunos resultados de clasificación y se han insinuado algunos rasgos que, con el conveniente desarrollo, podían llevar también a la inferencia de algunas conclusiones.

Palabras clave

Machine-Learning Autoaprendizaje Clasificación Inferencia

ABSTRACT

Machine Learning deeps into modern computer systems potential ability to learn. That self learning is understood as an improvement of the software algorithms they are controlled by based on the feedback of information when estimations of their model results are confronted against the information of the big data sets coming from physical reality.

The tools that have contributed to consolidate this discipline come from Mathematics and Statistics as scientific environment and realiza into technical and engineering terms, thanks to programming and development applications focused on those Maths and Stats. They comprehend the suitable functions to manage random variable and can statistically face fair amounts of metric or quantifiable information.

The basics needed to understand the different tools that allow software algorithms improvement have been described in this work. After that, a simple application developed in one of the most popular commercial setups has been implemented, in order to process data sets. This has been carried out using a synthetic data set and also using a masive real experience data base.

Eventually, the processed data have been used to extract some classification results and some features have been pointed out, features that suitably developed could also lead to some inferences.

KEYWORDS

Machine Learning Classification Inference

INDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

- - -

ABREVIATURAS, SIGLAS Y ACRÓNIMOS

Abreviaturas

Inf – Inferencia

Siglas

DS – Data Set

FFF – Fit, Form and Function

OOP – Object Oriented Programming

PCA – Principal Component Analysis

P/N – Part Number

Acrónimos

DIRACA – Director Académico

1 INTRODUCCIÓN AL AUTOAPRENDIZAJE DE ALGORITMOS

Es un hecho notorio, en todos los campos relacionados con los sistemas computacionales, que seguramente esta tecnología es la que registra un progreso más rápido a nivel global. Existen varios modos de establecer métricas que apoyen numéricamente esta sensación intuitiva [01] y por regla general el progreso tecnológico se asocia de manera indisociable con el informático. Así, a un horizonte de progreso de una década, no sólo el avance que se va registrando es de gran magnitud, de forma consolidada desde antes de mitad del siglo XX, sino que la dirección y el concepto de esos avances es difícil de predecir, ya que las prestaciones físicas (*hardware*), en continua mejora, que se ponen a disposición de los desarrolladores de aplicaciones, catalizan líneas de investigación *software* que unos pocos años antes parecían inalcanzables.

Ello es especialmente cierto aplicado al autoaprendizaje en sistemas computacionales, que es la definición básica del *Machine Learning*. Si se considera la expectativa de progreso tecnológico de hace medio siglo, se podían observar varios rasgos:

- Las posibilidades de computación existentes se consideraban espectaculares para la época, pero la tendencia era a considerarlas como actividades de cálculo no excesivamente complejas en sí mismas, siendo la novedad la rapidez y el volumen de información manejados, nunca vistos hasta entonces.
- Estas prestaciones computacionales eran ofrecidas por grandes computadores *mainframe*, con sistemas operativos multiusuario, estando en gran medida toda la comunidad de patrocinadores, propietarios y usuarios en los entornos universitario, militar, gubernamental o empresarial corporativo.
- El simple concepto de que un sistema computador fuera capaz de autoaprender pertenecía en aquella época al ámbito de lo fantástico, e incluso se le asociaban connotaciones socialmente inquietantes, desde el punto de vista de un poder excesivo que dichas plataformas pudieran llegar a adquirir como inteligencia no humana.

Con el paso del tiempo, como es bien sabido, se produjo la irrupción de la informática personal, en micro plataformas de coste asequible para todo tipo de instituciones e individuos, de cualquier perfil científico, técnico, investigador, empresarial o particular. Y no por ello se perdió la posibilidad de trabajo colaborativo, merced a la aparición de diversas redes precursoras del servicio *World Wide Web* y finalmente de Internet. Ello hizo posible una enorme multiplicidad de trabajos de investigación e implementación de sistemas capaces de autoaprender, sin cuya eclosión este campo no se encontraría en el estado pujante que registra, siendo ya una realidad los múltiples ejemplos de aplicaciones útiles que están actualmente en aprovechamiento.

Así pues, en cualesquiera tipo de plataforma o ámbito de uso, esta capacidad y el concepto de adquisición de experiencia en sistemas automáticos está comenzando a adquirir la madurez que se espera cuando ya han transcurrido unos años de su disrupción, y por tanto proporcionando valor añadido en multitud de usos y a millones de personas. En la actual situación de generación y gestión de cantidades masivas de información, cualquier sistema dispone de

multitud de Conjuntos de Datos (en adelante *Data Sets*, DS) sobre los que probar nuevos algoritmos así como su manera de evolucionar. Las situaciones en que dicha gestión de manera efectiva no dejan de aparecer en la literatura de investigación. Los ejemplos clasificación e inferencia de información, así como de otros ejemplos de explotación son destacadamente prometedores.

2 OBJETIVOS Y METODOLOGÍA

2.1 OBJETIVOS Y ALCANCE

Se pretende desarrollar este Trabajo Fin de Máster cubriendo varios objetivos concretos:

- A fin de partir de un conocimiento adecuado del *Machine Learning*, de cara a los usos que se les puede dar dentro del ámbito de la Gestión de Programas, se definirán las bases del aprendizaje de sistemas automáticos y se describirán sus fundamentos, procurando acotar qué aspectos serán los que más peso adquieran en nuestro estudio e investigación.
- Siendo el *Machine Learning* un campo de gran actividad investigadora, en el que el desarrollo y aparición de nuevas herramientas y técnicas es constante, se mencionarán unas cuantas de entre las más generalizadas, escogiéndose finalmente una línea y entorno concretos para trabajar en algunos casos prácticos, de complejidad baja o moderada y de perfil académico, que aparecerán en este Trabajo.
- Se pretende confeccionar un modelo sencillo de *Machine Learning*, para ilustrar los conceptos estudiados, procurando encuadrarlo en una situación asociada a la gestión de Programas e identificar algunos DS susceptibles de análisis con *Machine Learning*. Se pretende ejecutar una implementación sencilla de dicho análisis.
- En consonancia con el estudio de estas técnicas, y evaluación de su eficacia, se pretende recopilar lecciones aprendidas sobre las mísmas, incluyendo un estudio de validación de los resultados de los casos prácticos propuestos, así como posibles maneras de refinar el proceso utilizado.

2.2 METODOLOGÍA

Se propone la siguiente metodología para trabajar con series de datos cuantitativos:

- Sobre la base de un estudio teórico y descriptivo de las técnicas de *Machine Learning*, diseñar un experimento consistente en generar un DS simulado que reproduzca los resultados de un sistema en el que introduzcamos defectos o condicionantes conocidos y controlables.
- Analizar el DS con alguna técnica de *Machine Learning* implementable al grado de detalle de este Trabajo. Ver si las conclusiones y el modo en que el sistema autoaprende son coherentes con los condicionantes de partida.

- Repetir el ejercicio para llevar a cabo algún análisis adicional sobre datos simulados, procurando que el nivel de complejidad de la información se incremente.
- Si resulta factible, conseguir series de datos reales que se hayan producido en el ámbito de algún Programa conocido. Aplicar la misma metodología y recopilar los resultados.

3 ANTECEDENTES Y MARCO TEÓRICO

Hace ahora unos tres cuartos de siglo que comenzó la andadura de los primeros computadores electrónicos modernos. Desde entonces sus logros no han dejado de maravillar a su comunidad de creadores o usuarios, debido, entre otras razones, al crecimiento hiper-exponencial de sus prestaciones.

Una determinada concepción inicial del uso de computadores se fue basando en que eran capaces de ejecutar tareas sencillas, lejanas de la gran complejidad de la inteligencia humana. Solamente que lo podían hacer una cantidad enorme de veces, o con una enorme rapidez. Sin embargo, fue cuestión de poco tiempo que surgiese como hipótesis si estos dispositivos podían llegar a emular con éxito características previamente observadas sólo en la inteligencia humana, y por tanto poderse describir como “inteligencia artificial”. Entre algunas de estas características se cuentan la memoria integradora, la capacidad de abstracción o, muy singularmente, la capacidad de aprender.

3.1 FUNDAMENTOS BÁSICOS DE *MACHINE LEARNING*

Como se haría con cualquier disciplina, para fundamentar este Trabajo adecuadamente sobre el estado actual del *Machine Learning*, se recuerdan las definiciones básicas del mismo. Este documento no es un libro de enseñanza sobre la disciplina, pero conviene que en su texto estén presentes, al menos de forma resumida, algunos de los conceptos fundamentales sobre los que se apoyarán los casos prácticos de estudio que lo vertebran.

3.1.1 Definiciones

Como aclaración previa a las definiciones de este apartado, y aunque como regla general no se desea el uso generalizado de anglicismos, en este TFM se usará el término *Machine Learning* en inglés, dado lo extendido de su uso y el beneficio que aporta el evitar ambigüedades. No obstante se recuerda la traducción *Aprendizaje Automático* como una de las que con más asiduidad se maneja en textos en español [02].

A la hora de definir el propio concepto de *aprendizaje* aplicado a *Machine Learning* se han visto dos propuestas en [03]:

Por una parte, el *Machine Learning* se definiría como “el campo de estudio que da a las computadoras la capacidad de aprender sin haber sido explícitamente programadas para ello”. De esta manera se abre una serie de caminos y procedimientos para que los resultados arrojados por estos dispositivos aumenten su precisión y utilidad, sin necesidad de sofisticar apreciablemente su *software* ni su *hardware*.

De manera seguramente más fáctica, reuniendo parámetros relevantes y conectada con rasgos fácilmente medibles, la misma referencia cita a Tom Mitchell, destacado experto en esta materia de la Universidad Carnegie-Mellon: “Se dice que un software aprende de la experiencia E, respecto a una tarea T y un desempeño D, si su desempeño al realizar T, medido mediante D, mejora con la experiencia E”.

Es fácil observar que esta definición aporta muy poco valor si los tres conceptos respecto a los que se articula no se pueden manejar de manera totalmente mensurable y numérica. Por el contrario, si se consiguen establecer métricas adecuadas a este respecto, se posibilitará un entorno de investigación y prueba continua que no puede sino arrojar resultados de gran interés.

Por último, se añaden unas notas acerca del uso de los modelos estadísticos, aunque no correspondan a una definición ortodoxa de los mismos. Se suele atribuir a un conocido matemático y estadístico británico la cita de que “todos los modelos son erróneos, sin embargo, algunos son útiles” (Box y Draper, 1987). Sin resultar, como decimos, una definición o descripción estrictamente académica, sí nos sirve para reflexionar cómo en nuestro contexto un modelo no se buscará para que proporcione una explicación absolutamente identificada con la realidad a la que lo queremos referir, sino que nos bastará con que se aproxime lo suficiente y de esta manera nos permita extraer correlaciones útiles, aunque no sean perfectas.

Otra reflexión que cabe aportar, para ilustrar esta relación estrecha entre el *Machine Learning* y la Estadística, pero al mismo tiempo ayudar a no confundirlas, es que el *Machine Learning* es una rama de las ciencias y técnicas computacionales, mientras que la Estadística lo es de la Matemática, con todas las connotaciones de abstracción que ello conlleva.

Para evitar cualquier tipo de ambigüedad, se clarifica que el uso de modelos estadístico-matemáticos es una herramienta primordial en la disciplina del *Machine Learning*, en su actual desarrollo.

3.1.2 Historia

La reseña esquemática de la evolución histórica de esta disciplina, relativamente reciente, ha sido sacada de [04].

Como forma de designación, el término *Machine Learning* fue acuñado, al final de la década de los años 50 del siglo pasado, por el científico de la computación Arthur Samuel, de la corporación IBM.

Durante los siguientes veinte años los logros que se iban registrando en este campo, apoyados en la tecnología *hardware* existente, se iban orientando casi siempre al reconocimiento y clasificación de patrones. Para ello se analizaban las correlaciones y la aparición repetitiva de características al revisar cantidades muy grandes de datos. La condición de que se produzca una retroalimentación en función del análisis de grandes cantidades de datos reales, para ser considerado *Machine Learning*, se cumplía. Como se ha dicho en el punto 1. de “Introducción”, se aprovechaba el estado del arte del momento, de poder agregar una enorme cantidad de cálculos de complejidad baja o moderada, para lograr ese resultado.

El hito que se produce en 1981 pone en relación el *Machine Learning* con el empleo de redes neuronales, logrando por primera vez el reconocimiento de los caracteres de la tabla ASCII ASCII en una terminal de operación de una computadora. Ello representa un salto cualitativo en en las estrategias usadas para el autoaprendizaje.

En los últimos veinte años se ha ido consolidando una definición cada vez más precisa de los conceptos implicados en la materia, como clasificación, aplicada sobre datos disponibles, normalmente asociados a eventos ya producidos, o inferencia, que se refiere más bien a predicción de posibles tendencias o de eventos futuros.

3.2 *MACHINE LEARNING*: SITUACIÓN ACTUAL

Actualmente el *Machine Learning* se clasifica como una de las manifestaciones que más ampliamente reciben el nombre de Inteligencia Artificial. Se centra en aquellas técnicas que pueden confrontar las posibilidades de inteligencia actuales del sistema computerizado con cantidades más o menos grandes de datos, extrayendo experiencia y mejorando esta inteligencia. Por analogía con las capacidades humanas, es coherente llamar “aprendizaje” a este proceso.

Los datos que ayudarán al sistema a mejorar, su formato, el tamaño de los conjuntos en que se presenta, admiten mucha variabilidad. Incluso pueden provenir sin modificar de eventos o procesos físicos y reales, o pueden estar artificialmente generados o modificados para hacer más eficiente el citado proceso de aprendizaje.

El entorno computerizado del cual pretendemos extraer un resultado beneficioso, y que pretendemos por tanto que aprenda, trabaja mediante un determinado algoritmo que está regido por un modelo estadístico de interpretación de la realidad. Será confrontado con los datos disponibles y las conclusiones que se arrojen serán retroalimentadas al algoritmo y mejorarán el modelo. Se puede esquematizar el proceso mediante el diagrama de la Fig. 1.

La eficiencia o rapidez con que el sistema aprende depende fuertemente de la manera en que esta retroalimentación se incorpora. Para ello se emplean algunos procedimientos muy arraigados, entre otros, en la estadística y el análisis numérico clásicos. Por el contrario, otros son aportaciones relativamente recientes de la muy abundante actividad investigadora que actualmente caracteriza a esta disciplina.

Así, por ejemplo, se emplean técnicas de regresión, optimización de descriptores respecto al error cuadrático y análisis de convergencia en la solución, que son aplicables a la matemática clásica desde hace décadas e incluso siglos.

Por otra parte, los árboles de decisión, las técnicas de vectores de soporte y las redes artificiales (neuronales o bayesianas) son, entre otras, técnicas recientes en las que se basa el aprendizaje de sistemas automáticos.

