UNIVERSIDAD
TORCUATO DI TELLA

Tipo de documento: Tesis de maestría

CC BY NC SA

Escuela de Negocios. Master in Management + Analytics

Optimización del scouting en el fútbol argentino

Autoría: Barrán, Nicolás Marcelo
Año: 2024

¿Cómo citar este trabajo?
Barrán, N. (2024). "Optimización del scouting en el fútbol argentino". [Tesis de maestría. Universidad Torcuato Di Tella]. Repositorio Digital Universidad Torcuato Di Tella.
https://repositorio.utdt.edu/handle/20.500.13098/12915

El presente documento se encuentra alojado en el Repositorio Digital de la Universidad Torcuato Di Tella bajo una licencia Creative Commons Atribución-Compartir igual 4.0 Internacional Deed
Dirección: https://repositorio.utdt.edu

Biblioteca Di Tella

UNIVERSIDAD TORCUATO DI TELLA

MASTER IN MANAGEMENT + ANALYTICS

OPTIMIZACIÓN DEL SCOUTING EN EL FÚTBOL ARGENTINO

TESIS

Nicolás Marcelo Barrán

Mayo 2024

Tutor: Nicolás García Aramouni

Resumen

Históricamente en el fútbol profesional, el reclutamiento de jugadores es un proceso fundamental para los equipos. Durante los mercados de pases, se identifican los talentos que más se adecuan a las necesidades de los clubes para mejorar el rendimiento durante la temporada. Generalmente cada equipo establece objetivos de posiciones o jugadores que precisan cubrir y tratan de encontrar la mejor opción de contratación en base al presupuesto con el que cuentan.

Hoy en día, es común que los equipos europeos y sobre todo aquellos clubes de renombre que cuentan con amplios fondos, cuenten con equipos especializados de ingenieros y analistas de datos que se dedican a la optimización de la búsqueda y contratación de nuevos jugadores. La generación de datos con respecto al deporte ha crecido significativamente durante los últimos años, no solo se ha visto en el fútbol profesional, sino que también se ha extendido a muchos deportes de alto nivel tales como el básquet, el béisbol y el fútbol americano entre otros deportes de equipo.

Haciendo foco en el fútbol sudamericano y particularmente en el fútbol argentino, observamos una gran brecha en el uso del análisis de datos y estadísticas con el fin de mejorar los procesos de reclutamiento y contratación en comparación al fútbol europeo. Esta falta de aprovechamiento sobre la generación de información (cada vez mayor) termina siendo una limitación importante para los clubes argentinos en un entorno cada vez más competitivo.

Partiendo de este contexto, enfocaremos este trabajo final en aplicar metodologías de *clustering* y análisis predictivo sobre una base de datos y estadísticas de jugadores de fútbol proveniente de las ligas argentina, uruguaya, chilena, paraguaya y colombiana con el fin de proporcionar una herramienta innovadora para los *scouters* y así optimizar la etapa de reclutamiento durante los mercados de pases.

Empleando el lenguaje de programación R y haciendo uso de *Rstudio* aplicaremos técnicas de *clustering* tales como: *K-means, Partition Around Medoids (PAM), Single Linkage, Average Linkage y Complete Linkage* para poder agrupar a los jugadores y encontrar cuáles son los más similares (con respecto a sus características de juego) a un jugador en particular que queramos seleccionar.

Una vez encontrado un conjunto de al menos cinco jugadores similares, pasaremos a predecir el valor de mercado para estos utilizando y comparando las técnicas de: Regresión Lineal, Regresión *Ridge*, *Random Forest* y *XGBoost*. Podremos usar esta predicción para dar una recomendación de la cantidad de dinero a ofertar por el pase de estos jugadores y utilizando como fuente el reconocido valor de *transfermarkt*, podremos entender si éste se encuentra en un valor razonable, subvalorado o sobrevalorado en comparación a nuestro pronóstico.

OPTIMIZATION OF SCOUTING IN ARGENTINE FOOTBALL

Abstract

Historically in professional football, scouting has been a fundamental process for teams. During transfer windows, clubs try to identify the talents that best suit their needs to enhance performance during the regular season. Generally, each team sets goals for positions or players they need to fill and try to find the best hiring option based on their budget.

Nowadays, it is common for European teams, especially those with big budgets, to have specialized teams of engineers and data analysts dedicated to optimizing the search and recruitment of new players. The generation of data regarding sports has grown so much in recent years that this trend is not only seen in professional football but has also extended to many high-level sports such as basketball, baseball, and American football, among others.

Focusing on South American football, particularly Argentine football, we observe a significant gap in the use of data analysis and statistics to improve recruitment and hiring processes compared to European football. This lack of utilization of the growing information ends up being a significant limitation for Argentine clubs in an increasingly competitive environment.

Starting from this context, this work will focus on applying clustering methodologies and predictive analysis on a database of football players from the Argentine, Uruguayan, Chilean, Paraguayan, and Colombian leagues to provide an innovative tool for scouts to optimize the recruitment stage during transfer windows.

Using the R programming language and Rstudio, we will apply clustering techniques such as K-means, Partition Around Medoids (PAM), Single Linkage, Average Linkage, and Complete Linkage to group players and find those most similar (in terms of their playing characteristics) to a particular player we want to select.

Once we have found a set of at least five similar players, we will proceed to predict the market value for these players using and comparing techniques such as Linear Regression, Ridge Regression, Random Forest, and XGBoost. We can use this prediction to make a recommendation on the amount of money to offer for the transfer of these players, and using the renowned *transfermarkt* value as a source, we can understand if it is in a reasonable, undervalued, or overvalued state compared to our forecast.

Índice

Índice de Tablas

Índice de Figuras

1. Introducción

1.1. Contexto

La problemática que se quiere tratar en este trabajo final está relacionada al análisis de datos aplicados al deporte, en este caso en particular al fútbol masculino profesional. En base a contactos con *scouters* de un club perteneciente al fútbol argentino que mostraron interés en el análisis de datos aplicado al fútbol, se me comentó que, en la actualidad, tanto en Argentina como generalmente en Sudamérica, no se está aprovechando al máximo el uso de las estadísticas para reclutar nuevos jugadores a los equipos, mientras que los clubes de fútbol europeo ya cuentan con equipos de ingenieros y analistas de datos que se dedican en su totalidad a eficientizar y efectivizar la búsqueda y contratación de nuevos jugadores.

1.2. Problema

Hoy en día, el cuerpo técnico de este club utiliza una plataforma llamada *Wyscout*[1]. La misma es un proveedor de datos, estadísticas y video análisis para el fútbol profesional y cuenta con información de equipos, jugadores y partidos de más de 600 competiciones a nivel mundial y cuenta con información y estadísticas de más de medio millón de jugadores profesionales.

La metodología actual de trabajo consta de la exploración de jugadores directamente desde la herramienta a través del uso de filtros que esta misma provee. La principal problemática que se me comentó es la dificultad actual para encontrar jugadores similares utilizando un jugador de referencia. Este problema se podría atacar aplicando un algoritmo de *clustering* para encontrar un set de jugadores con similares características del jugador que se quiere encontrar.

Por otro lado, una variable importante que se puede descargar del sistema es el valor de mercado de los jugadores, el mismo es un valor que se obtiene desde la fuente *transfermarkt*[2] un sitio web especializado en la valoración de los jugadores de fútbol. Ésta no es siempre muy precisa ya que por distintos intereses puede ser que los valores estén sobrevalorados, es por esto por lo que se ampliará este trabajo a una segunda etapa y luego de identificar a los jugadores que podrían ser reclutados, también dar una recomendación de dinero a ofrecer por el pase de estos. Debemos destacar que, si bien la variable de valor de mercado está como opción dentro del sistema de *Wyscout*, hay varios jugadores, sobre todo de ligas menores o pertenecientes a equipos menos relevantes que tienen este dato nulo, por lo que la predicción que se quiere realizar también podría servir para valuar de cierta forma en estos casos particulares.

1.3. Objetivo

La finalidad a la que se apunta con este trabajo final es la de poder realizar en poco tiempo una recomendación, tanto al cuerpo técnico como al club con el que se está trabajando, de jugadores a contratar para reforzar el equipo, utilizando los modelos de

clustering, la idea sería poder brindarle al cuerpo técnico una recomendación de al menos 5 jugadores similares al perfil que desea el director técnico. A su vez, se expandirá este trabajo y además de realizar la recomendación de los jugadores, también se brindará una recomendación de los valores a ofertar por estos.

Para esto, consultaremos principalmente los siguientes trabajos como bibliografía:

Akhanli, S. E., & Hennig, C. (2023). Clustering of football players based on performance data and aggregated clustering validity indexes. *Journal of Quantitative Analysis in Sports*.

Yigit, A. T., Samak, B., & Kaya, T. (2020). An XGBoost-lasso ensemble modeling approach to football player value assessment. *Journal of Intelligent & Fuzzy Systems*.

En el primero se realizó un trabajo muy similar al que se quiere hacer en este trabajo final, donde los autores, utilizando datos estadísticos de la página web *whoscored.com*[3], los cuales son de acceso gratuito, tomaron distintas variables correspondientes a los jugadores, tales como variables personales (edad, altura, peso), variables de posición (asignando un valor dependiendo del rol que ocupa el jugador en el equipo), y variables de *performance* (cantidad de pases, cantidad de goles, quites de balón). A partir de todas estas variables se muestra cómo se realiza un proceso de escalado multidimensional y se procede a utilizar distintas técnicas de *clustering* para segmentar a los jugadores. Las técnicas utilizadas son varias, donde destacamos el *clustering* jerárquico, *clustering* por centroides y *clustering* por densidad entre otros.

En el segundo, se observa un trabajo de valuación de jugadores de fútbol utilizando información proporcionada por el simulador *Football Manager*[4]. Los autores de éste, también haciendo uso de datos intrínsecos y de performance de los jugadores, hacen uso de distintos modelos de predicción tales como regresiones lineales, *Ridge* y árboles de decisión *XGBoost* para lograr generar una valuación de cada uno de los jugadores con el fin de dar soporte a las decisiones de compra y venta de éstos.

El principal diferencial que queremos aportar con este proyecto es el de comenzar a aplicar estas técnicas de análisis de datos tanto para agrupar jugadores con el fin de encontrar similitudes en la forma de juego de estos cómo para valuarlos. De cierta forma se quiere combinar los trabajos anteriormente citados para poder tener ambos resultados en uno. Además, este trabajo aplicaría tanto para el fútbol argentino como para el fútbol sudamericano en sí, donde se carece del uso de estas técnicas.

2. Datos

Cómo se comentó anteriormente, los datos que se utilizarán para el trabajo final provienen directamente de la plataforma *Wyscout*. Esta plataforma cuenta con diversos datos y estadísticas de clubes y jugadores en más de 600 competencias internacionales. El acceso a la base de datos está provisto por los *scouters* del club, quienes a través de la herramienta pueden hacer una descarga masiva de datos y estadísticas de jugadores

en un período de tiempo seleccionado. Esta descarga de información se realizará en formato xls y luego se procederá a trabajar los diferentes ejercicios algorítmicos utilizando R y Rstudio.

Cuando hablamos de datos y estadísticas, nos referimos a la información general de los jugadores como por ejemplo pueden ser la altura, el peso o la edad. También contamos con otro tipo de información como, por ejemplo: equipo en el que juega/jugó en el período seleccionado y competencias de las que participó. Por otro lado, hablando de estadísticas del jugador, encontramos diversas métricas cómo, por ejemplo: acciones defensivas realizadas, duelos aéreos ganados, acciones de ataque exitosas, remates, asistencias, goles, entre otras. La cantidad de minutos en cancha es otro dato que podemos encontrar en el *dataset* y relacionado a la segunda acción que se quiere realizar en este trabajo final, desde *Wyscout* también podemos traer como una variable el valor actual del mercado provisto por el sitio web *transfermarkt*. Esta es la variable para la que se quiere generar una predicción para entender si el valor para cada jugador está sobre o subvalorado.

Si bien estas son algunas de las métricas que podemos encontrar en la plataforma, ésta cuenta con 110 variables diferentes que se pueden descargar por jugador. Debajo se puede observar una tabla que contiene las 51 variables seleccionadas que se utilizarán en este trabajo final. Las mismas fueron clasificadas en 8 categorías distintas con el fin de simplificar su comprensión y se añade un glosario en la sección "Apéndice A. Glosario de variables" donde se puede consultar la definición de cada una.

Tabla 1. Variables por categoría incluidas en el *dataset* a utilizar.

Categoría	Variables
Información general	i. Liga ii. Posición iii. Jugador (nombre) iv. Equipo durante el período seleccionado v. Posición específica vi. Edad vii. Valor de mercado (Transfermarkt) viii. País de nacimiento ix. Pie x. Altura xi. Peso
Tiempo de juego	i. Partidos Jugados ii. Minutos jugados
Acciones defensivas	i. Acciones defensivas realizadas/90 ii. Duelos defensivos/90 iii. Duelos defensivos ganados %
Acciones aéreas	i. Duelos aéreos en los 90 ii. Duelos aéreos ganados %
Recuperaciones	i. Posesión conquistada después de una entrada ii. Posesión conquistada después de una interceptación

Remates	i.	Goles, excepto los penaltis/90
	ii.	xG/90
	iii.	Remates/90
	iv.	Tiros a la portería %
Regates	i.	Acciones de ataque exitosas/90
	ii.	Regates/90
	iii.	Regates realizados %
	iv.	Duelos atacantes/90
	v.	Duelos atacantes ganados %
	vi.	Carreras en progresión/90
	vii.	Aceleraciones/90
	viii.	Desmarques/90
	ix.	Jugadas claves/90
Pases	i.	Toques en el área de penalti/90
	ii.	Pases recibidos /90
	iii.	Pases largos recibidos/90
	iv.	Pases hacia adelante/90
	v.	Precisión pases hacia adelante %
	vi.	Pases cortos / medios /90
	vii.	Precisión pases cortos / medios %
	viii.	Pases largos/90
	ix.	Precisión pases largos %
	x.	xA/90
	xi.	Asistencias/90
	xii.	Pases en el último tercio/90
	xiii.	Precisión pases en el último tercio %
	xiv.	Pases al área de penalti/90
	xv.	Pases hacía el área pequeña %
	xvi.	Pases en profundidad/90
	xvii.	Precisión pases en profundidad %
	xviii.	Pases progresivos/90

Al trabajar directamente con un club del fútbol profesional argentino de primera división, se construirá la base de datos con información principalmente proveniente de las principales ligas de fútbol sudamericano (argentina, uruguaya, colombiana, chilena, y paraguaya). Esta decisión sale a partir del poder adquisitivo del club, siendo que jugadores de otras ligas con mayor renombre, como por ejemplo la liga brasilera, podrían estar fuera del alcance económico del club a la hora de realizar una oferta formal.

Por otro lado, decidimos arbitrariamente no contemplar a aquellos jugadores que tuvieron menos de 1000 minutos de juego (quedándonos con el 45% de las observaciones), ya que entendemos que no son titulares en sus equipos y no cuentan con la participación suficiente para adaptarse rápidamente al plantel si fueran considerados. También hay que aclarar que en ningún momento de este ejercicio consideraremos a jugadores cuya posición sea arquero. Esto se debe a que los jugadores

que cumplen este rol cuentan con variables totalmente diferentes a los jugadores de campo y necesitaríamos construir una base únicamente para éstos.

Por último, con respecto a la dimensión del tiempo y competiciones consideradas, haremos uso de las estadísticas de cada jugador durante la última temporada (año natural 2023) y consideraremos cualquier tipo de competencia oficial en la que se haya participado, lo que incluye partidos de la liga nacional y partidos de competencias internacionales, excluyendo los partidos amistosos.

En conclusión, la base de datos final fue construida a conveniencia tanto con las métricas que se creen más relevantes, como así también que sea proveniente de ligas con similar poder económico al del campeonato argentino. El acceso total a la plataforma se encuentra garantizado por los *scouters*, quienes compartirán toda la información necesaria.

Comenzando con el análisis descriptivo de los datos, al estar enfrentando en primer lugar un problema de clústeres nos focalizaremos en los *features* que componen a éstos. Luego al querer resolver en segundo lugar un problema de predicción, analizaremos cuáles podrían ser los mejores predictores para la variable "Valor de mercado" de cada jugador.

Primero, se realizó un análisis de correlación entre las variables continuas de nuestra base de datos. Para esto, construimos una tabla de correlaciones a la cual le sumamos un mapa de calor para facilitar el análisis visual de estos resultados. Debido a la gran dimensión de esta tabla, se puede visualizar la misma en la sección "Apéndice B. *Features correlations*"

Analizando la tabla de correlaciones, nos encontramos con algunos resultados bastante esperados, donde los *features* de acciones defensivas tienen alta correlación positiva entre sí, como así también sucede lo mismo entre los *features* de las acciones ofensivas y que a su vez tienen correlaciones negativas entre ellos. Por ejemplo, podemos observar que "acciones defensivas realizadas/90" está altamente correlacionado positivamente con "duelos defensivos/90" (0.83) y "Posesión conquistada después de una interceptación" (0.88). Por otro lado, "Acciones de ataques exitosas" tiene gran correlación positiva con "Regates/90" (0.93) y "Duelos atacantes/90" (0.84). También, como hemos comentado, podemos ver las correlaciones negativas entre los *features* de acciones defensivas y ofensivas: "Acciones defensivas realizadas/90" tiene gran correlación negativa con "Goles/90" (0.70) y "Toques en el área de penalti/90" (0.73). Decimos que estos resultados eran bastante esperados ya que en un equipo de fútbol cada jugador tiene un rol muy predeterminado en el equipo, generalmente los jugadores defensivos no suelen participar de los ataques y viceversa, lo que termina generando que se den este tipo de correlaciones positivas entre defensivas por un lado y ofensivas por otro y a su vez negativas entre sí.

Segundo, construimos una tabla con los promedios de cada uno de los *features* y la dividimos por posición del jugador en la cancha separando entre defensores, mediocampistas y delanteros, para terminar de confirmar la hipótesis de que la posición

va a determinar la importancia que tiene cada *feature* para sí. Esta tabla se puede observar en el "Apéndice C. Promedio *features* por posición".

Tal como lo suponíamos, podemos ver en la tabla que para aquellos jugadores que son defensores, los *features* de acciones defensivas tienen mayor importancia y cuentan con promedios más altos por acción. En contraposición, sucede exactamente lo mismo con los jugadores que son delanteros, donde las acciones ofensivas cuentan con mayores promedios por acción, dando a entender que estas son más relevantes para la posición. Por otro lado, vemos que los mediocampistas se destacan en las *features* relacionadas a los pases, ya que en esta posición generalmente se busca mayoritariamente la generación de juego y como era de esperarse suelen estar en el intermedio entre las acciones defensivas y ofensivas.

Tercero, ampliando un poco más el análisis de los principales *features* continuos de nuestra tabla de base, exploramos la misma tabla que en el punto anterior, con la ligera modificación de ver los resultados por liga en lugar de posición para ver si logramos descifrar algún patrón interesante. Esta tabla también se puede observar en el "Apéndice D. Promedio *features* por liga".

En esta tabla, encontramos resultados bastante llamativos como que los jugadores de la liga colombiana cuentan con la mayor cantidad de *features* con el menor promedio entre las ligas. De los 38 *features* contemplados, la liga colombiana tiene el menor promedio en 20 de estos, seguido por la liga uruguaya con menor promedio en 10 de estos. Esto se puede deber principalmente a la competitividad entre los equipos de esta liga, donde los equipos son más parejos entre sí, en contraposición de compararlo con la liga argentina, donde al haber una mayor cantidad de equipos en el torneo de primera división, la competitividad de los equipos "grandes" se superpone sobre el resto de los equipos. Por otro lado, la liga chilena se muestra con el mayor promedio en 16 de los 38 *features*, lo cual nos hace pensar que el fútbol chileno cuenta con partidos con mayor cantidad de acciones y un juego mucho menos friccionado comparado con el resto de las ligas. También, al trabajar directamente con un club del fútbol argentino, lo que se puede analizar esta liga en particular es que esta cuenta con la mayor cantidad de promedios intermedios en 14 de los 38 *features*. Como comentamos anteriormente, podemos suponer que esto se puede deber a la disparidad de los equipos que conforman el torneo argentino, donde incluso también se resaltan preferentemente las acciones más defensivas, donde los equipos tienen un estilo de juego mucho más conservador y se lo considera un fútbol con mucha fricción.

Cuarto, comenzando a hacer hincapié en la variable de valor de mercado, construimos un gráfico para comparar estos por liga y por posición. El mismo lo podemos observar debajo.

Figura 1. Valor de mercado por liga y posición (en miles de dólares)

A simple vista podemos observar que los jugadores de la liga argentina son los más valiosos de todos, promediando un valor de $1.326.650 dólares por jugador, este valor es más que sorprendente, llegando a ser más del doble que los jugadores de la liga chilena (segunda liga con mayor valor de mercado por jugador). Más allá de la influencia de la liga argentina, podemos suponer que esto se debe también a la buena reputación de los jugadores argentinos en el mercado (los cuales predominan en el torneo del mismo país y muchos son buscados por equipos europeos con mayor poder adquisitivo). Haciendo mayor foco a nivel de posiciones, podemos ver claramente que en todas las ligas es mayor el valor de mercado asignado a los delanteros que a los defensores o mediocampistas. Esta pareciera ser una tendencia internacional y no solo sudamericana si observamos que los grandes fichajes de la historia (los de mayor paga por jugador) son por lo general hacia los delanteros, los mismos que siempre ocupan un mayor rol en las votaciones de premios individuales y a mejores jugadores (como por ejemplo en el galardón del balón de oro, donde este año 2023 los tres principales nominados fueron también delanteros).

Quinto, para terminar de confirmar este supuesto, se construyó otro gráfico de barras para comparar el valor de mercado por nacionalidad (considerando las mismas nacionalidades que las ligas utilizadas previamente). Los resultados fueron los siguientes.

Figura 2. Valor de mercado por nacionalidad (en dólares)

Aquí podemos terminar de confirmar que más allá de la liga, los jugadores de nacionalidad argentina suelen ser mejor valorados que el resto de las nacionalidades. Por otro lado, la gran sorpresa de este gráfico es el valor que muestran los jugadores de nacionalidad uruguaya, los cuales divididos únicamente por nacionalidad tienen un valor del 36% superior al promedio de su liga (ubicados en cuarta posición en el gráfico anterior).

Sexto, realizamos una tabla de correlación entre el valor de mercado y cada uno de los *features* continuos que analizamos al comienzo de este ejercicio. Esta tabla se puede observar en el "Apéndice E. Correlación valor de mercado".

En principio, observando el mapa de calor donde asignamos como valores extremos 1 y -1, se puede ver que ningún *feature* tiene una correlación lo suficientemente grande ya sea positiva o negativa para influir directamente en el valor del jugador, de hecho, todas las correlaciones son muy bajas. Esto nos hace suponer que entonces el valor de mercado estará más relacionado a la combinación de posición-*feature* donde seguramente la ponderación de las acciones jugará un rol clave en el valor de mercado de jugador dependiendo la posición que se esté buscando. A su vez también se puede apoyar el supuesto mencionado en el punto número cuatro, que, en el general de los jugadores, los *features* de acciones ofensivas correlacionan positivamente en mayor medida que el resto de los *features*, lo que indica que los delanteros suelen ser más "valiosos" que el resto de los jugadores.

Séptimo y último, se construyó otra tabla de correlación entre el valor de mercado y cada uno de los *features* continuos ahora abierta por posición para analizar el peso que tiene cada uno de estos dependiendo del rol del jugador en el equipo. Los detalles se pueden observar en la tabla 2 que se encuentra debajo:

Tabla 2. Correlación: valor de mercado por posición.

Correlación entre valor de mercado y *features* continuos	Delantero	Mediocampista	Defensor
Acciones defensivas realizadas/90	0.038	0.022	0.022
Duelos defensivos/90	0.037	0.035	0.001
Duelos defensivos ganados, %	0.035	0.029	0.062
Duelos aéreos en los 90	-0.095	-0.074	0.007
Duelos aéreos ganados, %	0.021	-0.012	-0.004
Posesión conquistada después de una entrada	0.095	0.059	0.140
Posesión conquistada después de una interceptación	0.052	0.079	0.151
Acciones de ataque exitosas/90	0.104	0.142	0.063
Goles, excepto los penaltis/90	0.180	0.132	0.086
xG/90	0.117	0.067	0.102
Remates/90	0.200	0.127	0.064
Tiros a la portería, %	-0.009	0.019	-0.004
Regates/90	0.104	0.141	0.069
Regates realizados, %	-0.044	0.034	-0.098
Duelos atacantes/90	0.084	0.130	0.067
Duelos atacantes ganados, %	0.035	0.128	-0.036
Toques en el área de penalti/90	0.095	0.074	0.072
Carreras en progresión/90	0.071	0.186	0.190
Aceleraciones/90	0.092	0.114	0.116
Pases recibidos /90	0.052	0.281	0.308
Pases largos recibidos/90	-0.033	0.023	0.164
Pases hacia adelante/90	-0.001	0.139	0.211
Precisión pases hacia adelante, %	0.009	0.116	0.202
Pases cortos / medios /90	0.031	0.281	0.330
Precisión pases cortos / medios, %	0.050	0.138	0.189
Pases largos/90	-0.003	-0.006	-0.039
Precisión pases largos, %	0.054	0.084	0.030
xA/90	0.076	0.068	0.031
Asistencias/90	0.073	0.051	0.008
Desmarques/90	0.005	0.085	0.175
Jugadas claves/90	0.104	0.076	0.091
Pases en el último tercio/90	-0.020	0.135	0.116
Precisión pases en el último tercio, %	0.081	0.152	0.184
Pases al área de penalti/90	0.042	0.092	0.046
Pases hacía el área pequeña, %	0.045	0.081	0.063
Pases en profundidad/90	0.014	0.089	0.051
Precisión pases en profundidad, %	-0.027	0.057	0.044
Pases progresivos/90	0.006	0.120	0.095

Tal como pensábamos anteriormente, vemos ligeros cambios sobre la correlación de los predictores con el valor a predecir cuándo se lo relaciona directamente con la posición. Para los delanteros vemos que los *features* más importantes son “Remates /90” y “Goles, excepto los penaltis/90”, dado que estos son los encargados de convertir los goles para ganar los partidos. Con respecto a los mediocampistas, al ser estos los responsables de la generación del juego, los predictores más influyentes son aquellos relacionados a los pases, tales como “Pases cortos / medios /90” y “Pases recibidos /90”. Por último, para los defensores, vemos dos espectros de atributos importantes, por un lado, aquellos referentes a las acciones defensivas, tales como “Posesión conquistada después de una entrada” y “Posesión conquistada después de una interceptación” y, por otro lado, aquellos que se refieren a la generación de juego como “Pases recibidos /90” y “Pases cortos / medios /90” al igual que los mediocampistas. Esto último puede deberse a que es muy amplio el espectro de jugadores defensivos, ya que no existen solo aquellos que se ocupan posiciones de zagueros centrales con un rol netamente defensivo, sino que también cuentan con los defensores laterales, quienes acompañan

en mayor medida a los mediocampistas y contribuyen bastante a la generación de juego además de dar soporte defensivo.

