UNIVERSIDAD DE GRANADA

Práctica 4. Agrupación de datos.

Tratamiento Inteligente de Datos.

Máster Profesional en Ingeniería Informática

Curso académico 2022/2023

Autor

José Alberto Gómez García

Índice

1. Introducción.

En esta práctica trataremos con algoritmos de clustering, o agrupamiento, si lo traducimos al español. El clustering es una técnica de aprendizaje automático no-supervisado que nos puede permitir encontrar patrones o estructuras ocultas en los datos. Esto puede ser útil en aplicaciones como la segmentación de clientes o la detección de anomalías, entre otras.

En esta práctica utilizaremos distintos algoritmos de agrupamiento para intentar extraer conocimiento de 3 bases de datos de diferentes temáticas. Trabajaremos sobre datos relativos a tipos de plantas iris, tipos de vinos y jugadores de la NBA en general.

A diferencia de prácticas anteriores, en esta ocasión el fichero "KNIME Workflow" sin resetear es ligero, por lo que se adjunta en la entrega en Prado.

2. Base de datos "Iris".

En primer lugar, en este apartado trataremos con una versión simplificada de la más que conocida base de datos Iris (en tanto que contiene 2 características en vez de 4), la cual ha sido modificada para contener outliers.

Para comenzar, echaremos un vistazo a como se encuentran distribuidos los datos. Sobre la representación, realizaremos una primera división correspondiente a cómo una persona podría realizar de forma intuitiva el agrupamiento de los datos sin conocer nada de estos.

En la parte superior se encuentra una nube de puntos bien diferenciada del resto, mientras que en la parte inferior podría considerarse que hay una nube de puntos de mayor tamaño, o dos nubes divididas por el medio, como se muestra. Nótese que en esta representación se han obviado a propósito los outliers que hay en las esquinas, y que trataremos más adelante.

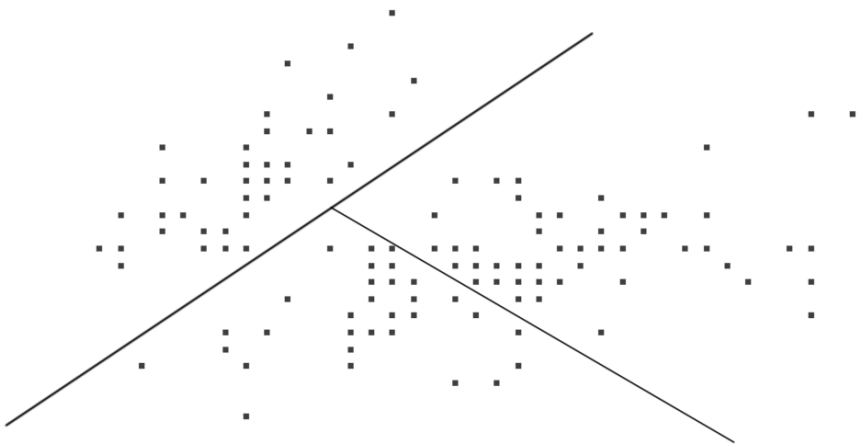

Imagen 1. Representación de los datos y primera idea de agrupamiento intuitivo.

En la imagen 2 se muestran los datos en diferentes colores, según el tipo de iris al que corresponde cada muestra. Podemos observar que la división intuitiva es acertada hasta cierto punto, pues la esquina superior izquierda corresponde a una determinada clase de iris (setosa), y la nube inferior contiene dos clases de iris (versicolor en azul, virginica en verde), que se separan más o menos por la diagonal que marcábamos anteriormente, aunque hay puntos de cada clase a uno y otro lado de dicha diagonal.

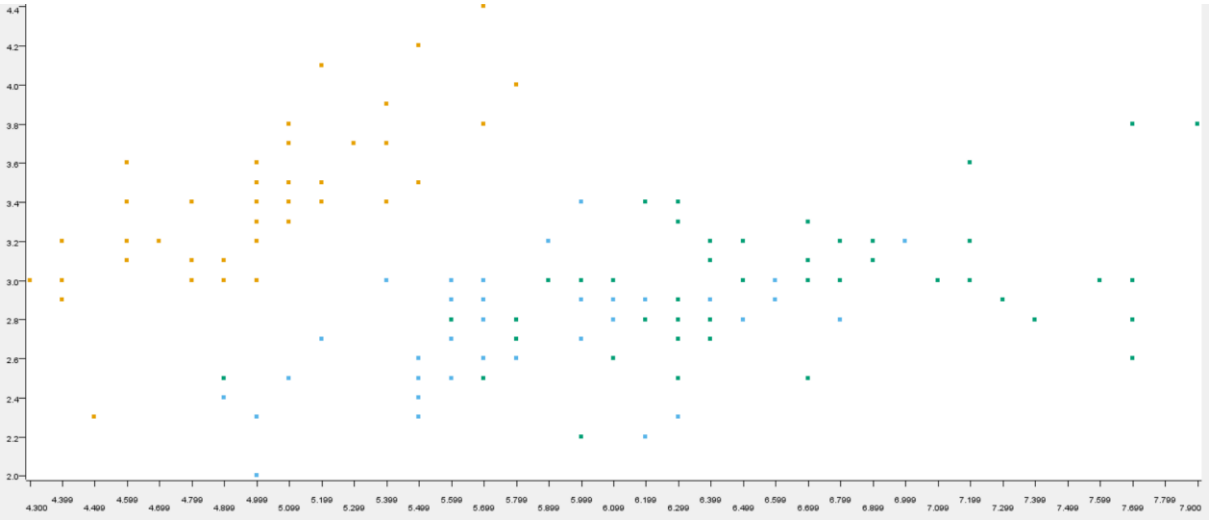

Imagen 2. Agrupamiento real de los datos en función del tipo de iris.

A continuación, y como se pide en el guion, realizaré un clustering jerárquico asociativo, haciendo uso del nodo de KNIME “Hierarchical Clustering”. Antes de esto, deberemos normalizar los datos para evitar la dominancia de una variable sobre la otra, pues se mueven en intervalos diferentes y la variable con mayor intervalo “dominaría” a la otra.

Si observamos el diagrama de distancias que se adjunta a continuación, podemos ver como el número de clústers apropiado parece ser entre 3 y 5, en tanto que son los números de clústers que se encuentran en el “codo” de la curva. Dado que conocemos la base de datos, sabemos que el número correcto de clústers es 3.

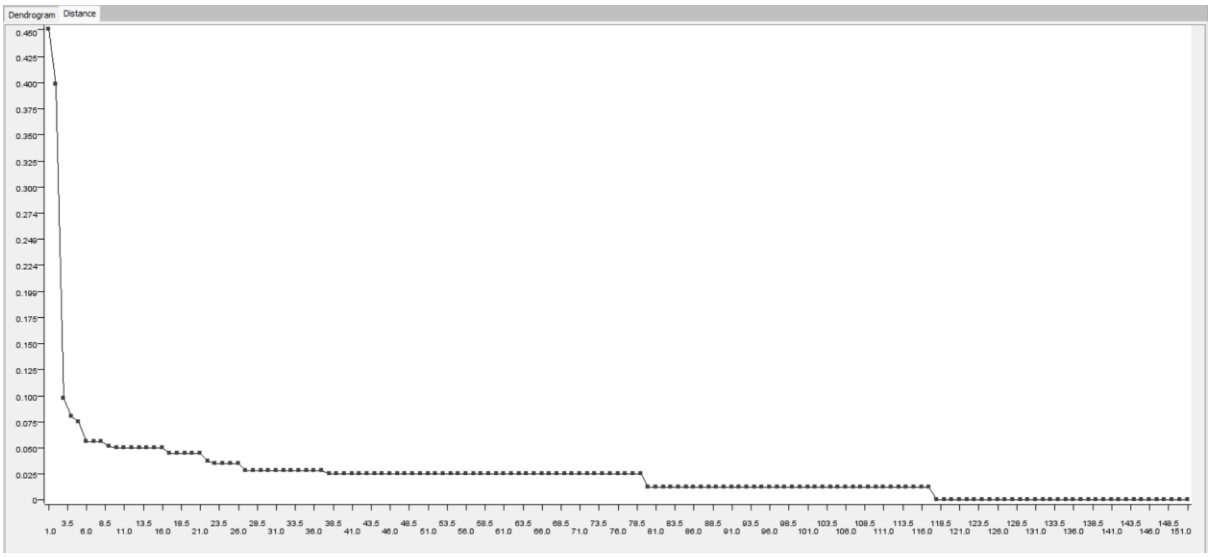

Imagen 3. Diagrama de distancias tras aplicar clustering jerárquico sobre iris-slw_outliers.dat.