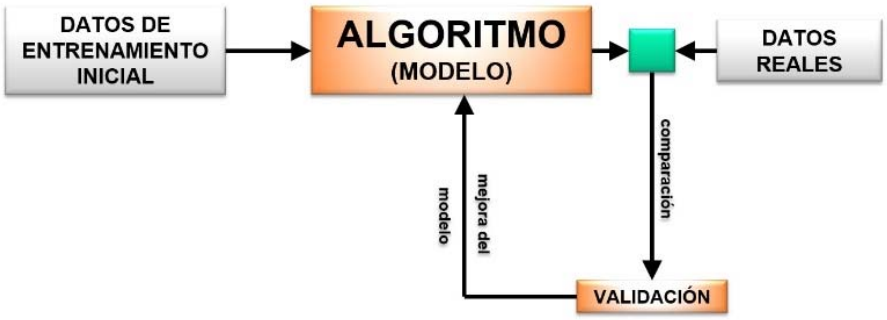


Fig. 1: *Ciclo de aprendizaje automático por corrección del algoritmo, con retroalimentación tras comparar con datos reales*

Se descarta para este trabajo incluir una explicación detallada de estas materias, sólo se mencionan como referencia general para el *Machine Learning* o en aquellos casos en que han sido usadas directamente en los supuestos prácticos de este Trabajo Fin de Máster.

3.3 PRINCIPALES RESULTADOS DEL MACHINE LEARNING: CLASIFICACIÓN E INFERENCIA

Para los sistemas automáticos con capacidad de aprendizaje se buscará que, al procesar los DS, proporcionen unos resultados útiles para comprender mejor los mecanismos que los producen y las tipologías de los mismos.

En este sentido, entre las actividades que los sistemas inteligentes permiten obtener como resultado al ser aplicados a los DS destacan dos, por el valor que aportan y lo amplio de su utilización: la clasificación y la inferencia.

La **clasificación** es el proceso por el que un sistema automático asigna categorías a un DS, atendiendo a unos determinados criterios. A menudo se habla, como sinónimos en la práctica de esas categorías, de etiquetación u objetivación.

Existen muchos aspectos relevantes en la manera de asignar categorías. Por ejemplo, a partir de la estructura de los datos de entrada en comparación con la de dichas categorías. Así, una categorización muy simple pero de gran importancia y uso muy extendido es la binaria: al sistema se le puede preguntar simplemente si determinado DS hace que la muestra entre o no entre en una determinada categoría. De ello puede haber ejemplos relativamente intrascendentes, como un decisor en base a datos recogidos de piezas de fruta, de si se escoge o no para empaquetado y consumo, o un evaluador, en base a las características de emails recibidos, de si son correo indeseado (*spam*) o no. Pero también puede haber ejemplos de gran trascendencia, incluso vitales, como sería predecir, a partir de la recolección de datos biomédicos en tiempo real, si se está produciendo o no un ataque cardiaco.

En [05] se muestran otros ejemplos de gran interés referidos a diferentes filosofías de

clasificación:

Multiclasificación: se puede clasificar en más de dos categorías. Por ejemplo, nuestra muestra de fruta se podría clasificar en calidad “alta”, “media”, “baja” y “desechar”.

Multietiquetado: una misma muestra puede recibir varias etiquetas diferentes. P. ej. una muestra de fruta “alta calidad” y “para zumo”.

Evaluación: el modelo de clasificación acompa su resultado con una indicación de la precisión de sus resultados, de manera que da una idea de la fiabilidad de dicho resultado.

La **inferencia** en *Machine Learning* se refiere a la capacidad de los procesos para hacer predicciones o sacar conclusiones elaboradas a partir de DS del mundo real, y por ende no procesados. Coloquialmente se habla de que inferir es “poner el aprendizaje automático a trabajar” y, de algún modo, es el valor añadido esencial que se espera obtener de los sistemas computacionales complejos con estas capacidades avanzadas.

En muchos casos, los conceptos que llevan a la inferencia parten de los previos de clasificación, pero añadiéndoles una elaboración que pretende emular las capacidades atribuibles de manera más clásica a la inteligencia humana, como pudiera ser la capacidad de abstracción.

Así, entre la multiplicidad de datos de entrada proporcionados al sistema automático, se puede hacer el ejercicio de clasificación binaria de si un paciente está sufriendo un infarto o no. Y esta clasificación puede hacerse de manera correcta. Pero yendo un paso más allá, el sistema puede procesar esos datos, aportarlos al modelo que implementa su algoritmo y de manera efectiva tener identificados elementos que caracterizan la salud del individuo, trascendiendo el “verdadero” o “falso” binario de si solamente nos preocupa si el ataque se va a producir.

Para ello los procedimientos que inciden sobre la mejora del modelo deben ser más sofisticado e incidir más sobre los aspectos predictivos.

3.4 ALGUNAS TÉCNICAS DE CLASIFICACIÓN: REGRESIÓN Y AGRUPAMIENTO

Existen diferentes estrategias para poder guiar de manera eficiente la gran capacidad de proceso de los sistemas de computación actuales hacia conclusiones que avancen hacia el Machuine Learning. Algunas de ellas se basan en conceptos matemáticos clásicos, desarrollados muchas décadas o incluso siglos atrás. Otros son relativamente recientes, surgidos en el ámbito de la actual gran actividad investigadora en casi todos sus campos.

Por la gran importancia que tienen para el tratamiento de grandes cantidades de datos, se van a describir con algún detalle dos de esos conceptos, uno de ellos clásico y otro de aparición más reciente. En ambos ha estado siempre clara la utilidad de su aplicación, sin embargo, el hecho de que existan ordenadores de cada vez mayor potencia, que realizan el trabajo repetitivo una vez se han planteado los bases del algoritmo correspondiente, multiplica su utilidad, la información procesada que podemos extraer (tanto en cantidad como en calidad) y las posibilidades de que el entorno automático donde se implementa, aprenda de la experiencia. Estamos hablando del método clásico de **Regresión Lineal**, propuesto ya por Gauss a principios

del s. XIX, y de los algoritmos de **Agrupamiento mediante “K – medias”**, que se comenzó a implementar hace algo más de cincuenta años.

Ambos procedimientos, desde su distancia temporal en la historia de la Matemática / Estadística, tienen sus características y causalidad común. El entorno que los motiva es el clásico, cuando se coleccionan ocurrencias de múltiples eventos de la realidad: el grupo de datos a estudiar puede presentarse en forma de “nube de puntos” más o menos numerosa, pero en cualquier caso irregular y aún no caracterizada, como suele corresponder a las realidades que se encuentran sin procesar en el mundo físico. Tomemos por ahora como ejemplo la de la Fig. 2.

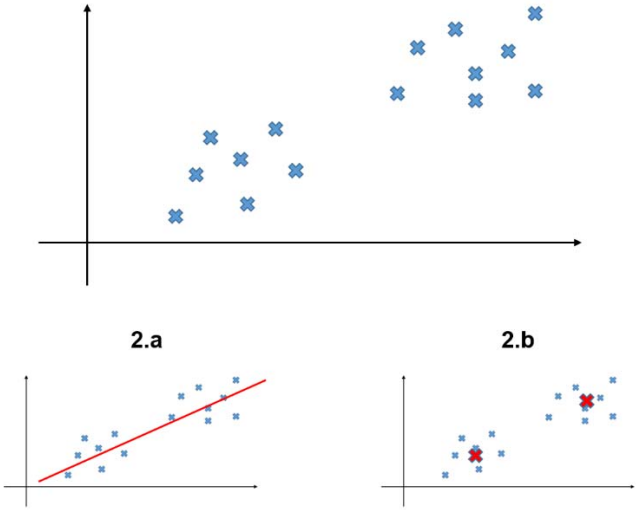


Fig. 2: *Algoritmos de tratamiento para nubes de puntos*

Nuestro propósito de modelización, como se ha comentado en 3.1.1, buscará asimilar tal nube a una cierta estructura matemática determinista.

Así, en el caso de la Regresión Lineal, se intentará con la que seguramente es la más simple que se podría aplicar: la línea recta, dentro del plano de aparición de los datos. Desde el punto de vista intuitivo está claro que se puede proponer la línea recta descrita en la Fig. 2.a como aproximadamente cercana y representativa de la totalidad del grupo de puntos.

El caso del Agrupamiento por “K-medias” es más sofisticado: se supondrá que la nube completa tiene algunas zonas de acumulación. De nuevo, la intuición lleva a reconocer que, al menos en el ejemplo de la Fig. 2, ello es así. Por eso en 2.b se proponen dos agrupamientos.

En ninguno de los casos se ha hablado de métodos analíticos para hallar los parámetros de la recta o las coordenadas de los agrupamientos correspondientes. Existe un modo de aproximación manual que para la apreciación humana une lo heurístico con lo intuitivo. Sin embargo el propósito de estas metodologías es proporcionar mejora de aprendizaje a sistemas no humanos, por lo que debe disponerse de un procedimiento analítico objetivo aplicable.

Se expone también brevemente la metodología de Análisis de Componente Principal

(PCA), que, en esencia, se propone identificar qué componentes dimensionales son los más densos en información estadística relevante, frente a otros métodos, que agrupan los eventos eventos estadísticos propiamente dichos.

A continuación se proporciona detalle adicional, para los tres métodos mencionados.

3.4.1 Regresión lineal

En este método el criterio de cercanía de la recta al conjunto de los puntos es el de que la suma de cuadrados de las diferencias recta – punto (en el eje de ordenadas) se minimice. Se puede ver esquemáticamente en la Fig. 3. Al basarse el cálculo en cuadrados de diferencias, y no importar por tanto el signo de las mismas, el método asegura un óptimo en el que se cumple el principio intuitivo de tener un número significativo de puntos tanto por encima como por debajo de la recta de regresión.

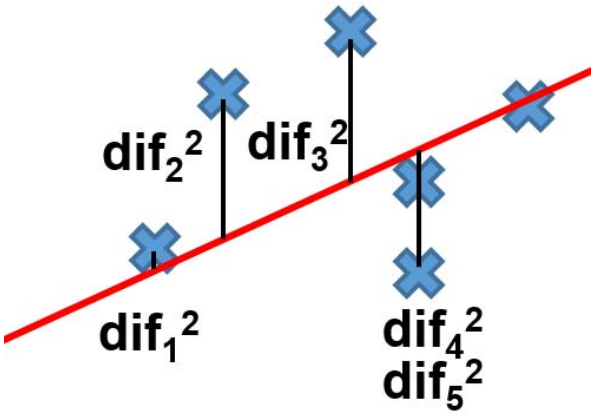


Fig. 3: *Diferencias cuadráticas en el cálculo de la regresión lineal*

En [03], citado también en 3.1.1, se ilustra un proceso sencillo de Machine Learning para optimizar la precisión de una recta de regresión a modo de ejemplo.

3.4.2 Agrupamiento mediante K-medias

Esta metodología se basa, tomando origen en su nombre, en que el algoritmo proponga que hay K puntos, llamados centroides, alrededor de los cuales se agrupan subconjuntos significativamente diferenciados de los puntos de la nube.

Existen diversos tipos de algoritmo para afinar la posición de los centroides. Se suelen basar en considerar K candidatos iniciales a ser centro de agrupamiento. Cada iteración de aplicación del algoritmo con la posición del centroide levemente variada se comprueba si el criterio de agrupamiento (p. ej. basado en la suma de distancias cuadráticas a los puntos) se hace más acentuado. Cuando transcurren unas determinadas iteraciones y la posición de los centroides se ha estabilizado, el problema de agrupamiento está resuelto. En la Fig. 4 se esquematiza este mecanismo de convergencia.

Obviamente podemos estar ante una nube de puntos que no tengan agrupamiento, o que el número K propuesto como hipótesis no responda a la realidad. En ese caso no se produce convergencia hacia posiciones estabilizadas de los centroides y el algoritmo no arroja un valor estable en K.

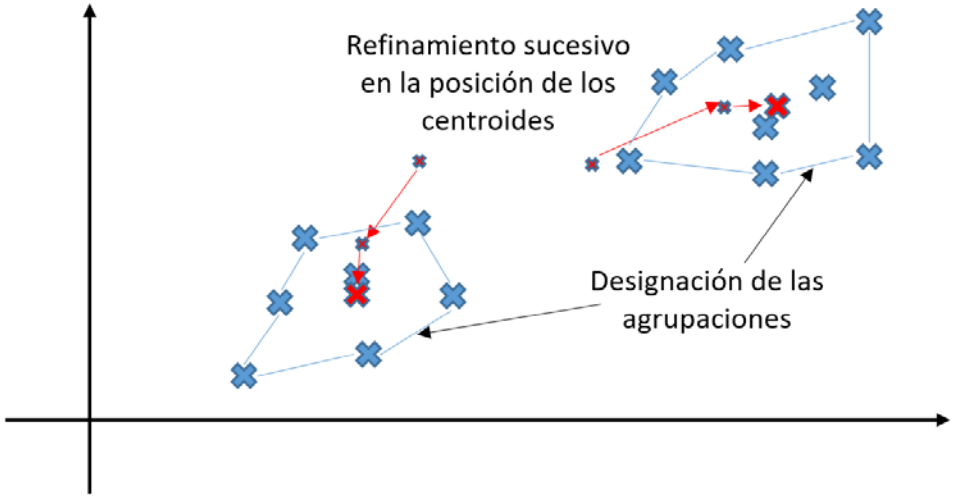


Fig. 4: *Determinación de la posición de los centroides. Ejemplo de agrupamiento “2-medias”.*

3.4.3 Análisis del Componente Principal (PCA)

Como se ha esbozado en 3.4, PCA se dirige a seleccionar dimensiones, más que puntos de la muestra de ocurrencias, que pueda ser más o menos extensa. Por ello no serán características de este análisis nubes de cientos o miles de puntos, sino listas de las diferentes variables usadas, que en el entorno de tablas o bases de datos más frecuentemente usado en la actualidad, se referirán a campos o columnas de las mismas.

Se proporciona en [06] una descripción de esta técnica, usando algunos ejemplos para comprender su potencial y aplicación adecuada.

Tras un estudio principalmente basado en el análisis de la varianza de la información, se calculan los autovalores de los diferentes componentes, “cuyo valor mide la cantidad de variación retenida por cada Componente Principal” (Kassambara, 2017). Después, algunas herramientas de representación presentan la información de forma que ayude a decidir qué componentes o variables se estudian más en detalle:

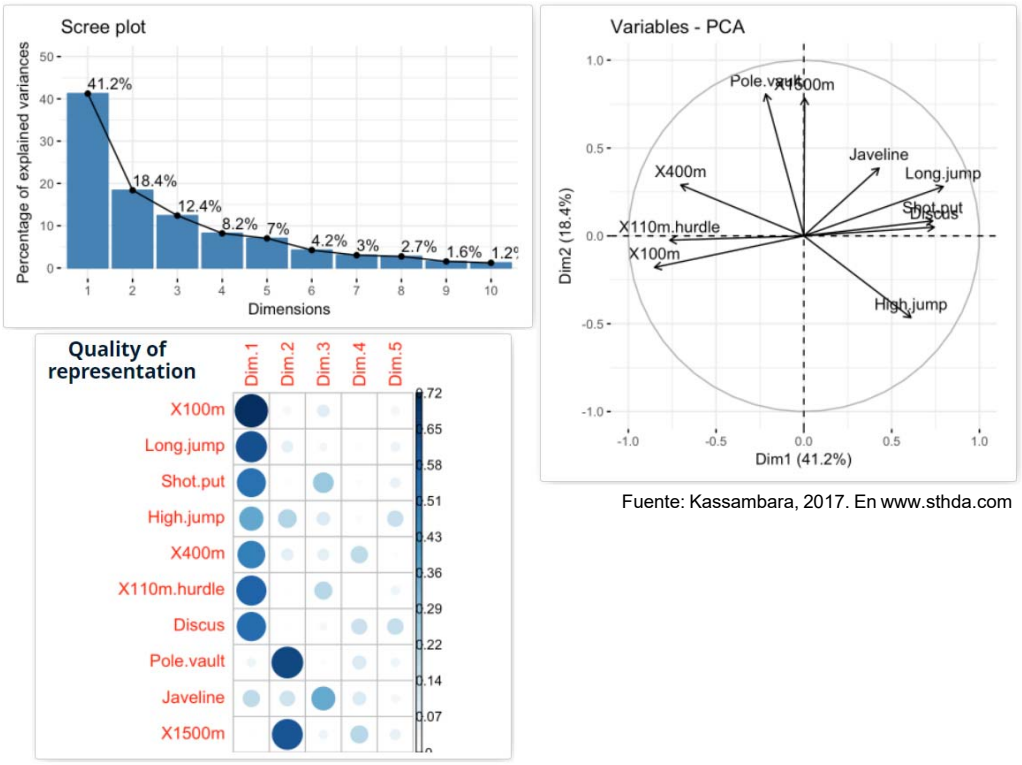
Gráfico de sedimentación: ordena todas las variables de mayor a menor relevancia en cuanto a si intervienen / explican la varianza observada en los datos. La situación ideal será aquella en que la relevancia de las variables sea alta en las dos o tres primeras y decaiga rápidamente en el resto.