3. Metodología

Cómo se comentó en la introducción, dividiremos este trabajo en dos etapas. En una primera instancia realizaremos un ejercicio de *clustering* para poder encontrar un set de jugadores similares a un jugador seleccionado. Para esto, utilizaremos 5 métodos diferentes de *clustering*:

i. *K-means*
ii. *Partition Around Medoids (PAM)*
iii. *Simple linkage*
iv. *Average linkage*
v. *Complete linkage*

En la segunda instancia nos enfocaremos en predecir el valor de mercado para el set de jugadores seleccionados a través del algoritmo anterior. En esta ocasión haremos uso de 4 métodos distintos:

i. Regresión lineal
ii. Regresión *Ridge*
iii. *Random Forest*
iv. *XGBoost*

Comenzaremos con la regresión lineal que nos servirá de referencia para luego poder hacer un *benchmark* contra los otros tres modelos y así poder comparar las mejoras que brinda cada uno y entender cuál es el que mejor se ajusta a nuestras necesidades.

Algoritmos de *clustering*

Previamente a comenzar a trabajar con los algoritmos de *clustering* seleccionados, debemos realizar un preprocesamiento de los datos.

En primer lugar, debido a cómo están representadas las variables directamente desde *Wyscout*, no será necesario realizar ningún tipo de cambio sobre la representación de estas. Akhanli & Hennig (2023)[5] realizaron un trabajo muy similar con un *dataset* de 1500 jugadores de fútbol provenientes de las 8 grandes ligas europeas (española, inglesa, italiana, alemana, francesa, rusa, neerlandesa y turca), utilizando como fuente la conocida plataforma *Whoscored*[3]. Cómo bien explican en su texto, las variables estadísticas de los jugadores en su base de datos se veían en números absolutos, por lo que, por un lado, tuvieron que cambiar la representación de estos para visualizar cada dato por partido jugado (dividiéndolo por la cantidad de minutos jugados y multiplicándolo por 90, lo que suele durar cada uno de estos). Esto se hace para salvar las diferencias que pueden surgir simplemente por el hecho de haber tenido más

minutos de participación en cancha. Por ejemplo, un jugador podría tener más remates que otro por el simple hecho de haber jugado más minutos, pero no quiere decir que sea un jugador que remate más al arco por partido jugado. Por otro lado, tuvieron que transformar a forma porcentual aquellas variables que fuesen subcategorías de otra, estas son variables tales como, "tiros a la portería %" o "precisión de pases en profundidad%" que se construyen cómo eventos exitosos sobre la cantidad total de ese evento. La razón de mostrar estas variables en porcentajes también se debe a que facilita la comparación entre los jugadores, ya que, si fuesen simplemente conteos, éstas estarían descontextualizadas y un jugador que haya tenido más remates a portería podría depender únicamente del hecho de contar con más remates globales. Cómo se puede observar en la sección de datos, tabla 1, nuestra base de datos ya cuenta con todas las variables estadísticas de cada jugador cada 90 minutos jugados y también con las variables de cada subcategoría expresadas en porcentaje de la variable global. Estos formatos ya vienen por defecto desde el software de *Wyscout*, como se explicó anteriormente.

En segundo lugar, realizaremos transformaciones a las variables de la categoría "información general" que queremos capturar, pero su formato original es categórico y queremos que este sea continuo. Para esto aplicaremos, específicamente a las variables "Liga", "Posición" y "Pie" la técnica de *one-hot-encoding*, la cual convierte las variables categóricas en variables binarias (0 o 1) creando una columna para cada categoría posible. La principal razón detrás de realizar esta transformación a estas variables yace en poder contar con esta información a la hora de realizar el *clustering* ya que hay metodologías que usaremos que no permiten este tipo de variables tal como K-medias.

En tercer lugar, removeremos aquellas variables que no son relevantes para esta primera etapa del trabajo. Al querer realizar un ejercicio de *clustering* para encontrar jugadores con similares características en el estilo del juego, no nos interesa contar con algunas variables que se encuentran dentro de la categoría "información general" (ver tabla 1), tales como: Equipo durante el período seleccionado, posición específica, edad, valor de mercado, partidos jugados, minutos jugados, país de nacimiento, altura y peso. Por otro lado, y cómo comentamos en el punto anterior, necesitaremos remover todas las variables que no sean numéricas para poder aplicar los algoritmos de *clustering*. En nuestro caso, luego de quitar las variables anteriormente nombradas nos quedaría únicamente como variable no numérica el nombre del jugador, que en nuestra base de datos cumple el rol de identificador de registro.

En cuarto lugar y antes de comenzar a aplicar las técnicas de *clustering*, estandarizaremos todas las variables. Esto se hace con el fin de evitar que las variables con valores nominales mayores, como en el caso de nuestra base de datos pueden ser aquellas variables porcentuales que tienen valores absolutos entre 0 y 100 dominen la distancia euclidiana, la cual es utilizada por más de uno de los algoritmos que aplicaremos, lo que nos asegurará que cualquiera sea la variable, contribuya de manera más equitativa a la distancia. También servirá para luego poder hacer una mejor interpretación de los resultados, ya que al estar todo en la misma escala, se puede hacer

una lectura directamente de la distancia a la que se encuentran las observaciones en términos de desviaciones estándares de la media de cada variable.

K-means

Para comenzar con la identificación de jugadores similares, primero aplicamos el algoritmo de *K-means*. Definido por James, Witten, Hastie & Tibshirani (2013)[6] cómo: "Un enfoque simple y elegante para particionar un conjunto de datos en "K" clústeres distintos y no superpuestos. Para realizar el *clustering K-means*, primero debemos especificar el número deseado de clústeres "K"; luego, el algoritmo *K-means* asignará cada observación a exactamente uno de los K clústeres.".

En otras palabras, *K-means*, es un método de agrupamiento cuyo objetivo es dividir al conjunto de datos en "K" grupos distintos, donde cada uno de éstos sea representado por un centroide. Suele ser un procedimiento muy útil para descubrir nuevas estructuras internas en los datos que no son inmediatamente perceptibles.

Para comenzar con su aplicación es necesario definir de antemano el número de clústeres "K" que se quiere identificar. Luego de haber definido este valor, el algoritmo de *K-means* procede a asignar cada una de las observaciones del conjunto de datos a uno de los "K" grupos, de tal manera que se minimice la variación dentro de cada clúster. Cabe resaltar que la variación mencionada anteriormente se mide usualmente utilizando la distancia euclidiana al cuadrado entre las observaciones y el centroide del clúster al que fue asignado.

En términos matemáticos, se empieza asignando los "K" centroides, habitualmente de manera aleatoria. Luego, cada observación se asigna al clúster cuyo centroide sea el más cercano, a partir del cálculo de la distancia euclidiana. Posteriormente, se vuelven a calcular los centroides de cada grupo como el promedio de las observaciones asignadas a dicho clúster. Por último, se repite iterativamente este proceso hasta que las asignaciones de las observaciones a cada grupo ya no cambian, indicándonos que *K-means* convergió.

La finalidad del algoritmo es minimizar la varianza intragrupo medida como la suma de las distancias euclidianas al cuadrado entre cada registro y el centroide de su clúster. En base a esto, podemos plantear formalmente que se busca minimizar la función objetivo:

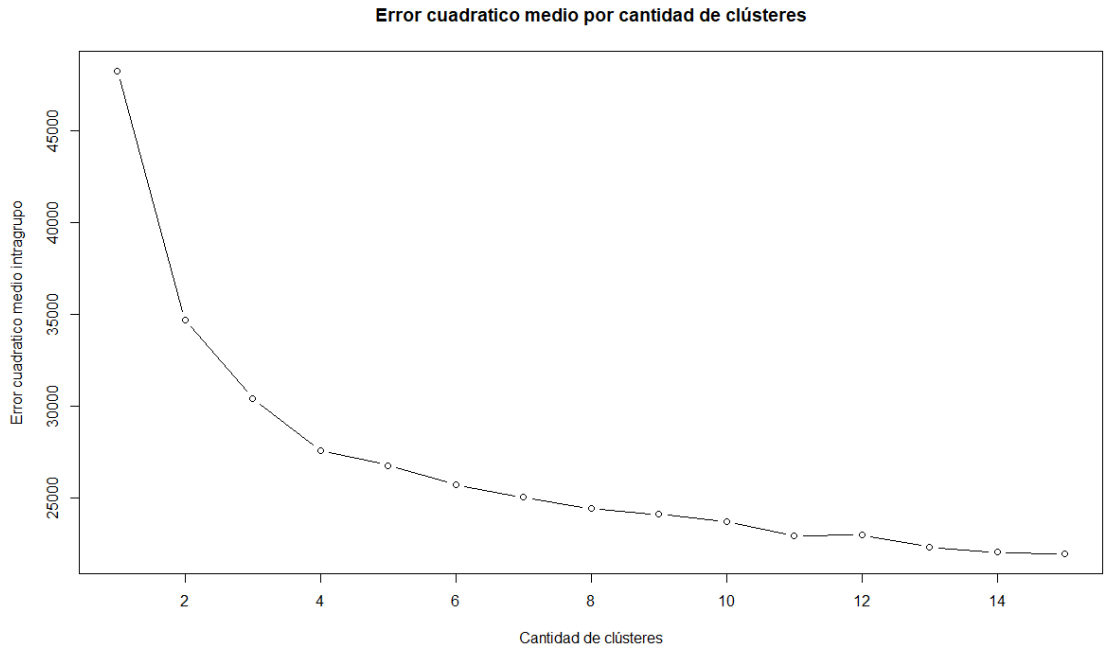
$$\min_{C_1,\dots,C_k} \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - \mu_k\|^2 \quad (1.1)$$

Donde $C_k$es el conjunto de observaciones en el clúster k, x_i es una observación en el clúster C_k y μ_k es el centroide del clúster k.

Para aplicar el algoritmo, comenzaremos determinando el número óptimo de clústeres a utilizar empleando el método del codo. Para esto calculamos la suma total del error cuadrático medio dentro de cada grupo para un rango de posibles números de clústeres (de 2 a 15 clústeres) y graficamos cómo varía este error en función del número de grupos. En nuestro caso, y como se puede observar en la figura 3, encontramos que

el número óptimo de clústeres era 4, ya que agregar más clústeres no reducía significativamente el error cuadrático medio intra-clúster.

Figura 3. Variación del ECM en función de la cantidad de clústeres (*K-means*).



Tal como se comentó previamente, podemos ver en la figura 3 como disminuye en gran medida el error cuadrático medio cuando pasamos de un clúster a dos, o de dos a tres y mismo de tres a cuatro. Luego de los cuatro clústeres la curva comienza a disminuir la pendiente y pierde sentido continuar agregando más grupos. Es este punto donde se forma el "codo" de la curva y el que a priori seleccionaríamos para continuar desarrollando este algoritmo.

Con el fin de asegurarnos que la elección de cuatro clústeres es óptima, realizamos una reducción de dimensionalidad con PCA y ploteamos los puntos con color según el número de clústeres. Para este algoritmo de *K-means*, generamos los gráficos para comparar la división entre 4 clústeres (figura 4) y 5 clústeres (figura 5) como podemos observar debajo.

Figura 4. Reducción de dimensionalidad con PCA: *K-means* 4 clústeres.

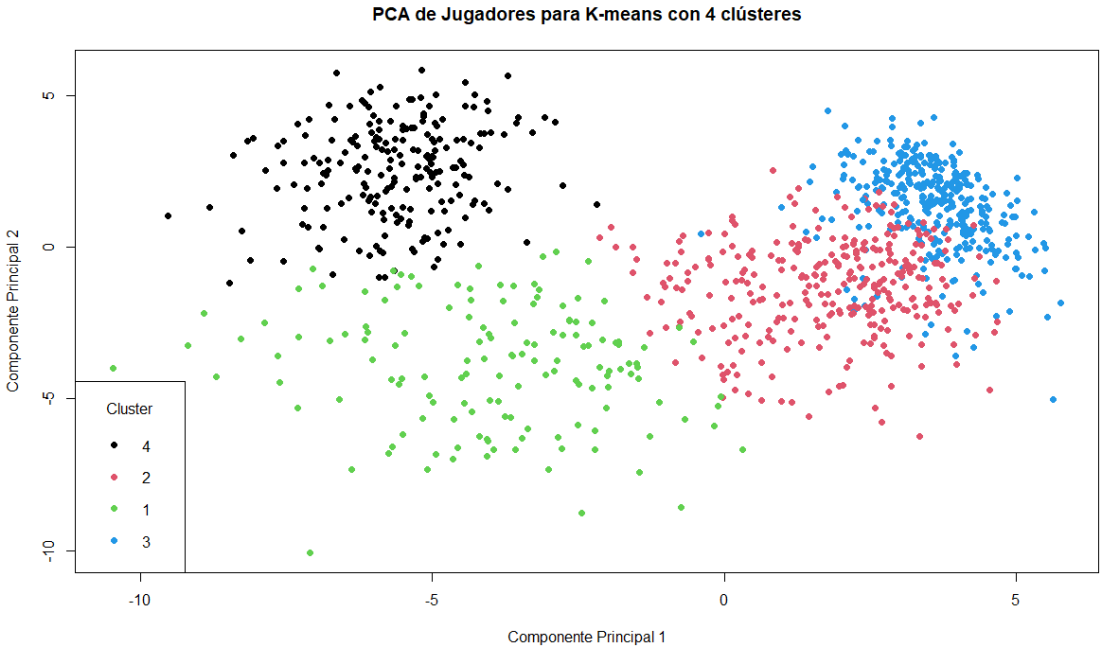
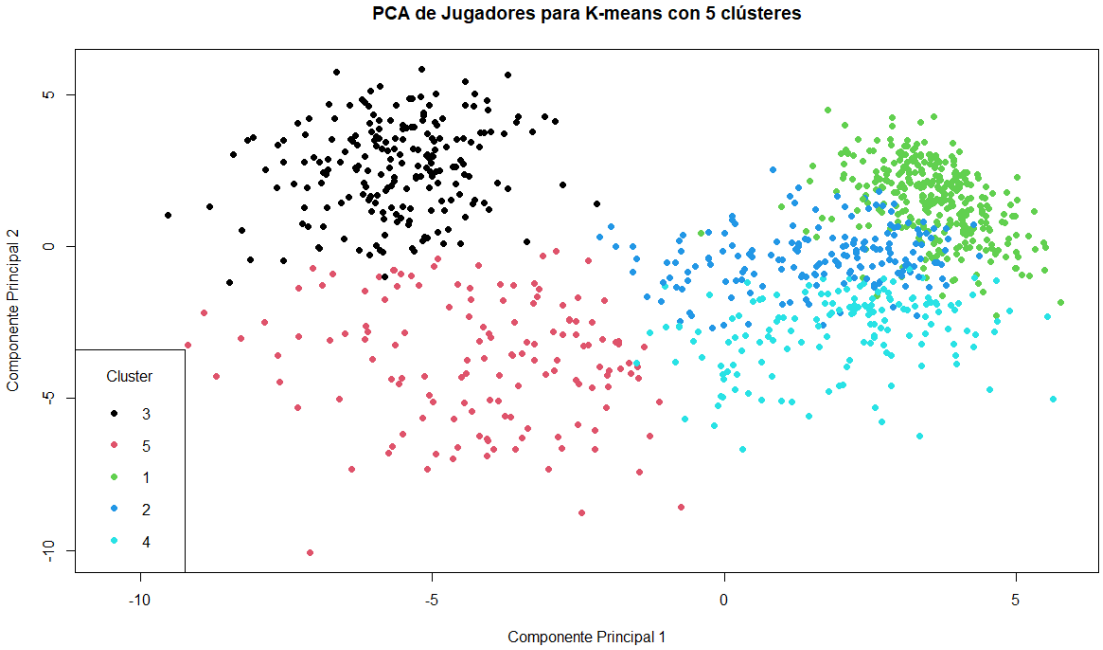


Figura 5. Reducción de dimensionalidad con PCA: *K-means* 5 clústeres.



Realizando un análisis un poco más exhaustivo de los resultados y adentrándonos en las observaciones, podemos decir que las divisiones que se producen son más bien posicionales, futbolísticamente hablando. Decimos esto, ya que los clústeres cuatro (figura 4) y tres (figura 5) que se generan en la parte superior izquierda de cada gráfico, corresponden a una agrupación principalmente predominada por jugadores con roles defensivos. Por otro lado, las observaciones que se encuentran en los clústeres tres

(figura 4) y uno (figura 5) que se localizan en la parte superior derecha de cada gráfico, prevalecen de jugadores ofensivos, primordialmente delanteros. En contraparte de todo esto, los restantes clústeres para ambos gráficos abundan de jugadores clasificados como mediocampistas. Cómo bien sabemos estos últimos tienen roles mixtos, por lo que podrían asemejarse a cada uno de los grupos anteriores dependiendo el rol que ejerza cada jugador y a esto se debe la ubicación que ocupan en estos planos.

En ambos ejercicios de *clustering*, ya sea *K-means* con cuatro o cinco clústeres, podemos detectar rápida y visualmente que la agrupación correspondiente a los defensores se conserva. Hemos probado incrementar el número de clústeres a seis y siete y las divisiones de estos continúan generándose en mayor medida hacia el sector derecho del gráfico dejando mayormente a las observaciones hacia la izquierda casi intactas.

El último factor que consideramos para terminar de confirmar el número de clústeres a utilizar es el *Average Silhouette Width (ASW)*. Cómo se especifica en otro de los trabajos de Akhanli & Hennig (2020)[7]: "*The Average Silhouette Width (ASW)* compara la disimilitud promedio de una observación con los miembros de su propio clúster con la disimilitud promedio con los miembros del clúster más cercano al que no está clasificado. Fue uno de los mejores métodos para estimar el número de clústeres en el estudio comparativo de Arbelaitz et al. (2012). Para i = 1, …, *n* define *ASW*:

$$S_i = \frac{b_i - a_i}{\max\{a_i, b_i\}} \in [-1,1],\text{"} \quad (2.1)$$

Siendo b_i la disimilitud promedio de la observación i con los miembros del clúster más cercano al que no pertenece y a_i la disimilitud promedio de la observación i con los otros miembros de su propio clúster.

"Donde:

$$a_i = \frac{1}{\mathrm{n}_{l_i} - 1} \sum_{x_j \in C_{l_i}} d(x_i, x_j), \quad (2.2)$$

$$b_i = \min_{h \neq l_i} \frac{1}{n_h} \sum_{x_j \in C_h} d(x_i, x_j).\text{"} \quad (2.3)$$

Definiendo a i como el índice que representa una observación particular en el conjunto de datos, j como el índice que representa otra observación diferente a i y que se utiliza para calcular la disimilitud con i. C_{l_i}es el clúster al que pertenece la observación i e C_h un clúster diferente a C_{l_i}. Por otro lado, n_{l_i} es el número de observaciones en el clúster C_{l_i} e n_h es el número de observaciones en el clúster C_h. Por último, el término $d(x_i, x_j)$ hace referencia a la disimilitud entre las observaciones x_i, x_j.

"Entonces *ASW* se define como:

$$I_{ASW}(C) = \frac{1}{n} \sum_{i=1}^{n} s_i \text{ ."}. \quad (2.4)$$

A partir del cálculo de ASW podemos decir que este oscila entre valores de -1 a 1, donde un valor más cercano a 1 significa que las observaciones están bien separadas entre clústeres y cohesionadas dentro de cada uno. Por otro lado, si el valor se asemeja

a 0, esto quiere decir que las observaciones se encuentran muy cerca del límite de decisión entre clústeres y los valores negativos hacen referencia a que hay registros que pueden haberse asignado de forma incorrecta.

Además de utilizar esta medida como ayuda para terminar de seleccionar la cantidad de clústeres, será una medida que emplearemos en todos los algoritmos de *clustering* para poder comparar los resultados y tomarla de ayuda para seleccionar la mejor opción. En este caso particular para *K-means*, si comparamos los resultados de ASW para cuatro y cinco clústeres, vemos que estos tienen un puntaje de 0.170 y 0.144 correspondientemente.

Finalmente, por estas razones mencionadas anteriormente, procedimos a realizar el ejercicio con un valor de cuatro clústeres. Los resultados finales se pueden apreciar de forma simplificada en la figura 4 y nos guardamos el valor de ASW de 0.170 para comparar el rendimiento con los próximos algoritmos.

Partition Around Medoids (PAM)

El segundo algoritmo que aplicaremos para identificar jugadores similares es *Partition Around Medoids (PAM)*. Kaufman & Rousseeuw (1990)[8], describieron que: "La idea de este algoritmo de particionamiento es la siguiente: Para obtener k clústeres, el método selecciona k objetos (llamados objetos representativos) en el conjunto de datos. Los clústeres correspondientes se encuentran luego asignando cada objeto restante al objeto representativo más cercano. Por supuesto, no toda selección de k objetos representativos da lugar a un agrupamiento "bueno". La clave es que los objetos representativos deben ser elegidos de manera que estén (en cierto sentido) centralmente ubicados en los clústeres que definen. Para ser exactos, la distancia promedio (o disimilitud promedio) del objeto representativo a todos los demás objetos del mismo clúster se minimiza. Por esta razón, llamamos a un objeto representativo óptimo el medoide de su clúster, y al método de particionamiento alrededor de medoides lo llamamos la técnica de k-medoides."

Al igual que *K-means, PAM* también pertenece a los llamados *partitioning clustering methods*. La principal diferencia radica en que *PAM* utiliza medoides en lugar de centroides, siendo un medoide una observación representativa que está centralmente ubicada dentro de un grupo, minimizando la disimilitud promedio entre él y los otros registros que se encuentran en el mismo clúster.

Para esto, el algoritmo de PAM se divide en dos fases principales: *Build* y *Swap*. En la primera, como bien indica su nombre, se comienza con la construcción de los grupos. Se inicia seleccionando a la primera observación que minimiza la disimilitud con el resto de las observaciones, la cual es la más central del conjunto de datos. Luego se elige el siguiente medoide que reduce la función objetivo lo máximo posible, siendo esta:

$$\min_{m_1,\ldots,m_k} \sum_{k=1}^{K} \sum_{i \in C_k} d(x_i, m_k) \quad (3)$$

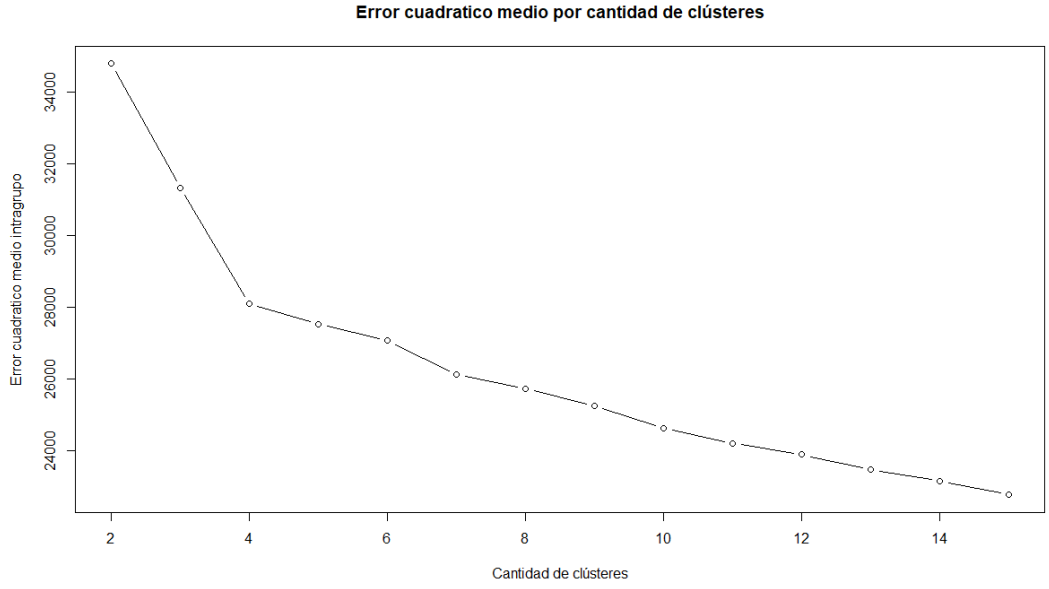
Donde m_k es el medoide del k-ésimo clúster, C_k es el conjunto de todos los objetos asignados al k-ésimo clúster y $d(x_i, m_k)$ representa la disimilitud entre la observación x_i y el medoide m_k.

La fase de construcción finaliza cuando cada objeto se asigna a su medoide más cercano, formando los clústeres iniciales. El algoritmo continúa con la fase de *Swap*, en la cual se intercambian los medoides conseguidos en la fase *Build*. Para mejorar el conjunto de medoides y el resultado del ejercicio de *clustering* se consideran todos los pares de objetos (m, i) donde m es un medoide e i no lo es y se evalúa el efecto de intercambiar m e i en la disimilitud total del clúster. Si el intercambio disminuye la disimilitud total, se realiza el cambio y se vuelve a iterar. Este proceso se repite hasta que ya no sea posible continuar recortando las disimilitudes totales.

A diferencia de *K-means*, se resalta que *PAM* es más robusto frente a valores atípicos, ya que minimiza las disimilitudes en lugar de las distancias cuadráticas, también permitiendo mayor flexibilidad a partir de la medida de disimilitud seleccionada.

Con relación a la empleabilidad del algoritmo *PAM* a nuestro trabajo final y al tener un funcionamiento muy similar a *K-means*, repetimos los pasos que utilizamos en éste y lo primero que hicimos fue aplicar el método del codo para tener una noción de la cantidad de clústeres a usar. Para esto, calculamos el error cuadrático medio intra-clúster para algoritmos *PAM* con valores de "k" clústeres de dos a quince. Los resultados se pueden apreciar en la figura 6 que se encuentra debajo.

Figura 6. Variación del ECM en función de la cantidad de clústeres (*PAM*).



Analizando los resultados obtenidos, y al igual que en *k-means*, encontramos el "codo" de la curva nuevamente a la cantidad de cuatro clústeres. Sin embargo, si miramos bien el gráfico se puede ver otra pendiente un poco más pronunciada cuando pasamos de seis a siete clústeres.

Siguiendo con la metodología propuesta, procederemos a examinar las representaciones de reducción de dimensionalidad con PCA ploteando los puntos con colores para el algoritmo *PAM* con cuatro y siete clústeres con el fin de esclarecer la conveniencia de la cantidad de grupos a seleccionar.

Figura 7. Reducción de dimensionalidad con PCA: *PAM* 4 clústeres.