Si observamos el dendrograma generado (imagen 4), podemos ver que la mayoría de muestras se encuentran bastante cerca y se van uniendo "de forma natural" dada la distancia entre ellas. Sin embargo, existen dos uniones para las cuales la distancia entre el clúster y la muestra es muy alta.

Este comportamiento nos hace sospechar de la existencia de dos outliers, dado que una vez generados los clústers con muestras cercanas entre sí, los outliers se unen en las últimas fases del proceso a aquel clúster que se encuentre más cerca, por lejos que se encuentren.

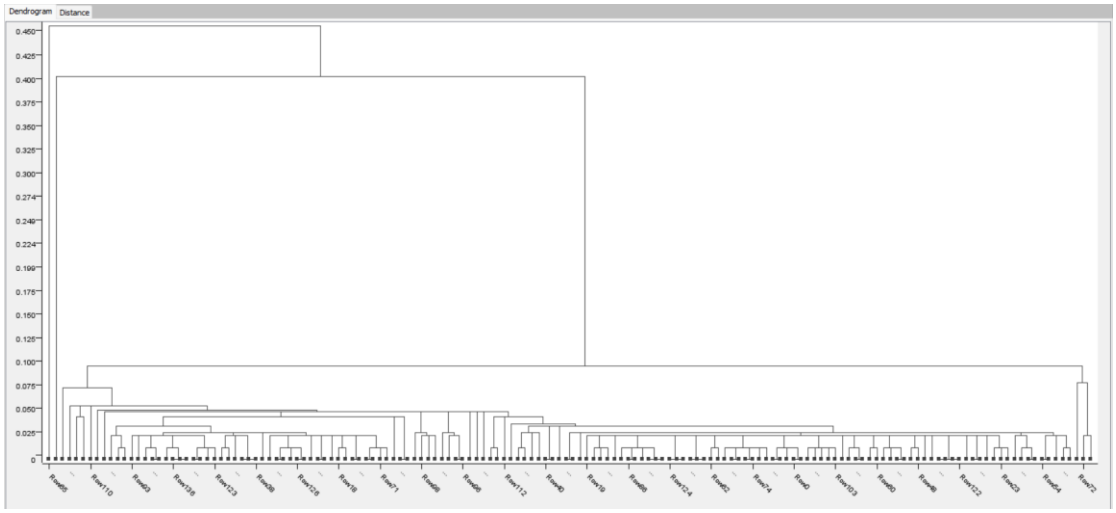

Imagen 4. Dendrograma resultado de aplicar clustering jerárquico sobre iris-slw_outliers.dat.

Si aplicamos el algoritmo de clustering "k-Means" especificándole que agrupe los datos en los 3 clústers que intuimos que hay dado el diagrama de distancias generado por el algoritmo de clustering jerárquico, obtenemos la representación mostrada en la imagen 5.

En esta podemos observar cómo los datos parecen agruparse bien en 3 clústers, de acuerdo con la idea intuitiva que presentamos en la primera imagen, y que dichos clústers se ajustan más o menos bien a la clasificación real de los datos que se mostraba en la imagen 2.

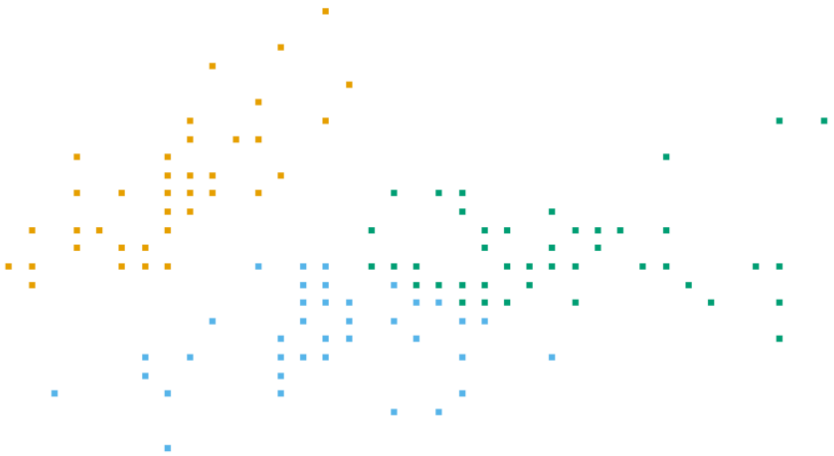

Imagen 5. Muestras agrupadas haciendo uso del algoritmo k-Means y 3 clústers.

No mostramos los outliers para facilitar la visualización del conjunto de datos. Sin embargo, cabe destacar que dichos outliers se encuentran en las esquinas inferior izquierda y superior derecha del gráfico. El primero se agrupa dentro del clúster azul, mientras que el segundo se agrupa en el clúster verde.

Estos outliers pueden influir en la generación de los clústers, por lo que haremos uso del nodo "Numeric Outliers" de KNIME para eliminarlos y repetir el experimento. Este proceso elimina únicamente los dos outliers que identificamos anteriormente, los cuales eran muy obvios y habían sido introducidos a propósito.

Si volvemos a ejecutar el algoritmo k-Means haciendo uso de 3 clústers, veremos como los resultados son muy similares a los obtenidos anteriormente. Las diferencias se encuentran en la frontera entre los clústers azul y verde, donde ahora existen un mayor número de instancias pertenecientes al clúster azul.

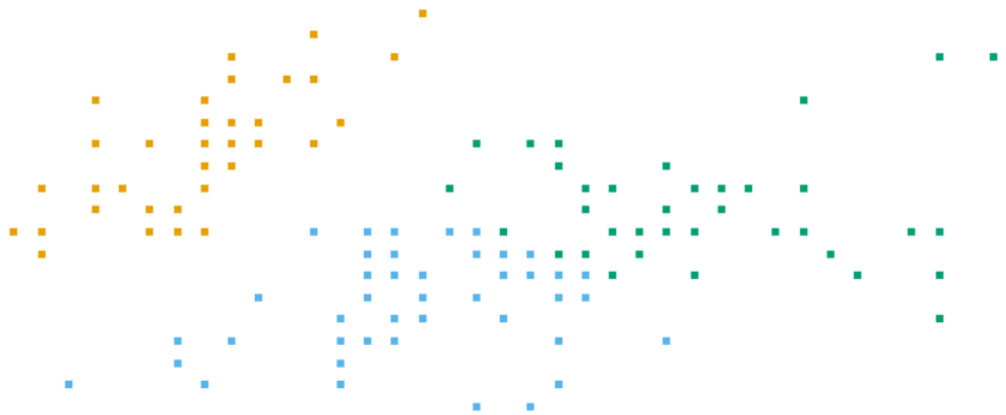

Imagen 6. Muestras agrupadas haciendo uso del algoritmo k-Means y 3 clústers. Outliers eliminados.