Circulo de correlación: Esta herramienta de representación es polar (ángulo y distancia respecto al origen). Pero al mismo tiempo presupone que en los ejes cartesianos hemos escogido unas dimensiones “Dim1 / Dim2”, muy frecuentemente las que el gráfico de sedimentación nos ha marcado como más relevantes. Con ese entendimiento, la posición polar da una idea de si las variables / componentes están muy correlacionadas entre sí

(según su cercanía angular) y de si las Dim1/Dim2 escogidas representan bien la información que portan (distancia desde el origen).

Calidad en la representación: Este gráfico confronta en forma de matriz las componentes de nuestro DS con las dimensiones que usamos para cuantificar. Dará idea de en qué dimensiones se representará mejor la información de las componentes.

En la Fig. 5 se muestran ejemplos de estas representaciones puestos a efectos ilustrativos por el artículo de referencia [06]. En 5.2.3 se muestran las que se han aplicado a ejemplos específicos de este Trabajo.



Fuente: Kassambara, 2017. En www.sthda.com

Fig. 5: *Ejemplos de herramientas de representación de Análisis de Componente Principal*

Este análisis es útil para seleccionar, en una serie de datos con multiplicidad de variables, en cuáles merece más la pena centrarse. Un ejemplo es el de las representaciones gráficas, ya que, cuando hay más de tres variables analizadas, es difícil que su representación arroje conclusiones fáciles de observar intuitivamente, y aun eso plasmando representaciones tridimensionales en perspectiva en el plano del papel.

En esas condiciones, el dar formato previo a los datos dejando disponible para representación un conjunto de pocas dimensiones, resulta de una utilidad evidente.

3.5 ENTORNOS DE IMPLEMENTACIÓN

Como reflexión inicial en este aspecto diremos que el tipo de sistemas automáticos que el

Machine Learning pretende dotar de la capacidad de autoaprendizaje es de naturaleza computacional. Por ello, en un principio, no se diferencia de manera radical de cualquier plataforma informática de propósito más general, aunque como en cualquier campo en que tecnologías son aplicables (siendo hoy día casi imposible encontrar una actividad humana que los computadores no tengan cabida) cuanta mayor es la potencia de los sistemas desplegados se consiguen resultados más notables.

Ya se ha mencionado la estrecha relación entre el *Machine Learning*, el manejo de DS de diferentes tamaños, pero abundando los de gran magnitud (que entre otros detalles, son los que dan nombre a este TFM) y la estadística. Por ello, la combinación de una plataforma *hardware* de suficiente potencia y un *software* flexible en cuanto a lenguaje de programación y dotado de funcionalidades avanzadas de cálculo estadístico será un entorno muy adecuado para depurar procedimientos de aprendizaje y finalmente programar aplicaciones que sean capaces de autoaprender. Se repasan someramente dos de los entornos más popularmente utilizados al día de la fecha en esta disciplina, los lenguajes R y Python.

3.5.1 R – Project for Statistical Computing

El Proyecto R para Computación Estadística recibe de forma coloquial entre los usuarios simplemente la denominación de “R”. Se trata de un entorno de desarrollo y programación basado en software libre y en código abierto. Se apoya en las licencias de GNU GPL, muy conocidas por su relación con el sistema operativo Linux. No obstante al nutrirse, debido a su carácter abierto, de las contribuciones de una multitudinaria comunidad de programaciones, existen multitud de librerías y aplicaciones para usarlo en muchas plataformas. Destaca la posibilidad de usar ficheros .xlsx de Excel que, de manera obvia dada la generalidad de utilización del paquete Office®, es una posibilidad muy frecuente de asociación.

En lo que respecta a las características intrínsecas de R como lenguaje, éste es de tipo interpretado, lo que le confiere una gran simplicidad y rapidez de uso, al no necesitar tiempos de compilación. R no se perfila como un entorno óptimo para gestionar bases de datos complejas desde un punto de vista relacional y en entornos de red distribuido, sin embargo tiene desarrolladas un gran número de herramientas estadísticas que hace sencillo el análisis de los datos en ese ámbito.

En el aspecto de programación de aplicaciones, las posibilidades de utilización de R en su entorno de consola simple como lenguaje interpretado se potencian en el entorno de Proyectos R-Studio, que permite general aplicaciones más complejas y modulares.

Como lenguaje específico de estadística, R es capaz de confrontar nubes de datos aleatorios con numerosos tipos de distribución, desde las más básicas, como la uniforme, exponencial, normal (gaussiana), binomial, hasta otras descriptivas de situaciones aleatorias más específicas, como la T, Gamma, Beta, etc. Sin olvidar las de Weibull, lo que supone una gran potencia como herramienta cuando se enfrentan situaciones y entornos relacionados con la fiabilidad.

R está capacitado para trabajar con los parámetros más característicos de cada uno de esos tipos de variable aleatoria, en lo que es la caracterización usual de las mismas, como sus valores de promedio, media, mediana, varianza, momentos de orden determinado, etc.

Para cerrar el ciclo de caracterización que el lenguaje ofrece, existe un conjunto muy nutrido de funciones gráficas que posibilitan mostrar con claridad los resultados de los análisis y facilitar las posibles interpretaciones de sus resultados, para así aplicar una retroalimentación hacia las propias herramientas de análisis y hacer posible el aprendizaje autónomo.

3.5.2 Python

Gran parte de lo descrito para R, en cuanto a características generales, es aplicable también a Python, por lo que no se repetirá en detalle. Igual que R, Python es también un lenguaje interpretado que no necesita compilación. La simplicidad de instalación y uso es una de sus características principales.

La orientación a objetos es un concepto avanzado de gran utilidad en programación. Permite aumentar de manera muy eficiente la eficiencia, ya que en multitud de ocasiones permite reutilizar bloques de código, sólo con unas pocas referencias que hagan compatible esas secciones de programa ya desarrolladas con los objetos que encapsulan la información. Python está de manera fundamental planteado para la Programación Orientada a Objetos (OOP), sin embargo R también ofrece unas prestaciones adecuadas para seguir esta filosofía de desarrollo de programas.

Esas prestaciones se sustancian en potentes librerías adicionales que dotan al entorno de una gran capacidad de manejo de estructuras, como Numpy, que es la principal librería que dota a Python de funcionalidades de manejo de variables vectoriales sobre las que se puede aplicar toda la variabilidad de funciones estadísticas.

Una vez procesada toda la información y el cálculo de dichas funciones en entorno estadístico, las librerías MATPLOTLIB y SEABORN son capaces de representar la información gráficamente con un rango de funcionalidades muy amplio.

4 ENTORNO PARA ANÁLISIS BÁSICO APLICADO A MACHINE LEARNING

Como se describe el el punto 2.1 de “Objetivos” para este TFM, uno de los propósitos del mismo será diseñar una aplicación o programa informático simple donde se puedan probar algunos de los conceptos básicos del aprendizaje automático. Para ello se va a utilizar el entorno de programación con R. Este lenguaje ofrece la posibilidad de utilizar multitud de funcionalidades de análisis estadístico, dentro de las prestaciones matemáticas que ofrece, todo ello embebido en un lenguaje de programación convencional interpretado que puede aplicarse sobre listas de datos en formato Excel, que es el que se utilizará como entrada.

4.1 ESPECIFICACIONES PRINCIPALES DEL SISTEMA

Se comenzará definiendo una aplicación de baja complejidad, que sea capaz de extraer conclusiones, principlamente basadas en un análisis matemático / estadístico, e incorporarse a los propios procedimientos de análisis para así autoaprender. Se comenzará definiendo figuras sencillas a partir de los datos agregados, pudiéndose definir correlaciones de mayor complejidad

a demanda del usuario. Se podrá escoger modos gráficos de representación que ayuden a la identificación de las relaciones entre las series de datos, resaltando las relaciones causales.

El resumen de las especificaciones planteadas se describe a continuación:

- Capacidad de admitir información en series, en formato Excel y diferentes tipos de datos.
- Extracción de parámetros estadísticos de las series o nubes de puntos referidas: media, mediana, moda, varianza, momentos de diverso orden, en función del tipo de variable aleatoria asimilable.
- Herramientas gráficas de representación, como ayuda a la extracción de conclusiones.

4.2 IMPLEMENTACIÓN DEL SISTEMA EN R

Con ayuda de las prestaciones de R se establecen algunas funciones sencillas, con un objetivo esencialmente didáctico y de demostración básica.

El sistema puede leer un DS y aplicarle determinadas fórmulas estadísticas a voluntad. Se ha puesto especial interés en que la aplicación pueda llevar a cabo agrupamiento (*clustering*) de los datos como el descrito en 3.4.2, confiando en que la potencia de esta metodología tanto en el dominio analítico como en el gráfico será una ayuda para caracterizar la información y posibilitar la clasificación de la misma.

5 CASOS PRÁCTICOS DE ANÁLISIS DE *MACHINE LEARNING*

La aplicación de las capacidades del *Machine Learning* a situaciones reales tiene ventajas evidentes. Por un lado permite extraer beneficios que solucionan problemas reales en ámbitos relacionados con programas y proyectos, ámbitos en los que el análisis estadístico y matemático son idóneos para aportar un gran valor explicativo del comportamiento de grandes cantidades de información, y la retroalimentación del autoaprendizaje hace avanzar el desempeño de las plataformas computerizadas que se busca sean expertas, ajustándolas a que puedan dar un servicio cada vez más robusto y tangible.

No obstante lo anterior, aplicar el análisis sobre DS artificialmente generados tiene un buen número de ventajas, en contextos que le son propios.

Cuando se genera un DS en un entorno numérico controlado, el control que se tiene sobre sus características, distribución estadística, muestras representativas y, en general, sus descriptores tanto determinísticos como estocásticos, puede ser completo. Cuando menos, todo lo acotado que el operador desee.

En esta situación, los DS que se pueden proporcionar como entrada a los algoritmos

funcionan como especímenes de laboratorio: la eficacia y eficiencia con que el sistema automático, al clasificar e inferir, logra identificar los sesgos que han sido previamente introducidos de manera consciente en las series, proporciona un criterio de la bondad del proceso proceso de aprendizaje del algoritmo.

Para el presente trabajo se ha tenido acceso a una información real de serie de datos, relacionada con el flujo logístico de reaprovisionamiento de determinadas piezas de repuesto para el mantenimiento para una plataforma aeronáutica. De esta manera, se conoce ya la estructura de los datos que tendrá esa información.

Por otra parte, según se ha visto, la estrategia de probar las herramientas de análisis con series de datos simuladas, con características conocidas, puede potencialmente arrojar beneficios claros: observar los resultados que las herramientas ponen de manifiesto se puede correlacionar con los sesgos y peculiaridades de la información que el usuario ha generado de manera consciente en esas simulaciones.

Por tanto, en los apartados siguientes se trabajará con dos series de datos que tendrán la misma estructura, y por tanto la herramienta software de análisis puede recibirlas como entradas de manera equivalente. La diferencia sustancial será que una es información sintetizada y limitada a un volumen pequeño, de información de muestra. La otra, mucho más masiva, es la que procede de un largo periodo de actividad real logística.

Se espera que el análisis comparativo del resultado de ambos análisis ayude a identificar características y a interpretar los datos del caso real. En cualquier caso se va a asumir que una técnica de clasificación, como puede ser el agrupamiento por K-medias, descrito en el punto 3.4.2, puede ser el punto inicial de obtención de información de caracterización del algoritmo y su mejora.

5.1 SIMULACIÓN DE UN CONJUNTO DE DATOS: TIEMPO LOGÍSTICO DE SERVICIO DE MATERIAL

El primer DS hace referencia al tiempo logístico de reaprovisionamiento de diversos tipos de piezas de repuesto usadas por una plataforma aeronáutica. Cualquier aeronave operativa necesita la facilitación de un gran número de componentes, para hacer frente a las actividades de mantenimiento, tanto del tipo de las programadas (normalmente asociadas a inspecciones periódicas descritas en el manual de mantenimiento) como no programadas, y por tanto imprevistas. Estas últimas suelen estar motivadas por averías sobrevenidas o por cualquier discrepancia o circunstancia similar.

En cualquier caso, el flujo logístico de piezas y materiales está tradicionalmente ligado al mantenimiento aeronáutico, permitiendo en algunas situaciones una planificación a largo plazo y otras veces necesitando reaccionar a necesidades rápidas. Los conceptos clásicos en logística de metodologías de flujo, nivel de reposición y control de stocks están presentes en el sector aeronáutico de manera similar a otros en los que también existe un control logístico. Todo ello se reflejará en la manera en la que se va a simular un caso hipotético asociado a varios materiales.

El procedimiento de designación de tipo de pieza necesaria para ser usada en apoyo de

mantenimiento como el descrito se basa en el concepto de “número de pieza” (*Part Number* – P/N). Un P/N identifica a cualquiera de las múltiples piezas individuales que son funcionalmente equivalentes entre sí para cumplir un cometido, dentro del funcionamiento de los sistemas de la aeronave. Aunque para ser designadas por el mismo P/N no se exige que todas las piezas de la población sean absolutamente idénticas, lo normal es que sean muy similares, ya que todas realizan la misma función. Se suele hablar de que tienen una similitud ‘FFF’: *Fit, Form & Function*. A cada pieza concreta fabricada, si por alguna razón es necesario realizarle un seguimiento individualizado, se le asignará también un “número de serie” – S/N. En principio no tendrán S/N elementos de gran simplicidad, o que no tienen una unidad de suministro discreta, o que tienen una naturaleza esencialmente efímera o consumible (p. ej. una arandela, metros lineales de cable o un envase de aceite lubricante). En cambio, elementos de mayor trascendencia, que convenga tener controlados en cuanto a ubicación, condición de funcionamiento o configuración, son generalmente seriados.

Los principales campos de la base de datos del DS bajo estudio se describen a continuación. En algunos casos se expone una valoración cualitativa de la importancia que tendrá su contenido en lo que respecta a aportar información distintiva del proceso que se quiere caracterizar. Los campos que no se mencionan tienen un papel de gestión meramente administrativa o no son cuantificables.

- Fecha de petición: aquella en la que surge la necesidad del material y se solicita el aprovisionamiento de la pieza.
- Fecha de despacho: aquella en la que la pieza se sirve al centro de trabajo. En muchos casos coincide con la anterior, ya que existe un almacen local y, si en el mismo hay existencias, la pieza se suministra de forma inmediata.