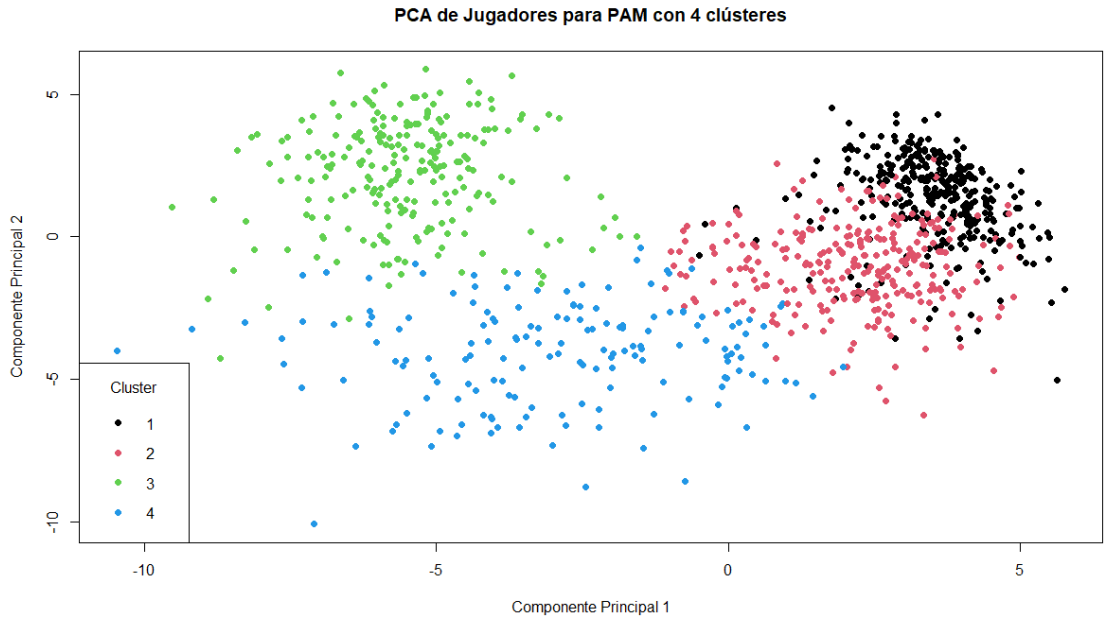
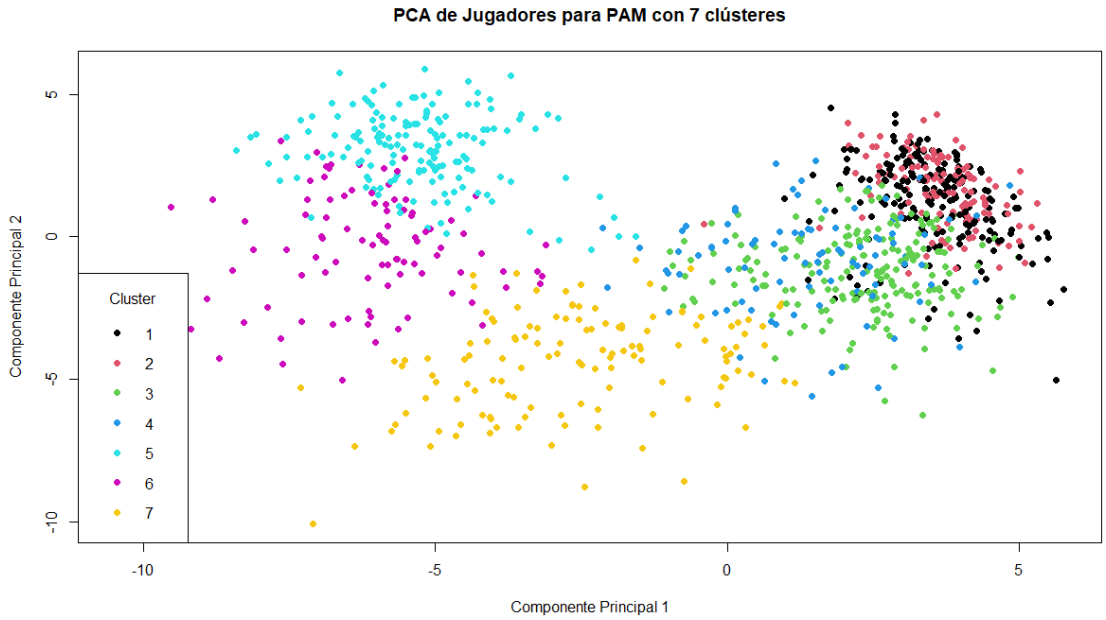


Figura 8. Reducción de dimensionalidad con PCA: *PAM* 7 clústeres.



La figura 7 nos muestra cómo quedarían compuestos los clústeres si realizamos *PAM* con un valor de "k" clústeres de cuatro, mientras que en la figura 8 se pueden ver los resultados para el algoritmo *PAM* con un valor "k" de siete clústeres.

Examinando la figura 7, notamos que los resultados son muy similares a la figura 4, donde habíamos realizado el ejercicio de K-means con cuatro clústeres. No existen diferencias relevantes y son muy poco significativas la cantidad de observaciones que cambian de clúster entre un ejercicio y otro. Por otra parte, si estudiamos la figura 8, también podemos relacionarla a las conclusiones que obtuvimos en *K-means*, donde la mayor complicación radica en el agrupamiento de los jugadores ofensivos, los cuales se encuentran en la parte superior derecha del gráfico. Al agregar más clústeres al ejercicio las conglomeraciones se propagan principalmente en esta zona mientras que las divisiones de los jugadores defensivos parecieran ser más simples y estar mejor segmentadas. A priori creeríamos que sumar más clústeres no enriquecería el estudio por lo que nos inclinaríamos por mantener el *PAM* con cuatro clústeres.

Para terminar de despejar las dudas con respecto a la cantidad óptima de clústeres a seleccionar y también tal como lo hicimos en *K-means*, aplicaremos el método de *ASW* para que nos ayude a tomar la mejor decisión en cuanto a qué valor asignarle a "k" y además poder contar con esta métrica para comparar los rendimientos de ambos algoritmos.

Usando *ASW*, obtuvimos un puntaje de 0.156 para *PAM* con cuatro clústeres y otro puntaje de 0.081 para *PAM* con siete clústeres. Tal como lo supusimos anteriormente, el rendimiento de cuatro clústeres es bastante superior al de siete, donde vemos una significativa baja del 48% entre el primer y segundo ejercicio. Por esta razón finalmente decidimos mantener el algoritmo de *Partition Around Medoids* con cuatro clústeres.

Single Linkage

Dejando de lado los algoritmos particionales, pasamos a trabajar con algoritmos jerárquicos. James, Witten, Hastie & Tibshirani (2013)[6] comentan que: "Una posible desventaja de *K-means* es que requiere que especifiquemos previamente el número de clústeres K. El *clustering* jerárquico es un enfoque alternativo que no requiere que nos comprometamos con una elección particular de "K" (cantidad de clústeres). El *clustering* jerárquico tiene una ventaja adicional sobre *K-means* en que resulta en una atractiva representación basada en árboles de las observaciones, llamada dendrograma.". "El término "jerárquico" se refiere al hecho de que los clústeres obtenidos al cortar el dendrograma a una altura determinada están necesariamente anidados dentro de los clústeres obtenidos al cortar el dendrograma a una altura mayor.".

Particularmente, *Single Linkage*, es un método de *clustering* jerárquico que agrupa las observaciones en función de la distancia mínima entre puntos. Se comienza seleccionando una medida de disimilitud entre cada par de registros del conjunto de datos, que como venimos mencionando en algoritmos previos, usualmente se elige a la distancia euclidiana. Se inicia el algoritmo considerando a cada observación como su propio clúster y se procede a iterar con los siguientes pasos. Primero se identifican los dos clústeres con la menor disimilitud entre sus miembros. Luego, se fusionan estos grupos para formar un nuevo clúster. Esto resulta en que ahora queden $k - 1$ clústeres,

siendo k la cantidad de clústeres. Este procedimiento se repite hasta que todas las observaciones del conjunto pertenezcan a un único clúster.

Matemáticamente hablando, el problema que se quiere resolver con *Single Linkage* es la minimización de la disimilitud de dos puntos pertenecientes a diferentes clústeres. Lo que se podría expresar de la siguiente forma:

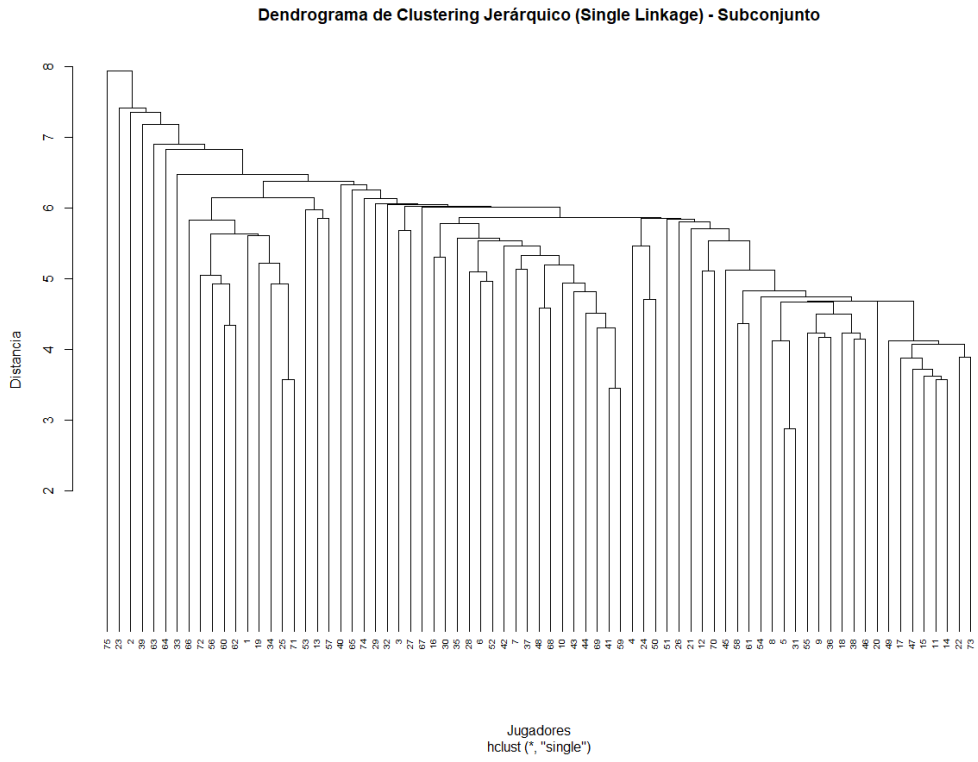
$$D_{SL}(A, B) = \min_{i \in A, j \in B} D_{ij} \ . \ (4)$$

Siendo D_{ij} la disimilitud entre las observaciones i e j

Podemos resaltar que este método es especialmente útil para crear grupos con estructuras alargadas y encadenadas, lo cual también podría ser una desventaja al formar clústeres en cascada donde las observaciones individuales se agrupan de una en una generando una alta sensibilidad a valores atípicos.

Para comenzar con la aplicación del algoritmo *Single Linkage* en nuestro trabajo final, y tal como se sugiere en la bibliografía, construimos un dendrograma. Debido a la gran cantidad de datos, se nos hace imposible generar un dendrograma lo suficientemente claro para estudiar y determinar a partir de éste la cantidad óptima de clústeres "K" a seleccionar. Es por esta razón que establecimos una semilla (con el fin de reproducibilidad del ejercicio) para componer un subconjunto aleatorio de setenta y cinco observaciones con el fin de utilizarlas para crear un dendrograma lo suficientemente claro para analizar. Una vez que conseguimos estos registros, aplicamos *clustering* jerárquico *Single Linkage* utilizando la distancia euclidiana y graficamos el dendrograma. Los resultados se pueden observar en la figura 9 debajo.

Figura 9. Dendrograma de *clustering* jerárquico (*Single Linkage*) para subconjunto



Cómo era de esperarse debido a la definición proporcionada de *Single Linkage*, vemos que hay ramas de gran extensión que generan fusiones individuales de observaciones. En otras palabras y analizando el dendrograma podríamos decir que a medida que aumentáramos la cantidad de clústeres, se suelen mantener clústeres individuales contra un clúster con el resto de las observaciones.

Para definir la cantidad de clústeres con la cual configurar el algoritmo final, nos basamos en las elecciones previas para *K-means* y *PAM*. Hicimos tal elección con el fin de contar con valores de *ASW* razonables, ya que como vimos previamente en el cálculo de esta medida, a mayor cantidad de clústeres es más probable que el puntaje descienda.

A continuación, graficamos las reducciones de dimensionalidad con PCA para obtener un mejor detalle de cómo resultaría cada clúster. Utilizamos valores para "K" de cuatro y diez clústeres. Los resultados pueden observarse debajo en las figuras 10 y 11 correspondientemente.

Figura 10. Reducción de dimensionalidad con PCA: *Single Linkage* 4 clústeres.

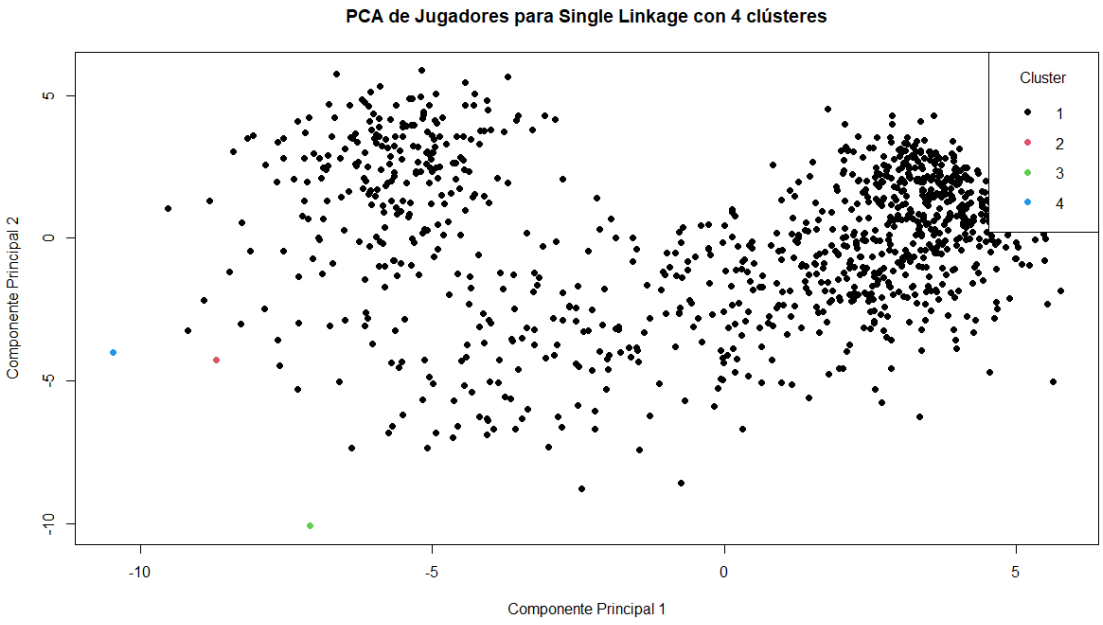
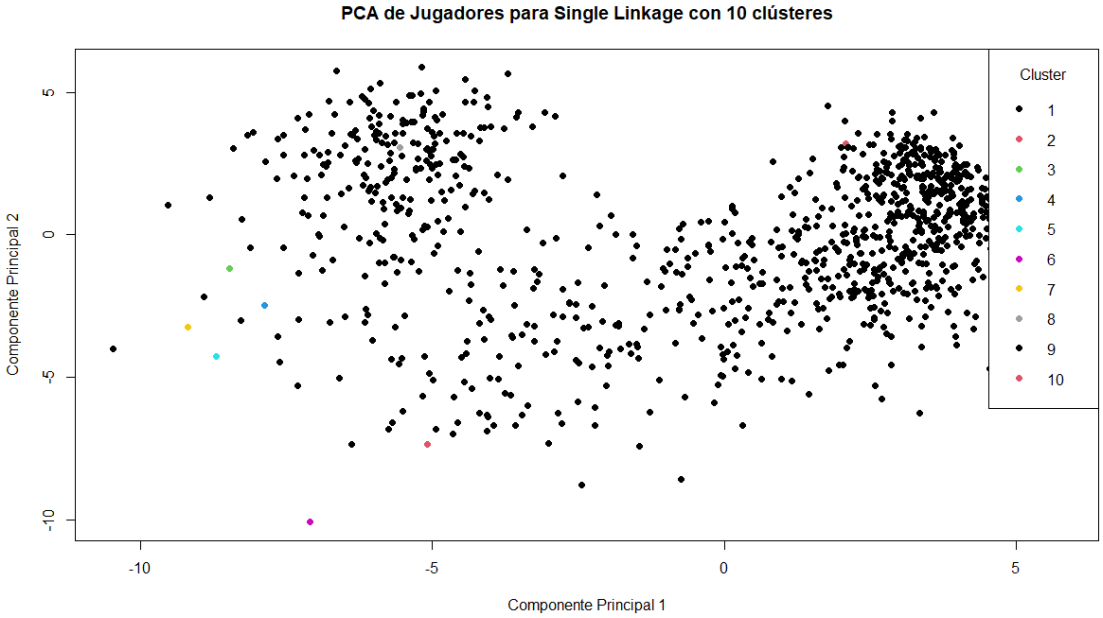


Figura 11. Reducción de dimensionalidad con PCA: *Single Linkage* 10 clústeres.



A simple vista y en ambos gráficos podemos ver que se cumple lo predicho. Examinando la figura 10, podemos notar cómo el algoritmo separa a todas las observaciones en el clúster número uno, mientras que detecta aquellos registros que se encontraban más alejados del resto y les asigna a cada uno de estos un clúster individual. Luego, a pesar de haber añadido seis clústeres más para probar el ejercicio con un valor "K" de diez, continuamos viendo que el algoritmo genera los agrupamientos de la misma manera. Se vuelve a percibir que el clúster número uno engloba a la gran mayoría de registros y se generan otros clústeres con observaciones individuales. Estos resultados podrían ser buenos si quisiéramos encontrar *outliers* en nuestra base de datos, pero definitivamente no lo son para la finalidad de nuestro ejercicio.

Por último, computamos los valores de *ASW* para ambos casos. Para el primer caso, *Single Linkage* con cuatro clústeres, obtuvimos un valor de 0.304, mientras que, para el segundo caso, *Single Linkage* con diez clústeres, el puntaje desciende estrepitosamente a 0.032. A pesar de ver un muy buen valor de *ASW* para el primer caso, visualizando la figura 10 y contemplando lo analizado previamente, procederemos a descartar este algoritmo para la comparación final.

Average Linkage

Tal como se comentó en la sección previa, *Average Linkage* también pertenece a la familia del *clustering* jerárquico. La diferencia con *Single Linkage* radica en la forma en que se calculan las disimilitudes entre clústeres, lo cual afectará directamente a la generación del dendrograma y por ende a los resultados de las asignaciones de clústeres.

Siguiendo con las definiciones de James, Witten, Hastie & Tibshirani (2013)[6], la descripción para *Average Linkage* es: "Disimilitud media entre clústeres. Calcular todas

las disimilitudes entre pares de observaciones en el clúster A y las observaciones en el clúster B, y registrar el promedio de estas disimilitudes.”.

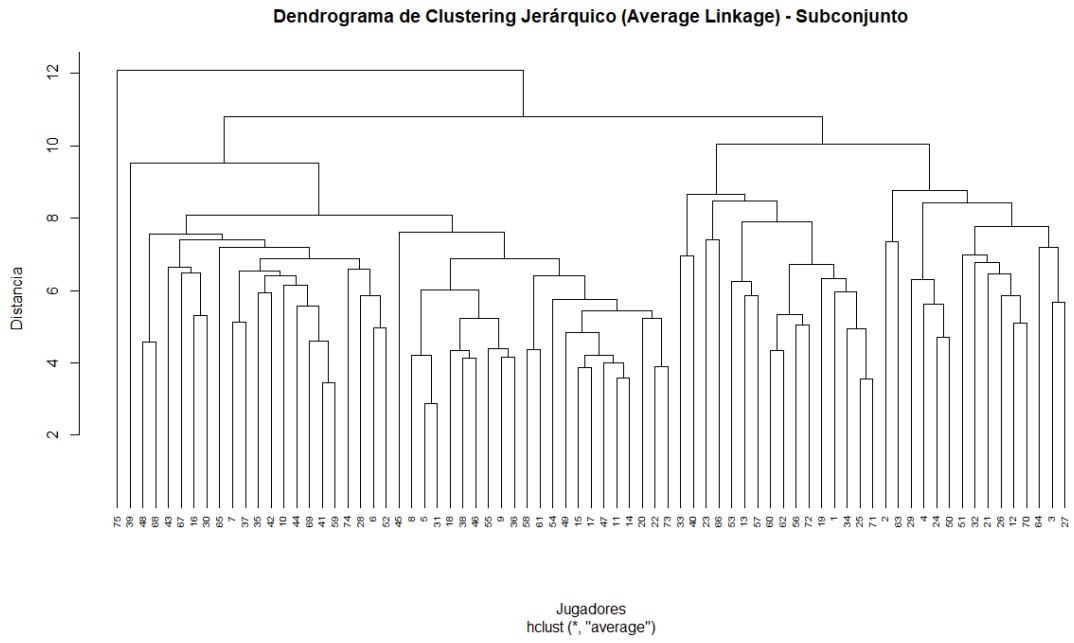
Retomando la explicación en términos matemáticos, en *Average Linkage* el problema que se quiere resolver es el siguiente:

$$D_{AL}(A,B) = \frac{1}{|A| \cdot |B|} \sum_{i \in A, j \in B} D_{ij} \ . (5)$$

En otras palabras, la distancia de enlace entre dos clústeres se calcula como el promedio de las disimilitudes entre todos los pares de puntos, donde cada uno de estos puntos pertenece a un clúster diferente. Ejemplificando, sean dos clústeres “A” y “B”, la distancia de *Average Linkage* será el promedio de todas las disimilitudes entre cada observación asignada al clúster “A” y cada observación asignada al clúster “B”. Es por esto, que este algoritmo tiende a producir clústeres más compactos y equilibrados cuando comparamos los resultados contra un enfoque de *Single Linkage* ya que ahora consideramos la relación promedio entre las observaciones de cada clúster en lugar de la distancia mínima entre estos. Por esta razón los autores comentan que: “Por lo general, se prefieren los algoritmos de *Average Linkage* y *Complete Linkage* por sobre *Single Linkage*, ya que tienden a producir dendrogramas más equilibrados.”.

Continuando con la metodología de aplicación de *Average Linkage* en este trabajo final, seguimos los pasos de lo realizado con *Single Linkage*. Al contar con tantas observaciones en nuestra base de datos y con el fin de obtener una mejor visualización del dendrograma, optamos nuevamente por establecer una semilla (aplicamos la misma que usamos anteriormente, con el fin de poder analizar una comparación de los resultados obtenidos) para contar con setenta y cinco registros al azar. Sobre estos, generamos el dendrograma utilizando *Average Linkage* con distancia euclidiana. Estos resultados se pueden ver en la figura 12 que se encuentra debajo:

Figura 12. Dendrograma de *clustering* jerárquico (*Average Linkage*) para subconjunto



Examinando el dendrograma obtenido y comparándolo contra el de *Single Linkage*, podemos decir que a priori obtuvimos mejores resultados. Si bien en esta muestra se observa primero la separación de un *outlier* como primer clúster (registro 75), luego se puede ver cómo se producen otros dos clústeres más uniformes en tamaño y forma. A su vez, en estos últimos dos clústeres que mencionamos, podemos percibir otra división en dos de cada uno de estos. Esta primera división vuelve a dejarnos un clúster de una única observación (registro 39), mientras que siguen existiendo otros tres grupos más equilibrados.

Siguiendo la línea de análisis, graficaremos la reducción de dimensionalidad con PCA para tener una mejor noción de cómo quedarían conformadas distintas cantidades de clústeres. Para este caso de *Average Linkage*, basándonos en el análisis del dendrograma hecho anteriormente, probaremos dos variantes de cuatro y seis clústeres los cuales podemos visualizar en las figuras 13 y 14.

Figura 13. Reducción de dimensionalidad con PCA: *Average Linkage* 4 clústeres.

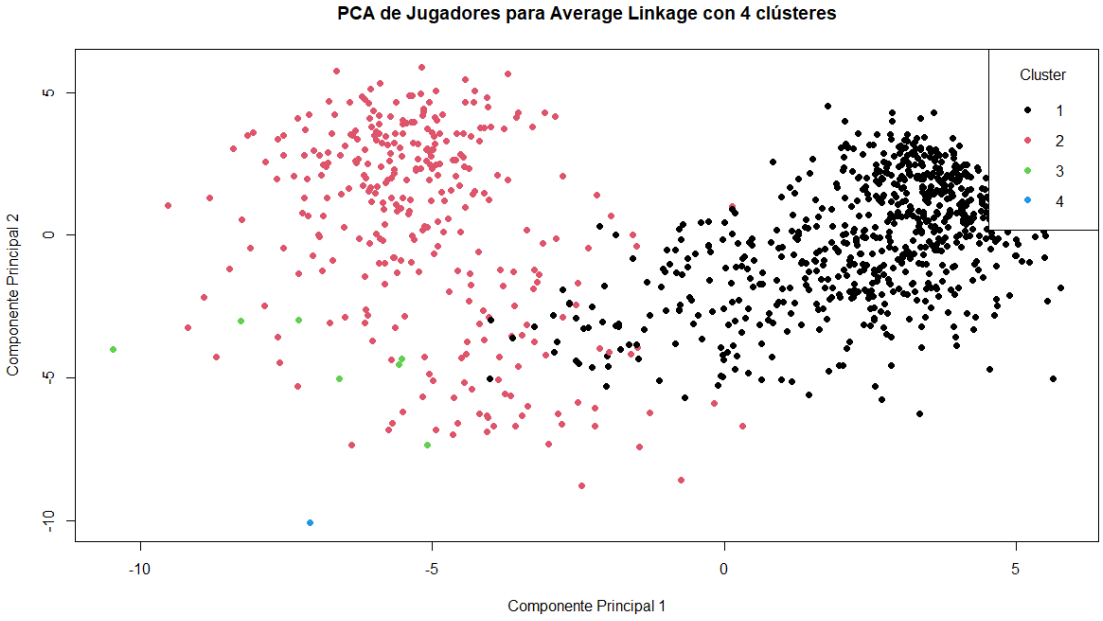
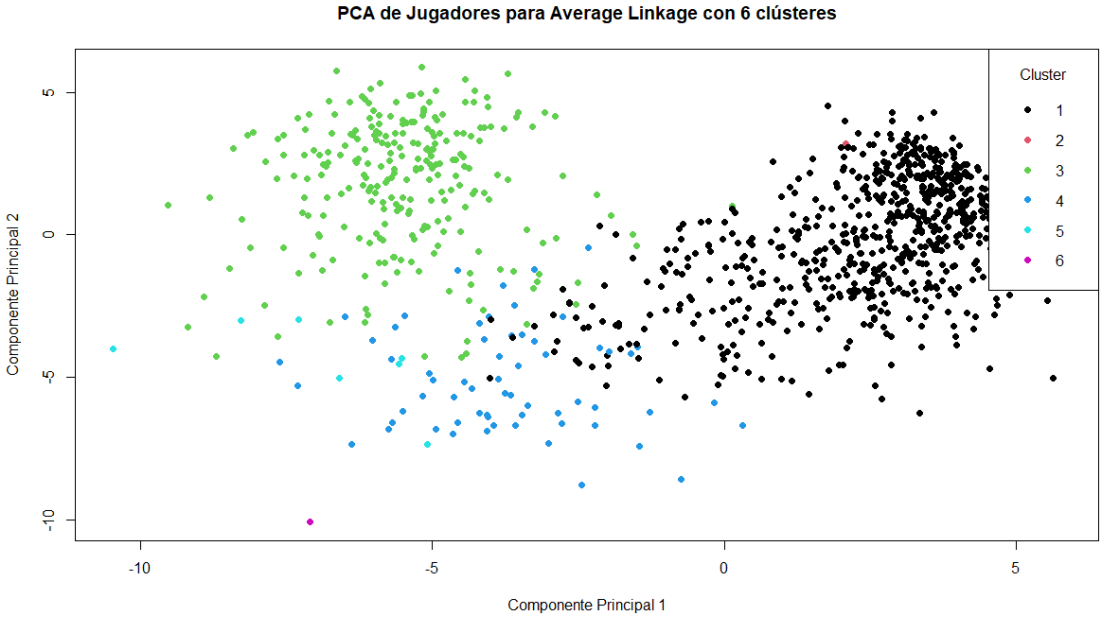


Figura 14. Reducción de dimensionalidad con PCA: *Average Linkage* 6 clústeres.