Para terminar este apartado, cargaremos los datos sin outliers del fichero correspondiente y haremos uso del algoritmo de agrupamiento k-Medoids. Además de añadir el nodo correspondiente a este algoritmo, también deberemos añadir el nodo "Distance Matrix Calculate", que calcula las distancias entre cada pareja de filas de la base de datos, ya que esta información es requerida por el algoritmo de agrupamiento.

No ahondaremos demasiado en la teoría, pero a diferencia de k-Means, K-Medoids utiliza una función de optimización en la que iterativamente se intercambian los medoides con otras muestras del clúster, con el objetivo de encontrar el centroide que proporcione la configuración óptima. Esto lo hace más resistente ante la existencia de outliers. En todo caso, vamos a hacer uso de la base de datos sin outliers; y lo compararemos los resultados

proporcionados contra los generados por k-Means sobre la misma base de datos sin outliers.

Si observamos la imagen 7, veremos las agrupaciones que genera el algoritmo k-Medoids son prácticamente idénticas a las generadas por k-Means. Concretamente, sólo existen diferencias en 4 muestras, rodeadas en rojo para identificarlas más fácilmente. Las dos muestras más a la izquierda, anteriormente enmarcados en el clúster verde, ahora han pasado a ser del clúster azul; mientras que para las dos muestras más a la derecha ha sucedido lo contrario.

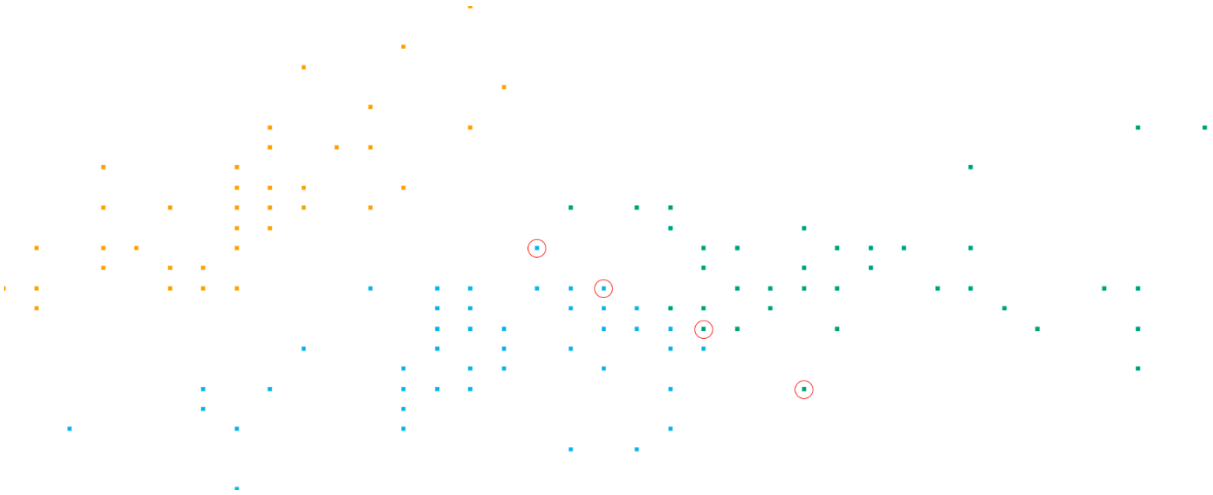

Imagen 7. Muestras agrupadas haciendo uso del algoritmo k-Medoids y 3 clústers.

3. Base de datos “Wine”.

Esta base de datos contiene datos sobre 13 características químicas de 178 vinos de una región de Italia. Adicionalmente, se tiene una columna adicional que indica el tipo de vino en función de su nivel de alcohol, teniéndose 3 posibles clases. La descripción de los atributos se encuentra en el fichero *wine_names.txt*. Este fichero nos dice también que no hay valores perdidos en la base de datos, por lo que no tendremos que realizar ningún tipo de gestión al respecto.

Lo primero que haremos será cargar el contenido del fichero *wines.data* en KNIME y utilizar un nodo “Column Rename” para asignar a cada columna el nombre de la característica a la que corresponde, de manera que nos es más sencillo trabajar con los datos.

Hecho esto, normalizaremos cada característica haciendo uso del algoritmo “Min-Max”, de manera que los valores se encuentren en el rango [0,1]. Así, podremos aplicar un algoritmo de clustering jerárquico agregativo.

El diagrama de distancias resultado de aplicar dicho proceso de clustering se puede consultar en la imagen 8. Si nos fijamos en el codo de la gráfica, el cual se encuentra en torno a altura 0.66, este nos dice que sería conveniente usar 3 o 4 clústers, pudiendo considerarse incluso el usar 5 clústers.

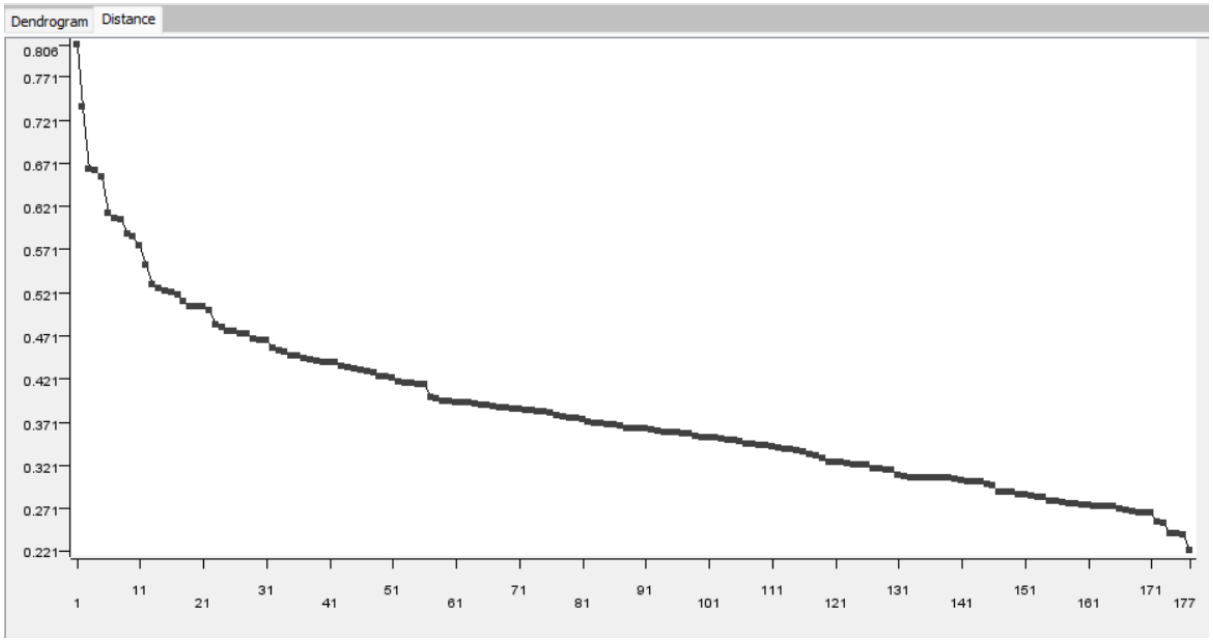

Imagen 8. Diagrama de distancias tras aplicar clustering jerárquico sobre wines.data.

A continuación, analizaremos los valores concretos que toman los datos. Con esto, intentaremos descubrir patrones en los datos que nos puedan ser de utilidad, así como localizar outliers y eliminarlos.