Se genera una variable intermedia [F. Despacho] - [F. Petición], que será nuestra métrica del número de días que tarda en servirse la pieza. En el caso comentado de suministro inmediato, esta variable valdrá cero.

- Part Number (P/N). Permitirá la trazabilidad acerca de qué piezas serán las que tengan alguna característica distintiva sobre su dificultad o no en ser servidas, o se agrupen de cualquier otra manera que sea útil individualizar.
- ML: Nivel de Mantenimiento (Maintenance Level). Permitirá caracterizar si este ML tiene una correlación con los rasgos característicos comentados.
- Cantidad: la referida al número de piezas solicitado en la petición. En principio parece obvio que resultará más complicado servir pedidos por una gran cantidad de piezas, que por pocas, o una, unidad.

5.1.1 Hipótesis de simulación de conjunto de datos

Se va a generar la simulación de peticiones de seis tipos (Part Number – P/N) diferentes de componente, necesitados en actividades de mantenimiento. Se cumplirán las siguientes condiciones:

- Se pondrán órdenes de compra en cantidad variable. Mientras la mayoría tendrán

una cadencia regular, otras serán no programadas.

- Se establecerá una relación entre petición en número elevado y dificultad en satisfacer la cantidad deseada.
- Se introducirá algún evento aleatorio que dificulte la fabricación.
- Se supondrá que existe alguna fuente alternativa de adquisición, con un periodo de carencia en la activación de la fabricación.

5.1.2 Series de datos simuladas

Con los condicionantes mencionados en el apartado anterior, se genera una serie de datos, correspondiente a un histórico de pedidos de un número de 10 P/Ns. De cada P/N se han supuesto 10 eventos de solicitud de material. Por ello, la serie de datos comprende un total de 100 eventos.

Los campos o columnas más característicos desde el punto de vista de cálculo numérico que se usarán para este supuesto se refieren a la mecánica típica de un proceso de acopio de material y servicio logístico, y se han descrito al inicio de este subapartado. La información que se ha supuesto en este caso se muestra a continuación, con una pequeña explicación cuando ella es necesaria:

- Fecha de pedido: normalmente es el momento temporal desde el que comienzan a contar tiempos logísticos, p. ej. los que tienen consecuencias contractuales para el proveedor.
- Cantidad pedida
- Fecha de entrega: puede ser fijada por contrato, como incremento a contar desde la fecha del pedido, o bien el proveedor realiza una estimación cada vez.

5.1.3 Inspección preliminar de datos. Estrategia de clasificación o inferencia

Se pueden intuir algunas características en la eficacia del aprovisionamiento de piezas en relación al tipo de demanda y la mecánica de los pedidos. Lo normal es que una demanda regular de piezas permita a los centros de manufactura del proveedor planear su actividad de fabricación y eliminar o mitigar problemas en la misma.

Sin embargo, la fabricación es sólo una de las actividades de la cadena de suministro, e incluso estaría en discusión si es la más importante o no de ellas, en virtud de un enfoque simplista de que, si determinado material existe físicamente fabricado en las instalaciones del proveedor, ubicarlo en las del cliente es comparativamente una tarea menor. Existen multitud de fases, relativas al vinculo contractual, a licencias de uso / exportación / importación, caracterización documental, de Calidad (más la de Aeronavegabilidad, en el ámbito aeronáutico), transporte, almacenamiento, etc. que se deben asegurar para que el cliente disponga del material en sus instalaciones, en condición útil para su uso.

En cualquier caso, para este conjunto de datos se puede esperar encontrar que los pedidos realizados fuera de la periodicidad atribuible a un uso regular de la flota de aeronaves, o si por alguna razón se realizan pedidos por un número de piezas mayor al acostumbrado, existirán en

general mayores dificultades para satisfacer dicho pedido, lo que se debería notar en un mayor tiempo logístico para aprovisionar y también en una mayor frecuencia de pedidos fragmentados (el material no se recibe de una sola vez).

5.1.4 Análisis de datos

La aplicación en R descrita en 4.2, tiene como principal propósito implementar y analizar agrupamientos en K-medias (*K-means Clustering*). Se ha suministrado como entrada el fichero simulado objeto de este apartado 5.1 y el resultado es la representación gráfica de la Fig. 6.

Como primera apreciación, debe decirse que la interpretación de la nube de puntos, así como las conclusiones que se pueden extraer de su representación gráfica, de su agrupamiento y, por tanto, del intento de clasificación asociado, no son triviales. Esto último no es sorprendente, dada la naturaleza compleja de la información logística, si se quiere tener toda ella en cuenta para decidir qué partes tienen mayor relevancia.

El diagrama es el resultado de proponer como hipótesis inicial un *clustering* de 4 agrupamientos. El algoritmo *4-means* da como resultado una concentración algo definida, pero con solapamiento entre dos de los agrupamientos.

También llama la atención el hecho de que uno de los puntos está tan alejado del resto que ha adquirido carácter de *cluster* por sí solo, ya que esa distancia hace difícil que comparta propiedades métricamente cercanas al resto, según los criterios que sigue el algoritmo KMEANS().

Se ha querido llevar a cabo un análisis más detallado de lo que puede significar este punto: la interpretación es que posiblemente esté representando la superposición de dos (2) puntos de un mismo P/N simulado en la serie (PN007), para el cuál se ha pensado un retraso extremo del plazo de servicio de la correspondiente pieza: 120 días. Ello llevaría a la interpretación de que en el diagrama x-y la ubicación de puntos a la izquierda está asociado a grandes retrasos en el suministro de piezas, mientras que que la parte derecha de la gráfica representa a los eventos que se han llevado a cabo con poco o ningún retraso.

5.1.5 Conclusiones del caso

El análisis K-means de una población de eventos logísticos teóricos (lista simulada) nos permite, en primera hipótesis, clasificarlos en tres *clusters* y ponerlos en relación con el tiempo logístico que han tardado en servirse. Podríamos hablar de un primer grupo de “puntuales”, otro de “intermedios” y otro de relativamente “impuntuales”. Existiría un caso aislado (o dos, según interpretación) en el que el retraso en el servicio es extremo, de forma que atrae para sí un *cluster* propio.

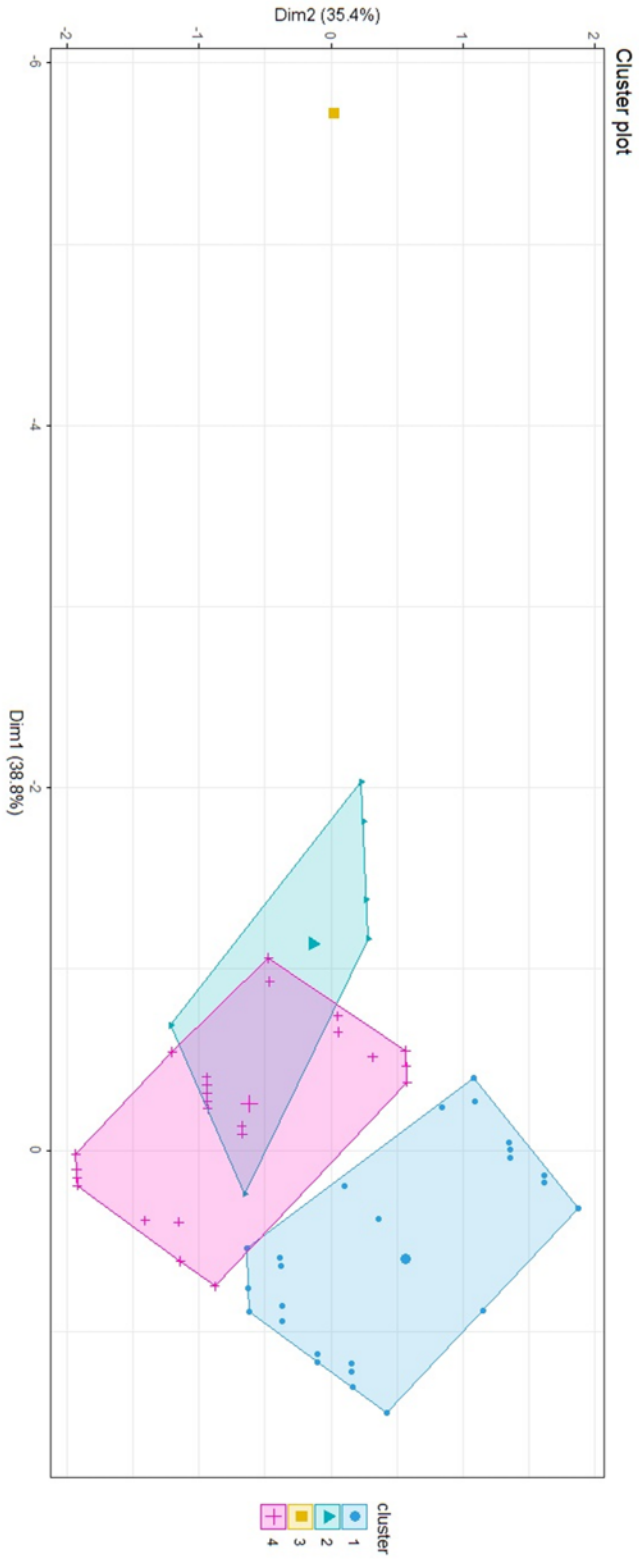


Fig. 6: *Análisis de Agrupamiento en K-Medias para serie de datos simulada*

5.2 UN CONJUNTO REAL DE DATOS: TIEMPO LOGÍSTICO DE SERVICIO DE REPUESTOS

Se dispone de un DS recogido en una actividad real de aprovisionamiento logístico, para una plataforma aérea. En esencia y con la información en bruto, se dispone de la lista de peticiones de piezas en un lapso de tiempo de aproximadamente 38 meses, entre las fechas de 19/03/2019 y 30/05/2022.

5.2.1 Series de datos de trabajo

Los datos han sido suministrados mediante una sola lista de información sin procesar. Entre los campos o columnas de la tabla que se procesará, hay algunas de carácter textual o alfanumérico, con las cuales es inicialmente difícil alimentar cálculos matemáticos o estadísticos, ya que van más dirigidos a dar información administrativa a los gestores logísticos de las secciones de apoyo a la aeronave implicada.

Como sugerencia para futuras líneas de investigación sobre este tipo de *Data Sets*, de los campos que son puramente textuales se podría extraer información convertible a métricas numéricas. P. ej. en los campos de observaciones se podrían aplicar rutinas de reconocimiento de texto y contabilizar si aparecen términos de relevancia habituales en mantenimiento aeronáutico y logística, como “rotura”, “grieta”, “pérdida”, “sustitución”, etc. Sin embargo, la complejidad software necesaria para ello lo deja fuera del alcance pensado para este TFM.

5.2.2 Inspección preliminar de datos. Estrategia de clasificación o inferencia

A fin de que se pueda extraer un análisis comparativo de con el apartado anterior, se ha optado por el mismo tipo de estrategia: se observará si un Agrupamiento 4-Medias registra un comportamiento comparable sobre una serie que en este caso contiene más de 3.700 eventos logísticos.

Por supuesto, no debe esperarse que un conjunto de datos de origen real y de tamaño masivo se comporte como nuestro primer ejemplo, ni debe considerarse una disfunción del método el hecho de que se manifiesten diferencias. Por el contrario, éstas deberían ser estudiadas y encuadradas en el comportamiento del algoritmo.

5.2.3 Análisis de datos

Análisis de Agrupamiento K-Means (K = 4)

Al suministrar la nueva serie de datos masivos a la aplicación parametrizada para ajustarse a un Agrupamiento 4-Means se ha obtenido la salida gráfica de la Fig. 7.

Antes de evaluar si, en efecto, el algoritmo ha individuado de manera efectiva 4 grupos, se pueden mencionar algunas ideas preliminares. Tomando las conclusiones más obvias del caso de serie de datos simulada, podemos asumir que los puntos de más a la izquierda se refieren a los eventos logísticos de mayor retraso en recibir los materiales.

No son los únicos significantes gráficos que se configuran, como también es de esperar

cuando hay más de 3.000 puntos desplegados en el diagrama: la aparición de los segmentos de puntos, casi fusionados, que aparecen en varios lugares de la representación hacen referencia seguramente a los diferentes P/N en los que se divide la representación. De algunos de esos P/N hay más ejemplos y de otros menos, como se puede ver en el fichero de información no procesado. Ello es coherente con que la longitud de dichos segmentos es muy variable. De algunos de los P/N hay sólo 1, ó en todo caso muy pocas, apariciones, lo que correspondería a la multiplicidad de puntos aislados.

Análisis de Componente Principal (PCA)

El agrupamiento obtenido sobre los datos reales en bruto ha proporcionado poca información gráfica interpretable. Por ello se va a aplicar la metodología PCA, que en esencia se puede considerar un preformateo o preparación de las variables sobre las que aplicar otros métodods, por lo que no ees excluyente de aplicar de nuevo K-Means. A este respecto, se analizará si aplicar valores diferentes de 4 mejora la interpretabilidad.

Adicionalmente, se añaden al análisis algunas otras columnas de la serie de datos, de manera que pasan a ser las siguientes cinco (5), las variables consideradas:

- Descripción de la pieza: Campo de texto. Puede contener expresiones del tipo “arandela”, “módulo compresor”, “motor”, etc.
- Origen: La pieza procede del stock local o de alguno de los proveedores industriales
- Condición: Útil / reparable / Para baja / Otras
- Dif. fechas ent / sal: Días que ha tardado en servirse la pieza
- Tipo de Mantenimiento: Nivel de Mantenimiento 1 ó 2 (ML1 / ML2), o bien otro, generalmente asociado a reparaciones imprevistas.

En estas condiciones, se ha pasado el DS por el algoritmo de PCA y éste ha caracterizado las variables de la siguiente manera, reflejada en la Fig. 8:

El Gráfico de sedimentación no concentra dimensionalmente la información todo lo que sería deseable. Habría que tomar 4 dimensiones para acumular un 85% de la varianza en la información. En Círculo de correlación, la correlación mutua está bastante dispersa entre variables, sin embargo sí da idea de solidez el hecho de que los “Días Ent/Sal” (tiempo que tarda la provisión de la pieza) está bien representada, básicamente en la Dim3 (gran distancia al origen y alineación con el eje horizontal).

La información anterior se complementa con la del gráfico de Calidad en la representación: confirma que “Días Ent/Sal” (que es nuestro campo logístico de máximo interés) concentra casi todo su peso en la Dim3, lo cual nos da pistas para un segundo análisis.

De todas formas se ha hecho un K-Means sobre los datos con este formato, representando Dim2 y Dim3 en el plano. Se ha escogido K=10 para, de algún modo, admitir que el algoritmo puede solapar y amalgamar varios *clusters* que nos darán poca información, pero al elevar el número de K le damos la oportunidad de que extraiga algunos de pequeño tamaño de la nube general. Y en efecto, en la Fig. 8 se puede ver como el *cluster* 3 y, parcialmente el 4, se separan del resto y pueden ser tratados de manera aislada. Sus valores de Dim3 son más extremos, lo que los relaciona con pedidos que tardan muchos días en servirse.

Para el segundo análisis PCA al que nos referimos, seguiremos los siguientes criterios:

- Convertimos la variable “Descripción de la pieza” en “Primera palabra del campo Descripción de la pieza”, a fin de eliminar ruido en el reconocimiento del texto.
- Observaremos las gráficas PCA y elegiremos las dimensiones DimN relevantes.
- Repetiremos el algoritmo de Agrupamiento 10-Means.