En línea con lo comentado previamente a partir del dendrograma, se puede ver en ambas figuras 13 y 14 cómo existe en ambos casos al menos un clúster conformado por una única observación. Tanto como para la primera figura, como para la segunda, el registro que pareciera ser un *outlier*, es siempre el mismo, el cual tiene el menor valor de todos para el componente principal 2. También, podemos percibir que en ambos gráficos siguen existiendo dos clústeres muy grandes y remarcados: los grupos uno y dos en la figura 13 y los grupos uno y tres en la figura 14. Retomando lo analizado de las agrupaciones en el algoritmo de *K-means*, podemos afirmar que tanto el grupo de jugadores defensivos, cómo el grupo de jugadores ofensivos suelen mantener su agrupación mientras que a medida que añadimos nuevos clústeres, éstos se generan para los mediocampistas con roles mixtos de juego.

Para cerrar el análisis de *Average Linkage*, calculamos los valores de *ASW* para ambos casos diagramados. Para el clúster Average Linkage de cuatro clústeres, obtuvimos un puntaje de 0.264, mientras que para la versión con seis clústeres alcanzamos una puntuación de 0.208. Si bien se obtienen valoraciones un poco más altas que con los algoritmos particionales, cuando comparamos los gráficos de reducción de dimensionalidad con PCA, vemos que las divisiones no parecieran ser tan equilibradas.

Complete Linkage

El último algoritmo de *clustering* que aplicaremos en este trabajo final es *Complete Linkage*. Cómo ya mencionamos en las últimas dos secciones, este también pertenece a la familia del *clustering* jerárquico. Nuevamente, la principal diferencia contra los algoritmos de *Single Linkage* y *Average Linkage* consiste en el cálculo de la medida de disimilitud entre los grupos, impactando directamente en la generación del dendrograma.

En este caso, la definición de James, Witten, Hastie & Tibshirani (2013)[6] es: "Disimilitud máxima inter-clúster. Calcular todas las disimilitudes entre pares de observaciones en el clúster A y las observaciones en el clúster B, y registrar la mayor de estas disimilitudes.".

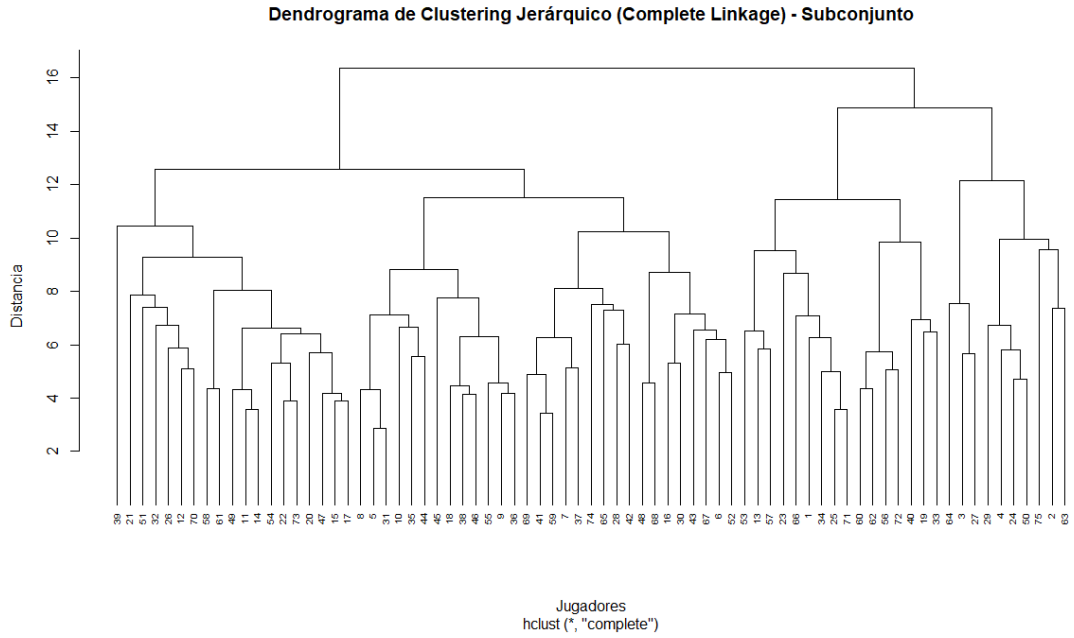
Partiendo de esta definición, podemos decir que se puede utilizar la siguiente expresión matemática para el cálculo de disimilitud en *Complete Linkage*:

$$D_{CL}(A, B) = \max_{i \in A, j \in B} D_{ij} \ . \ (6)$$

Tomando en consideración estas definiciones, se puede afirmar que el algoritmo de *Complete Linkage* tiene mayor tendencia a formar clústeres más compactos y gracias a que toma la distancia máxima entre todos los pares de puntos de cada grupo, es menos sensible a las observaciones *outliers*, en comparación a *Single Linkage* y *Average Linkage*. A partir de esto esperamos, ver un dendrograma que nos muestre distintos grupos bien definidos y con mayor separación entre sí, evitando caer en los clústeres de una única observación, tal como vimos en con los algoritmos jerárquicos previos.

Continuando con la metodología de aplicación del algoritmo de *Complete Linkage*, procedimos con los mismos pasos detallados en las secciones anteriores de *clustering* jerárquico. Establecimos la misma semilla para obtener el mismo subconjunto de setenta y cinco observaciones que utilizamos para los análisis de *Single Linkage* y *Average Linkage* con el fin de poder diagramar un dendrograma claro para poder hacer un primer análisis descriptivo de este y así definir la cantidad de clústeres con la que avanzaremos a correr el algoritmo final. El resultado de este dendrograma generado usando *Complete Linkage* con distancia euclidiana se puede apreciar en la figura 15 que se encuentra debajo.

Figura 15. Dendrograma de *clustering* jerárquico (*Complete Linkage*) para subconjunto



Tal como esperábamos, en primer lugar, podemos ver que no se genera ningún clúster individual en las primeras divisiones tal como sucedía con *Single Linkage* y *Average Linkage*. Esto no quiere decir que luego en la aplicación final del algoritmo a nuestra base de datos no suceda, ya que solo estamos visualizando un subconjunto, pero entendemos que es menos probable este tipo de comportamiento a través de *Complete Linkage*. En segundo lugar, también como se mencionó previamente, destacamos que las agrupaciones son más compactas y están mejor delimitadas producto de la medida de disimilitud. Si observamos la escala del gráfico vemos que las uniones se realizan a mayor distancia causando este efecto de separación, en el cual esperamos obtener clústeres con observaciones más cohesivas entre sí.

Procediendo de la misma manera que lo hicimos con todos los algoritmos previos, y teniendo en cuenta lo examinado en el dendrograma, diagramaremos la reducción de dimensionalidad con PCA con el fin de visualizar cómo quedarían compuestos los clústeres aplicando *Complete Linkage* a nuestra base de datos completa. En esta ocasión, utilizaremos los valores de cuatro y seis clústeres, similar a lo que hicimos con el resto de las técnicas. Se pueden visualizar ambos gráficos en las figuras 16 y 17 que se encuentran a continuación.

Figura 16. Reducción de dimensionalidad con PCA: *Complete Linkage* 4 clústeres.

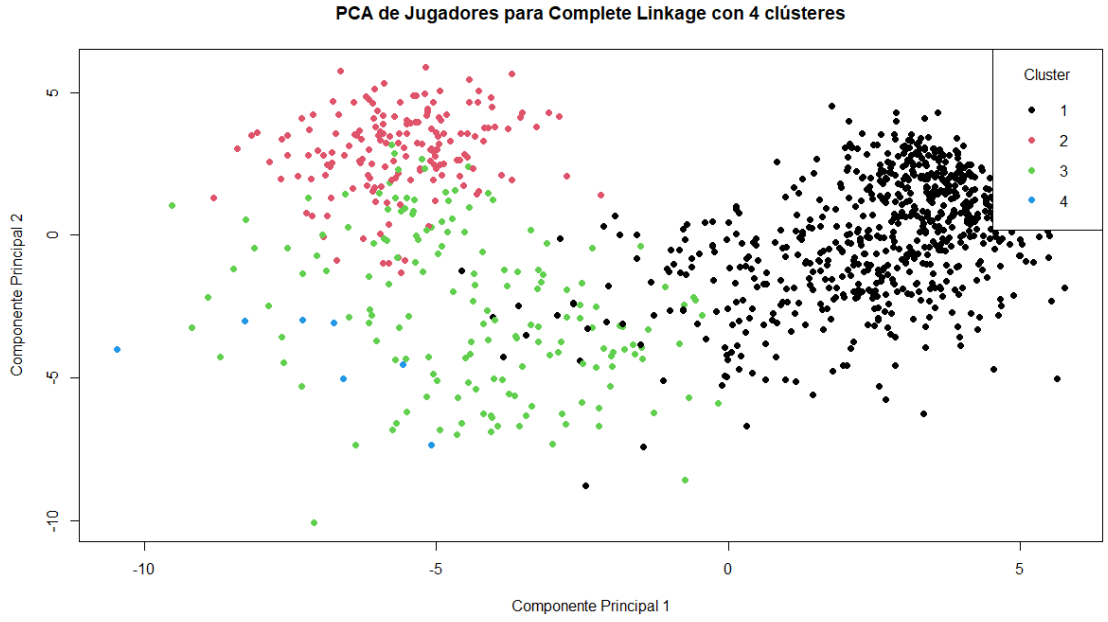
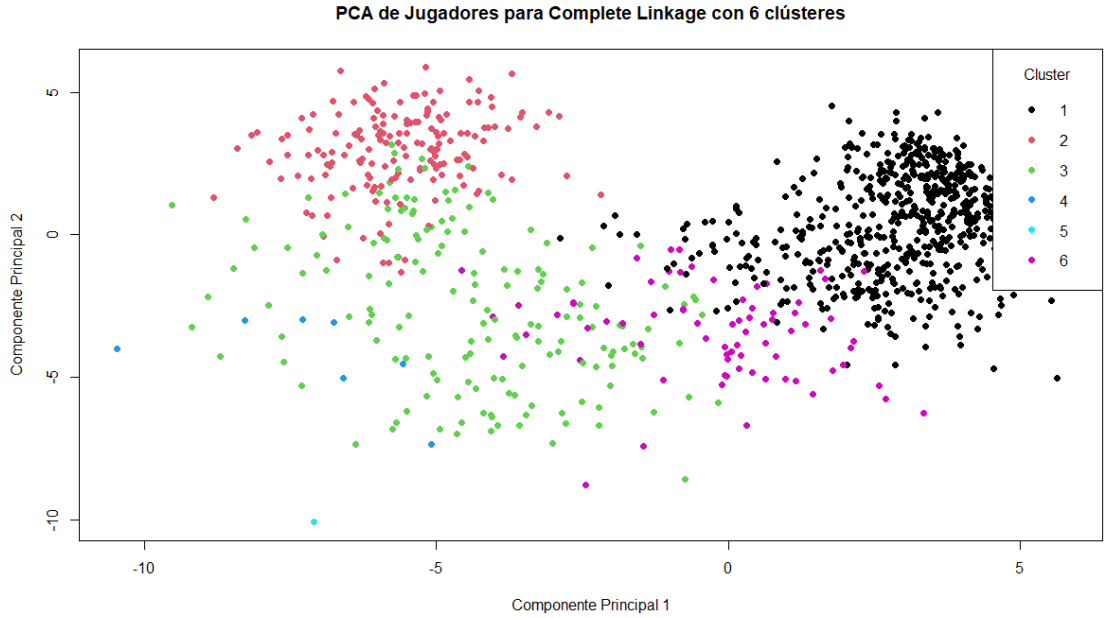


Figura 17. Reducción de dimensionalidad con PCA: *Complete Linkage* 6 clústeres.



Enfocándonos en la figura 16, notamos la primera gran diferencia de los resultados de *Complete Linkage* de cuatro clústeres contra *Single Linkage* y *Average Linkage* de cuatro clústeres, tal cual comentamos previamente, si bien el clúster número cuatro tiene muy pocas observaciones, éste no está compuesto de un único registro. Por otro lado, a medida que vamos expandiendo la cantidad de clústeres, volvemos a ver la tendencia de que haya al menos una observación *outlier* que conforma un clúster por sí misma. En todos los casos de *clustering* jerárquico es el mismo registro, el cual tiene el valor más bajo del componente principal 2. También continuamos percibiendo la tendencia de todos estos algoritmos jerárquicos por mantener principalmente en el clúster con mayor cantidad de observaciones a los jugadores con características ofensivas, mientras que las principales divisiones se ven en primera medida sobre los mediocampistas y luego, como se ve en la figura 17 se comienza a expandir hacia los defensores (principalmente ubicados en la parte superior izquierda del gráfico).

Para finalizar, computamos la medida que estamos utilizando para valorar y comparar cada uno de estos algoritmos, *ASW*. Con respecto al ejercicio de *Complete Linkage* con cuatro clústeres, el puntaje fue de 0.238, mientras que para el mismo con seis clústeres alcanza un total de 0.141. Si bien se percibe un descenso aproximado del 41%, el puntaje para el segundo es bastante similar a los que observamos en los ejercicios de *clustering* particional de cuatro clústeres, con la ventaja de poder contar con dos clústeres extra y con la posibilidad de advertir acerca de posibles *outliers*.

Algoritmos predictivos

Para esta segunda parte de la tesis, nos inspiramos en lo hecho por Yigit, Samak y Kaya (2020)[9] quienes siguieron la siguiente metodología: “En primer lugar, se recopilaron los datos suficientes. Después de eso, se analizaron dichos datos; se dieron forma y manipularon en una forma que proporciona la mejor respuesta al problema, y

éstos quedaron listos para la construcción del modelo. Finalmente, con los datos necesarios y apropiados; se construyeron modelos de aprendizaje automático que tienen como objetivo predecir el valor de los jugadores. Se construyeron, evaluaron y compararon diferentes modelos supervisados.”.

Antes de comenzar a desarrollar esta parte del trabajo, donde intentaremos predecir el valor de mercado de los jugadores a través de la variable “Valor de mercado (Transfermarkt)”, tenemos que aclarar que no utilizaremos la base de datos que pre - trabajamos para la sección anterior, sino que haremos un nuevo preprocesamiento de los datos sobre la base original.

Primero, comenzamos visualizando la base de datos completa con la que iniciamos a trabajar. Utilizando la función “*summary*” en R, notamos que existen jugadores que tienen un valor de mercado igual a cero. En la realidad, esto supondría que dichos jugadores no tienen valor y otros equipos podrían adquirirlos gratuitamente. Cómo no es cierto que los jugadores tengan un valor nulo de venta, lo primero que haremos es remover todas estas observaciones aplicando un filtro con la condición de que el campo “Valor de mercado (Transfermarkt)” sea estrictamente mayor a 0.

Segundo, volvemos a utilizar la misma función para obtener nuevamente un resumen de la base de datos campo por campo y nos encontramos con que hay un grupo de jugadores que si bien cumplen con la condición mencionada en el primer paso, no contamos con ningún dato estadísticos sobre el juego de estos. Principalmente notamos que la causa de la ausencia de información se debe a que estos jugadores cuentan con poco tiempo de juego en el período seleccionado que abarca nuestra base de datos. Al no poder contar con estas variables, las cuales necesitamos crucialmente para utilizar de predictores, decidimos eliminar de la base todas las observaciones que tienen valores “N/A” en las variables estadísticas de juego. Cabe destacar que los jugadores cuentan o no estrictamente con toda la información estadística de su juego y no hay casos en los que haya información parcial para un grupo de estas variables.

Tercero, examinamos todas las variables que están incluidas en nuestra muestra. Pasaremos a eliminar aquellos campos que no consideramos relevantes para la predicción que queremos realizar. La primera variable que quitamos fue “posición específica”. Recordemos que esta variable detalla las posiciones específicas dentro de la cancha en las cuales se suele desenvolver el jugador. Ejemplificando, un jugador defensivo podría desarrollarse principalmente como defensor central, pero también podría tener un estilo de juego y contar con la condición física que le permita cubrir la posición de lateral derecho. La razón detrás de eliminar esta variable es la de simplificar el análisis a partir de que capturamos dicha información sobre la posición principal en el campo de juego en la variable “Posición” y mantener esta variable podría verse redundante. La segunda variable que eliminamos fue “Equipo durante el período seleccionado". Esta fue una razón totalmente arbitraria ya que sabemos perfectamente que el club al que pertenece el jugador suele influir directamente en el valor de mercado de este, ya que no es lo mismo jugar en un equipo de renombre tal cómo “River Plate” o “Boca Juniors” que hacerlo en equipos cómo “Central Córdoba” o “Barracas Central”

que cuentan con presupuestos mucho más pequeños y a su vez no tienen el mismo poder de negociación. La principal razón de esta acción radica en que esta variable es categórica y además cuenta con demasiados valores ya que contamos con todos los equipos de primera división de cinco grandes ligas sudamericanas. Por otro lado, tal como explicamos en la variable posicional, podemos capturar esta información resumidamente desde el campo "Liga". La tercera variable que sacamos de la base de datos fue "Pie" debido a que directamente creemos que no es un dato relevante para calcular el valor de mercado de los jugadores. Las últimas dos variables que descartamos fueron "Altura" y "Peso" también debido a que no queremos que dichas características físicas puedan llegar a interferir en el cálculo predictivo.

Cuarto, notamos que, en la base de datos resultante luego de todas las modificaciones previas, aún contamos con tres variables categóricas "Liga", "Posición" y "País de nacimiento". Al saber que trabajaremos con algunos algoritmos que no tienen la posibilidad de procesar variables categóricas, aplicamos la técnica de *one-hot-encoding* sobre estas con el fin de poder utilizar la misma base de datos en cada uno de estos.

El resultado de todas estas transformaciones aplicadas sobre la base de datos inicial nos da como resultado final la base que utilizaremos para el ejercicio de predicción, la cual cuenta con el nombre de cada jugador que funciona como *ID* de cada observación, el valor de mercado correspondiente a cada uno y otras 68 variables que servirán como predictivas.

Una vez que contamos con esta base final para la valuación de jugadores, nuestro siguiente paso fue realizar la división de ésta en dos: una base de entrenamiento, a la que le asignamos el 80% de las observaciones para entrenar los modelos y una base de prueba, a la que le asignamos el restante 20% con el fin de ser utilizada para validar las predicciones. Sirve aclarar que para este paso se usó una semilla para poder reproducir los resultados obtenidos.

Antes de comenzar a entrenar y probar los modelos, aplicamos la metodología de selección de *features* automático *backward stepwise* con la finalidad de poder reducir la cantidad de características en nuestro modelo y poder quedarnos con las más relevantes para generar las predicciones. Este enfoque que utilizamos empieza usando todas las variables de nuestra base de datos y va eliminando iterativamente una a una las que tengan menor impacto en la predicción del valor de mercado del jugador. Para nuestro caso, le solicitamos a este algoritmo que cómo máximo mantenga cincuenta variables y que finalmente nos muestre el modelo que haya logrado el valor de R^2 más alto. El resultado final fue un modelo que incluye 34 variables, de las cuales 25 corresponden a variables estadísticas de su *performance* en el juego, 4 hacen referencia al país de nacimiento, 2 corresponden a variables *dummy* de la liga donde participan, 2 corresponden a la cantidad de partidos y minutos jugados, y la última hace referencia a la edad. El R^2 final de este modelo fue de aproximadamente 0.34. El detalle de las variables seleccionadas se puede ver en la tabla 3 que se publica debajo.

Tabla 3. Variables seleccionadas a partir de *backward stepwise*.

Variables seleccionadas a partir de Backward Stepwise			
1. Edad	2. Partidos jugados	3. Minutos jugados	4. Acciones defensivas realizadas /90
5. Duelos defensivos /90	6. Duelos defensivos ganados %	7. Tiros a la porteria %	8. Posesion conquistada despues de una interceptacion
9. Goles excepto los penaltis /90	10. xG /90	11. Remates /90	12. Posesion conquistada despues de una entrada
13. Regates realizados %	14. Duelos atacantes ganados %	15. Pases recibidos /90	16. Precision pases hacia adelante %
17. Pases cortos medios /90	18. Precision pases cortos medios %	19. Pases largos /90	20. Precision pases largos %
21. Desmarques /90	22. Jugadas claves /90	23. Pases en el ultimo tercio /90	24. Precision pases en el ultimo tercio %
25. Pases al area de penalti /90	26. Pases hacia el area pequeña %	27. Pases en profundidad /90	28. Precision pases en profundidad %
29. Liga Argentina	30. Liga Colombia	31. País de nacimiento Chile	32. País de nacimiento Costa Rica
33. País de nacimiento Italy	34. País de nacimiento Uruguay		

A continuación, comenzamos con las aplicaciones. Repasando empezaremos con un modelo de regresión lineal, continuaremos con una regresión *Ridge,* seguiremos con un modelo de *Random Forest* y terminaremos con *XGBoost*.

Regresión lineal

Iniciamos aplicando regresión lineal para la predicción del valor de mercado de los jugadores. Si bien es el modelo más simple de todos los que utilizaremos en esta tesis, decidimos que sea el primero para que sus resultados nos sirvan de *benchmark* contra los demás modelos.

Como bien sabemos, la regresión lineal es una técnica estadística que se utiliza principalmente para comprender la relación entre una variable que seleccionamos como dependiente y una o más variables independientes que servirán como predictores de la primera. Refiriéndonos a nuestro caso, la variable dependiente será el campo "Valor de mercado (Transfermarkt)" y contaremos con múltiples variables independientes las cuales quedaron seleccionadas en el paso anterior donde aplicamos la metodología de selección de *features* automático *backward stepwise*.

Al contar con más de una variable predictora, técnicamente estamos aplicando una regresión lineal múltiple. James, Witten, Hastie & Tibshirani (2013)[6] comentan que: "En lugar de ajustar un modelo de regresión lineal simple separado para cada predictor, un enfoque mejor es extender el modelo de regresión lineal simple para que pueda acomodar directamente múltiples predictores. Podemos hacer esto dando a cada predictor un coeficiente de pendiente separado en un solo modelo:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon.\text{"} \quad (7.1)$$

Donde Y es la variable dependiente, X_p son las p variables independientes, β_0 es la ordenada al origen, $\beta_1, \ldots, \beta_p$ son los coeficientes de los predictores, que cuantifican la asociación entre cada predictor y su respuesta y ε es el término de error.

Cada coeficiente β_p se interpreta como el efecto promedio en Y de un aumento de una unidad en X_p, manteniendo fijos todos los demás predictores, lo que permite entender el impacto individual de cada predictor en la variable de respuesta, posibilitando una visión más clara de las relaciones subyacentes en los datos.

Al desconocer los coeficientes $\beta_0, \ldots, \beta_p$, estos se deben estimar a partir de los datos. Para esto, se utiliza el método de mínimos cuadrados, donde se van a seleccionar

aquellos valores para que los coeficientes minimicen la suma de los residuos al cuadrado:

$$RSS = \sum_{i=1}^{n}(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \cdots - \hat{\beta}_p x_{ip})^2 \text{ (7.2)}$$

Donde y_i es el valor observado de la variable dependiente para la i-ésima observación, x_{ip} son los valores observados de los predictores para la i-ésima observación y $\hat{\beta}_p$ son los coeficientes estimados que minimizan la expresión de *RSS*

Por último, al ya contar con todas las estimaciones de los coeficientes $\hat{\beta}_p$, podemos hacer predicciones para nuevas observaciones usando la fórmula:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p \text{ (7.3)}$$

La aplicación en R de este modelo es bastante simple y *Rstudio* ya lo trae por defecto. Entrenamos el modelo utilizando la función "*lm*" a la que le indicamos que nuestra variable dependiente será "Valor de mercado (Transfermarkt)" y que debe utilizar la base de datos de entrenamiento, resultante del método *backward stepwise*.

Una vez entrenado el modelo, lo exponemos a nuestra base de datos de prueba para que genere predicciones sobre observaciones que no ha conocido previamente. Para esto, simplemente hacemos uso de la función "*predict*" a la que le indicamos que utilice el modelo entrenado previamente y le compartimos la base de datos de prueba con las variables seleccionadas anteriormente.

Contando con las predicciones para cada observación en la base de datos de prueba, definimos una métrica de calidad pero que también nos sirva de comparación con los algoritmos que entrenaremos a continuación. Así como elegimos *ASW* para los algoritmos de *clustering*, escogimos *WAPE (Weighted Absolute Percentage Error)* como la medida de error a usar con nuestros algoritmos predictivos.

Cómo su nombre lo indica, *WAPE* mide el error calculando la suma de errores absolutos del pronóstico, es decir que a los valores reales le resta los predichos, y luego divide esto por la suma de los valores reales. Luego se multiplica el resultado por cien para expresarlo de manera porcentual.

$$WAPE = \frac{\sum_{i=1}^{n}|A_i - F_i|}{\sum_{i=1}^{n}|A_i|} \times 100 \text{ (8)}$$

Donde A_i es el valor real de la i-ésima observación y F_i es el valor predicho para la i-ésima observación.

La razón por la que elegimos esta métrica es porque nos brinda una medida de error relativa al modelo de predicción con relación al tamaño de los valores reales. Además, su lectura es bastante sencilla e intuitiva ya que, al estar expresada en porcentaje, sabemos que un resultado "x%" significa que, en promedio, las predicciones estarán desviadas "x%" de los valores reales. Este valor oscilará entre 0% y 100% y cuanto menor sea significa que el modelo mejor predice. También es útil ya que podemos aplicarlo a

todas las metodologías que utilizaremos por lo que es muy ventajoso para la finalidad que queremos darle.

Dicho esto, por último, calculamos el *WAPE* del modelo de regresión lineal que entrenamos. El resultado es de aproximadamente 73.6%, número que parece bastante sencillo de superar por los algoritmos restantes.

Regresión *Ridge*

Para continuar, aplicaremos la regresión *Ridge* para intentar encontrar un modelo que en la comparativa genere una mejor predicción que el modelo de regresión lineal usado previamente.