Para la detección de outliers he utilizado histogramas, pues permiten ver rápidamente si hay algún valor fuera de lo normal. Los nodos de KNIME con este propósito no me proporcionan todas las opciones que me gustaría a la hora de generar el gráfico, no siendo la visualización tan rápida e intuitiva como esperaba, no se adjuntan en esta memoria por el motivo expuesto. La información visualizada en los histogramas la he complementado con el nodo "Value Counter", para ver con más detalle los datos que parecían ser outliers.

Las conclusiones a las que he podido llegar respecto de cada variable son las siguientes:

- Los vinos de las clases 1 y 2 parece que se pueden diferenciar por el nivel de alcohol. Los vinos de tipo 1 tienen más de 13º de alcohol, mientras que la mayoría de los vinos de tipo 2 tienen 13º o menos. Los vinos de la clase 3 tienen un nivel de alcohol entre los 12 y 14º.
- Una gran cantidad de los vinos de tipo 1 tiene un nivel de ácido málico entre 1.5 y 2. Para las otras dos clases de vino este valor fluctúa mucho más. No parece haber outliers.
- En el caso de la ceniza sí que hay outliers claros. Estos son tres, con valores 1.36, 3.22 y 3.23, respectivamente.
- En el caso del magnesio si parecen existir dos outliers, con valores 151 y 162.
- En el caso de los flavonoides parece haber un outlier muy claro, con valor 5.08. La mayoría de los vinos de clase 3 parecen tener un valor inferior a 1 de este índice.

- Los valores de la intensidad de color se mueven en un rango más amplio y no hay outliers claros. En todo caso, podríamos considerar dos outliers con valores 11.75 y 13.
- El tono (entendemos que del color) presenta un outlier con valor 1.71.
- El rango de prolina está repartido para cada clase de vino, pero sólo los vinos de clase 1 superan consistentemente el valor 1000, pudiendo alcanzar valores en torno a 1500. Podríamos llegar a considerar outlier el valor 1680.
- Para el resto de variables no parece haber outliers ni patrones claros.

Hasta el momento hemos identificado algunos outliers aparentemente claros y algunos dudosos, pero sin seguir ningún criterio matemático concreto. Utilizaremos el nodo "Numeric Outliers" para detectar y eliminar outliers. Este nodo considerará outliers aquellos valores que caigan fuera del rango $R = [Q_1 - k(IQR), Q_3 + k(IQR)]$, siendo IQR la diferencia entre el tercer y primer cuartil, Q_3 y Q_1 dichos cuartiles y k una constante positiva, a la que le asignaremos el valor 1.5. Se eliminarán aquellas observaciones en las que al menos una de sus características tenga un valor considerado fuera de lo normal.

Así pues, eliminamos 16 muestras, quedando un total de 162. En la imagen 9 se muestra la salida del nodo "Numeric Outliers", el cual nos dice cuántos outliers ha detectado para cada variable.

Row ID	S Outlier ...	I Membe...	I Outlier ...	D Lower bound	D Upper bound
Row0	Alcohol	178	0	10.362	15.663
Row1	Malic acid	178	3	-0.61	5.27
Row2	Ash	178	3	1.693	3.072
Row3	Alcalinity of ...	178	4	10.625	28.025
Row4	Magnesium	178	4	59.5	135.5
Row5	Total phenols	178	0	0.125	4.405
Row6	Flavanoids	178	0	-1.43	5.45
Row7	Nonflavanoi...	178	0	0.01	0.69
Row8	Proanthocy...	178	2	0.188	3.007
Row9	Color intensity	178	3	-1.325	10.715
Row10	Hue	178	1	0.27	1.63
Row11	OD280/OD315	178	0	0.057	5.037
Row12	Proline	178	0	-227.5	1,712.5

Imagen 9. Salida detallada del nodo "Numeric Outliers".

Podemos destacar que se han eliminado muchos de los outliers que habíamos considerado, como los correspondientes a la ceniza, magnesio, flavonoides, intensidad del color y tono, y algunos adicionales que no habíamos considerado, como tres valores en el ácido málico, cuatro valores en la alcalinidad de las cenizas, dos adicionales en el magnesio y uno adicional en la intensidad de color.

Tras haber eliminado estos outliers, repetimos la ejecución del algoritmo de clustering jerárquico. El diagrama de distancias mostrado en la figura 10 nos indica que el número de clústers apropiado podría ser 3 o 4. No hay gran diferencia respecto del diagrama de

distancias anteriormente expuesto, pero si es cierto que el usar 5 o 6 clústers se descarta con mayor facilidad al estar más lejos del primer codo de la curva.

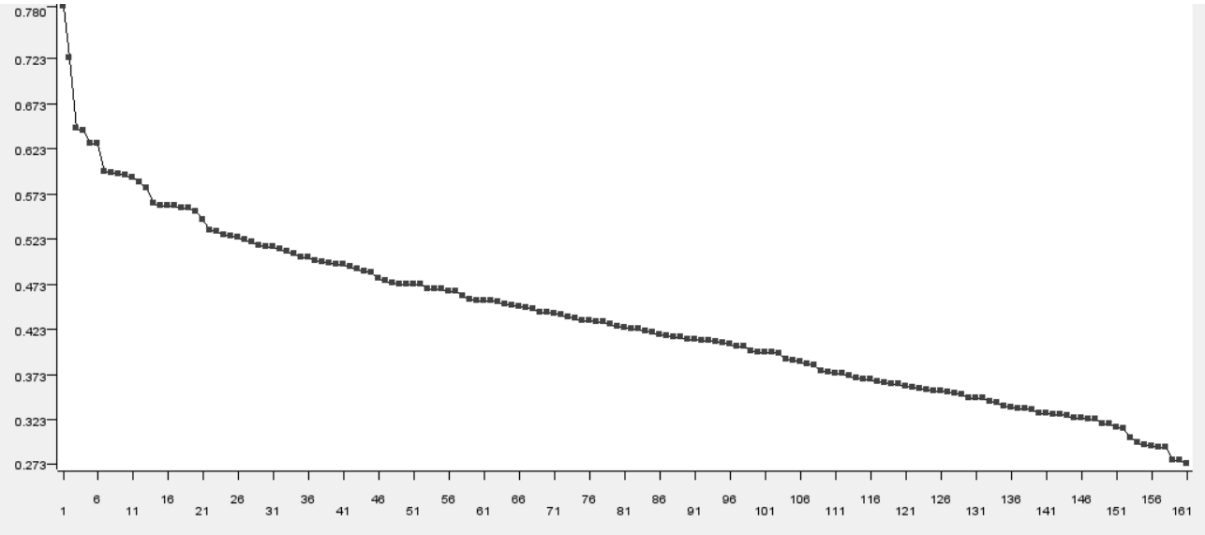

Imagen 10. Diagrama de distancias tras aplicar clustering jerárquico sobre wines.data sin outliers.

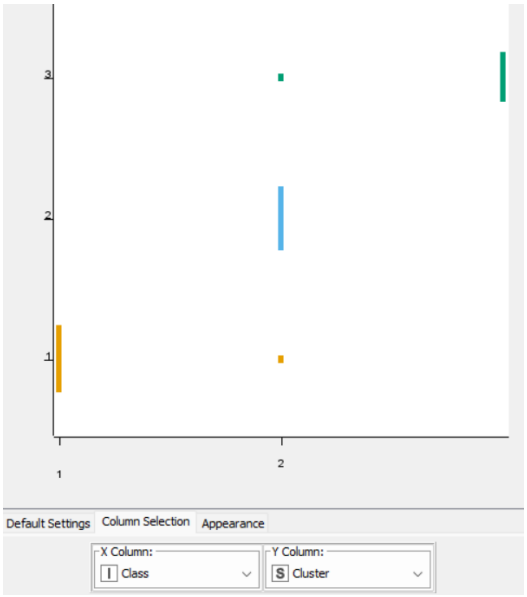

Imagen 11. Clústers vs asignación original.