La caracterización de variables pasa a ser la reflejada en las gráficas de la Fig. 10. En resumen:

- El Gráfico de sedimentación no cambia sustancialmente. Ningún par o tríada de dimensiones concentra la variación de información.
- El Círculo de correlación sigue reflejando una cierta correlación entre “Origen” de la pieza y “Días fecha ent/sal”. Lo significativo es que esta gráfica y la Matriz de Calidad consolidan la relevancia de esas dos variables (y sugieren que las otras tres no lo son tanto) y la importancia de la Dim3 y 4. Esas dimensiones serán las que se empleen para un nuevo *clustering* 10-Means, que se muestra en la Fig. 11.

En este segundo análisis PCA, la interpretabilidad de las gráficas y su posibilidad de relacionarlas con rasgos visualizables en la realidad se ha incrementado en alguna medida. Se ha confirmado que Dim3 está relacionada con el retardo logístico en servir la pieza, con lo cual las ocurrencias de los *clusters* 3 - 4, en la Fig. 9, y 6 - 10 en la Fig. 11, serán los más críticos para minimizar la carencia de material en la actividad de mantenimiento. Esa información puede servir para extraer experiencia en nuestra organización o para exigir un mejor cumplimiento del contrato de suministro a nuestros proveedores.

5.2.4 Conclusiones del caso

En este caso, confeccionado con datos brutos recuperados de un entorno real, el algoritmo de Agrupamiento también arroja un primer resultado de 4 *clusters* o grupos. Dichos clusters están todavía más solapados que en el caso de datos teóricos. En en principio ello no debe invalidar el análisis, pero de todas formas se ha recurrido al uso de una nueva metodología que previsiblemente ayudará a la interpretación. También para reforzar la vertiente descriptiva de técnicas de este Trabajo.

El Análisis según Componentes Principales (PCA) permite, entre otras cosas, pasar del concepto de variables o campos / columnas de la información, al más general de Dimensiones. Se ha identificado la dimensión más relacionada con la tardanza en el servicio y se han ejecutado nuevos agrupamientos que resaltan *clusters* de pedidos afectados por dicha tardanza, lo que puede ayudar al diagnóstico de debilidades en nuestra cadena de suministro.

Se aprecia que lograr una interpretabilidad de las representaciones de los DS, además de no ser una tarea sencilla, no debe ser un objetivo que se persiga con excesiva impaciencia desde los primeros pasos del análisis. La multiplicidad de herramientas disponibles para *Machine Learning* posibilitan pasos intermedios analíticos o de cambios en el formato de los datos, que muchas veces no parecen dar una intuición inicial, o que se encuentran tras algunas iteraciones de prueba – error.

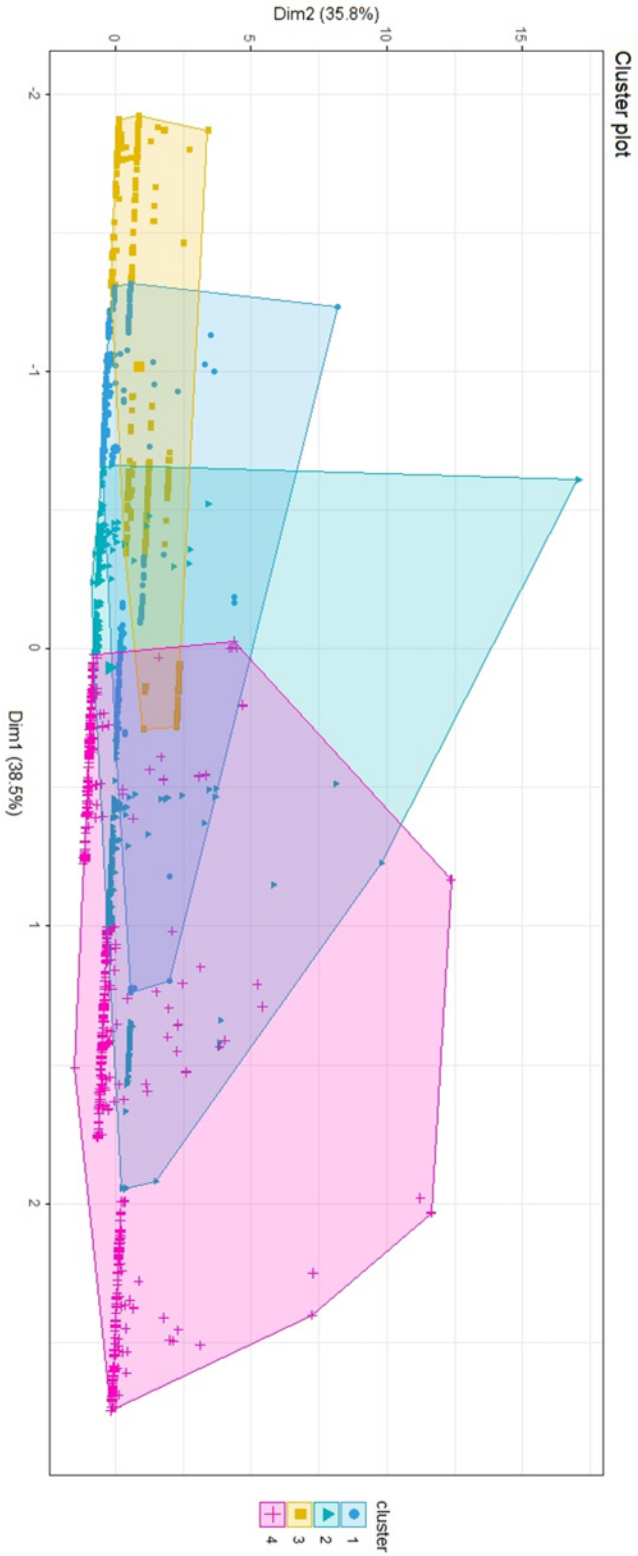


Fig. 7: *Análisis de Agrupamiento en K-Medias para serie de datos masiva real*

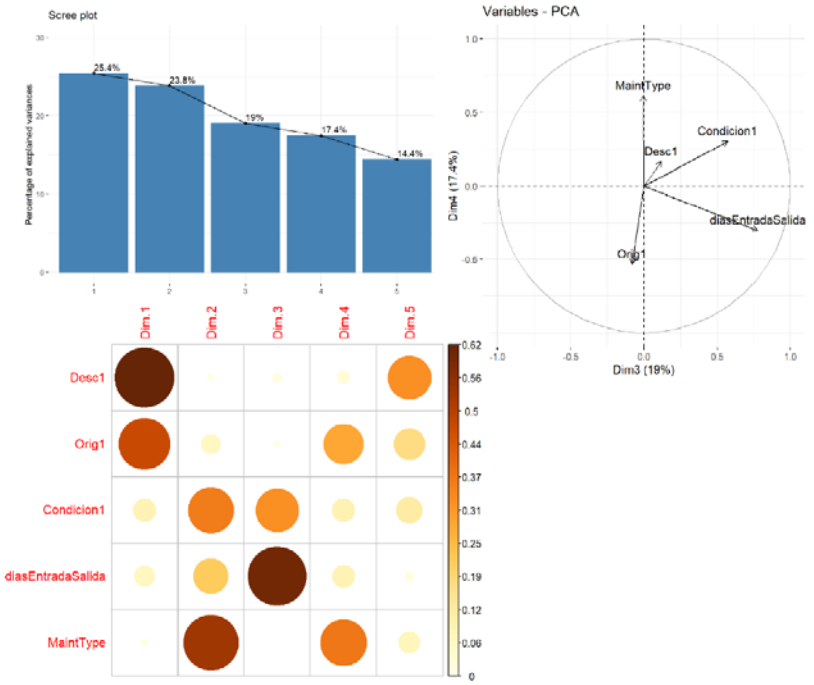


Fig. 8: *Primer Análisis de Componentes Principales sobre serie de datos masiva real*

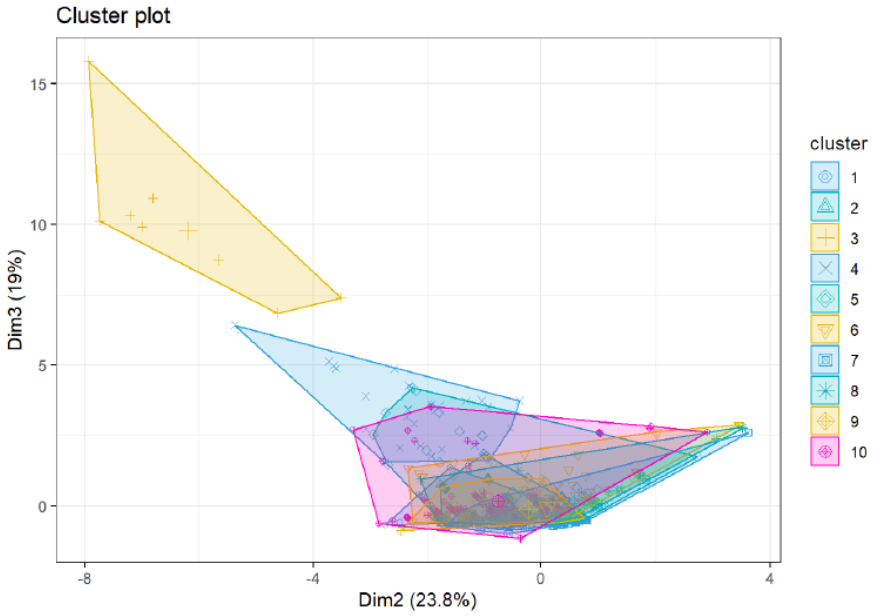


Fig. 9: *Agrupamiento 10-Means correspondiente al primer análisis PCA*

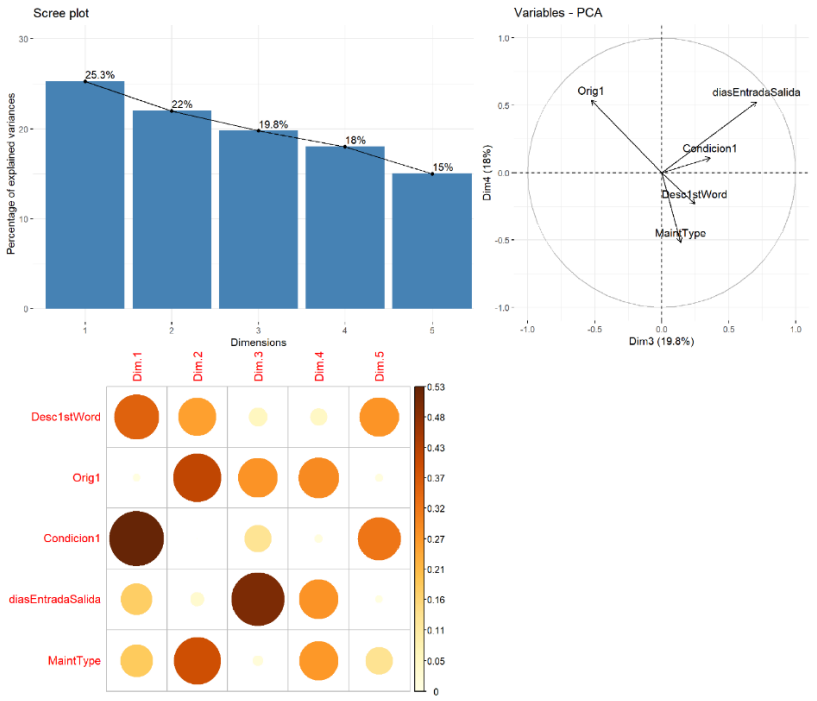


Fig. 10: *Segundo Análisis de Componentes Principales sobre serie de datos masiva real: modificación del campo “Descripción” y reinterpretación del las dimensiones*

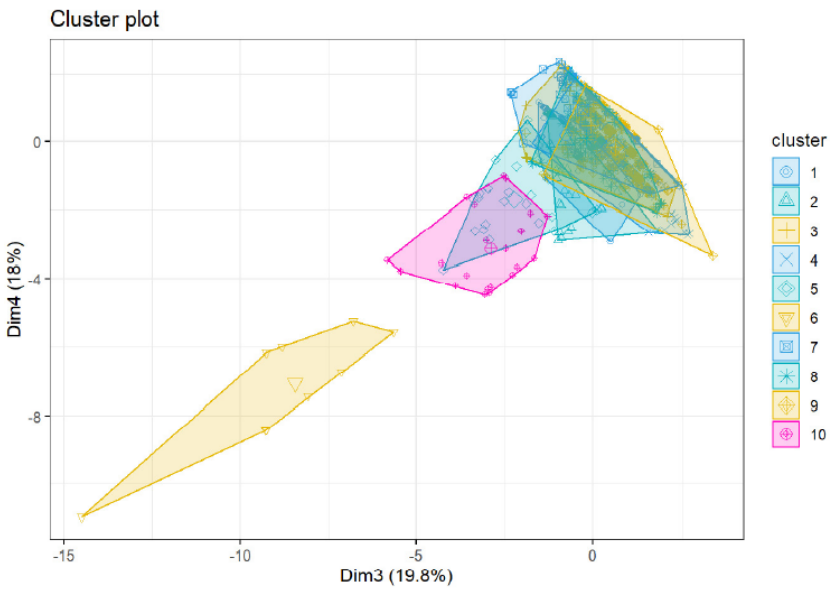


Fig. 11: *Agrupamiento 10-Means correspondiente al segundo análisis PCA*

5.3 ESTUDIO COMPARATIVO ENTRE AMBOS CASOS DE APLICACIÓN DE *MACHINE LEARNING*

Los rasgos que se pueden considerar comparativamente se refieren al origen de los datos, teóricos y simulados en un caso y con la irregularidad y variación de la realidad física en el otro. También a la masividad de los mismos, de manera que en la primera serie de datos simulados, que también es más pequeña, la aparición de puntos discretos es más notoria, mientras que cuando hay miles de puntos en la representación gráfica, como es el caso del segundo DS, se produce una agregación de gran número de ellos en representaciones sin solución de continuidad.

La mayor facilidad para reconocer los sesgos y variaciones del primer caso, ya que han sido introducidos intencionalmente, constituyen finalmente una ayuda para interpretar el segundo caso real, lo cual debería constituir el objetivo más coherente de todo el estudio.

5.3.1 Categorizaciones de clasificación

Se ha interpretado en los diagramas de las Fig. 6 y 7, y se ha confirmado en las Fig. 9 y 11, que los clusters encontrados hacen referencia a la puntualidad en el suministro logístico. De esa manera, se podría asignar una clasificación entre P/Ns que no tienen problemas apreciables en cuanto a fecha de entrega, mientra que en el extremo opuesto se identificarían piezas que sufren retraso generalizado en su suministro.

Aunque puede servir como hipótesis inicial, para confirmar esa interpretación serían necesarios análisis y variaciones adicionales, ya que en ocasiones dicha interpretación sobre salidas gráficas después de la aplicación de un algoritmo pueden no ser totalmente intuitivas.

5.3.2 Inferencias prácticas

Se debe decir que no se han podido identificar inferencias en este estadio del análisis. Sin embargo con alguna actividad adicional se estima que se podrían inferir algunos comportamientos futuros, p. ej. en el caso de los P/N, en función de la evolución gráfica de los segmentos que se identifican con ellos (ver Fig.7).