Haciendo una introducción sobre la regresión *Ridge*, podemos decir que pertenece a los conocidos métodos de reducción, cuya principal característica es que buscan reducir la varianza a cambio del aumento en el sesgo con el fin de poder mejorar la capacidad de generalizar el modelo ante nuevos datos. En el caso particular de *Ridge*, esta se basa en la regresión lineal para regularla e introduce una restricción sobre los coeficientes de regresión: las sumas de las pendientes al cuadrado tienen que ser menores a un hiperparámetro (λ) que seleccionemos. Mientras menor sea éste, más se restringe al modelo, lo que reduce la varianza, pero genera un incremento del sesgo.

$$\sum_{i=1}^{p} \beta_i^2 \leq \lambda^2 \text{ (9.1)}$$

Seguimos haciendo uso del manual *ISLR* de los autores James, Witten, Hastie & Tibshirani (2013)[6] quienes explican que: "La regresión *Ridge* es muy similar a mínimos cuadrados, excepto que los coeficientes *Ridge* se estiman minimizando una cantidad ligeramente diferente. En particular, los coeficientes estimados de regresión *Ridge* $\hat{\beta}^R$ son los valores que minimizan:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}\beta_j^2 = RSS + \lambda \sum_{j=1}^{p}\beta_j^2 \text{ ." (9.2)}$$

Donde y_i es la variable dependiente, x_{ij} son las variables independientes, β_0 la ordenada al origen, β_j son los coeficientes de las variables independientes y λ es el hiperparámetro que controla la magnitud de la penalización.

La principal motivación detrás de la penalización es que, al aumentar el hiperparámetro λ, se fuerza a los coeficientes a acercarse a cero, lo que reduce su magnitud y tiene el efecto de simplificar el modelo, ayudando a mejorar su capacidad de generalización y evitar el sobreajuste. Cuando $\lambda = 0$, decimos que el modelo de regresión Ridge pasa a ser una regresión lineal múltiple, mientras que, por otro lado, a medida que este aumenta su valor, los coeficientes β_j se reducen y eventualmente pueden acercarse a 0, simplificando el modelo predictivo final.

En resumidas cuentas, la regresión *Ridge* tiene la ventaja de disminuir la varianza y mejorar la generalización del modelo hacia datos nuevos. También se observa un buen manejo de la multicolinealidad, reduciendo el impacto de las variables altamente correlacionadas al tener la posibilidad de contraer los coeficientes hacia cero dada la

restricción, lo que a su vez genera una mayor estabilidad de los coeficientes resultantes. Sin embargo y como comentamos anteriormente, todo lo bueno de lograr disminuir la varianza se logra en detrimento de la introducción de un aumento en el sesgo. Además, algo que podría ocasionar el uso de esta regresión es que los parámetros estimados cambien el signo al variar λ, lo que podría generar inconsistencia en el efecto del *feature* sobre la variable dependiente. Si la comparamos contra la regresión lineal múltiple la cual ya utilizamos para predecir la valuación de los jugadores, podríamos decir que *Ridge* debería dar mejores resultados al ser más efectivo frente a casos de multicolinealidad, a su facilidad para la generalización de los datos reduciendo la varianza y la estabilidad que puede generar sobre los coeficientes obtenidos.

Para aplicar *Ridge* en R, utilizamos la librería "*glmnet*". Antes de entrenar el modelo, debimos convertir nuestros *datasets* de entrenamiento y prueba al formato de matriz, ya que la función que vamos a usar para el modelo sólo admite datos de este tipo. Ajustamos el modelo de regresión *Ridge* aplicando la función "*cv.glmnet*", a la que le compartiremos la base de datos de entrenamiento (la que resultó luego de haber aplicado *backward stepwise*) en formato matriz y la variable dependiente. El prefijo "*cv*" en la función hace alusión a que se utilizara el método de *cross validation* al entrenar el modelo. A su vez para que esta haga uso de *Ridge* debemos especificar un parámetro "*Alpha*" igual a 0. Con respecto al valor de λ dejaremos que el modelo pruebe y decida por sí mismo cuál es el valor óptimo gracias al proceso de *cross validation*.

Una vez finalizado el entrenamiento del modelo, pasamos a predecir las observaciones que se encuentran en nuestra base de datos de prueba. Para esto, usamos la función "*predict*" a la que le solicitamos que prediga con el modelo *Ridge* sobre el *dataset* de prueba. Agregamos a esta función el parámetro "*s = "lambda.min*"" para pedirle al algoritmo predictivo que haga uso del valor de λ que minimizó el error de validación cruzada en el proceso de ajuste del modelo. En nuestro caso vemos que *lambda* toma un valor de *39.029*.

Por último, como hemos comentado al terminar la aplicación de la regresión lineal, calculamos el *WAPE* del modelo *Ridge* usando los valores predichos y la valuación real de mercado para los jugadores en la base de prueba. El resultado de esta métrica es de un 71.2%, por lo que podemos decir que hubo una pequeña mejora de 2.4 puntos porcentuales contra el modelo de regresión lineal.

Random Forest

Dejamos de lado por un momento las regresiones lineales y *Ridge*, y pasamos a trabajar con modelos de árboles de decisión.

Hablando particularmente de *Random Forest*, podemos decir que este es un modelo de ensamble donde se combinan múltiples árboles de decisión para ajustar un modelo más robusto y preciso. Cuando hablamos de modelos de árboles de decisión, básicamente nos referimos a algoritmos de aprendizaje automático que hacen uso de la estructura de árbol (nodo, ramas y hojas) para representar y tomar decisiones basadas en datos. Esto los hace muy intuitivos y logran buenos ajustes en contextos de variables

mixtas, datos perdidos y atípicos. Que sea un modelo de ensamble, quiere decir que se propone construir varios modelos de árboles de decisión por separado para luego ensamblarlos y que cada uno de estos genere una predicción, donde la decisión final dependerá de promediar los resultados de cada árbol. El efecto que tiene esto es la reducción de varianza con el fin de disminuir el error final del modelo.

James, Witten, Hastie & Tibshirani (2013)[6] coinciden con lo comentado previamente y luego introducen los términos de *Bagging* y *Random Forest*: "Recordemos que dado un conjunto de n observaciones independientes $Z_1, \ldots, Z_n$, cada una con varianza σ^2, la varianza de la media $\bar{Z}$ de las observaciones se da por σ^2/n. En otras palabras, el promedio de un conjunto de observaciones reduce la varianza. Por lo tanto, una forma natural de reducir la varianza y aumentar la precisión de predicción de un método de aprendizaje estadístico es tomar muchos conjuntos de entrenamiento de la población, construir un modelo de predicción separado utilizando cada conjunto de entrenamiento y promediar las predicciones resultantes."

Con respecto al concepto de *Bagging*, hacen referencia a la posibilidad de poder construir B modelos de predicción, usando B conjuntos de entrenamiento separados y promediándolos para obtener un único modelo con baja varianza:

$$\hat{f}_{avg}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^B(x) \text{ . (10)}$$

Donde, B indica la cantidad de modelos y $\hat{f}^B(x)$ es la predicción del modelo b-ésimo para la entrada x.

Sin embargo, como en la práctica no se suele disponer de múltiples conjuntos de entrenamiento, se utiliza la técnica de *Bootstrap*, la cual permite generar varios conjuntos de entrenamiento a partir de un único conjunto de datos inicial. El método *Bootstrap*, implica tomar múltiples muestras con reemplazo del conjunto de datos original, lo que generaría B diferentes conjuntos de datos de entrenamiento. Finalmente, como se ve en la fórmula 10.1, podríamos promediar las predicciones de cada uno de estos conjuntos para obtener el resultado final.

Por otro lado, se introduce el método de *Random Forest* como una mejora al método de *Bagging*. *Random Forest* añade un pequeño ajuste aleatorio que decorrelaciona los árboles. Si bien se comienza construyendo un "bosque" de árboles de decisión a partir de muestras de entrenamiento *Bootstrap*, se selecciona aleatoriamente un subconjunto de m predictores del conjunto completo de predictores (p), para cada uno de los árboles. El valor de m suele ser $\sqrt{p}$.

La justificación a que cada árbol de decisión nos considere la mayoría de los predictores disponibles, surge de que, si existiese un predictor muy fuerte en el conjunto de datos, la mayoría de los árboles en un modelo de *Bagging* utilizarían dicho predictor fuerte en la primera división, generando muchos árboles similares entre sí, resultando en predicciones con alta correlación, la cual no permitiría la reducción de varianza deseada. En otras palabras, en promedio $(p - m)/p$ de las divisiones no considerarán

al predictor fuerte, lo que permite que otros predictores tengan más oportunidad de ser considerados, lo que decorrelaciona los árboles y alcanza el fin de disminuir la varianza.

Haciendo unos comentarios finales acerca de *Random Forest*, podemos decir que es un modelo que presenta menor error esperado que las regresiones comentadas previamente, nos permite computar la importancia de cada variable, podemos estimar la performance del modelo al mismo tiempo que lo entrenamos, y con relación a otros modelos de aprendizaje automático, tiene pocos hiperparámetros. Se suma también que *Random Forest* decorrelaciona los modelos (al no usar todos los *features* en todos los árboles), lo que reduce la varianza del modelo en sí. También gracias a esta eficiencia de utilizar menos variables, se reduce el costo computacional. En contraparte, tiene la desventaja en el punto de interpretabilidad, debido a que es un modelo complejo de naturaleza no lineal y por su inclusión de varios árboles de decisión. Además, esta complejidad que hablamos trae consigo la necesidad de requerir más memoria y aumentar el tiempo de entrenamiento. En comparación con los modelos previos, esperamos seguir mejorando los resultados obtenidos previamente al implementar un método más sofisticado y robusto como hemos presentado.

Con respecto a la aplicación de *Random Forest* en R, utilizamos la librería "*randomForest*". Al igual que *Ridge*, precisa que las bases de datos se encuentren en formato de matriz, por lo que comenzamos realizando dichas transformaciones de las bases de entrenamiento y prueba (que fueron procesadas luego de haber usado *backward stepwise*). Para ajustar el modelo hacemos uso de la función "*randomForest*" a la que le compartimos la base de datos de entrenamiento comentada previamente, los valores de mercado de dichas observaciones e incluimos los hiperparámetros "*ntree*" e "*importance*". El primero, define la cantidad de árboles a utilizar, el cual establecimos en 10.000, y el segundo lo programamos en "*True*" para computar la importancia de las variables en el cálculo de la variable dependiente.

Para la predicción, volvemos a utilizar la función "*predict*" a la que le compartimos la base de datos de prueba y le indicamos que haga la predicción con el modelo recién ajustado de *Random Forest*.

Finalizamos los comentarios de este modelo calculando el *WAPE* obtenido para las predicciones. En este caso, obtuvimos un puntaje de 61.8%, lo que nos indica que seguimos mejorando nuestras predicciones, en este caso la mejora contra el modelo previo, *Ridge,* es de 9.4 puntos porcentuales.

XGBoost

El último modelo en el que nos vamos a adentrar es *XGBoost*. Al igual que *Random Forest*, este es un modelo de la familia de árboles de decisión.

Al igual que con el resto de los modelos, comenzamos con una breve introducción a *XGBoost*. El nombre de este se debe a que es un algoritmo de Extreme *Gradient Boosting Machine*, el cual es un método de aprendizaje automático que basa la construcción de su modelo predictivo uniendo predicciones de otros modelos, en el caso particular de

XGBoost, son árboles de decisión, tal como vimos previamente en *Random Forest*. La principal diferencia con el modelo anterior radica en que *XGBoost* genera los árboles de decisión de manera secuencial y no en paralelo como *Random Forest*. Esto quiere decir que ahora los árboles de decisión se harán sucesivamente y cada modelo nuevo tomará los aprendizajes del anterior e intentará corregir los errores para lograr una predicción más acertada.

La explicación que dan los autores James, Witten, Hastie & Tibshirani (2013)[6] en *An introduction to statistical learning* acerca de *Boosting* es la siguiente: "El *boosting* funciona de manera similar al *Bagging*, excepto que los árboles se desarrollan de manera secuencial: cada árbol se desarrolla utilizando información de los árboles previamente desarrollados. El *boosting* no implica muestreo *bootstrap*; en su lugar, cada árbol se ajusta en una versión modificada del conjunto de datos original."

Estos explican que la idea principal es mejorar la precisión del modelo a través de un proceso iterativo gradual, remarcando que el *boosting* aprende de manera incremental. Primero se empieza con un único árbol de decisión. Luego cada nuevo árbol de decisión se ajusta a los residuos del modelo actual. Estos residuos representan las partes del dato que el modelo actual no logra predecir correctamente, por lo que el nuevo árbol se enfoca en mejorar las deficiencias del modelo actual. El nuevo árbol ajustado a los residuos se agrega al modelo existente y se actualizan dichos residuos. Este proceso se itera con el fin de que cada árbol adicional mejore las predicciones del modelo.

Además, es importante resaltar que el boosting cuenta con tres hiperparámetros principales. En primer lugar, el número de árboles B que, en el caso particular de este modelo, si B es demasiado grande, se podría sobreajustar el modelo. En segundo lugar, el parámetro de contracción λ, el cual va a controlar la rapidez de aprendizaje del *boosting*. λ suele ser un número positivo pequeño y mientras menor sea, se requerirá de un B más alto para lograr un mejor rendimiento. Por último, el hiperparámetro d, el cual señala el número de divisiones en cada árbol, por lo que representa la profundidad de la interacción determinando la cantidad de variables que puede involucrar cada árbol.

Concluyendo, en base a la explicación del algoritmo, podemos decir que *XGBoost* propone de cierta forma una mejora versus al modelo de *Random Forest* que aplicamos anteriormente. Haciendo foco en los errores residuales que va dejando cada árbol de decisión, esperamos que este algoritmo pueda producir modelos de alta calidad y eficiencia. Si bien podemos decir que *XGBoost* apunta a ser un modelo preciso al aprender lentamente acerca de los datos de entrenamiento, esto también se podría considerar una desventaja al necesitar mayor poder computacional y demorar los resultados. Por lo comentado y debido a su gran familiaridad con *Random Forest*, esperamos obtener mejores resultados que este último o al menos similares.

En cuanto a la aplicación de *XGBoost* en R, usamos la librería "*xgboost*". Al igual que los modelos previos, comenzamos convirtiendo las bases de datos de entrenamiento y prueba (*backward stepwise*) a formato matriz. Utilizamos la función "xgboost" para

ajustar el modelo al cual le compartimos la base de datos de entrenamiento y los valores correspondientes de mercado para cada jugador en esa base. Entre los hiperparámetros únicamente ajustamos el valor de "*nrounds*" en 1000, lo que quiere decir que iterará esa cantidad de veces y establecimos como objetivo "*reg:squarederror*" por lo que el modelo intentará minimizar el error cuadrático de las estimaciones.

Para realizar las predicciones, al igual que en todos los modelos que probamos, usamos la función "*predict*" a la cual le especificamos que utilice el *dataset* de prueba aplicando el modelo *XGBoost* recientemente entrenado.

Finalizando, computamos el puntaje de *WAPE* obtenido por el modelo. Para *XGBoost*, este es del 61.2%, lo que lo posiciona por encima de *Random Forest* a 0.6 puntos porcentuales.

4. Resultados

Luego de haber comentado acerca de la metodología que aplicamos para este trabajo final, pasaremos a interpretar los resultados obtenidos y a realizar las comparaciones pertinentes entre los rendimientos que alcanzaron cada uno de los modelos, en primera instancia aquellos algoritmos de *clustering* y luego los algoritmos predictivos. Además, también comentaremos cuáles son los pasos que ejecutamos a continuación para cerrar el ejercicio completo que se propuso desde un inicio: encontrar cinco jugadores similares a un jugador en cuestión y generar una predicción de su propio valor de mercado para compararla con la real y entender si ese valor se encuentra sub, sobrevalorado o lo vemos en montos razonables.

Algoritmos de *clustering*

En primer lugar, con respecto a los algoritmos de *clustering*, recordemos que aplicamos los siguientes: *K-means*, *PAM*, *Single Linkage*, *Average Linkage* y *Complete Linkage*. Si bien ya comentamos acerca de los resultados obtenidos durante la ejecución de estos, nos gustaría hacer una comparación entre los rendimientos y justificar por qué seleccionaríamos uno de estos para efectuar la búsqueda de los jugadores más similares.

Cómo se vio previamente, ya de antemano descartaremos el uso de *Single Linkage* y *Average Linkage*. La razón detrás de esto es que en sus resultados obtenidos podemos ver cómo se generan clústeres individuales o con muy pocas observaciones ocasionando en contraposición grandes clústeres con muchos registros, lo que hace parecer que no hay divisiones entre estos. Si bien mencionamos que el criterio a utilizar para definir con qué metodología avanzar es *ASW*, en estos casos por más valores altos que hayan logrado en comparación al resto, vemos en los diagramas de reducción de dimensionalidad con PCA que es a costa de generar estas pequeñas divisiones.

A partir de esto, nos queda una definición más reducida contando con dos algoritmos de *clustering* particionales: *K-means* y *PAM* versus un algoritmo de *clustering* jerárquico: *Complete Linkage*. Para avanzar con la comparación, creamos una tabla de

doble entrada donde computamos los puntajes obtenidos de *ASW* para distintas cantidades de clústeres. La misma se puede observar debajo:

Tabla 4. Puntaje *ASW* por metodología de *clustering* y cantidad de clústeres.

Metodología / Cantidad de clústeres	4	5	6	7	10
K-Means	0,17	0,15	0,13	0,11	0,09
PAM	0,16	0,13	0,11	0,08	0,09
Complete Linkage	0,24	0,23	0,14	0,13	0,05

Analizando los resultados, notamos a simple vista que *Complete Linkage* logra grandes diferencias favorables en términos de su puntaje *ASW* cuando lo comparamos contra los otros dos algoritmos de *clustering* particionales cuando el valor de "k" clústeres es de cuatro o cinco. Luego, esta ventaja se va emparejando a medida que aumentamos la cantidad de clústeres a seis o siete y se desploma por completo cuando generamos grandes cantidades de clústeres, iguales o mayores a diez.

Con el fin de continuar disminuyendo la cantidad de algoritmos para determinar con cual sería más conveniente avanzar, examinamos con mayor detalle los resultados obtenidos por los algoritmos de *clustering* particionales. Observando la tabla 4, vemos que, si bien los resultados son bastante parejos, K-means logra siempre posicionarse con puntajes superiores en la *performance* de *ASW*. Si comparamos los resultados obtenidos en las reducciones de dimensionalidad con PCA que compartimos en las figuras 4,5,7 y 8 también se puede ver que las agrupaciones que se realizan sobre los datos son muy similares, donde como hemos comentado previamente, se empiezan delimitando los jugadores por posiciones: defensores, mediocampistas y delanteros. Luego se van generando los demás clústeres, principalmente sobre los mediocampistas, quienes pueden ocupar tanto roles más defensivos como ofensivos. Dados estos resultados y al no notar grandes diferencias entre una metodología y otra, más que como se definen los centroides /medoides, nos terminamos basando en su puntaje de *ASW* para descartar *PAM* y seguir nuestro trabajo con *K-means*.

A esta instancia llegamos con un método de *clustering* particional (*K-means*) y un método de *clustering* jerárquico (*Complete Linkage*). Si comparamos mano a mano sus puntajes *ASW* en la tabla 4, vemos que Complete Linkage sobresale cuando establecemos valores bajos de cantidad de clústeres, pero este se desploma cuando asignamos valores altos. Volviendo a analizar las figuras 16 y 17 donde diagramamos las reducciones de dimensionalidad con PCA para *Complete Linkage* vemos que la mejor performance en niveles bajos de cantidad de clústeres se logra debido a que, como en el resto de los algoritmos jerárquicos, esta metodología agrupaciones más desbalanceadas en cantidad de observaciones. También una singular diferencia que apreciamos a partir de los gráficos de reducción de dimensionalidad con PCA es que en el algoritmo de *K-means* se suele mantener como clúster con más observaciones al que

agrupa principalmente a los jugadores que ocupan rol de defensores (quienes se encuentran en la parte superior izquierda de los gráficos) mientras que *Complete Linkage* mantiene en su clúster con mayor observaciones a los jugadores con roles de delanteros (los que se posicionan en la parte superior derecha del gráfico).

Para terminar de definir qué algoritmo quisiéramos determinar como el más apto para la finalidad de nuestro trabajo, vamos a comentar los pasos a seguir luego de definir la metodología a utilizar y realizaremos una prueba de desempeño para cada uno de estos con el objetivo de ver cual nos entrega los resultados más coherentes.

Una vez que ya estén generados los clústeres de la forma en la que se explicó en la sección metodológica, pasamos a definir un jugador de la base de datos para el cual quisiéramos encontrar los cinco jugadores más similares en su forma de juego. Luego de haber definido al jugador objetivo, delimitamos la búsqueda a aquellos jugadores que se encuentran en el mismo clúster asignado por el algoritmo. Después calculamos las distancias euclidianas entre el jugador objetivo y el resto de los jugadores pertenecientes al mismo clúster y las ordenamos de menor a mayor. El último paso consta simplemente en observar este *ranking* y conocer quiénes son los cinco jugadores que se encuentran más cercanos al jugador objetivo.

En el primer ejercicio, consideramos un jugador cuya posición es defensor central y utilizamos *K-means* y *Complete Linkage* con cuatro y seis clústeres cada uno para comparar los cuatro distintos resultados que podríamos llegar a obtener. Durante este proceso, a este jugador seleccionado pasaremos a llamarlo “jugador objetivo”. Para el análisis de estos resultados, vamos a utilizar el campo de posición específica correspondiente a la información general del jugador y nuestro conocimiento general del juego y los jugadores para determinar si los resultados pueden ser considerados buenos reemplazos para éste. Debajo se muestra un resumen de los resultados obtenidos, donde podemos ver al jugador objetivo y los jugadores similares hallados. En ésta se indica con una cruz que algoritmo lo encontró, la posición específica detallada en la base de datos y algunas métricas que nos parecen relevantes para la posición de defensor central.

Tabla 5. Jugadores similares – defensor central

Jugador	K-means (4)	K-means (6)	Complete Linkage (4)	Complete Linkage (6)	Posición específica	Acciones defensivas realizadas /90	Duelos defensivos ganados (%)	Duelos aéreos en los 90	Posesión conquistada después de una interceptación
Jugador objetivo					**Centrodefensor, Centrodefensor izquierdo**	**9.11**	**69%**	**5.23**	**7.66**
Jugador A	X	X	X	X	Centrodefensor, Centrodefensor derecho	9.95	58%	3.38	6.26
Jugador B	X	X			Centrodefensor, Centrodefensor izquierdo	11.77	61%	6.53	8.61
Jugador C	X	X			Centrodefensor, Centrodefensor izquierdo	8.46	75%	3.4	4.44
Jugador D	X	X			Centrodefensor, Centrodefensor derecho	9.13	63%	4.29	7.65
Jugador E	X	X			Centrodefensor, Centrodefensor izquierdo	10.81	69%	3.58	7.14
Jugador F			X	X	Centrodefensor, Centrodefensor derecho	8.49	90%	3.61	6.92
Jugador G			X		Centrodefensor, Centrodefensor derecho	12.4	68%	4.31	6.4
Jugador H			X	X	Extremo derecho, Lateral derecho	7.07	63%	4.6	4.24
Jugador I			X	X	Extremo derecho, Lateral derecho	8.15	56%	3.18	3.75
Jugador J				X	Extremo derecho, Lateral derecho	8.15	59%	3.86	5.78

En primer lugar, hablando de los resultados de *K-means*, vemos que este algoritmo nos arroja cinco jugadores bien defensivos y con posiciones específicas idénticas al jugador seleccionado, todos defensores centrales que podrían ocupar posiciones tanto del lado izquierdo, como derecho. A su vez, se puede ver que tienen características muy similares correspondientes a métricas de gran relevancia para la posición. Cabe destacar que, para ambos casos de cuatro y seis clústeres, los resultados fueron exactamente los mismos. En segundo lugar, hablando de los resultados de *Complete Linkage*, en el caso de cuatro clústeres el algoritmo nos arroja tres defensores, también centrales que ocupan este rol principalmente en el sector derecho de la defensa y luego aparecen dos extremos, actualmente con roles un poco más ofensivos, pero que han sabido ocupar el rol de defensores laterales. Para el caso de seis clústeres, figuran casi los mismos jugadores, salvo que se cambia uno de los defensores centrales sugeridos por otro jugador, también particularmente extremo, pero que podría tomar un rol de defensor lateral. En el caso de los extremos encontrados, podemos decir que se percibe una baja notable versus todos los valores de las métricas defensivas consideradas con respecto al "jugador objetivo".

En el segundo ejercicio, consideramos un jugador cuya posición es delantero central y también utilizamos *K-means* y *Complete Linkage* con cuatro y seis clústeres para la comparación. Al igual que en el análisis anterior, creamos una tabla resumen para mostrar los resultados obtenidos.

Tabla 6. Jugadores similares – delantero central

Jugador	K-means (4)	K-means (6)	Complete Linkage (4)	Complete Linkage (6)	Posición específica	Goles excepto los penaltis /90	Remates /90	Toques en el área de penaltis	Pases recibidos /90
Jugador objetivo					**Centrodelantero**	**0.33**	**2.05**	**3.31**	**11.38**
Jugador A	**X**	**X**	**X**	**X**	Centrodelantero	0.31	2.15	3.42	7.46
Jugador B	**X**	**X**	**X**	**X**	Centrodelantero	0.27	1.91	4.45	9.03
Jugador C	**X**	**X**	**X**	**X**	Centrodelantero	0.3	2.99	3.97	8.68
Jugador D	**X**	**X**	**X**	**X**	Centrodelantero	0.29	2.11	3.6	8.76
Jugador E	**X**	**X**			Wing izquierdo, Centrodelantero	0.18	1.54	2.54	14.25
Jugador F			**X**	**X**	Centrodelantero	0.3	2.18	4.74	9.13

Empezando por *K-means* el algoritmo vuelve a ser consistente y arroja los mismos resultados para su versión de cuatro y seis clústeres. De los cinco jugadores propuestos, cuatro son plenamente delanteros centrales como el "jugador objetivo" mientras que el restante se puede desempeñar tanto como delantero central como extremo izquierdo. Para este último jugador, notamos que tiene valores más bajos que el resto de los jugadores hallados en todas las métricas seleccionadas, salvo por aquella de pases recibidos. Por otro lado, analizando los resultados de *Complete Linkage*, al igual que *K-means* en esta ocasión vemos también resultados consistentes manteniendo los mismos jugadores sugeridos tanto para el algoritmo de cuatro clústeres como para el de seis clústeres. De los cinco jugadores seleccionados como recomendación, todos cumplen el mismo rol de ser plenamente centrodelanteros con características de juegos muy similares. Se da la particularidad que ambos algoritmos coinciden en cuatro jugadores y difieren en la selección de uno de ellos, aquel que en *K-means* no es plenamente delantero central.

Con relación a los resultados obtenidos directamente en el ejercicio final, podemos decir que la *performance* de *K-means* se ve sólida para ambos casos, tanto para el primero donde buscamos jugadores defensivos, como para el segundo donde buscamos un jugador plenamente ofensivo. En cambio, *Complete Linkage* no obtuvo los resultados esperados cuando el jugador objetivo fue un defensor central. Esto era de esperarse por lo comentado previamente cuando analizamos sus diagramas de reducción de dimensionalidad con PCA (figuras 16 y 17). Tal como comentamos durante los desarrollos de los algoritmos, creemos que *K-means* podría estar funcionando mejor ya que tiende a formar clústeres más compactos que podrían reflejar mejor las similitudes en métricas clave, particularmente en este escenario planteado donde los vimos que los jugadores se terminan agrupando de manera clara por roles ofensivos y defensivos. Es por esta razón que definimos que el mejor algoritmo a aplicar para realizar *clustering* con la finalidad de encontrar jugadores más similares es *K-means*. Con respecto a la cantidad de clústeres a aplicar, sugerimos la opción de cuatro, ya que ésta cuenta con el mayor puntaje de *ASW*.