Si aplicamos el algoritmo k-Means, especificando que se agrupen los datos 3 clústers, obtenemos el resultado que se muestra en la figura 11. Podemos ver como la mayoría de los datos se clasifican correctamente, habiendo seis instancias de los tipos de vino 1 y 3 (tres para cada clase) que se clasifican como vinos de tipo 2. La precisión de la clasificación es del 96.3%. Estos corresponden a los puntos naranjas y verdes en la columna central. El que los datos se clasifican correctamente es quizás más fácil de ver en una tabla en la que se enfrenta la clasificación original y el clúster asignado.

A continuación, probaremos a reducir la dimensionalidad de los datos.

En primer lugar, calcularemos la matriz de correlación para ver si parejas de variables con una correlación lineal (positiva o negativa) alta, de manera que podamos eliminar una de ellas. Eliminaremos una de las variables cuando el grado de correlación sea mayor a 0.6 en valor absoluto. Haciendo uso de los nodos “Linear Correlation” y “Correlation Filter” realizamos dicho filtrado.

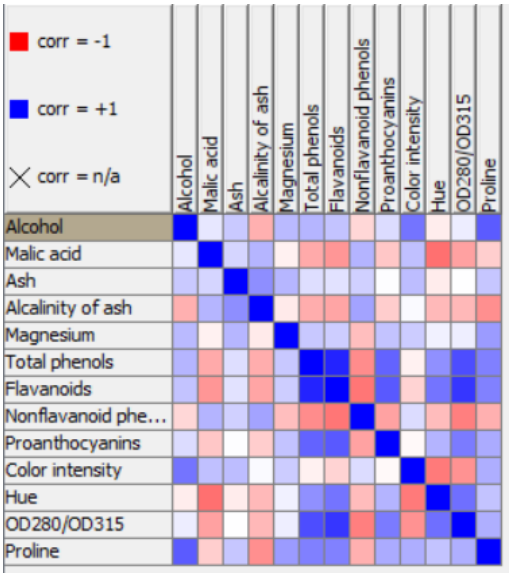

Imagen 12. Matriz de correlación.

En la figura 12 se muestra la matriz de correlación. A partir de ella, obtenemos las siguientes parejas de variables con correlación mayor a 0.6 (valor exacto entre paréntesis):

- Flavonoides y los fenoles totales (0.8646)
- Alcohol y prolina (0.6437)
- Fenoles y OD280/OD315 (0.6999)
- Flavonoides y OD280/OD315 (0.7872)
- Fenoles y proantocianinas (0.6124)
- Flavonoides y proantocianinas (0.6527)

Eliminaremos las columnas correspondientes a los flavonoides, proantocianinas, OD280/OD315 y prolina; quedándonos así 9 de las 13 características originales.

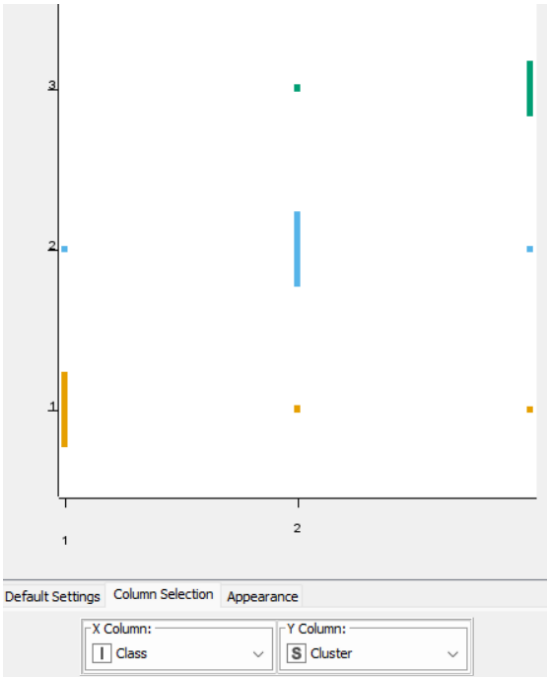

Imagen 13. Clústers vs asignación original.

Si volvemos a aplicar el algoritmo k-Means con 3 clústers, obtenemos la clasificación que se representa de forma gráfica en la figura 13.

En esta ocasión, se producen 7 errores de clasificación en lugar de 6. Hay dos vinos de la clase 2 que se clasifican como de la clase 1, dos vinos de la clase 2 que se clasifican como clase 3, un vino de la clase 3 que se clasifica como clase 1 y otro que se clasifica como clase 2, y un vino de la clase 1 que se clasifica como clase 2. La precisión total de la clasificación realizada es del 95.7%.

También, probaremos a aplicar PCA (análisis de componentes principales) para reducir la dimensionalidad de la base de datos sin outliers, pero con las 13 características originales.

Si reducimos a 9 dimensiones, obtenemos una precisión en la clasificación del 95.7%. De los 7 errores, seis de ellos corresponden a clasificar vinos de la clase 2 como vinos de la clase 3, siendo el restante clasificar un vino de la clase 2 como vino de la clase 1.

En el caso de que reduzcamos a 5 dimensiones, la precisión aumenta hasta el 96.9%. En esta ocasión, tres vinos de la clase 2 serán clasificados como de la clase 3, y dos vinos de la

clase 2 serán clasificados como de la clase 1. La precisión es ligeramente mayor que si usáramos la base de datos, con todas sus características, tras haber eliminado los outliers.

No se muestran la distribución de los clústers respecto a las clases de la primera columna para dimensionalidades 5 y 9, pues en el primer caso esta es idéntica a la mostrada en la imagen 11, y en el segundo caso es prácticamente idéntica, existiendo una única muestra mal clasificada (de otro color) en la columna central.

Por último, haremos que los datos se resuman en 2 variables. En esta ocasión, la precisión es del 96.3%, dado que hay una muestra adicional de la clase 2 clasificada como si fuera de la clase 1. El porcentaje de acierto es el mismo que si aplicáramos el algoritmo de agrupación sobre la base de datos sin outliers, con todas las características para cada muestra de esta.

La ventaja de aplicar PCA de forma que los datos se resuman en dos variables es que podamos visualizar las agrupaciones generadas por el algoritmo k-Means en función de dichas variables en un plano. Esto se muestra en la figura 14, en la cual se pueden ver como los datos se agrupan en 3 grupos bien diferenciados.

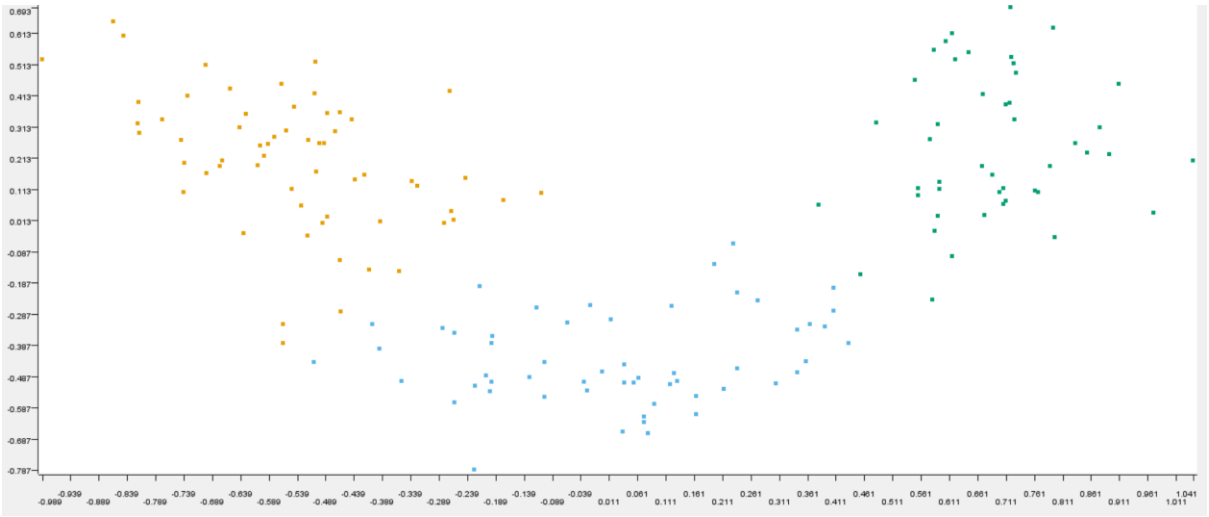

Imagen 14. Muestras agrupadas haciendo uso del algoritmo k-Means y 3 clústers.