6 CONCLUSIONES

A lo largo de este Trabajo Fin de Máster se han considerado los aspectos básicos relacionados con el *Machine Learning*, que se puede traducir como “autoaprendizaje de sistemas automáticos”. No siendo este TFM un libro de texto de esta disciplina, se ha expuesto lo suficiente como para resaltar su utilidad en el marco de la Gestión de Programas, en concreto en aspectos logísticos y de apoyo a sistemas.

Merced al avance y asequibilidad de los sistemas telemáticos, que resulta una ventaja en multitud de áreas, el *Machine Learning* también puede fácilmente aportar valor en un

amplio rango de actividades. Los casos prácticos de este trabajo han propuesto, aunque sea en un estadio introductorio, métodos que permiten entrever como, específicamente, se puede dar inteligencia más integrada a la actividad logística, mediante hardware y software al alcance de cualquier estructura de gestión.

En ese mismo estadio introductorio, se ha llevado a cabo un análisis básico de información característica de la gestión de programas, específicamente en el campo logístico. Se ha implementado en una modalidad de datos simulados, como aprendizaje inicial del sistema, y sometiéndolo a datos masivos reales, lo que en un futuro desarrollo permitiría observar autoaprendizajes tangibles.

Se ha constatado como lección aprendida que la complejidad de la materia es apreciable, tanto en el formato necesario de los datos de entrada como en la interpretación de resultados de salida. En esa misma línea, se debe afirmar que, así como es relativamente asequible llevar a cabo clasificaciones básicas, pasar al estadio de inferencias eficaces exige un esfuerzo sustancialmente mayor.

7 REFERENCIAS BIBLIOGRÁFICAS

[01] Chandler, David L. (2013). “How to predict the progress of technology”. M.I.T., 6 de marzo. Disponible en: https://news.mit.edu/2013/how-to-predict-the-progress-of-technology-0306 [consultado el 04-06-2022].

[02] Aprendizaje automático (2022). En: Wikipedia, la enciclopedia libre, 9 de abril. Disponible en: https://es.wikipedia.org/wiki/Aprendizaje_automático [consultado el 04-06-2022].

[03] McCrea, Nick (2014). “An Introduction to Machine Learning Theory and Its Applications: A Visual Tutorial with Examples”, 8 de agosto. Disponible en: https://www.toptal.com/machine-learning/machine-learning-theory-an-introductory-primer [consultado el 04-06-2022].

[04] Machine Learning (2022). En: Wikipedia, The Free Encyclopedia, 8 de marzo. Disponible en: https://en.wikipedia.org/wiki/Machine_learning [consultado el 04-06-2022].

[05] Waseem, Mohammad (2022). “How To Implement Classification In Machine Learning?”, 28 de marzo. Disponible en: https://www.edureka.co/blog/classification-in-machine-learning/ [consultado el 04-06-2022].

[06] Kassambara, Alboukadel (2017). “ PCA - Principal Component Analysis Essentials”, 23 de septiembre. Disponible en: http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/ [consultado el 10-06-2022].

ANEXOS

Anexo I

Código interpretado de la aplicación R

```
loadDataFromExcel <- function(fname){
    library("xlsx")
    dfout <- read.xlsx2(fname, 1, header=TRUE)

    #rename some ugly column names to asvoid problems:
    colnames(dfout)[1] <- "FEntradaPeticion"
    colnames(dfout)[2] <- "FSalidaDespacho"
    colnames(dfout)[3] <- "FEntradaParaDinGraf"
    colnames(dfout)[4] <- "FSalidaParaDinGraf"
    colnames(dfout)[5] <- "XPteEnvioOReparando"
    colnames(dfout)[6] <- "NSN"
    colnames(dfout)[7] <- "NSNSinIPLWRKPNRInfo"
    colnames(dfout)[8] <- "PNIPLWRKPNRInfo"
    colnames(dfout)[9] <- "DescripcionIPLWRKPNRInfo"
    colnames(dfout)[10] <- "PNSL2000"
    colnames(dfout)[11] <- "EspaciosPN"
    colnames(dfout)[12] <- "DescripcionSL2000"
    colnames(dfout)[13] <- "SNLOTE"
    colnames(dfout)[14] <- "PNContenedorAsociado"
    colnames(dfout)[15] <- "ITY_ITPInfo"
    #[16] "ML"
    #[17] "AOG"
    colnames(dfout)[18] <- "MFC_ITPInfo1"
    colnames(dfout)[19] <- "MFC_ITPInfo2"
    colnames(dfout)[20] <- "MRPC_ITPInfo"
    #[21] "QEC"
    colnames(dfout)[22] <- "MFC_SL2000"
    colnames(dfout)[23] <- "Ubicacion"
    #[24] "Origen"
    #[25] "Destino"
    colnames(dfout)[26] <- "AVOMotor"
    colnames(dfout)[27] <- "Condicion"
    #[28] "Cantidad"
    colnames(dfout)[29] <- "RefCOC"
    colnames(dfout)[30] <- "FCaducidad"
    colnames(dfout)[31] <- "ObservacionesTPRTP400"
    colnames(dfout)[32] <- "Observaciones2TPRTP400"
    colnames(dfout)[32] <- "Observaciones2TPRTP400"
    colnames(dfout)[33] <- "MOTIVODELAREPARACION"
    colnames(dfout)[34] <- "OBSERVACIONESCAMO1"
    colnames(dfout)[35] <- "OBSERVACIONESCAMO2"
    colnames(dfout)[36] <- "RPOWC"
    colnames(dfout)[37] <- "StatusParaEnvíoAReparacionMALOGOP"
    colnames(dfout)[38] <- "AccionPendienteEnvioReparacionMALOGOP"
    colnames(dfout)[39] <- "FDDAccionPendienteEPI"
    colnames(dfout)[40] <- "LRU_S_CADUCIDAD_AlmacenNew"
    colnames(dfout)[41] <- "LRU_S_CADUCIDAD_PreserCumplimiento"
    colnames(dfout)[42] <- "LRU_S_CADUCIDAD_FechaEntradaSL2000"
    colnames(dfout)[43] <- "PNsinEspacios"
    colnames(dfout)[44] <- "StockSL2000TotalporNSNdeDinM01SL2000"
    colnames(dfout)[45] <- "CondicionSNSL2000deDinM01SL2000"

    #remove empty rows
```

```
        dfout<-subset(dfout, FEntradaPeticion!="")

        return(dfout)
    }#end function loadLataFromExcel()


    anonimizePN <- function(dfloc){
        #PNSL2000
        locVals <- unique(dfloc$PNSL2000)
        dfloc <- dfloc %>%
                mutate(PN1 = match(dfloc[,"PNSL2000"],locVals))

        #NSN
        locVals <- unique(dfloc$NSN)
        dfloc <- dfloc %>%
                mutate(NSN1 = match(dfloc[,"NSN"],locVals))

        #DescripcionSL2000
        locVals <- unique(dfloc$DescripcionSL2000)
        dfloc <- dfloc %>%
                mutate(Desc1 = match(dfloc[,"DescripcionSL2000"],locVals))

        #DescripcionSL2000 - 1ª palabra
        dfloc <- dfloc %>%
                mutate(Desc1stWord = gsub("([A-Za-z]+).*", "\\1", dfloc[,"DescripcionSL2000"]))
        locVals <- unique(dfloc$Desc1stWord)
        dfloc <- dfloc %>%
                mutate(Desc1stWord = match(dfloc[,"Desc1stWord"],locVals))

        #Ubicacion
        locVals <- unique(dfloc$Ubicacion)
        dfloc <- dfloc %>%
                mutate(Ubic1 = match(dfloc[,"Ubicacion"],locVals))

        #Origen
        locVals <- unique(dfloc$Origen)
        dfloc <- dfloc %>%
                mutate(Orig1 = match(dfloc[,"Origen"],locVals))

        #Condicion
        locVals <- unique(dfloc$Condicion)
        dfloc <- dfloc %>%
                mutate(Condicion1 = match(dfloc[,"Condicion"],locVals))
        return(dfloc)
    } #end function anonimizePN()

    convertDates <- function(dfloc){
        dfloc <- dfloc %>%
                mutate(FEntradaPet = as.Date(as.numeric(dfloc[,"FEntradaPeticion"]), origin = "1899-12-30"))
        dfloc <- dfloc %>%
                mutate(FSalidaDesp = as.Date(as.numeric(dfloc[,"FSalidaDespacho"]), origin = "1899-12-30"))
        dfloc <- dfloc %>%
                mutate(FEntradaDinGraf = as.Date(as.numeric(dfloc[,"FEntradaParaDinGraf"]), origin = "1899-
12-30"))
        dfloc <- dfloc %>%
                mutate(FSalidaDinGraf = as.Date(as.numeric(dfloc[,"FSalidaParaDinGraf"]), origin = "1899-12-
30"))
        dfloc <- dfloc %>%
                mutate(XPteEnvioReparando = as.Date(as.numeric(dfloc[,"XPteEnvioOReparando"]), origin =
```

```
"1899-12-30"))

        dfloc <- dfloc %>%
                mutate(diasEntradaSalida                                                                    =
ifelse(!is.na(dfloc[,"FSalidaDesp"]),as.numeric(difftime(dfloc[,"FSalidaDesp"],dfloc[,"FEntradaPet"],units = c("days"))),-1))


        dfloc <- dfloc %>%
                mutate(contadorDias = as.numeric(dfloc[,"FEntradaPeticion"])-43543)

        return(dfloc)
    } #end function convertDates()

    otherOps <- function(dfloc){
        dfloc <- dfloc %>%
                mutate(MaintType = rep(0, nrow(dfloc)))
                dfloc$MaintType <- ifelse(dfloc$ML == "ML1", 1, ifelse(dfloc$ML == "ML2", 2, 3))
        return(dfloc)
    } #end function otherOps()

    getSimplifiedDF <- function(dfin){

        dfout <- dfin[,c("contadorDias","PN1","Desc1","Orig1","diasEntradaSalida","MaintType","Condicion1")]

        return(dfout)
    } # end function getSimplifiedDF

    library(tidyverse) #includes ggplot2
    library(lubridate)
    library(ggplot2)
    library(plotly)

    #Estas 6 líneas hay que "descomentarlas" para generar los datos:
    df1 <- loadDataFromExcel("movimientos taller motores.xlsx")
        initialColCount <- length(df1)
        initialRowCount <- nrow(df1)
    initialdf <- df1

    df1 <- anonimizePN(df1)
    df1 <- convertDates(df1)
    df1 <- otherOps(df1)

    dfsimple <- getSimplifiedDF(df1)
    #df2 <- df1[,c("contadorDias","PN1","diasEntradaSalida","MaintType")]
    #por alguna razón a la función k-means clustering no le gusta la variable "contador días"... la quito:
    df2 <- df1[,c("Desc1","Orig1","Condicion1","diasEntradaSalida","MaintType")]

    set.seed(123)
    km.res <- kmeans(df2, 10, nstart = 25)

    #library(ggpubr)
    #library(factoextra)
    #dibujo3 <- fviz_cluster(km.res, data = df2,
    #            palette = c("#2E9FDF", "#00AFBB", "#E7B800", "#FF00BF"),
    #            geom = "point",
    #            ellipse.type = "convex",
    #            ggtheme = theme_bw()
    #            )
```

```
library(rmarkdown)
render("informe1.Rmd")
```

Anexo II

Informe PCA

(renderización mediante función libr. “RMarkdown”)

PCA analysis with dfsimple

10 junio 2022

Introducción

Vamos a tratar de solucionar el problema de la representación bidimensonal de datos de k-means usando Principal Component Analysis (PCA) Pulsar aquí para ver el artículo usado como referencia para PCA. (http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/)

PCA permite resumir y visualizar la información en un conjunto de datos que contiene múltiples variables (columnas del DF) cuantitativas interrelacionadas. Cada variable podría considerarse como una dimensión diferente, por lo que si hay más de 3 no se puede hacer una adecuada representación gráfica. En nuestro caso, el DF utilizado en el primer ejemplo tiene 5 columnas (dimensiones).

PCA permite extraer la información importante de un DF de múltiples variables y expresar esta información como un conjunto menor de variables nuevas, denominadas componentes principales. Estas nuevas variables corresponden a una combinación lineal de las originales. El número de componentes principales es menor o igual al número de variables originales.

El objetivo de PCA es identificar los componentes principales a lo largo de los cuales la variación en los datos es máxima.

En otras palabras, PCA reduce la dimensionalidad de los datos multivariados a dos o tres componentes principales para poder visualizarlos gráficamente, con la menor pérdida de información.

Estandarización de datos

En PCA es normal tener que escalar (estandarizar) las variables para que sean más fácilmente comparables. En general, se fuerza que su media sea cero y su desviación estándar sea la unidad. En R se puede estandarizar con `scale()`, pero el algoritmo PCA que usamos al incluir `library("FactoMineR")` ya efectua la estandarización por defecto.

El siguiente código contiene la llamada a PCA con las variables de interés. El resultado muestra los objetos que genera la llamada y que contienen los parámetros del PCA resultante:

```
res.pca1 <- PCA(df1[,c("Desc1","Orig1","Condicion1","diasEntradaSalida","MaintType")], graph
 = FALSE)
print(res.pca1)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 3754 individuals, described by 5 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

Generamos otros con otras variables: Sustituimos la descripción textual del SL2000 por únicamente la primera palabra:

```
res.pca2 <- PCA(df1[,c("Desc1stWord","Orig1","Condicion1","diasEntradaSalida","MaintType")],
 graph = FALSE)
print(res.pca2)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 3754 individuals, described by 5 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

Valores propios / Varianzas

Los valores propios (eigenvalues) miden la cantidad de variación retenida por cada componente principal (PC). Los valores propios son grandes para las primeras PC y decrecen con las PC posteriores. Es decir, las primeras PC corresponden a las direcciones con la máxima cantidad de variación en el conjunto de datos.

```
eig.val <- get_eigenvalue(res.pca1)
eig.val
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  1.2686536         25.37307                    25.37307
## Dim.2  1.1893256         23.78651                    49.15958
## Dim.3  0.9513134         19.02627                    68.18585
## Dim.4  0.8710423         17.42085                    85.60670
## Dim.5  0.7196651         14.39330                   100.00000
```

La suma de todos los valores propios da una varianza total de 10.