Para seguir ahondando más en nuestro análisis de *clustering*, profundizaremos un poco más sobre el algoritmo seleccionado para recolectar más información acerca de este con relación a nuestra base de datos. Para entender mejor cómo se compone cada uno de los cuatro clústeres generados por *K-means*, creamos una tabla para resumir la información de cada uno. En la misma, añadimos el porcentaje de defensores, mediocampistas y delanteros que hay en cada uno de ellos. A su vez, exploramos acerca de las variables más importantes para la asignación de los clústeres. Para obtener la información acerca de los *features* más importantes para la designación de clústeres, ejecutamos un algoritmo de *Random Forest* al cual le establecimos como variable dependiente el número de clúster asignado, resultando en un modelo de clasificación multiclase. A este le solicitamos que compute la importancia de cada variable para luego tomar las cinco más representativas y mostrar el promedio de cada una de estas para cada grupo. Por último, nos pareció relevante agregar el promedio del valor de mercado para cada clúster ya que en este proyecto también estamos trabajando en la predicción de éste. Los detalles se pueden visualizar a continuación en la tabla 7 y los mismos también corresponden a la figura 4 donde se había realizado la reducción de dimensionalidad con PCA.

Tabla 7. Detalles por clúster de *K-means*.

Variables	Clúster asignado			
	1	2	3	4
Defensores (%)	1.0%	0.0%	0.0%	99.1%
Mediocampistas (%)	98.7%	56.6%	0.9%	0.9%
Delanteros (%)	0.3%	43.4%	99.1%	0.0%
Acciones de ataque exitosas /90 (promedio)	1.28	3.82	2.38	0.58
Duelos atacantes /90 (promedio)	4.56	10.63	9.46	1.67
Carreras en progresión /90 (promedio)	0.75	2.07	0.86	0.60
Pases al área de penalti /90 (promedio)	1.39	3.08	1.35	0.65
Pases hacia adelante /90 (promedio)	13.40	10.64	3.95	15.98
Valor de mercado Transfermarkt (promedio)	$ 809,167	$ 1,042,975	$ 947,321	$ 607,950

A partir de estos resultados concretos, terminamos de confirmar varias ideas planteadas previamente. En primer lugar, la agrupación que realiza *K-means* priorizando las posiciones de los jugadores. Se puede observar muy marcada la pertenencia específica de cada posición en los clústeres 1,3 y 4, siendo cada uno de estos casi exclusivamente para jugadores de la posición predominante la cual en todos los casos representa aproximadamente el 99% de las observaciones. El clúster número 2 se distingue de éstos al contar con jugadores principalmente mediocampistas, pero también incluyendo en gran medida a delanteros.

En segundo lugar, notamos que las primeras cuatro variables más relevantes a la hora de asignar un clúster son estadísticas principalmente ofensivas, pero quienes más destacan en estas son aquellos jugadores del clúster 2 el cual no es donde predominan los delanteros sino la mezcla de estos con los volantes más ofensivos. A su vez, confirmamos lo analizado previamente en el apéndice E, donde podíamos observar que

son las variables ofensivas las que más se correlacionan positivamente con el valor de mercado del jugador, siendo el clúster 2 no solo el que mejor *performance* tiene en estas, sino que también son los de mayor valor de mercado promedio.

En tercer y último lugar, también a partir de observar cuales son las variables más importantes para designar clúster, entendemos por qué el algoritmo de *K-means* se concentra en realizar divisiones principalmente sobre las observaciones de los mediocampistas ofensivos y delanteros, ya que cuando ejecutamos éste incrementando la cantidad de grupos, las particiones correspondientes a los defensores y mediocampistas con características más defensivas se mantenían casi intactas. Todo esto se debe a que el algoritmo considera más relevantes aquellas relacionadas al juego ofensivo.

En base a este último ejercicio, y continuando con el hilo de la primera conclusión planteada, nos gustaría terminar de despejar la duda de si la variable "posición" a la cual incluimos en este análisis a través del uso de *one-hot-encoding* está influyendo de manera significativa en la distancia euclidiana entre los jugadores. Para esto, nos planteamos probar como resulta nuestro algoritmo de *K-means* de cuatro clústeres cuando lo exponemos a la base de datos sin información de la variable "posición". Con el fin de poder comparar los resultados con el ejercicio previo, decidimos recrear tanto la reducción de dimensionalidad con PCA como la tabla resumen mostrando los porcentajes de participación de cada posición en cada uno de los clústeres, el valor de mercado promedio de cada grupo y volveremos a utilizar la metodología de *Random Forest* para determinar los nuevos *features* más significativos para realizar el *clustering*, de los cuales tomaremos los cinco más importantes para conocer sus promedios en cada clúster. Los resultados se pueden apreciar en la figura 18 y la tabla 8 debajo:

Figura 18. Reducción de dimensionalidad con PCA: *K-means* 4 clústeres (sin posición)

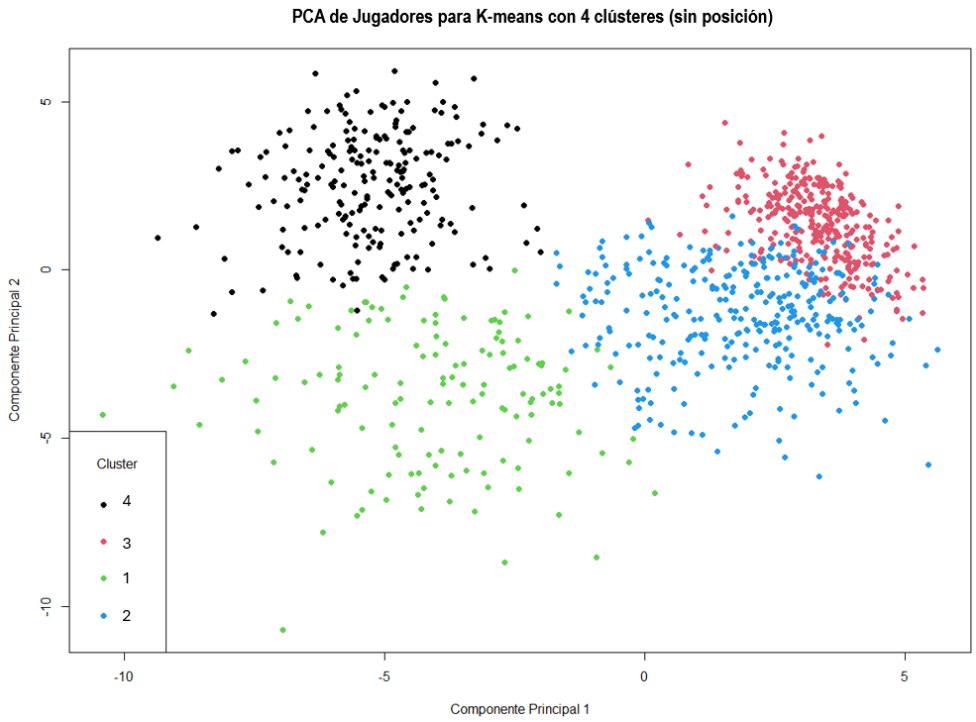


Tabla 8. Detalles por clúster de *K-means* (sin posición)

Variables	Clúster asignado			
	1	2	3	4
Defensores (%)	13.5%	0.0%	0.0%	87.8%
Mediocampistas (%)	86.2%	54.1%	3.3%	12.2%
Delanteros (%)	0.3%	45.9%	96.7%	0.0%
Duelos atacantes /90 (promedio)	4.47	10.71	9.35	1.72
Asistencias /90 (promedio)	0.55	1.39	0.73	0.14
Acciones de ataque exitosas /90 (promedio)	1.32	3.83	2.34	0.54
Pases hacia adelante /90 (promedio)	14.38	10.48	4.00	15.15
Desmarques /90 (promedio)	0.25	0.74	0.29	0.04
Valor de mercado Transfermarkt (promedio)	$ 896,691	$ 1,059,274	$ 921,429	$ 531,814

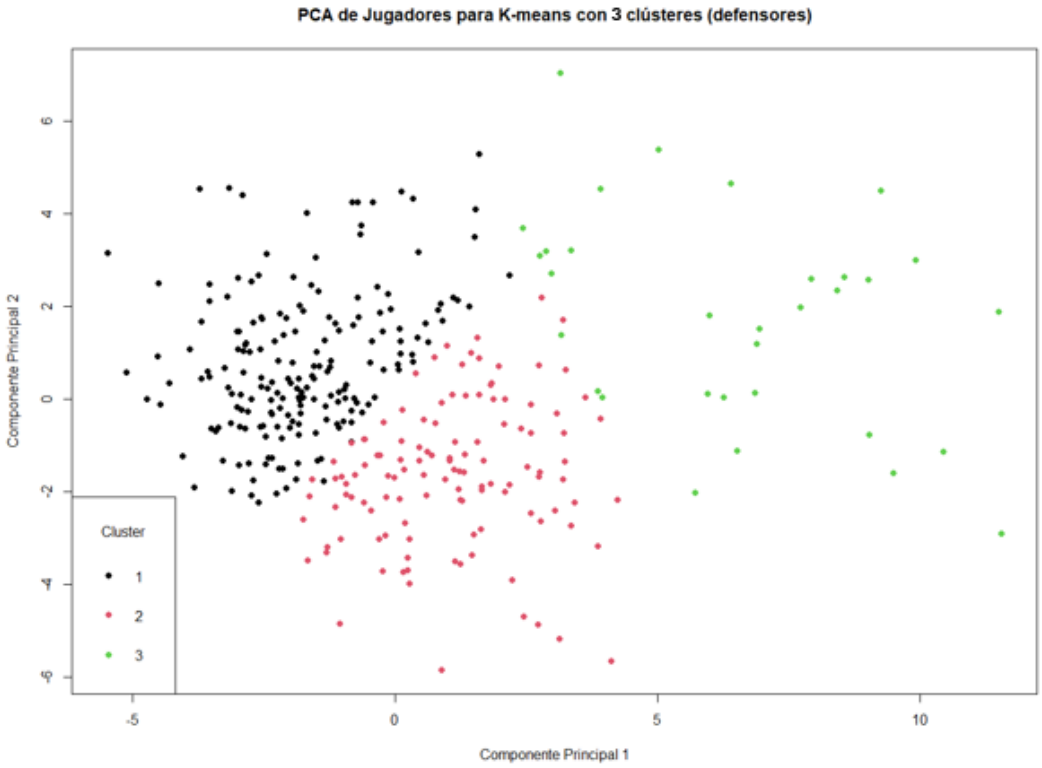
Considerando la información obtenida, podemos afirmar que los jugadores continúan agrupándose principalmente por posiciones dentro del campo de juego, incluso aún cuando la variable "posición" queda por fuera del ejercicio. Las segmentaciones de este experimento son muy similares al del escenario inicial, aunque se puede percibir que en los clústeres que anteriormente había una predominancia excesiva de una única posición, cómo en los grupos 1 y 4, ahora la posición dominante se ha visto reducida. Con respecto a las variables más importantes para predecir la asignación al clúster, estás continúan siendo mayoritariamente ofensivas, lo que genera la agrupación de los mediocampistas ofensivos y delanteros en el clúster 2. Por último, podemos resaltar que se mantienen las conclusiones respecto a los promedios de valor de mercado para cada uno de los segmentos. En base a este análisis, concluimos que la variable "posición" no influye significativamente sobre los resultados de las agrupaciones y estas se realizan primordialmente en base a las características de juego de cada jugador.

En vista de estos resultados, nos parece relevante extender el análisis de *clustering* dentro de cada posición, lo que podría darnos más información acerca de los estilos particulares de juego dentro de cada una de estas con el fin de mejorar la interpretación sobre las distintas modalidades de los jugadores segmentados. Para esto, dividiremos nuestra base de datos en tres conjuntos diferentes donde cada uno de ellos contenga únicamente jugadores correspondientes a una sola posición (defensores, mediocampistas, delanteros) y continuaremos utilizando la misma metodología con la que venimos trabajando. Utilizando el algoritmo de *K-means*, primero determinaremos el número de clústeres a través del método del codo, segundo realizaremos el agrupamiento, tercero graficaremos los resultados haciendo uso de la reducción de dimensionalidad con PCA y finalizaremos calculando el valor de ASW para cada uno de estos.

Empezando con la base de datos correspondientes a los defensores, el método del codo da como resultado que la mejor agrupación es la correspondiente a tres clústeres. Conforme a esto, ejecutamos el algoritmo de *K-means* con un valor de "K" clústeres de

tres y graficamos los resultados usando la reducción de dimensionalidad con PCA (figura 19). El valor de ASW para este ejercicio es de 0.08.

Figura 19. Reducción de dimensionalidad con PCA: *K-means* 3 clústeres para defensores



Con el fin de hacer una mejor interpretación de estos resultados, decidimos analizar algunas métricas promedio con las que cuenta cada clúster. Para la elección de estas, utilizamos nuevamente una predicción de *Random Forest* para detectar variables importantes y nuestro conocimiento del juego para terminar de seleccionarlas. Además, contemplamos el campo de "posición específica" relacionada a cada jugador para dividirlos a estos principalmente entre defensores "centrales" y defensores "laterales". Resumimos estos hallazgos en la tabla 9 que se encuentra abajo.

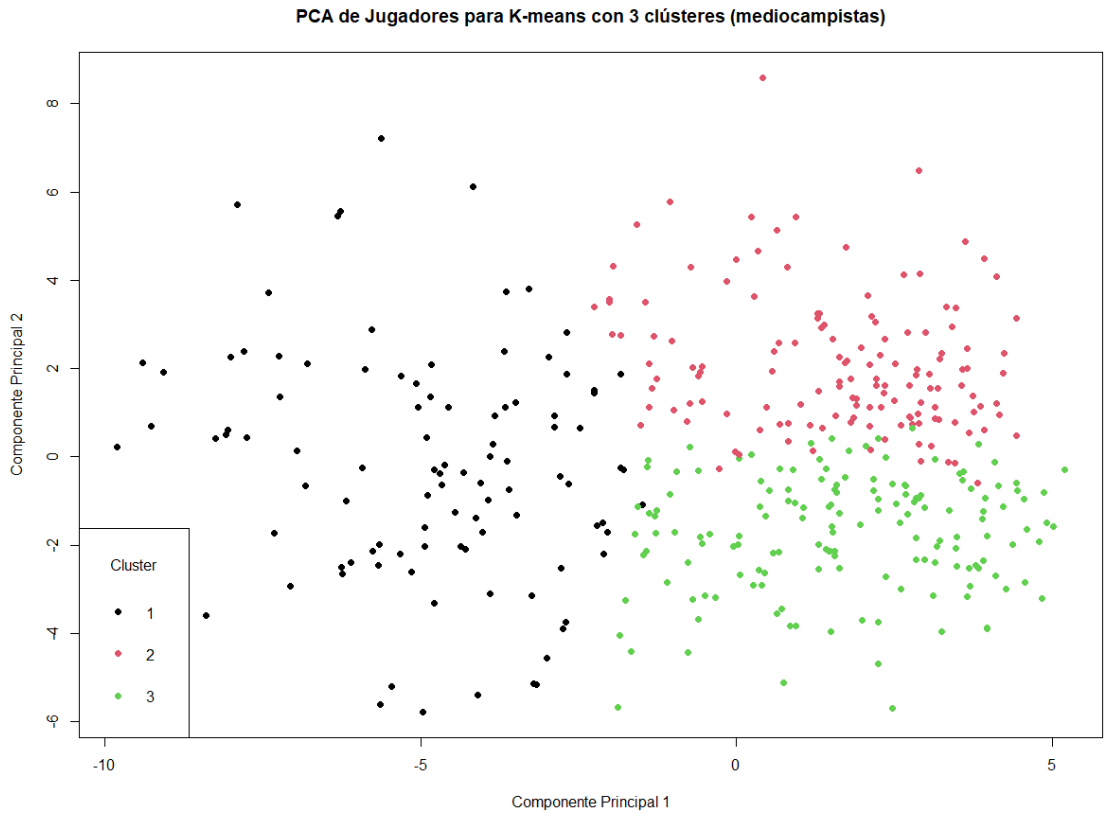
Tabla 9. Detalles por clúster de *K-means* para defensores

Variables	Clúster asignado		
	1	2	3
Defensores centrales (%)	93.0%	89.2%	56.3%
Defensores laterales (%)	7.0%	10.8%	43.8%
Pases recibidos /90 (promedio)	16.11	26.81	26.50
Pases en el último tercio /90 (promedio)	3.95	5.55	7.24
Duelos aéreos en los 90 (promedio)	4.82	4.43	3.65
Acciones de ataque exitosas /90 (promedio)	0.42	0.61	1.57
Carreras en progresión /90 (promedio)	0.35	0.84	1.20
Valor de mercado Transfermarkt (promedio)	$ 414,032	$ 680,833	$ 869,531

Las principales conclusiones acerca de la conformación de los clústeres que se pueden alcanzar con respecto a los datos obtenidos son: el primer clúster se compone principalmente de jugadores centro-defensivos, quienes tienen un rol más agresivo a la hora de defender, no reciben tantos pases y suelen tener más duelos aéreos al ubicarse más próximos a la propia portería. El segundo clúster cuenta con jugadores híbridos, que, si bien en su mayoría se desempeñan como defensores centrales, estos pueden ser capaces de tomar roles un poco más ofensivos, participando activamente del juego, con promedios significativamente mayores en la recepción y distribución de pases. El tercer clúster ya da muestras de una mayor inclusión de defensores laterales, que los dos anteriores. Estos se destacan tanto en las acciones de pases, como en las acciones de ataque y carreras en progresión. Por último, nos parece significativo de cara a los algoritmos predictivos destacar que se repite nuevamente el patrón de contar con valores de mercado promedio más altos para los jugadores con características más ofensivas.

Proseguimos con el conjunto de datos respectivo a los mediocampistas, para estos, el método del codo también señala que debemos usar un valor "K" clústeres de tres. Implementamos *K-means* con tres grupos y diagramamos los resultados nuevamente haciendo uso de la reducción de dimensionalidad con PCA (figura 20). El valor computado de ASW para este ejercicio es de 0.10.

Figura 20. Reducción de dimensionalidad con PCA: *K-means* 3 clústeres para mediocampistas



Continuando con la misma metodología que aplicamos para la base de defensores, generamos un resumen correspondiente a los resultados que vemos para la

segmentación de mediocampistas utilizando las predicciones de *Random Forest* acerca de las variables más importantes y nuestro conocimiento de fútbol. En este caso también usamos la variable "posición específica" pero dado que las posiciones de los mediocampistas suelen ser más complejas que las de los defensores, tomando roles muy variados, decidimos segmentar la participación en tres categorías a las que llamamos "ofensivos", "defensivos" y "de creación". Los resultados se pueden ver en la tabla 10.

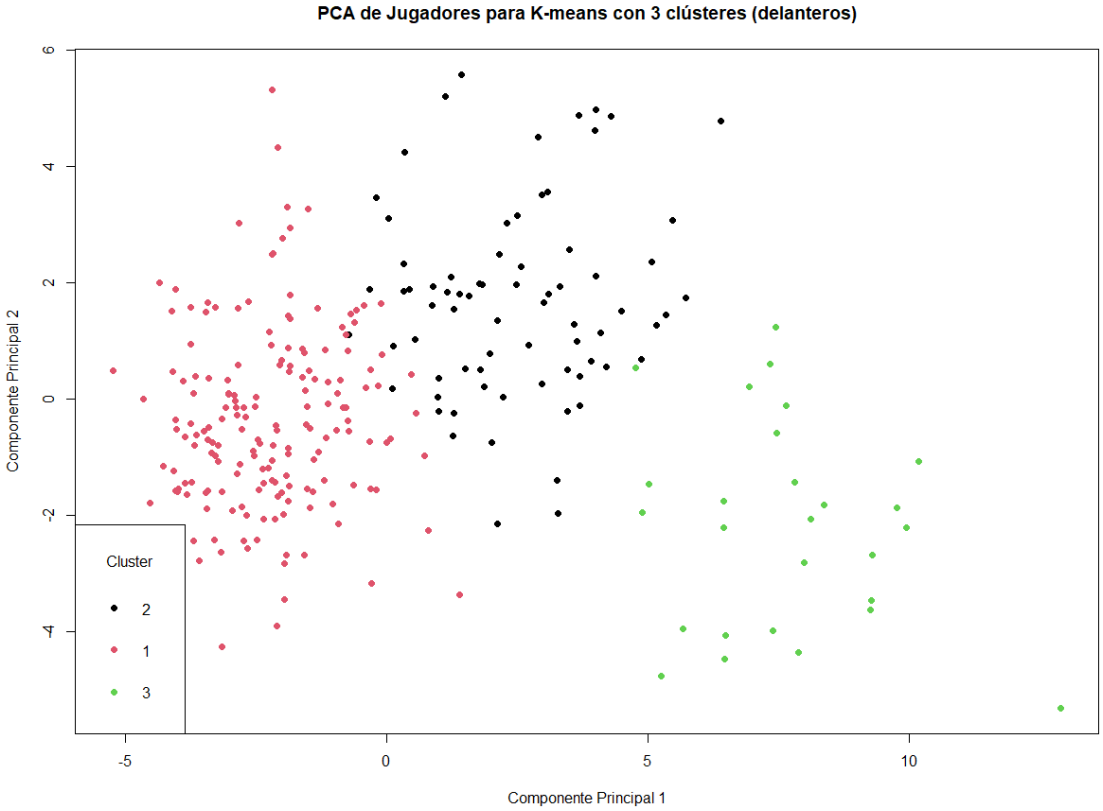
Tabla 10. Detalles por clúster de *K-means* para mediocampistas

Variables	Clúster asignado		
	1	2	3
Mediocampistas ofensivos (%)	52.5%	2.6%	1.4%
Mediocampistas defensivos (%)	2.0%	38.1%	54.3%
Mediocampistas de creación (%)	45.5%	59.4%	44.3%
Pases recibidos /90 (promedio)	26.00	30.84	19.71
Pases cortos / medios /90 (promedio)	31.21	39.67	28.44
Acciones de ataque exitosas /90 (promedio)	3.25	1.51	1.20
Acciones defensivas realizadas /90 (promedio)	6.48	9.11	10.05
Regates /90 (promedio)	3.57	1.39	1.25
Asistencias /90 (promedio)	1.25	0.58	0.46
Valor de mercado Transfermarkt (promedio)	$ 1,062,037	$ 939,634	$ 660,000

Las observaciones más importantes sobre la formación de los clústeres basadas en los datos recolectados son: el primer clúster se compone mayoritariamente de volantes ofensivos, quienes tienen métricas intermedias en lo que respecta a la creación de juego, haciendo referencia a los pases, y muestran un valor muy por encima de los demás grupos en variables plenamente ofensivas como acciones de ataque exitosas, regates y asistencias. El segundo clúster se destaca principalmente por contar en su mayoría con jugadores creativos, quienes sobresalen en la recepción y distribución de los pases. En el tercer clúster predomina la presencia de volantes defensivos, quienes se destacan en el *feature* de acciones defensivas realizadas por partido y tienen características más similares a los jugadores correspondientes a la posición de defensor. No es menor que notamos nuevamente la correlación positiva entre el valor de mercado y los estilos de juego más ofensivos.

Por último, repetimos este enfoque sobre la base de datos relacionada a los delanteros. Empezando por el método del codo, determinamos que éste también debería realizarse en tres segmentos. Pusimos en marcha el algoritmo de *K-means* de tres clústeres y graficamos los resultados nuevamente a través de la reducción de dimensionalidad con PCA (figura 21). El cálculo de ASW en este caso muestra 0.16.

Figura 21. Reducción de dimensionalidad con PCA: *K-means* 3 clústeres para delanteros



Aplicando el mismo método utilizado tanto para defensores como mediocampistas, elaboramos un resumen de los resultados obtenidos en la agrupación de delanteros mediante las predicciones de *Random Forest* sobre las variables más relevantes y nuestro conocimiento del juego. También basándonos en la información de la variable "posición específica" definimos tres estilos de juegos predominantes en los delanteros "centrodelantero", "extremo" y "mediapunta". El resumen se observa en la tabla 11 debajo.

Tabla 11. Detalles por clúster de *K-means* para delanteros

Variables	Clúster asignado		
	1	2	3
Centrodelantero (%)	42.5%	94.2%	14.8%
Extremo (%)	43.8%	4.0%	7.4%
Mediapunta (%)	13.7%	1.7%	77.8%
Pases progresivos /90 (promedio)	3.51	1.67	6.82
Pases al área de penalti /90 (promedio)	2.55	1.17	3.82
Toques en el área de penalti /90 (promedio)	2.90	3.36	1.82
Goles excepto los penaltis /90 (promedio)	0.20	0.28	0.14
Regates /90 (promedio)	4.91	2.11	3.49
Asistencias /90 (promedio)	1.82	0.69	1.34
Valor de mercado Transfermarkt (promedio)	$ 1,155,303	$ 903,125	$ 694,231

Los hallazgos clave sobre la formación de los clústeres, derivados del análisis de los datos recolectados muestran que: el primer clúster está principalmente formado por delanteros con funciones de extremos, donde se acentúan las habilidades de regates y asistencias. Estos son jugadores que suelen desbordar y asistir a los centrodelanteros. El segundo clúster está compuesto casi exclusivamente por centrodelanteros, quienes se destacan por tocar más veces el balón en el área rival y marcar los goles. El tercer clúster muestra una gran presencia de jugadores clasificados como "mediapunta". Estos jugadores, suelen ubicarse detrás de los centrodelanteros concentrándose en la creación de juego y de oportunidades de gol. A nivel de *features* estos resaltan en aquellos de creación como pases progresivos y pases al área de penalti. Por último, comentando acerca de los valores de mercado, aquí notamos que las características relacionadas a los jugadores que se posicionan como extremos son las más valoradas de todas, incluso por encima de los centrodelanteros.

Concluyendo este último análisis respectivo a los ejercicios de *clustering*, podemos afirmar que los clústeres dentro de cada posición nos ofrecen una visión más detallada de cada grupo, revelando estilos de juego más particulares que pueden facilitar la identificación de jugadores aún más similares que mediante la clasificación inicial que realizamos por posición. No obstante, al adoptar este enfoque, existe el riesgo de segmentar excesivamente y perder de vista jugadores que, aunque no ocupen el rol idéntico al jugador objetivo, podrían representar una opción viable.

Algoritmos predictivos

En segundo lugar, con respecto a los algoritmos predictivos, recordemos que aplicamos los siguientes (utilizando las variables seleccionadas por *backward stepwise*, tabla 3): Regresión lineal, Regresión *Ridge*, *Random Forest* y *XGBoost*. Durante la sección de metodología comentamos que íbamos a utilizar como medida de *performance* y comparación el *Weighted Absolute Percentage Error* (*WAPE*). A medida que fuimos implementando cada uno de los modelos computamos el *WAPE* obtenido por lo que decidimos resumirlo en la tabla 12 que se aprecia debajo.

Tabla 12. *WAPE* por metodología predictiva.