Continuaremos este apartado aplicando el algoritmo de agrupamiento DBSCAN sobre el conjunto de datos original normalizado. Para poder utilizar este algoritmo deberemos elegir dos parámetros, el número mínimos de puntos para ser núcleo de un clúster y el diámetro de los entornos (eps), y la elección de estos parámetros no siempre es sencilla.

Para seleccionar el número mínimo de puntos se suele utilizar el logaritmo neperiano del número total de muestras. En nuestro caso, $\ln(178) \cong 5$. Por otra parte, la decisión del parámetro eps no es tan sencilla. Por lo que he podido consultar, se utilizan diagramas de distancias kNN y se comprueba cuando se produce un codo, eligiéndose el menor valor posible. Este procedimiento nos recuerda a cómo elegimos el número de clústers a partir del diagrama de distancias generado por un algoritmo de clustering jerárquico. Sin embargo, KNIME no parece disponer de nodos para realizar esta representación, por lo que

hemos seguido un procedimiento de prueba y error. Además, el guion nos indica que utilicemos el nodo DBSCAN de Weka, pero este no tiene salida de datos, de manera que no podemos generar gráficos como el mostrado en la figura 13. Por tanto, utilizaremos el nodo del paquete "Analytics > Mining" integrado en KNIME, que genera los mismos agrupamientos.

Manteniendo el valor 5 para el número mínimo de puntos variamos eps, obteniendo los siguientes resultados.

- Eps = 0.1 → 0 clústers, por lo que se consideran las 178 muestras como ruido.
- Eps = 0.2 → 0 clústers, por lo que se consideran las 178 muestras como ruido.
- Eps = 0.3 → 0 clústers, por lo que se consideran las 178 muestras como ruido.
- Eps = 0.4 → 4 clústers, de manera que hay 10, 69, 8 y 14 instancias en cada uno. 78 instancias clasificadas como ruido.
- Eps = 0.5 → 104 muestras en un clúster, 51 en otro clúster, 23 muestras como ruido.
- Eps = 0.6 → 167 muestras en un clúster, 11 muestras como ruido.

En la imagen 15 se muestran los diagramas para eps = 0.4 (izquierda) y eps =0.5 (derecha).

Imagen 15. Clústers generador por DBSCAN vs asignación original; eps = 0.4 (izq) y eps = 0.5 (der).

A la luz de los resultados expuestos, podemos concluir que DBSCAN no funciona bien para este conjunto de datos, obteniéndose resultados mucho mejores con el algoritmo k-Means, apoyándonos en clustering jerárquico para obtener el número apropiado de clústers. En este caso conocíamos este último valor de antemano, pero no siempre será así.

4. Base de datos “NBA_RegularSeason_2021_2022”.

Antes de cargar esta base de datos en KNIME, modificamos el nombre de las columnas en Microsoft Excel, dado que algunas tienen nombres excesivamente largos, al contener la descripción de la variable. Durante el proceso de carga en KNIME, obviamos las columnas “RANK”, “FULL_NAME” y “TEAM”. La primera de ellas está vacía, la segunda no nos proporciona información útil alguna, pues el desempeño de un jugador no depende de cómo se llame, y la tercera se elimina por la dificultad de codificar X equipos (con un número de jugadores similar) de manera que sea útil, además de que creo que no me aportará demasiada información.

En el marco del análisis exploratorio, detectamos valores nulos en cinco columnas. Dado que los valores perdidos en dichas columnas son relativamente pocos (oscilan entre el 1 y el 7%), nos limitaremos a aplicar un mecanismo de imputación por media.

Posteriormente, decidimos lidiar con posibles outliers. Dado que en este caso tenemos un gran número de variables no nos detenemos a comentar cada una de ellas de manera individual y pormenorizada. En su lugar, aplicamos el nodo “Numeric Outliers” para eliminar aquellos valore que caigan fuera del rango $R = [Q_1 - k(IQR), Q_3 + k(IQR)]$, siendo IQR la diferencia entre el tercer y primer cuartil, Q_3 y Q_1 dichos cuartiles y k una constante positiva.

Si utilizamos k=1.5 como en el apartado anterior, el algoritmo considera la mitad de la base de datos como outliers, pasando de tener cerca de 720 observaciones a apenas 370, lo cual creemos que dista mucho de la realidad. Si aumentamos el valor de k a 3, de manera que el algoritmo sea más permisivo, nos quedamos con unas 590 observaciones. Particularmente, las variables que según el algoritmo tienen más outliers son la siguientes:

- 2P% (porcentaje de tiros de 2 puntos), con 72 outliers.
- eFG% (porcentaje efectivo de tiros), con 41 outliers.
- TS% (porcentaje real de tiro), con 42 outliers.
- ORTG (valoración ofensiva), con 25 outliers.

Dado que me parece una cantidad excesiva de outliers, especialmente en las tres primeras características, decidimos no eliminar ninguno y lidiar con ellos en el proceso de clustering. Así pues, no se eliminan jugadores considerados estrellas, tanto ofensivas (Lebron James, Kevin Durant o Luca Doncic) como defensivas (Rudy Gobert o Paul George), y jugadores que debido a su poco tiempo de juego no tienen estadísticas demasiado altas.

En el baloncesto hay 5 posiciones, pero un mismo jugador puede jugar en varias posiciones en una misma temporada, para un mismo o diferentes equipos. Inicialmente pensaba que había datos repetidos, hasta que me fijé mas a fondo en este hecho. A pesar de haber 5 posiciones, en la base de datos se especifican 7 posiciones. Hay 3 posiciones “básicas” (nótese que no se hace referencia a los bases ni aleros):

- F: ala-pívot o ala-caza. → Son, por lo general, jugadores altos que juegan cerca del aro, hábiles en el rebote y con buen tiro en posiciones dentro de la zona.
- C: pívot o centro. → Son, por lo general, los jugadores más altos del equipo, jugando en posiciones cercanas al aro, tanto en ataque como en defensa.
- G: escolta. → Jugadores cuya función es dar apoyo al base a la hora de subir el balón, y por otro, en desempeñar las funciones de anotador desde el perímetro.

Y otras 4 posiciones compuestas por dos letras, en las que entendemos que la primera letra hace referencia al rol principal y la segunda letra a un rol secundario que pueden tomar puntualmente según el desarrollo del partido o la mecánica del equipo. Entendemos que en estos grupos se enmarcan bases y aleros.

Dada esta clasificación, a mi juicio no demasiado clara, vamos a intentar agrupar los jugadores en tres grandes grupos, jugadores principalmente ofensivos, en tanto que suelen estar más avanzados en el campo enemigo, principalmente defensivos, en tanto que están más cerca de la canasta del equipo y balanceados o mixtos, que jugarán por los laterales y se moverán entre ataque y defensa de manera constante, por lo que suelen estar en el centro del campo.