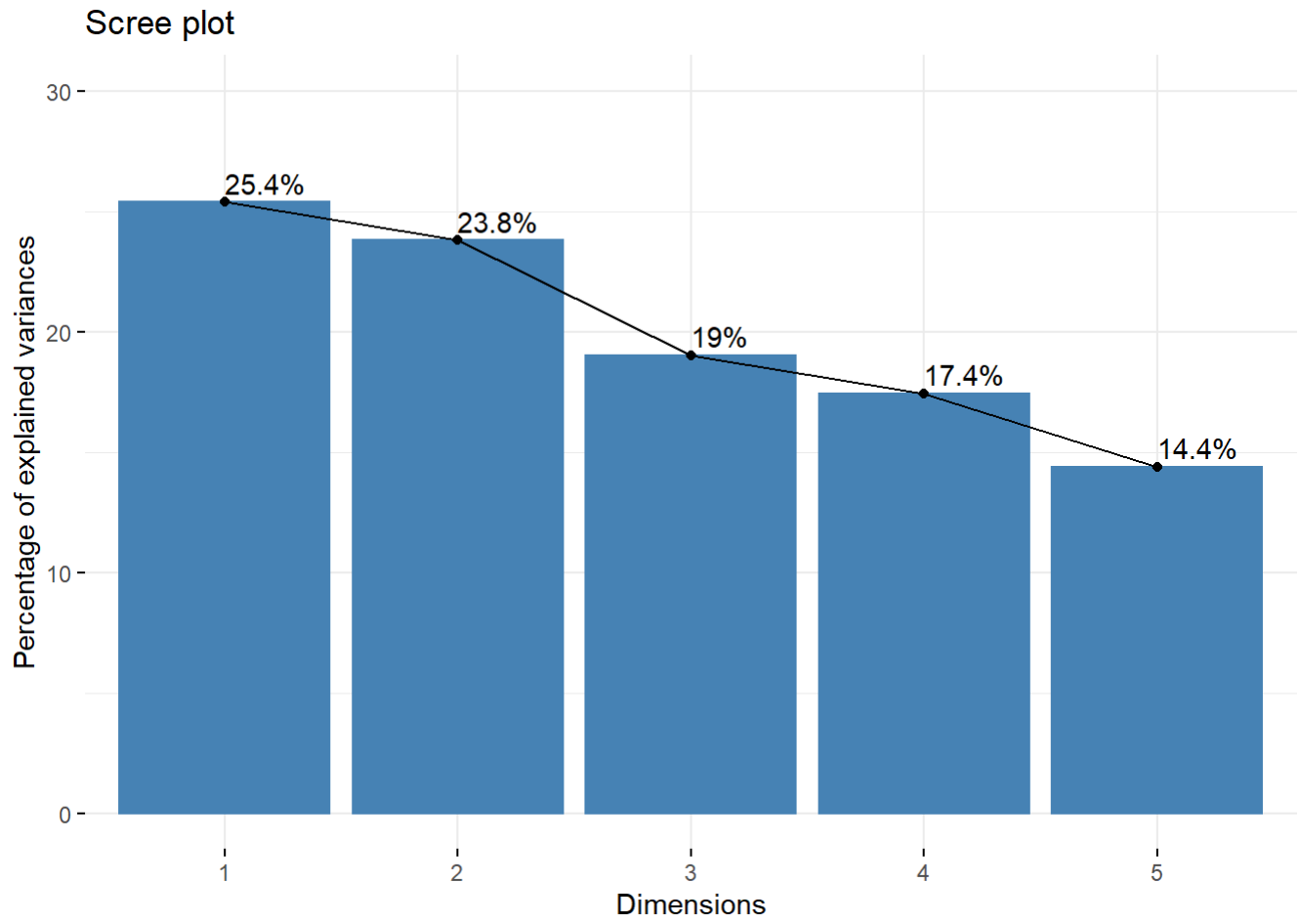
La proporción de variación explicada por cada valor propio se da en la segunda columna.

Los valores propios se pueden utilizar para determinar el número de componentes principales a retener después de PCA (Kaiser 1961) :

- Un valor propio > 1 indica que las PC explican más varianza que la explicada por una de las variables originales en los datos estandarizados. Esto se usa comúnmente como un punto de corte para el cual se retienen las PC. Esto es cierto solo cuando los datos están estandarizados.
- También se puede limitar el número de componentes al número que represente una determinada fracción de la varianza total. Por ejemplo, si está satisfecho con el 70 % de la varianza total explicada, utilice el número de componentes para lograrlo.

A continuación se muestra el gráfico de sedimentación obtenido mediante la función `fviz_eig()` de paquete `factoextra` .

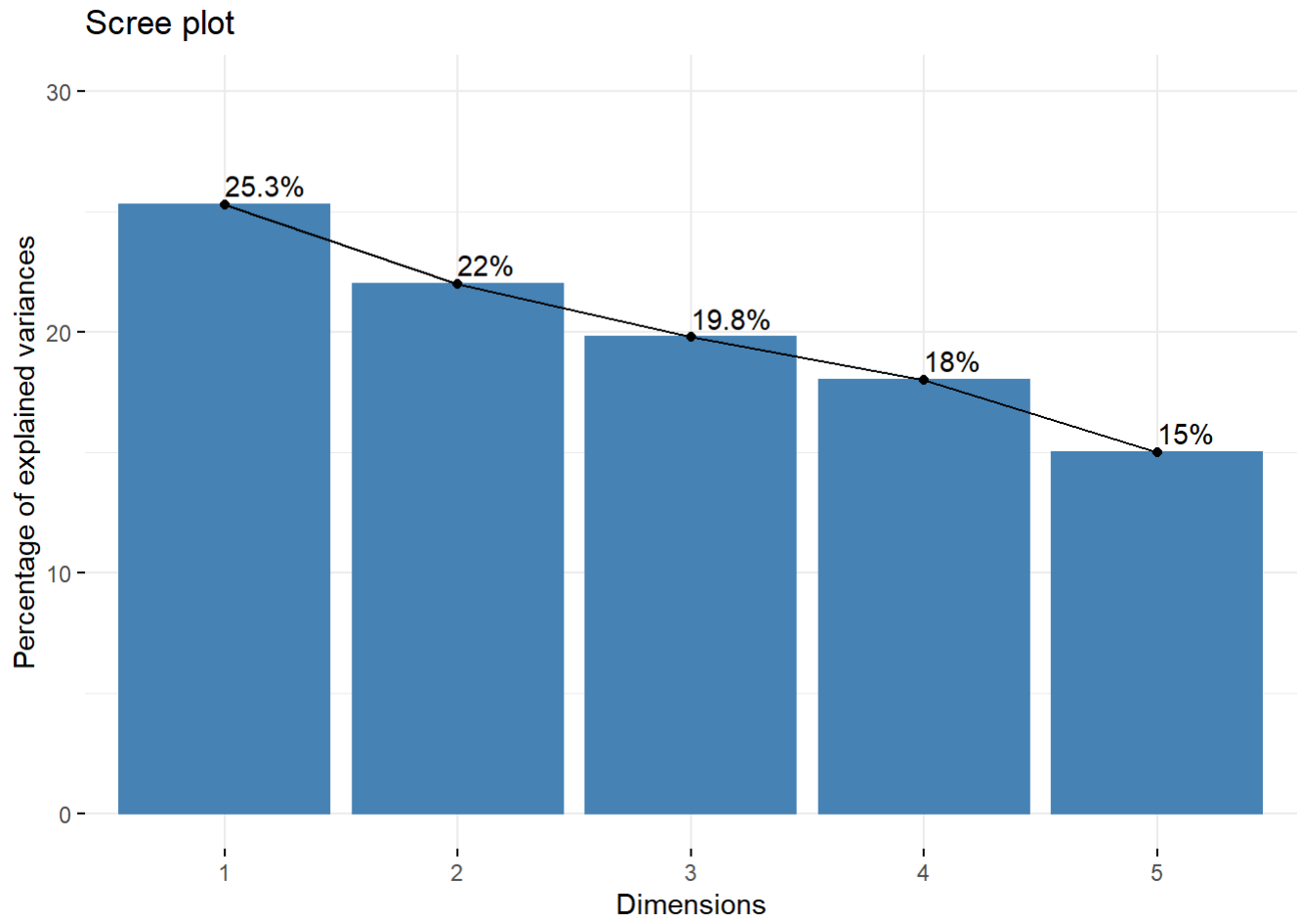
```
fviz_eig(res.pca1, addlabels = TRUE, ylim = c(0, 30))
```



De la gráfica anterior, podríamos detenernos en el 4º PC. El 85.6% de la información (varianzas) contenida en los datos es retenida por los primeros cuatro PCs.

Veamos si hay mejoras en el gráfico de sedimentación cuando sólo evaluamos la 1ª palabra de la descripción SL2000:

```
fviz_eig(res.pca2, addlabels = TRUE, ylim = c(0, 30))
```



Lamentablemente no se ha conseguido reducir el peso específico de las varianzas para ninguna de las dimensiones.

Gráfico de variables

Un método simple para extraer los resultados de una salida de PCA es usar la función `get_pca_var()` (paquete `factoextra`).

Esta función proporciona una lista de matrices que contienen todos los resultados de las variables activas (coordenadas, correlación entre variables y ejes, coseno al cuadrado y contribuciones)

```
var <- get_pca_var(res.pca1)
var
```

```
## Principal Component Analysis Results for variables
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

Los componentes de `get_pca_var()` se pueden utilizar en la gráfica de variables de la siguiente manera:

`var$coord` : coordenadas de variables para crear un diagrama de dispersión `var$cos2` : representa la calidad de representación de las variables en el mapa de factores. Se calcula como las coordenadas al cuadrado: `var.cos2 = var.coord * var.coord` . `var$contrib` : contiene las contribuciones (en porcentaje) de las

variables a los componentes principales. La contribución de una variable (var) a un componente principal dado es (en porcentaje) : (var.cos2 * 100) / (cos2 total del componente). Tenga en cuenta que es posible trazar variables y colorearlas según:

- su calidad en el mapa de factores (cos2)
- sus valores de contribución a los componentes principales (contrib).

Se puede acceder a los diferentes componentes de la siguiente manera:

```
# Coordinates
head(var$coord)
```

```
##                      Dim.1      Dim.2        Dim.3
## Desc1              0.7889663  0.0822031  0.119515205
## Orig1              0.6858784  0.2560472 -0.080766630
## Condicion1         0.3035746 -0.6050541  0.571840076
## diasEntradaSalida -0.2727976  0.4650169  0.776832781
## MaintType          0.0958155  0.7312161 -0.006004622
##                      Dim.4      Dim.5
## Desc1              0.1666381 -0.5733435
## Orig1             -0.5325425  0.4169963
## Condicion1         0.3058932  0.3481097
## diasEntradaSalida -0.3014435 -0.1224882
## MaintType          0.6125632  0.2843813
```

```
# Cos2: quality on the factore map
head(var$cos2)
```

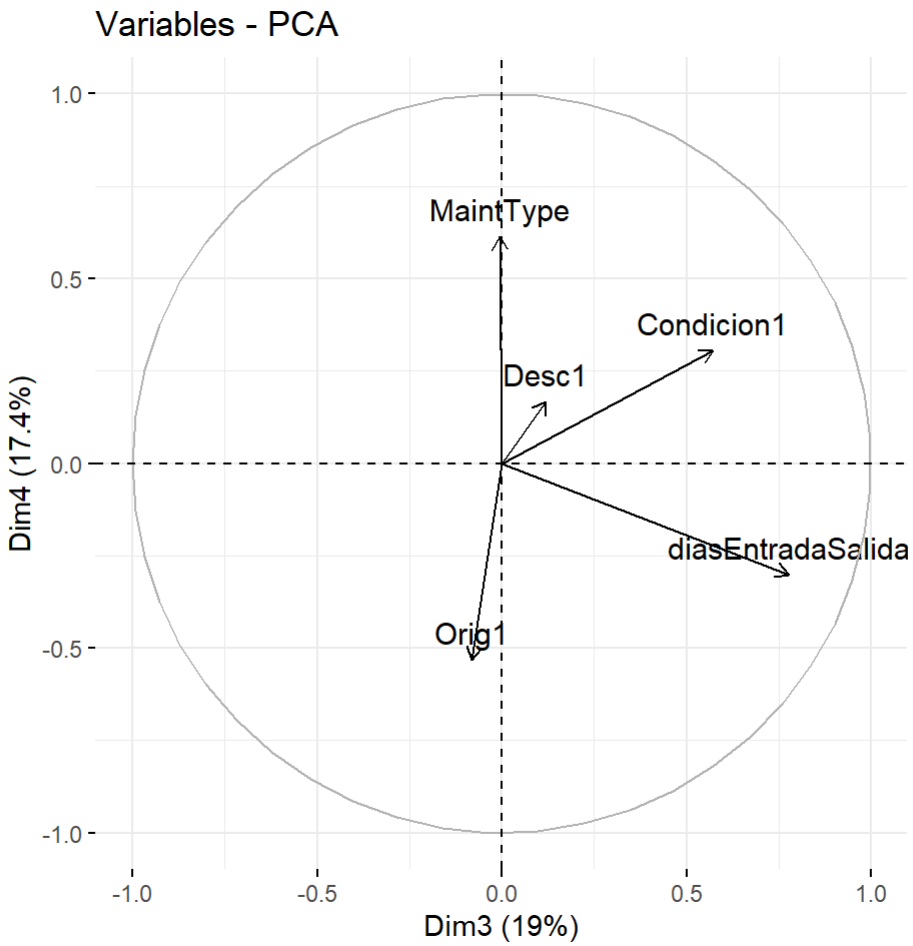
```
##                      Dim.1      Dim.2        Dim.3
## Desc1             0.62246778 0.00675735 1.428388e-02
## Orig1             0.47042912 0.06556017 6.523249e-03
## Condicion1        0.09215752 0.36609040 3.270011e-01
## diasEntradaSalida 0.07441855 0.21624073 6.034692e-01
## MaintType         0.00918061 0.53467693 3.605548e-05
##                      Dim.4      Dim.5
## Desc1             0.02776827 0.32872272
## Orig1             0.28360152 0.17388595
## Condicion1        0.09357062 0.12118039
## diasEntradaSalida 0.09086819 0.01500336
## MaintType         0.37523370 0.08087270
```

```
# Contributions to the principal components
head(var$contrib)
```

```
##                       Dim.1      Dim.2       Dim.3     Dim.4
## Desc1             49.0652287  0.5681665  1.501490870  3.187935
## Orig1             37.0809750  5.5123819  0.685709704 32.558869
## Condicion1         7.2641988 30.7813445 34.373642005 10.742374
## diasEntradaSalida  5.8659476 18.1817939 63.435367347 10.432122
## MaintType          0.7236499 44.9563132  0.003790074 43.078700
##                       Dim.5
## Desc1             45.677179
## Orig1             24.162065
## Condicion1        16.838441
## diasEntradaSalida  2.084769
## MaintType         11.237547
```

Círculo de correlación