Metodología	WAPE (%)
Regresión Lineal	73.6%
Regresión *Ridge*	71.2%
Random Forest	61.8%
XGBoost	61.2%

Examinando los resultados, vemos que los algoritmos de *Random Forest* y *XGBoost* son los que mejor rendimiento tienen. Si bien la regresión *Ridge* logra una mejora de 2.4 puntos porcentuales versus la regresión lineal, su *performance* queda lejos de la de los modelos ensamblados con árboles de decisiones con casi 10 puntos porcentuales de diferencia contra estos. Por esta razón decidimos descartar tanto el modelo de regresión lineal como la regresión *Ridge*.

A continuación, dado que los mejores resultados los obtuvimos de *Random Forest* y *XGBoost*, propondremos algunas ideas y las probaremos con el fin de lograr alguna mejora sobre estos.

Lo primero que probamos, es volver atrás la modificación propuesta por *backward stepwise*, lo que nos deja con nuestro dataset inicial trabajado, pero sin ningún recorte de variables por este método automático de selección de *features* que habíamos implementado. A esta prueba la llamamos "*Dataset* completo". Ajustamos nuevamente ambos modelos y los resultados de *WAPE* para las predicciones son de 60.7% para *Random Forest* y 61.7% para *XGBoost.*

Lo segundo que hicimos, es probar nuevamente el ajuste de los modelos con la base completa de *features* con el agregado de escalar todas las variables independientes. A este caso lo llamamos "*Dataset* completo normalizado" y los resultados del *WAPE* para sus predicciones son de 60.7% para *Random Forest* y 61.7% para *XGBoost*, obteniendo resultados casi idénticos que el caso anterior.

Lo tercero que intentamos, es aprovechar que tanto el modelo de *Random Forest* como *XGBoost*, permiten computar la importancia de cada *feature* en la predicción de la variable dependiente. Utilizando los datos de la prueba "*Dataset* completo" computamos los 30, 20, 15 y 10 *features* más importantes para cada uno. Entre estas variables destacamos la aparición repetida de la *dummy* "LigaArgentina", "Edad", "Minutos jugados", "Pases recibidos /90" y "Pases cortos/medios /90". Resumimos los resultados en la tabla 13 que se aprecia debajo, donde también añadimos los rendimientos conseguidos por los casos de "*Dataset* completo" y "*Dataset* completo normalizado".

Tabla 13. Resultados de *WAPE* con modificaciones sobre *Random Forest* y *XGBoost*

Metodología	Inicial	Dataset completo	Dataset completo normalizado	30 Variables más importantes	20 Variables más importantes	15 Variables más importantes	10 Variables más importantes
Random Forest	61.8%	60.7%	60.7%	59.8%	59.8%	60.7%	64.3%
XGBoost	61.2%	61.7%	61.7%	61.5%	64.3%	61.4%	69.0%

Analizando los resultados obtenidos, vemos rendimientos diferentes tanto para *Random Forest* como para *XGBoost*. Para el primero vemos que se logra mejorar la *performance* obtenida inicialmente en casi todos los casos, salvo en el modelo

entrenado con las 10 variables más importantes, el cual retrocede del escenario inicial. Viendo el valor de *WAPE* alcanzado con todos sus decimales, decimos que el mejor modelo para *Random Forest* es aquel que se entrena con las 20 variables que este considera más importantes cuando se le comparte la base de datos con todos los *features*, logrando superar al modelo inicial en 2 puntos porcentuales. Por otro lado, en el caso de *XGBoost*, de los nuevos modelos ajustados el que mejor rendimiento tiene, es aquel que selecciona las 15 variables más importantes para este algoritmo. Sin embargo, este alcanza un *WAPE* de 61.4% quedando 0.2 puntos porcentuales por encima del escenario inicial.

Partiendo de los resultados de la tabla 13 y lo comentado previamente, podemos confirmar que el algoritmo con mejores rendimientos es *Random Forest*. Observando la comparación modelo a modelo contra *XGBoost*, de las siete variantes que planteamos, en seis logra mejor *performance*. Además, no es un dato menor que en cinco de los ajustes planteados supera también al mejor modelo de *XGBoost*, el cual es el inicialmente planteado con las variables definidas por *backward stepwise*.

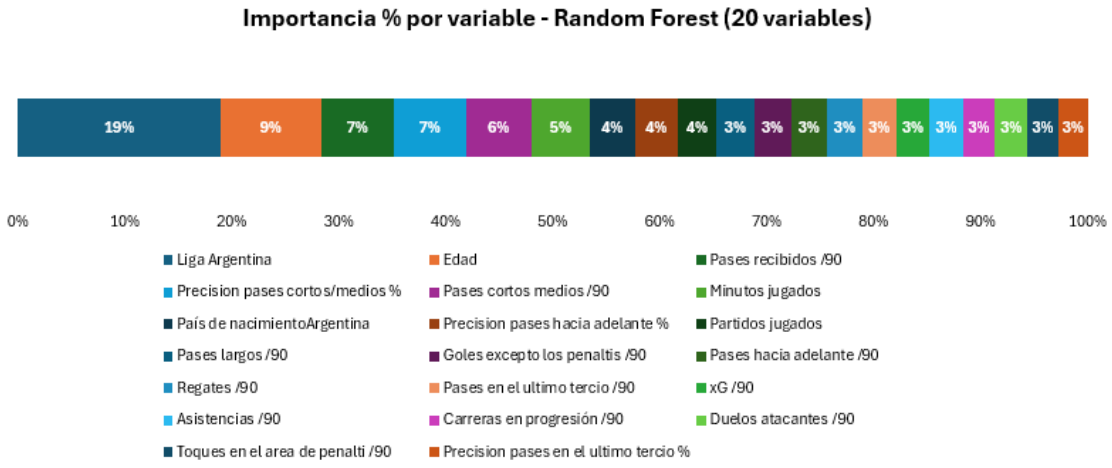
Finalizando la comparación entre estos dos algoritmos predictivos, podemos decir en función de los resultados que la mejor opción para predecir el valor de mercado sobre una base general de jugadores es *Random Forest*. En base a esto, decidimos expandir nuestro análisis para explorar con mayor profundidad el rendimiento del modelo que es más preciso en las predicciones, para nuestro trabajo, el algoritmo de Random Forest con las 20 variables más importantes definidas por éste.

En primer lugar, nos parece importante esclarecer cuales son las 20 variables más importantes seleccionadas por el modelo y también entender qué relevancia tiene cada una para la predicción; por lo que a continuación construimos una tabla que señala cuáles son estos *features* en conjunto con su correspondiente porcentaje de aumento en el error cuadrático medio (*%incMSE*) calculado por el algoritmo y un gráfico de barra apilada donde podemos ver el porcentaje de importancia que tiene cada uno de estos para el modelo.

Tabla 14. Las 20 variables más importantes para *Random Forest*

Rank	Variable	%IncMSE
1	Liga Argentina	97.14
2	Edad	48.11
3	Pases recibidos /90	35.06
4	Precision pases cortos/medios %	34.42
5	Pases cortos medios /90	31.69
6	Minutos jugados	27.70
7	País de nacimientoArgentina	21.81
8	Precision pases hacia adelante %	20.50
9	Partidos jugados	18.66
10	Pases largos /90	17.92
11	Goles excepto los penaltis /90	17.69
12	Pases hacia adelante /90	17.39
13	Regates /90	16.83
14	Pases en el ultimo tercio /90	16.25
15	xG /90	16.15
16	Asistencias /90	15.90
17	Carreras en progresión /90	15.53
18	Duelos atacantes /90	15.48
19	Toques en el area de penalti /90	14.79
20	Precision pases en el ultimo tercio %	14.13

Figura 22. Importancia % por variable – *Random Forest* (20 variables)

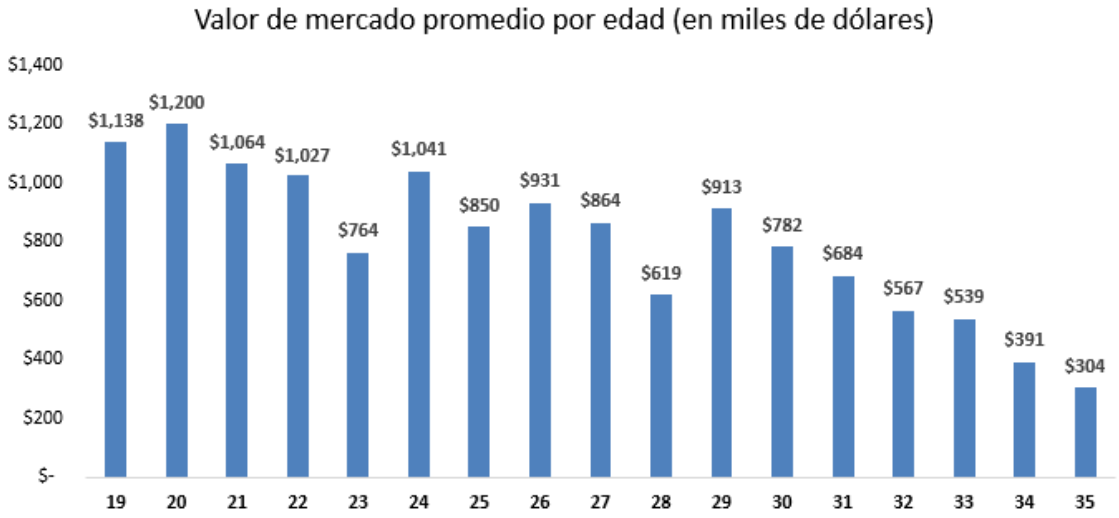


En base a estos resultados, confirmamos la importancia de la variable *dummy* "Liga Argentina", la cual no solo es la variable más relevante para el modelo, sino que, sin ésta, Random Forest podría casi duplicar el error cuadrático medio si no se la incluyera, convirtiéndose en un *feature* crucial para la predicción. Recordemos que el valor de mercado promedio por jugador para la liga argentina es más del doble que el de la liga chilena y hasta más del triple que del resto de las ligas incluidas en este análisis.

Por otro lado, notamos la aparición de la variable de información general "Edad", lo cual tiene mucho sentido ya que bien se sabe que aquellos jugadores que estén en una

edad más cercana al retiro tendrán una valuación menor ya que pierden valor de reventa para el club que los contrate. Debajo, añadimos un gráfico de barras que enseña el valor de mercado promedio por edad para esclarecer estos dichos (figura 23). También, tal como pensábamos, observamos que en la tabla 14 se encuentran ambas variables de tiempo de juego "Minutos jugados" y "Partidos jugados" ya que, a mayor cantidad de ambos, podríamos suponer que los jugadores son titulares en sus equipos, lo que aumentaría el valor de mercado para estos dada la relevancia que tienen en sus clubes.

Figura 23. Valor de mercado promedio por edad (en miles de dólares)



Otro punto para resaltar y relacionado con el análisis de *clustering* realizado previamente, es la gran presencia de variables relacionadas al armado de juego y al juego ofensivo, principalmente relacionada a mediocampistas atacantes y delanteros. Cuando exploramos con mayor profundidad los resultados de *K-means* veíamos este mismo patrón de priorizar las características ofensivas y un mayor valor de mercado promedio por jugador para aquel clúster que estaba integrado por volantes de ataque y delanteros.

En segundo lugar, decidimos continuar explorando variantes de este modelo con el fin de entender si es posible mejorar aún más los valores de *WAPE* alcanzados. Apoyándonos en los resultados obtenidos recientemente tanto de los ejercicios de *clustering*, como los de predicción y también en el análisis de datos inicial de este trabajo final, plantearemos distintas aperturas de la base de datos y predeciremos el valor de mercado para cada una de ellas utilizando nuestro *Random Forest* con las 20 variables más importantes.

La primera apertura realizada es por posición, donde dividimos la base de datos en tres: una para defensores, otra para mediocampistas y la última para delanteros. Debajo se pueden observar los resultados de *WAPE* para cada una de éstas y además incluimos el porcentaje de observaciones con el que se queda cada base de datos y el dato de coeficiente de variación con el fin de entender si esto puede estar afectando a las predicciones.

Tabla 15. Resultados de *WAPE* por posición para *Random Forest* (20 variables)

Posición	% de observaciones	Wape %	Coeficiente de variación
Defensores	32.2%	58.0%	1.28
Mediocampistas	37.6%	67.8%	1.54
Delanteros	30.2%	78.5%	1.51
Global	**100.0%**	**59.8%**	**1.50**

Considerando los resultados de la tabla 15, notamos rendimientos realmente diferentes para cada caso. El modelo que predice el valor de mercado de los defensores logra superar a su versión que incluye a todos los jugadores, mientras que los modelos para mediocampistas y delanteros tienen mayor dificultad para predecir que cualquiera de los modelos anteriores de este mismo algoritmo. La posible razón detrás de estos valores de *WAPE* en los últimos dos casos es la gran dispersión de valores de mercados que hay para esas posiciones, muy similares a la dispersión de la base global, pero contemplando una cantidad considerablemente menor de observaciones. Adentrándonos en nuestra base de datos podemos ver que los valores de mercado para mediocampistas pueden alcanzar los 11 millones de dólares y para los delanteros un total de 16 millones de dólares cuando los defensores no logran superar el valor de 4.5 millones de dólares.

La segunda segmentación que realizamos es por liga, donde realizamos una división para cada una de las presentes en nuestra base de datos: argentina, chilena, colombiana, paraguaya y uruguaya. Tomamos esta decisión basada en las grandes diferencias que notamos a lo largo de este trabajo en el valor de mercado promedio por jugador para cada una de ellas, principalmente de la liga argentina con el resto. Los resultados de esta se pueden ver a continuación.

Tabla 16. Resultados de *WAPE* por liga para *Random Forest* (20 variables)

Liga	% de observaciones	Wape %	Coeficiente de variación
Argentina	30.4%	64.6%	1.26
Chile	16.2%	48.4%	0.68
Colombia	24.2%	47.5%	0.77
Paraguay	13.0%	79.7%	1.46
Uruguay	16.3%	52.4%	1.34
Global	**100.0%**	**59.8%**	**1.50**

Con respecto a los resultados de la tabla 16, también vemos un amplio rango de valores de *WAPE*. Notamos que hay modelos que superan considerablemente los resultados alcanzados por el modelo global. Aquellos que muestran un mejor rendimiento son los que se ejecutan con los jugadores de la liga de Chile y Colombia. Esto no es casualidad ya que también son estos los que tienen un menor coeficiente de variación. Además, a pesar de que el coeficiente de variación es mayor para el modelo de la liga colombiana que para el de la chilena, creemos que el primero logra mejores resultados en sus predicciones dado que supera en gran medida la cantidad de observaciones para su entrenamiento.

Por otro lado, y en contraposición con lo comentado anteriormente, el modelo que predice los valores de mercado para los jugadores de la liga de Paraguay es el que peor *performance* logra al contar con un coeficiente de variación bastante alto y muy similar al del modelo global y asimismo cuenta con muy pocas observaciones.

Con respecto al modelo de la liga argentina, podemos decir que su resultado es consistente a los valores que muestra de porcentaje de observaciones y coeficiente de variación. Si bien reduce el coeficiente en comparación al modelo global, este solo cuenta con un 30% de las observaciones y el modelo tiene un rendimiento similar a las pruebas que realizamos con *Random Forest* (tabla 13).

Por último, el caso de la liga de Uruguay nos resulta bastante particular ya que consideramos que cuenta con un coeficiente de variación alto y un bajo porcentaje de las observaciones totales, sin embargo, su rendimiento logra superar al modelo global. Ahondando más en la base de datos, notamos que esta liga tiene la particularidad de contar con un 9% de jugadores de alto valor de mercado en comparación con el resto de sus observaciones, lo que genera el aumento desmedido del coeficiente de variación. Si quitáramos dichos registros y computáramos nuevamente este coeficiente, obtendríamos un valor de 0.63, lo que explica que, a pesar de los valores señalados en la tabla, este modelo alcanza un gran rendimiento.

La tercera división que exploramos fue por cuartiles de valor de mercado. Ordenamos nuestra base de datos por esta variable de menor a mayor y realizamos las divisiones correspondientes. De este modo garantizamos el 25% de observaciones a cada grupo y podemos comprender de mejor manera donde se encuentran las dificultades para nuestro modelo.

Tabla 17. Resultados de *WAPE* por cuartiles de valor de mercado para *Random Forest* (20 variables)

Cuartiles de valor de mercado	% de observaciones	Wape %	Coeficiente de variación
0%-25%	25.0%	35.9%	0.49
25%-50%	25.0%	13.8%	0.16
50%-75%	25.0%	17.2%	0.21
75%-100%	25.0%	51.6%	0.87
Global	**100.0%**	**59.8%**	**1.50**

Examinando los resultados de la tabla 17, vemos claramente la correlación existente entre el coeficiente de variación y el *WAPE* obtenido. Los modelos que capturan los datos centrales de valor de mercado tienen una excelente *performance* y un coeficiente de variación muy bajo. Yendo a los datos numéricos el cuartil 25%-50% se compone de jugadores que rondan un valor de $250.000 hasta $400.000 dólares y el cuartil 50%-75% cuenta con jugadores que van desde los $400.000 hasta los $750.000 dólares ambos sin contar con ningún *outlier*.

Por otro lado, las predicciones se dificultan más en el primer cuartil 0%-25% y sobre todo en el último de 75%-100% donde aumenta considerablemente el coeficiente de variación con respecto a los otros cuartiles y donde mayor presencia de datos atípicos existe, sin embargo, éste se mantiene muy por debajo del modelo global y logra superar en ambos casos su resultado de *WAPE*. No es menor que el último cuartil cuenta con una gran variedad de valores de mercado que van desde los $750.000 hasta los $16.000.000, lo que dificulta más al modelo a la hora de hacer sus predicciones.

Si bien estos rendimientos eran esperables debido a los coeficientes de variación capturados en cada cuartil, creemos que es importante mostrarlo ya que habla del nivel de confiabilidad de los resultados. Aquellos resultados de valoración media, donde el modelo es más preciso, pueden inspirar más confianza al cuerpo técnico del club para utilizar el modelo cuando no se trate de contrataciones de jugadores peculiarmente baratos o caros.

Para finalizar con el análisis de los resultados obtenidos, generamos dos gráficos de dispersión: uno para analizar la relación de la variable estadística de juego más importante para nuestro modelo y el valor real de los jugadores y segundo otro para comparar los valores predichos del modelo con los valores reales de los jugadores. En ambos casos añadiremos con color el clúster al que pertenecen en la ejecución de *K-means* y solo consideraremos a aquellos jugadores que se encuentren en nuestra base de datos de prueba, para los que contamos con un valor predicho.

Figura 24. Gráfico de dispersión de valor de mercado real vs pases recibidos /90

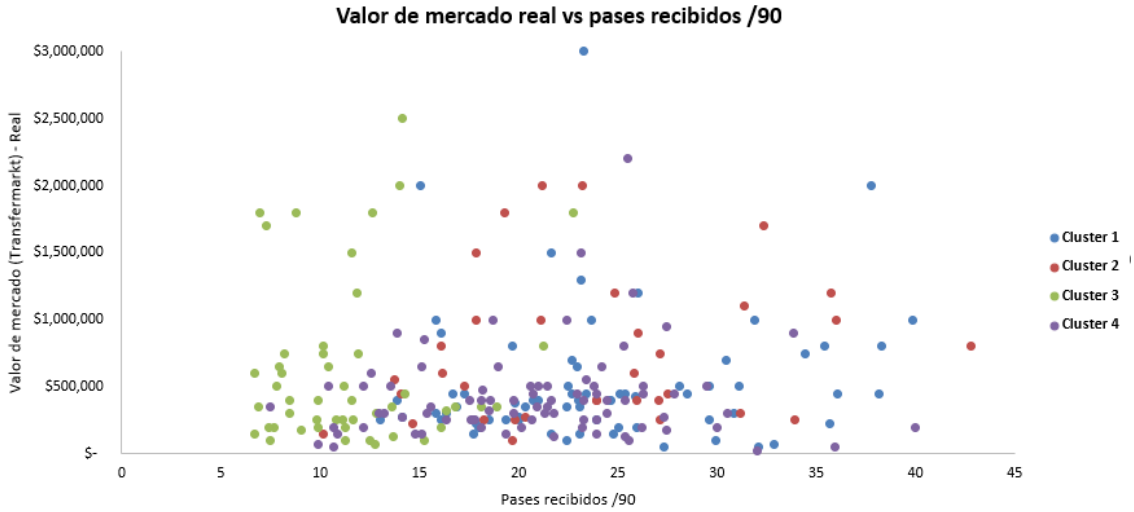
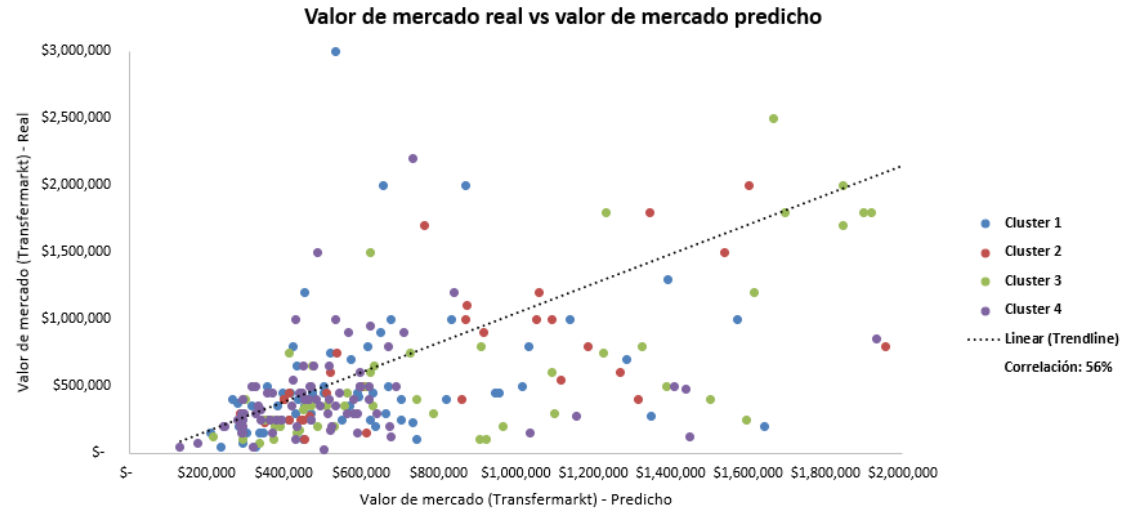


Figura 25. Gráfico de dispersión de valor de mercado real vs valor de mercado predicho



Con respecto al primer gráfico de dispersión, notamos cómo principalmente los delanteros, pertenecientes al clúster 3, logran valores altos de mercado a pesar de no lograr un buen rendimiento de la variable “pases recibidos /90” el cual es el *feature* estadístico de juego más relevante. A su vez, gracias a este gráfico también podemos observar lo comentado acerca del clúster 4 correspondiente mayoritariamente a los defensores, quienes parecieran que sin importar sus respectivos valores de “pases recibidos /90”, mantienen generalmente valores de mercado relativamente bajos. Por otro lado, los clústeres 1 y en mayor medida el 2, donde predominan los mediocampistas, parecen ser los que mejor correlacionan estas variables mostrando valores de mercado altos para jugadores con valores de “pases recibidos /90” generalmente superiores al promedio.

Por último, examinando el segundo gráfico de dispersión, podemos repetir lo anteriormente mencionado acerca del clúster 4, cuyos jugadores se encuentran principalmente en el cuadrante inferior izquierdo, estando menos dispersos y teniendo

valores muy similares de valor de mercado real y valor de mercado predicho. Es por esta razón que los mejores resultados de *WAPE* en la apertura del modelo por posiciones se alcanzan en aquel que considera a los jugadores con rol defensivo. Con respecto a los demás clústeres no se alcanza a notar algún patrón en singular, pero a nivel general del modelo podemos ver a partir del gráfico como nuestro algoritmo tiende a sobrevalorar a los jugadores siendo que el valor de mercado predicho se ve generalmente superior al valor de mercado real. Un buen ejemplo para ilustrar esta última idea y relacionarla con los análisis realizados, es el caso del jugador uruguayo Enzo Larrosa quien se desempeñó como delantero central en Godoy Cruz de Mendoza (liga argentina) durante el período analizado. Éste tiene un valor real en *Transfermarkt* de $400.000 dólares, pero al participar de la liga argentina, tener tan solo 22 años y contar con un rol plenamente ofensivo, nuestro modelo termina valuándolo en $1.500.000 dólares.

5. Conclusiones

A lo largo de este trabajo final, hemos comentado acerca de la importancia del proceso de reclutamiento de nuevos jugadores, principalmente para los planteles de fútbol masculino profesional. Como bien mencionamos, las ventanas de transferencia que se dan durante los recesos de verano e invierno son momentos cruciales para que los equipos puedan recurrir a los mercados de pases y conseguir nuevos jugadores ya sea para reemplazar a algún jugador que esté lesionado o que vaya a abandonar la institución por diferentes motivos como también buscar jugadores que comiencen a aportar sobre distintas falencias que tenga el equipo.

En el caso particular del fútbol argentino y mayoritariamente en el sudamericano, tanto los cuerpos técnicos, cómo los *managers* deportivos de cada equipo son los principales responsables de buscar y conseguir las mejores opciones a las que los equipos puedan acceder. La problemática yace en que tanto los cuerpos técnicos, como los gerentes deportivos, tienen un sinfín de tareas además del reclutamiento de nuevos jugadores, lo que les impide poder enfocarse por completo en la importante labor de conseguir las mejores contrataciones.

Para desempeñar mejor esta tarea, equipos de las grandes ligas europeas, con presupuestos mucho mayores a los del fútbol sudamericano, han comenzado a contratar equipos especializados de analistas e ingenieros de datos con el fin de optimizar no solo el tiempo de la búsqueda, sino la calidad de las contrataciones realizadas en cada mercado de pases.

En base a esto, nos propusimos colaborar con la optimización de los procesos de reclutamiento actuales en el fútbol argentino. Mediante un contacto que forma parte del cuerpo técnico de un equipo de la primera división del torneo argentino, conseguimos una base de datos con información general y estadísticas de jugadores de fútbol de distintas ligas sudamericanas (Argentina, Chile, Colombia, Paraguay y Uruguay). La finalidad de este trabajo siempre apuntó a optimizar los tiempos en conseguir jugadores similares a un jugador de referencia utilizando metodologías de

clustering para acelerar y automatizar el proceso y, por otro lado, a partir de la información provista, predecir el valor de mercado para los jugadores sugeridos con el fin de dar una recomendación monetaria a ofrecer por dichos jugadores.

Comenzamos presentando los datos obtenidos y luego realizamos un análisis descriptivo de los datos que nos serviría para después orientar nuestro trabajo con los algoritmos de *clustering* y predictivos.

Con respecto a los primeros, los algoritmos de *clustering*, evaluamos cinco opciones diferentes, dos relacionadas al *clustering* particional: *K-means* y *PAM,* y tres técnicas pertenecientes al *clustering* jerárquico: *Single Linkage*, *Average Linkage* y *Complete Linkage*. Para todos los casos, comentamos cómo funciona cada uno de los algoritmos y explicamos como realizar la aplicación en R, mostrando y comparando cada resultado obtenido entre sí. Definimos no solo utilizar la medida *ASW* como métrica de calidad para las comparaciones, sino que también usamos la reducción de dimensionalidad con PCA para apreciar de mejor manera los resultados de cada algoritmo.