En primer lugar, filtraremos algunas de las características de la base de datos, como la edad, los partidos y minutos jugados que no creemos que nos den demasiada información al respecto. También obviamos algunas características que se pueden calcular a partir de otras, como eFG%, TS% o TRB%. A las restantes, entre las que se encuentran los tiros libres, canastas de 2 y 3 puntos, rebotes, asistencias, robos y bloqueos, aplicamos un análisis de componentes principales, de manera que los datos se resuman en 2 dimensiones.

Serán estas dos dimensiones las que pasaremos al algoritmo k-Means, el cual clasificará los datos en 3 clústers. En la imagen 16 podemos observar un diagrama que enfrenta las posiciones de los jugadores y su rol.

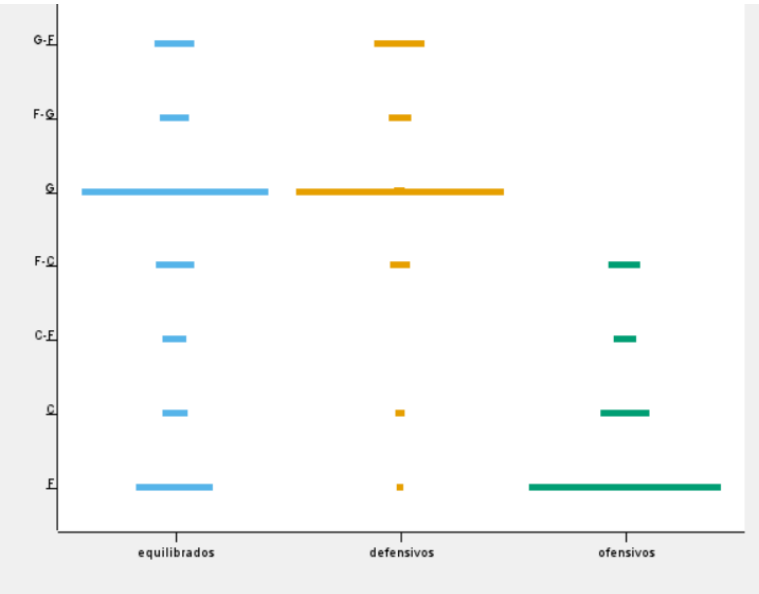

Imagen 16. Clasificación de los jugadores según su rol principal.

Encontramos una división bastante clara. Los jugadores ofensivos suelen ser ala-pívot o pívots, y en dicho grupo no se encuentra ningún escolta o base. Por el contrario, la mayoría de jugadores defensivos son bases o escoltas, con algún ala-pívot muy puntual. Sin embargo, uno podría esperar que los jugadores que más alternan entre defensa y ataque fueran los G-F o F-G (bases y aleros en teoría), pero parecen ser los escoltas "puros" y parte de los ala-pívot.

El número de muestras en el clúster de jugadores ofensivos es de 208, mientras que en los clústeres defensivos y equilibrados hay 219 y 289 muestras, respectivamente.

En la imagen 17 se muestran en el eje X los clústers creados y en el eje Y la variable que refleja el número de tiros triples. Podemos ver como la mayoría de dichos tiros triples son lanzados por los jugadores que se mueven por los laterales. Sin embargo, como podemos ver en la imagen 18, el porcentaje de éxito es relativamente similar entre las diferentes agrupaciones. Parece lógico que los jugadores que más triples han lanzado, los equilibrados, también sean quienes más fallen.

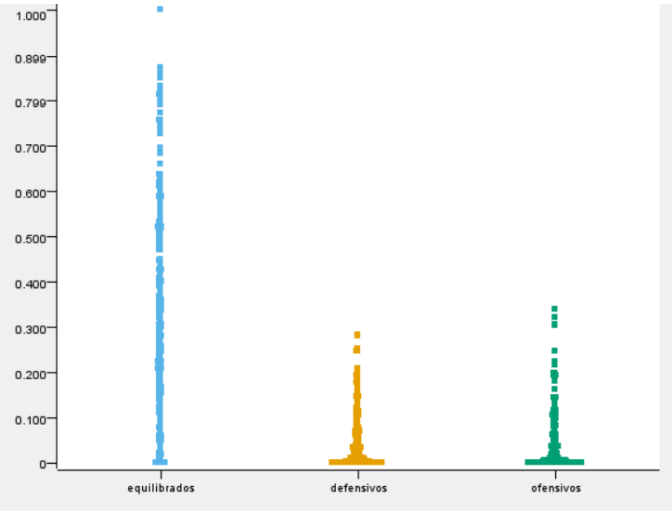

Imagen 17. Triples lanzados en función del grupo de jugador.

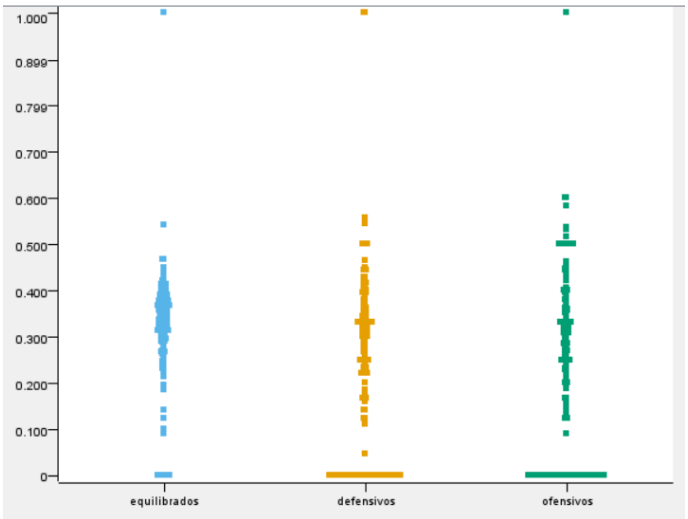

Imagen 18. Triples encestados en función del grupo de jugador.

Algo similar sucede en el caso de los tiros libres lanzados y encestados, así como los tiros a canasta de 2 puntos lanzados y encestados.

Son los jugadores equilibrados quienes más lanzan, y, aunque no hay grandes diferencias entre las tres agrupaciones, son los jugadores ofensivos quienes más encestan.

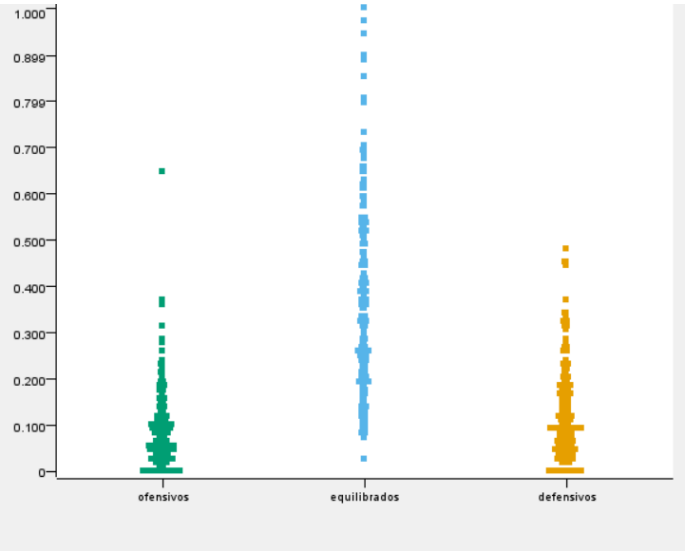

Imagen 19. Asistencias realizadas en función del grupo de jugador.