```
fviz_pca_var(
    res.pca1,
    axes = c(3, 4),
    col.var = "black"
)
```



El gráfico anterior también se conoce como gráficos de correlación de variables. Muestra las relaciones entre todas las variables. Se puede interpretar de la siguiente manera:

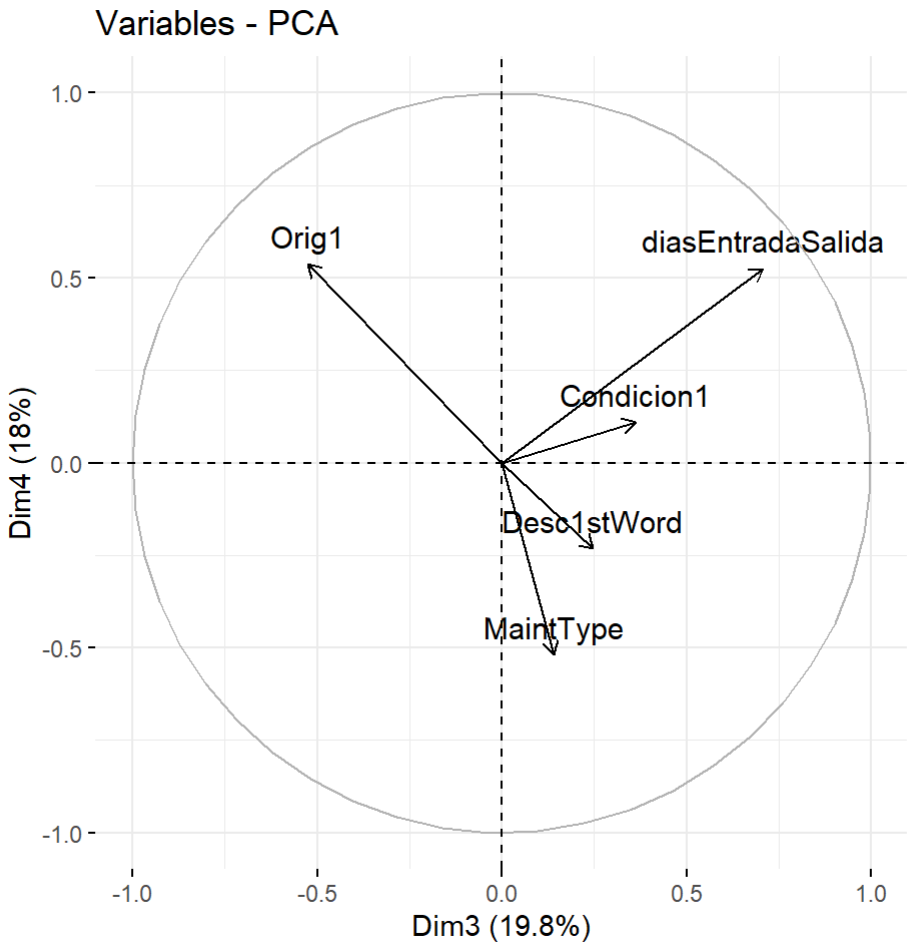
-Las variables correlacionadas positivamente se agrupan.

-Las variables correlacionadas negativamente se colocan en lados opuestos del origen del gráfico (cuadrantes opuestos).

-La distancia entre las variables y el origen mide la calidad de las variables en el mapa de factores. Las variables que están alejadas del origen están bien representadas en el mapa de factores.

Representamos ahora con el otro dataset, considerando solo la primera palabra de la descripción SL2000:

```
fviz_pca_var(
    res.pca2,
    axes = c(3, 4),
    col.var = "black"
)
```



No es fácil de explicar, pero parece que hemos mejorado algo el círculo de correlación, pues las flechas son más largas.

Calidad de representación

La calidad de representación de las variables en el mapa de factores se llama cos2 (coseno cuadrado, coordenadas cuadradas). Puede acceder a cos2 de la siguiente manera:

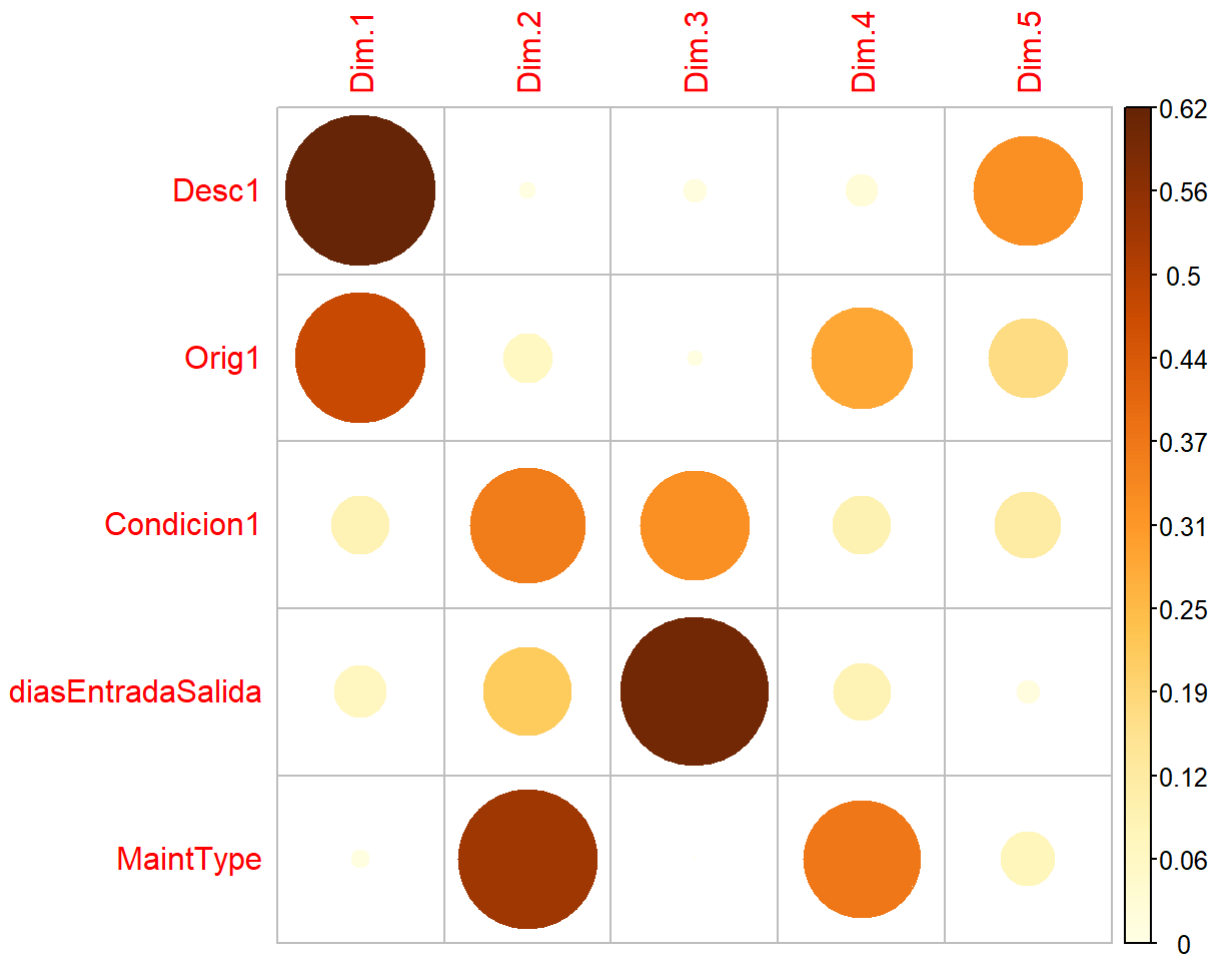
```
head(var$cos2, 4)
```

```
##                       Dim.1      Dim.2       Dim.3      Dim.4
## Desc1             0.62246778 0.00675735 0.014283884 0.02776827
## Orig1             0.47042912 0.06556017 0.006523249 0.28360152
## Condicion1        0.09215752 0.36609040 0.327001073 0.09357062
## diasEntradaSalida 0.07441855 0.21624073 0.603469169 0.09086819
##                       Dim.5
## Desc1             0.32872272
## Orig1             0.17388595
## Condicion1        0.12118039
## diasEntradaSalida 0.01500336
```

Para una variable dada, la suma de los cos2 de todos los componentes principales es igual a uno. Si una variable está perfectamente representada por solo dos componentes principales (Dim.1 y Dim.2), la suma del cos2 en estas dos PC es igual a uno. En este caso las variables se posicionarán sobre el círculo de correlaciones.

Puede visualizar el cos2 de las variables en todas las dimensiones usando el paquete `corrplot` :

```
library("corrplot")
corrplot(var$cos2, is.corr=FALSE)
```



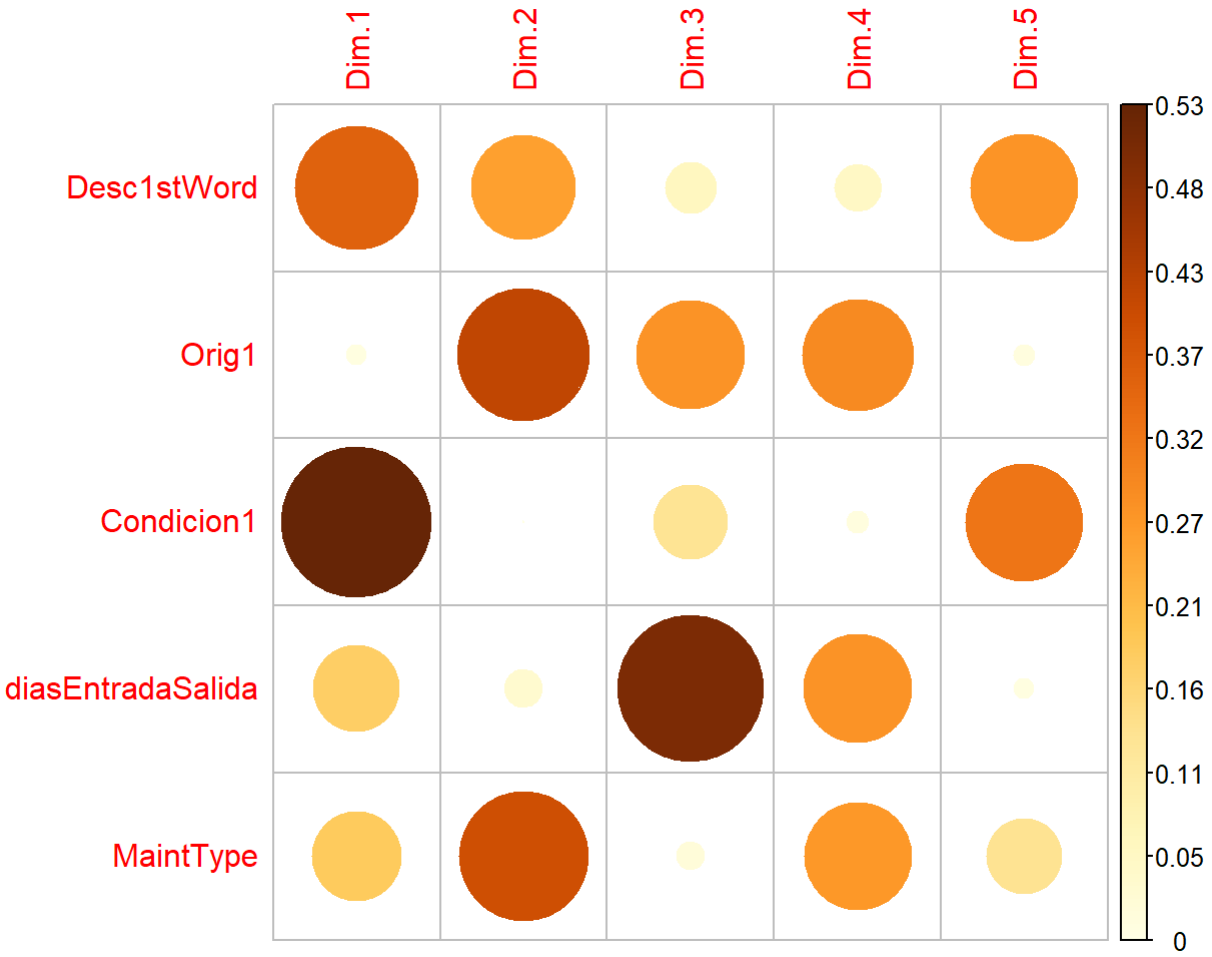
-Los valores de cos2 se utilizan para estimar la calidad de la representación.

-Cuanto más cerca esté una variable del círculo de correlaciones, mejor será su representación en el mapa de factores (y más importante será interpretar estos componentes)

-Las variables que están cerca del centro de la gráfica son menos importantes para los primeros componentes.

Repetimo con los otros datos:

```
var <- get_pca_var(res.pca2)
corrplot(var$cos2, is.corr=FALSE)
```



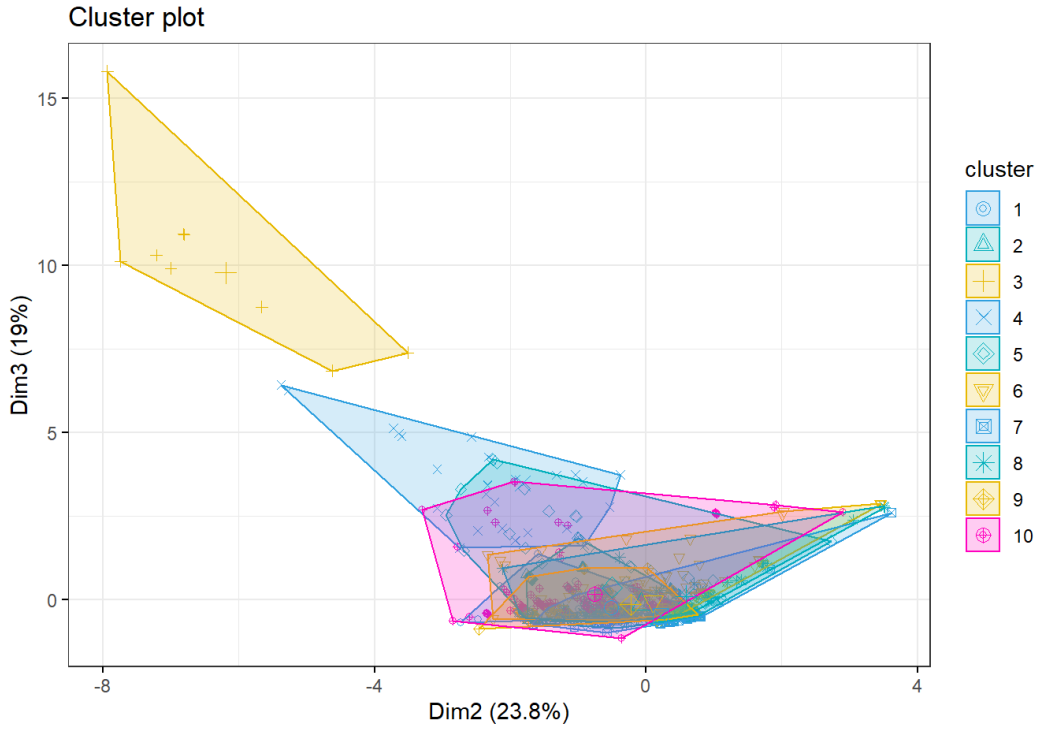
No parece que hayamos mejorado mucho, aunque como aspecto positivo podemos ver que la variable `diasEntradaSalida` está ligeramente mejor representada en la D3 (dimensión 3) de lo que lo estaba en la D2 antes, por lo que podemos intentar centrar la representación del clustering:

- (D2,D3) con el nombre completo.
- (D3,D4) con solo la primera palabra.

Volvemos a representar gráficamente k-means

La representación gráfica 1:

```
    km.res <- kmeans(df1[,c("Desc1","Orig1","Condicion1","diasEntradaSalida","MaintType")], 1
0, nstart = 25)
    fviz_cluster(km.res, data = df1[,c("Desc1","Orig1","Condicion1","diasEntradaSalida","Main
tType")],
    palette = c("#2E9FDF", "#00AFBB", "#E7B800","#2E9FDF", "#00AFBB", "#E7B800","#2E9FDF", "#
00AFBB", "#E7B800", "#FF00BF"),
    geom = "point",
    axes = c(2, 3),
    ellipse.type = "convex",
    ggtheme = theme_bw()
    )
```



La representación gráfica 2:

```
    km.res <- kmeans(df1[,c("Desc1stWord","Orig1","Condicion1","diasEntradaSalida","MaintTyp
e")], 10, nstart = 25)
    fviz_cluster(km.res, data = df1[,c("Desc1stWord","Orig1","Condicion1","diasEntradaSalida"
,"MaintType")],
    palette = c("#2E9FDF", "#00AFBB", "#E7B800","#2E9FDF", "#00AFBB", "#E7B800","#2E9FDF", "#
00AFBB", "#E7B800", "#FF00BF"),
    geom = "point",
    axes = c(3, 4),
    ellipse.type = "convex",
    ggtheme = theme_bw()
    )
```