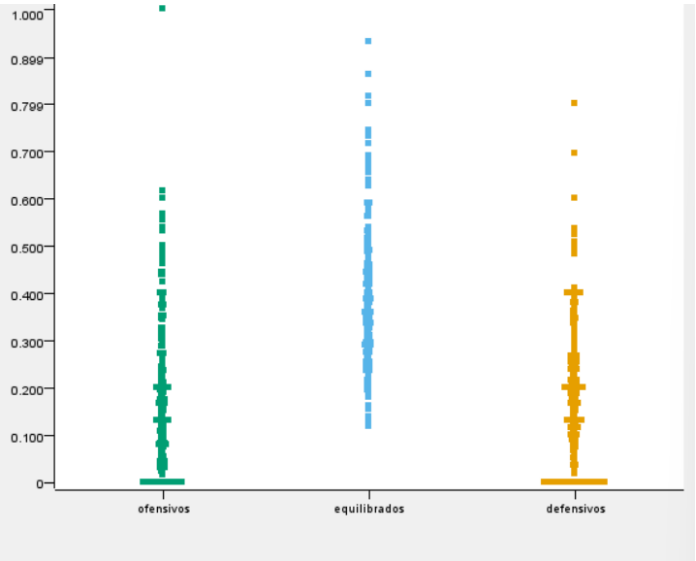

Imagen 20. Robos de balón realizados en función del grupo de jugador.

Si nos fijamos en la imagen 19, veremos que los jugadores equilibrados son quienes más asistencias realizan por partido, no habiendo demasiada diferencia entre atacantes y defensores.

Por otra parte, en la imagen 20 podemos observar que también son los jugadores que se mueven por el centro del campo los que más robos de balón realizan, siendo los siguientes los atacantes, y los últimos los defensores.

En la imagen 21 comparamos los minutos jugados (eje X) y los puntos anotados (eje Y) por parte de los jugadores. Como podemos ver, tanto jugadores ofensivos como defensivos son los que permanecen menos tiempos en el campo, no llegando a jugar la mayoría de los casos ni el 50% del partido. No se aprecian demasiadas diferencias en el número de puntos anotados.

A igualdad de tiempo jugado, los jugadores equilibrados parecen anotar algunos puntos más. Dado que los jugadores de este grupo tienen de jugar más minutos por partido, también son los que más puntos anotan; de ahí esa predominancia de las muestras azules hacia la parte superior derecha del gráfico, aunque siempre hay alguna excepción.

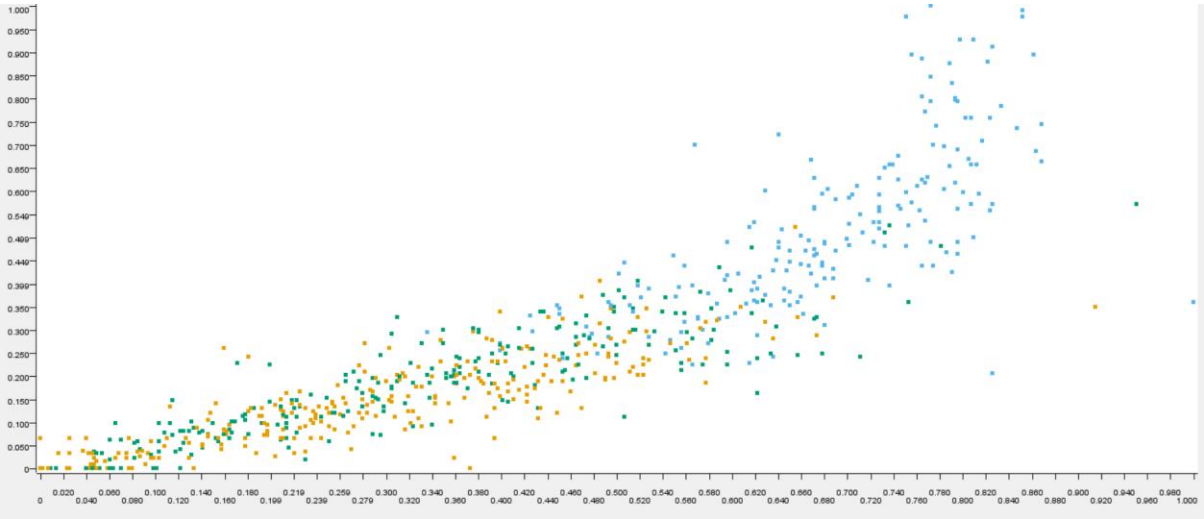

Imagen 21. Minutos jugados (eje X) contra puntos anotados (eje Y).

En la imagen 22 se pueden ver las muestras en el plano formado por las dos dimensiones del PCA. Podemos ver cómo los clústers están bastante diferenciados, encontrándose los jugadores equilibrados a la izquierda, mientras que los ofensivos están en la parte derecha superior y los defensivos en la parte derecha inferior.

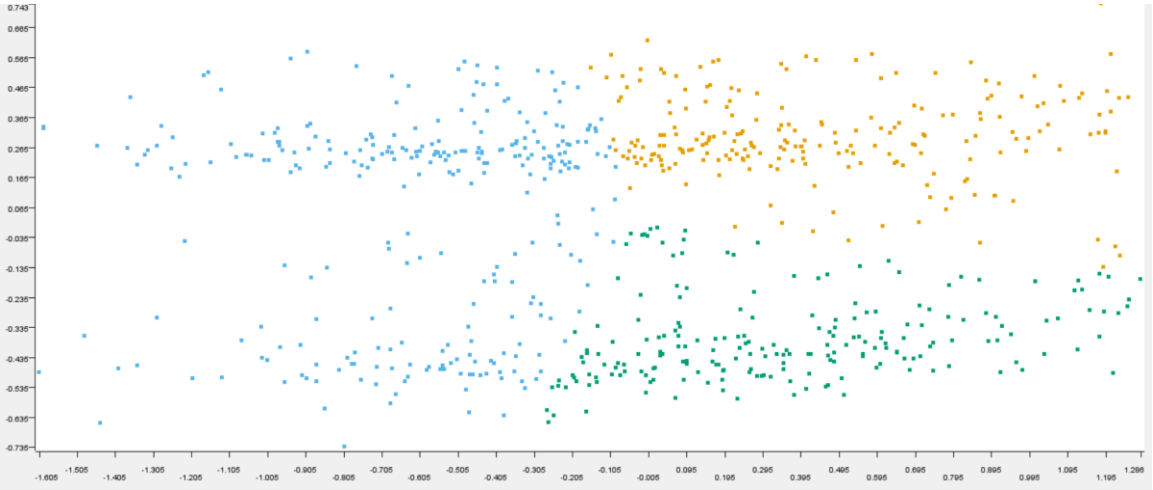

Imagen 22. Jugadores representados en el plano formado por las componentes principales.

Para el resto de características consideradas no parece haber diferencias notables entre las agrupaciones. La polivalencia de los jugadores y el hecho de que varias posiciones compartan los mismos atributos dificulta mucho la diferenciación en grupos claros de los jugadores a partir de las características dadas.

Se intentaron realizar otras agrupaciones haciendo uso de diferentes sets de atributos. En una de las series de experimentos se utilizaron únicamente los atributos referidos a características ofensivas, como puntos marcados, tiros libres y triples encestados, rebotes, etc. En la otra serie de experimentos, se utilizaron atributos más orientados a la defensa y utilidad (de los cuales hay menos), como el número de asistencias, robos, bloqueos y

recuperaciones de balón. En ninguno de los dos casos se consiguieron resultados distintos de los mostrados anteriores, de manera que no pudimos encontrar ningún patrón o grupo claramente diferenciable entre los jugadores principalmente atacantes y aquellos que son principalmente defensores, más allá de la posición. Esto en parte lo atribuyo a que apenas hay estadísticas que indiquen como de bueno es un jugador defensivamente, por lo que cuesta diferenciarlos de los demás.