Las principales conclusiones que alcanzamos de este ejercicio fueron las siguientes: Los algoritmos de *clustering* jerárquico *Single Linkage* y *Average Linkage* no tienen buenos resultados para la finalidad de este trabajo. Ambos generan agrupaciones desbalanceadas de observaciones con grupos de hasta un solo registro contra otros que contienen más del 40% de los jugadores. Sin embargo, podría llegar a ser interesante si quisiéramos detectar jugadores con alguna particularidad específica. Los dos algoritmos de *clustering* particional que aplicamos generaron resultados muy similares debido a su funcionamiento, si bien en la bibliografía consultada se prioriza el método *PAM*, en base a los rendimientos por *ASW* cómo métrica de calidad definimos que *K-means* lograba mejores agrupaciones. Poniendo a prueba finalmente *K-means* y *Complete Linkage* en el ejercicio final de este trabajo, notamos a partir de los resultados presentados por cada algoritmo que *K-mea*ns fue más coherente con la selección de jugadores que *Complete Linkage*. Decimos que fue más coherente ya que el algoritmo nos arrojó jugadores que podrían suplir directamente al jugador objetivo seleccionado, el cual cumple con la finalidad de este trabajo. A pesar de esto, creemos de todos modos que el algoritmo de *Complete Linkage*, por su naturaleza, podría ser de gran ayuda para detectar jugadores que, si bien normalmente no pertenecen a la posición buscada, podrían llegar a adaptarse, cumpliendo un rol híbrido en el plantel. Es por todas estas razones que nuestra recomendación para la búsqueda de jugadores más similares a un jugador objetivo es *K-means*. Por último, una vez definido el algoritmo, exploramos la posibilidad de ahondar el análisis particularmente para cada posición, lo que podría ofrecer una vista más detallada dentro de cada segmento.

Con relación a los segundos, los algoritmos predictivos, evaluamos cuatro opciones diferentes, comenzamos con lo más simple, un algoritmo de regresión lineal. Seguimos con algo un poco más sofisticado, un algoritmo de regresión *Ridge*, el cual es conocido como un método de reducción. Finalmente, pasamos a los algoritmos de ensamble y árboles de decisión donde primero ajustamos un modelo de *Random Forest* y terminamos con la aplicación del algoritmo de *boosting XGBoost.* Al igual que con los

algoritmos de *clustering*, mantuvimos el mismo procedimiento, donde comentamos el funcionamiento de cada uno de estos y explicamos cómo se realizó la aplicación en R para luego interpretar y comparar los resultados obtenidos. En esta ocasión hicimos uso del cálculo *WAPE* para tener una métrica de error de los modelos con el fin de medir la calidad de estos y poder contar con una medición uniforme para la comparación.

Las conclusiones más destacadas que alcanzamos en este ejercicio predictivo fueron: Si bien sabíamos que el modelo de regresión lineal no lograría el mejor desempeño, este nos brindó un primer acercamiento a las predicciones. Como bien explicamos, la imposibilidad de este algoritmo para capturar relaciones no lineales hace que su rendimiento sea el peor en un caso tan complejo como el que se planteó. Sin embargo, fue un gran punto de partida para conocer un valor de *WAPE* inicial y en base a esto lograr las mejoras deseadas. Por otro lado, la utilización de la regresión *Ridge*, no mostró grandes mejoras contra el valor inicial de la regresión lineal, apenas superando a la primera por 2.4 puntos porcentuales de la métrica *WAPE*. En base a esto, se decidió no continuar explorando este camino y pasamos a trabajar con los modelos de ensamble, árboles de decisión y *boosting*. Tal como sucede en la bibliografía consultada, estos modelos fueron los que mostraron un mejor desempeño a la hora de predecir el valor de mercado de los jugadores. *Random Forest* fue sin lugar a duda el algoritmo que mejor *performance* tuvo en las predicciones, especialmente en su variante a la que le aplicamos su propio método de selección de variables automático en base al cálculo de los veinte *features* que éste considero como los más relevantes para la predicción cuando le enseñamos la base de datos completa. Luego exploramos más variantes de este modelo enseñándole diferentes segmentaciones de la base de datos original para entender donde había mejores y peores rendimientos. Así como nos indicaba la intuición, el modelo predecía de mejor forma en aquellos segmentos donde no había tanta dispersión en los valores reales de mercado de los jugadores y de peor manera en los casos contrarios y más si a su vez la segmentación generaba una muestra más pequeña.

En base a todos estos análisis y resultados, la recomendación en este caso, para la predicción de valor de los jugadores, es la utilización del modelo de *Random Forest* que seleccionó por sí mismo las veinte variables más importantes. Además, luego de ver su buena *performance* cuando se lo expuso a una gran cantidad de datos, lo óptimo a realizar una vez encontrado los cinco jugadores más similares al jugador objetivo sería separar únicamente a estas observaciones dentro de la base de datos de prueba y ajustar el modelo de *Random Forest* con la base de datos que contenga al resto de los jugadores.

Por otro lado, quisiéramos sugerir una expansión que se podría realizar sobre lo que ya se hizo en este trabajo final. Con el fin de cerrar el ciclo completo de la tarea de reclutamiento, creemos que podría ser una buena idea explorar la predicción de salario a ofrecer al jugador según su rendimiento. Esto puede ser un poco más complicado debido a la dificultad de conseguir los datos salariales a causa de la sensibilidad de esta información. Sin embargo, si se llegara a contar con una variable que refleje el salario

del jugador, creemos que se podría aplicar la misma metodología para predecir esto y hacer el ofrecimiento correspondiente en el contrato.

Para finalizar, creemos que los resultados obtenidos en esta tesis ofrecen una base sólida para futuras investigaciones y aplicaciones en diversos deportes y contextos. Se podría llegar a replicar este trabajo considerando otros deportes o ligas de fútbol siempre y cuando sea posible contar con datos estadísticos y de valor de mercado para poder encontrar jugadores similares y valuarlos. La principal dificultad yace en este último punto ya que la recopilación de datos es un reto constante, particularmente en deportes no profesionalizados, donde la calidad y disponibilidad de datos puede variar considerablemente comparándolo con deportes o ligas profesionales. Además, dependiendo el enfoque que se quiera dar, pensamos que podría sería razonable expandir aún más los análisis correspondientes al *clustering* específico a cada posición, donde incluso se podría volver a repetir el ejercicio sobre una posición específica, como, por ejemplo, “centrodelantero” y entender qué estilos particulares de centrodelanteros existen, lo que permitiría dar con un abanico de opciones más concretas.

Referencias

1. *Wyscout*: https://www.hudl.com/en_gb/products/wyscout

2. *Transfermarkt*: https://www.transfermarkt.com/

3. *Whoscored*: https://www.whoscored.com/

4. *Football Manager*: https://www.footballmanager.com/

5. Akhanli, S. E., & Hennig, C. (2023). Clustering of football players based on performance data and aggregated clustering validity indexes. *Journal of Quantitative Analysis in Sports*.

6. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (1st ed.). Springer.

7. Akhanli, S. E., & Hennig, C. (2020). Comparing clusterings and numbers of clusters by aggregation of calibrated clustering validity indexes. Statistics and Computing.

8. Kaufman, L. and P. J. Rousseeuw (1990). Finding groups in data: An introduction to cluster analysis. New York: Wiley.

9. Yigit, A. T., Samak, B., & Kaya, T. (2020). An XGBoost-lasso ensemble modeling approach to football player value assessment. *Journal of Intelligent & Fuzzy Systems*.

Apéndice A. Glosario de variables

Variable	Definición
Liga	País del equipo en el que participa el jugador
Posición	Rol del jugador en el campo de juego
Jugador (nombre)	Nombre y apellido del futbolista
Equipo durante el período seleccionado	Club al que pertenece el jugador en el período seleccionado
Posición específica	Función detallada del jugador en el campo de juego
Edad	Edad del jugador
Valor de mercado (Transfermarkt)	Valor de mercado del jugador estimado por Transfermarkt
País de nacimiento	Nacionalidad del jugador
Pie	Pie preferido del jugador (izquierdo o derecho)
Altura	Estatura del jugador en centímetros
Peso	Peso del jugador en kilogramos
Partidos jugados	Número de partidos en los que el jugador ha participado
Minutos jugados	Total de minutos jugados por el jugador
Acciones defensivas realizadas/90	Número de acciones defensivas efectuadas por partido
Duelos defensivos/90	Número de enfrentamientos defensivos por partido
Duelos defensivos ganados %	Porcentaje de duelos ganados
Duelos aéreos en los 90	Número de duelos aéreos por partido
Duelos aéreos ganados %	Porcentaje de duelos aéreos ganados
Posesión conquistada después de una entrada	Número de veces que el jugador recupera la posesión tras una entrada
Posesión conquistada después de una interceptación	Número de veces que el jugador recupera la posesión tras una intercepción
Goles, excepto los penaltis/90	Número de goles no provenientes de penaltis por partido
xG/90	Goles esperados por partido
Remates/90	Número de remates realizados por partido
Tiros a la portería %	Porcentaje de remates que van a la portería
Acciones de ataque exitosas/90	Número de acciones ofensivas exitosas por partido
Regates/90	Número de regates intentados por partido
Regates realizados %	Porcentaje de regates exitosos
Duelos atacantes/90	Número de enfrentamientos ofensivos por partido

Duelos atacantes ganados %	Porcentaje de duelos ofensivos ganados
Carreras en progresión/90	Número de carreras hacia adelante con el balón por partido
Aceleraciones/90	Número de aceleraciones realizadas por partido
Desmarques/90	Número de desmarques realizados por partido
Jugadas claves/90	Número de jugadas clave creadas por partido
Toques en el área de penalti/90	Número de toques dentro del área rival por partido
Pases recibidos/90	Número de pases recibidos por partido
Pases largos recibidos/90	Número de pases largos recibidos por partido
Pases hacia adelante/90	Número de pases hacia adelante realizados por partido
Precisión pases hacia adelante %	Porcentaje de pases hacia adelante exitosos
Pases cortos / medios /90	Número de pases cortos / medios realizados por partido
Precisión pases cortos / medios %	Porcentaje de pases cortos / medios exitosos
Pases largos/90	Número de pases largos realizados por partido
Precisión pases largos %	Porcentaje de pases largos exitosos
xA/90	Asistencias esperadas por partido
Asistencias/90	Número de asistencias realizadas por partido
Pases en el último tercio/90	Número de pases realizados en el tercio final del campo por partido
Precisión pases en el último tercio %	Porcentaje de pases en el último tercio exitosos
Pases al área de penalti/90	Número de pases realizados al área rival por partido
Pases hacía el área pequeña %	Porcentaje de pases realizados al área rival exitosos
Pases en profundidad/90	Número de pases en profundidad realizados por partido
Precisión pases en profundidad %	Porcentaje de pases en profundidad exitosos
Pases progresivos/90	Número de pases progresivos realizados por partido

Apéndice B. *Features correlations*

	Acciones defensivas realizadas/90	Duelos defensivos/90	Duelos defensivos ganados, %	Duelos aéreos en los 90	Duelos aéreos ganados, %	Posesión conquistada después de una entrada	Posesión conquistada después de una intercepción	Acciones de ataque exitosas/90	Goles, excepto los penaltis/90	xG/90	Remates/90	Tiros a la portería, %	Regates/90	Regates realizados, %	Duelos atacantes/90	Duelos atacantes ganados, %	Toques en el área de penalti/90	Carreras en progresión/90	Aceleraciones/90	Pases recibidos/90	Pases largos recibidos/90	Pases hacia adelante/90	Precisión pases hacia adelante, %	Pases cortos / medios/90	Precisión pases cortos / medios, %	Pases largos/90	Precisión pases largos, %	xA/90	Asistencias/90	Desmarques/90	Jugadas claves/90	Pases en el último tercio/90	Precisión pases en el último tercio, %	Pases al área de penalti/90	Pases hacia el área pequeña, %	Pases en profundidad/90	Precisión pases en profundidad, %	Pases progresivos/90
Acciones defensivas realizadas/90	1.000																																				
Duelos defensivos/90	0.831	1.000																																			
Duelos defensivos ganados, %	0.558	0.248	1.000																																		
Duelos aéreos en los 90	-0.232	-0.328	-0.026	1.000																																	
Duelos aéreos ganados, %	0.515	0.238	0.469	0.095	1.000																																
Posesión conquistada después de una entrada	0.430	0.411	0.029	-0.156	0.060	1.000																															
Posesión conquistada después de una intercepción	0.887	0.561	0.535	-0.137	0.586	0.262	1.000																														
Acciones de ataque exitosas/90	-0.496	-0.228	-0.376	-0.153	-0.527	-0.129	-0.552	1.000																													
Goles, excepto los penaltis/90	-0.574	-0.471	-0.351	0.217	-0.356	-0.205	-0.527	0.400	1.000																												
xG/90	-0.703	-0.594	-0.422	0.377	-0.395	-0.249	-0.639	0.425	0.772	1.000																											
Remates/90	-0.701	-0.494	-0.506	0.167	-0.505	-0.195	-0.683	0.671	0.679	0.809	1.000																										
Tiros a la portería, %	-0.279	-0.264	-0.094	0.154	-0.133	-0.133	-0.242	0.245	0.378	0.345	0.236	1.000																									
Regates/90	-0.402	-0.130	-0.335	-0.165	-0.498	-0.095	-0.478	0.932	0.300	0.309	0.546	0.148	1.000																								
Regates realizados, %	0.191	0.116	0.193	-0.103	0.219	-0.015	0.214	-0.050	-0.183	-0.215	-0.206	-0.067	-0.120	1.000																							
Duelos atacantes/90	-0.590	-0.280	-0.488	0.062	-0.590	-0.130	-0.660	0.842	0.448	0.528	0.681	0.213	0.867	-0.204	1.000																						
Duelos atacantes ganados, %	0.435	0.288	0.330	-0.276	0.361	0.117	0.459	-0.156	-0.358	-0.444	-0.385	-0.132	-0.170	0.532	-0.379	1.000																					
Toques en el área de penalti/90	-0.734	-0.578	-0.447	0.425	-0.447	-0.258	-0.680	0.545	0.702	0.854	0.782	0.315	0.462	-0.228	0.663	-0.473	1.000																				
Carreras en progresión/90	-0.235	-0.036	-0.171	-0.281	-0.311	-0.099	-0.273	0.730	0.134	0.102	0.325	0.095	0.747	-0.017	0.587	0.004	0.234	1.000																			
Aceleraciones/90	-0.274	-0.061	-0.217	-0.207	-0.358	-0.120	-0.333	0.729	0.155	0.148	0.351	0.111	0.760	-0.056	0.611	-0.055	0.273	0.807	1.000																		
Pases recibidos /90	0.325	0.337	0.159	-0.490	0.161	0.233	0.349	-0.051	-0.324	-0.427	-0.245	-0.172	-0.089	0.172	-0.233	0.332	-0.461	0.222	0.037	1.000																	
Pases largos recibidos/90	-0.686	-0.522	-0.414	0.357	-0.455	-0.236	-0.653	0.571	0.554	0.670	0.667	0.284	0.484	-0.214	0.640	-0.442	0.797	0.315	0.335	-0.313	1.000																
Pases hacia adelante/90	0.690	0.479	0.495	-0.328	0.507	0.223	0.731	-0.417	-0.555	-0.666	-0.611	-0.236	-0.406	0.264	-0.610	0.509	-0.744	-0.057	-0.220	0.709	-0.629	1.000															
Precisión pases hacia adelante, %	0.456	0.360	0.346	-0.425	0.301	0.179	0.461	-0.296	-0.422	-0.521	-0.478	-0.188	-0.258	0.203	-0.405	0.384	-0.569	0.000	-0.108	0.594	-0.527	0.591	1.000														
Pases cortos / medios /90	0.470	0.443	0.241	-0.438	0.261	0.270	0.490	-0.187	-0.413	-0.515	-0.384	-0.210	-0.189	0.195	-0.322	0.376	-0.557	0.124	-0.050	0.958	-0.443	0.766	0.654	1.000													
Precisión pases cortos / medios, %	0.528	0.390	0.395	-0.452	0.389	0.180	0.554	-0.425	-0.482	-0.594	-0.564	-0.216	-0.379	0.223	-0.533	0.427	-0.670	-0.084	-0.194	0.644	-0.641	0.659	0.822	0.697	1.000												
Pases largos/90	0.603	0.365	0.458	-0.250	0.481	0.154	0.647	-0.421	-0.486	-0.584	-0.536	-0.219	-0.423	0.215	-0.605	0.445	-0.674	-0.143	-0.254	0.494	-0.564	0.863	0.366	0.503	0.512	1.000											
Precisión pases largos, %	0.105	0.046	0.080	-0.172	0.094	0.068	0.133	-0.075	-0.060	-0.095	-0.086	-0.041	-0.104	0.048	-0.144	0.130	-0.135	-0.015	-0.045	0.240	-0.129	0.185	0.339	0.248	0.220	0.168	1.000										
xA/90	-0.446	-0.248	-0.327	-0.152	-0.426	-0.071	-0.480	0.679	0.305	0.324	0.514	0.175	0.557	-0.120	0.555	-0.181	0.407	0.499	0.484	0.080	0.486	-0.267	-0.234	-0.056	-0.333	-0.247	-0.019	1.000									
Asistencias/90	-0.468	-0.227	-0.387	-0.197	-0.481	-0.064	-0.518	0.683	0.299	0.315	0.543	0.157	0.568	-0.135	0.597	-0.205	0.391	0.495	0.470	0.141	0.479	-0.272	-0.186	0.007	-0.290	-0.275	0.021	0.898	1.000								
Desmarques/90	-0.362	-0.151	-0.331	-0.238	-0.410	-0.015	-0.414	0.602	0.208	0.222	0.459	0.089	0.505	-0.090	0.513	-0.120	0.242	0.449	0.398	0.238	0.307	-0.115	-0.102	0.124	-0.205	-0.176	0.015	0.605	0.668	1.000							
Jugadas claves/90	-0.484	-0.285	-0.346	-0.081	-0.438	-0.109	-0.500	0.669	0.377	0.405	0.546	0.191	0.564	-0.163	0.587	-0.249	0.518	0.489	0.477	0.009	0.561	-0.355	-0.252	-0.123	-0.366	-0.342	-0.041	0.793	0.773	0.569	1.000						
Pases en el último tercio/90	0.445	0.437	0.161	-0.438	0.213	0.270	0.412	-0.116	-0.408	-0.502	-0.293	-0.212	-0.147	0.143	-0.286	0.344	-0.585	0.113	-0.035	0.764	-0.442	0.774	0.422	0.755	0.461	0.713	0.191	0.040	0.112	0.224	-0.069	1.000					
Precisión pases en el último tercio, %	0.076	0.183	-0.041	-0.370	-0.074	0.139	0.034	0.083	-0.115	-0.156	-0.056	-0.057	0.090	0.037	0.032	0.088	-0.163	0.209	0.141	0.429	-0.139	0.166	0.576	0.435	0.402	-0.014	0.367	0.088	0.169	0.170	0.080	0.297	1.000				
Pases al área de penalti/90	-0.297	-0.073	-0.291	-0.315	-0.418	0.016	-0.356	0.733	0.141	0.129	0.422	0.064	0.614	-0.071	0.531	-0.075	0.219	0.591	0.548	0.274	0.388	-0.055	-0.157	0.117	-0.231	-0.046	0.001	0.723	0.754	0.696	0.656	0.303	0.136	1.000			
Pases hacía el área pequeña, %	-0.154	-0.058	-0.180	-0.080	-0.179	0.025	-0.178	0.239	0.132	0.152	0.208	0.056	0.169	-0.059	0.206	-0.084	0.169	0.175	0.154	0.068	0.157	-0.109	0.039	0.034	-0.060	-0.125	0.171	0.251	0.288	0.219	0.286	0.030	0.220	0.215	1.000		
Pases en profundidad/90	-0.099	0.013	-0.155	-0.329	-0.183	0.083	-0.139	0.362	-0.012	-0.036	0.218	-0.031	0.276	-0.032	0.221	0.055	-0.076	0.339	0.244	0.436	0.055	0.241	0.018	0.324	0.025	0.269	0.045	0.489	0.541	0.768	0.403	0.560	0.115	0.682	0.123	1.000	
Precisión pases en profundidad, %	0.034	0.024	0.038	-0.073	-0.017	0.065	0.033	0.050	-0.012	-0.032	-0.019	-0.038	0.026	-0.031	-0.001	0.015	-0.041	0.060	0.047	0.106	-0.007	0.062	0.129	0.104	0.050	0.042	0.180	0.084	0.099	0.101	0.115	0.112	0.175	0.096	0.222	0.123	1.000
Pases progresivos/90	0.637	0.462	0.431	-0.377	0.428	0.199	0.659	-0.296	-0.524	-0.638	-0.531	-0.229	-0.310	0.242	-0.533	0.490	-0.705	0.021	-0.131	0.679	-0.557	0.945	0.524	0.701	0.573	0.893	0.212	-0.155	-0.150	-0.050	-0.248	0.821	0.199	0.092	-0.047	0.322	0.096	1.000

Apéndice C. Promedio *features* por posición

Promedio features por posición	Defensor	Mediocampista	Delantero
Acciones defensivas realizadas/90	9.69	8.81	4.08
Duelos defensivos/90	6.02	7.00	3.67
Duelos defensivos ganados, %	69.70%	60.68%	55.68%
Duelos aéreos en los 90	4.57	2.78	5.62
Duelos aéreos ganados, %	56.65%	44.07%	33.30%
Posesión conquistada después de una entrada	0.62	0.87	0.42
Posesión conquistada después de una interceptación	7.31	5.76	2.59
Acciones de ataque exitosas/90	0.60	1.72	2.81
Goles, excepto los penaltis/90	0.03	0.06	0.24
xG/90	0.05	0.07	0.31
Remates/90	0.42	0.99	2.09
Tiros a la portería, %	30.61%	27.06%	40.15%
Regates/90	0.55	1.89	3.00
Regates realizados, %	62.38%	53.05%	46.42%
Duelos atacantes/90	1.70	5.62	9.99
Duelos atacantes ganados, %	47.68%	42.03%	31.84%
Toques en el área de penalti/90	0.55	0.82	3.09
Carreras en progresión/90	0.61	0.99	1.18
Aceleraciones/90	0.14	0.38	0.50
Pases recibidos /90	20.89	25.25	14.53
Pases largos recibidos/90	0.44	0.65	2.09
Pases hacia adelante/90	16.03	12.99	5.17
Precisión pases hacia adelante, %	72.06%	70.60%	59.62%
Pases cortos / medios /90	28.53	33.11	18.62
Precisión pases cortos / medios, %	88.32%	86.23%	77.45%
Pases largos/90	6.16	4.30	1.37
Precisión pases largos, %	55.00%	56.18%	53.07%
xA/90	0.01	0.06	0.10
Asistencias/90	0.15	0.70	0.91
Desmarques/90	0.05	0.34	0.42
Jugadas claves/90	0.06	0.22	0.39
Pases en el último tercio/90	4.83	6.47	2.50
Precisión pases en el último tercio, %	62.65%	70.67%	63.53%
Pases al área de penalti/90	0.67	1.68	1.80
Pases hacía el área pequeña, %	34.55%	44.35%	45.28%
Pases en profundidad/90	0.37	0.71	0.54
Precisión pases en profundidad, %	29.89%	33.67%	32.18%
Pases progresivos/90	7.73	6.58	2.67

Apéndice D. Promedio *features* por liga

Promedio features por liga	Argentina	Chile	Colombia	Paraguay	Uruguay
Acciones defensivas realizadas/90	8.45	7.97	7.22	7.44	7.72
Duelos defensivos/90	6.20	5.86	5.35	5.39	5.81
Duelos defensivos ganados, %	62.64%	63.53%	61.57%	62.10%	61.97%
Duelos aéreos en los 90	4.38	3.65	3.20	5.73	4.10
Duelos aéreos ganados, %	45.98%	45.65%	44.81%	44.55%	45.56%
Posesión conquistada después de una entrada	0.87	0.60	0.59	0.53	0.56
Posesión conquistada después de una interceptación	5.75	5.27	5.05	5.48	5.43
Acciones de ataque exitosas/90	1.64	1.85	1.42	1.68	1.70
Goles, excepto los penaltis/90	0.09	0.12	0.09	0.11	0.10
xG/90	0.12	0.14	0.12	0.14	0.12
Remates/90	1.04	1.16	1.12	1.15	1.06
Tiros a la portería, %	30.83%	34.32%	29.90%	32.34%	33.40%
Regates/90	1.84	1.89	1.41	1.72	1.91
Regates realizados, %	51.83%	55.09%	56.60%	55.06%	54.68%
Duelos atacantes/90	5.80	5.96	4.70	5.38	5.64
Duelos atacantes ganados, %	40.00%	40.71%	42.51%	40.58%	42.46%
Toques en el área de penalti/90	1.29	1.50	1.18	1.59	1.29
Carreras en progresión/90	0.93	1.06	0.82	0.85	0.91
Aceleraciones/90	0.34	0.36	0.26	0.38	0.33
Pases recibidos /90	20.53	22.04	22.53	19.56	19.13
Pases largos recibidos/90	0.94	1.07	0.83	1.16	0.95
Pases hacia adelante/90	11.86	12.15	11.66	11.86	12.05
Precisión pases hacia adelante, %	66.87%	68.79%	72.02%	66.41%	65.65%
Pases cortos / medios /90	27.82	28.66	28.51	26.62	25.91
Precisión pases cortos / medios, %	83.71%	84.89%	87.46%	82.70%	83.27%
Pases largos/90	4.03	4.33	3.66	4.39	4.56
Precisión pases largos, %	55.46%	55.69%	54.56%	56.58%	52.09%
xA/90	0.05	0.07	0.05	0.06	0.05
Asistencias/90	0.54	0.64	0.57	0.63	0.52
Desmarques/90	0.24	0.30	0.28	0.29	0.21
Jugadas claves/90	0.20	0.26	0.19	0.25	0.18
Pases en el último tercio/90	4.69	4.96	4.76	4.97	5.00
Precisión pases en el último tercio, %	65.53%	66.89%	68.56%	65.16%	63.20%
Pases al área de penalti/90	1.33	1.54	1.21	1.53	1.37
Pases hacía el área pequeña, %	41.07%	43.20%	39.60%	44.63%	39.09%
Pases en profundidad/90	0.50	0.64	0.55	0.63	0.48
Precisión pases en profundidad, %	33.06%	31.02%	30.09%	32.57%	33.25%
Pases progresivos/90	5.86	6.14	5.59	6.02	6.12

Apéndice E. Correlación valor de mercado

Correlación entre valor de mercado y features continuos	Valor de mercado (Transfermarkt)
Acciones defensivas realizadas/90	-0.039
Duelos defensivos/90	0.000
Duelos defensivos ganados, %	-0.034
Duelos aéreos en los 90	-0.058
Duelos aéreos ganados, %	-0.060
Posesión conquistada después de una entrada	0.078
Posesión conquistada después de una interceptación	-0.014
Acciones de ataque exitosas/90	0.149
Goles, excepto los penaltis/90	0.157
xG/90	0.110
Remates/90	0.163
Tiros a la portería, %	0.018
Regates/90	0.147
Regates realizados, %	-0.055
Duelos atacantes/90	0.136
Duelos atacantes ganados, %	-0.009
Toques en el área de penalti/90	0.102
Carreras en progresión/90	0.158
Aceleraciones/90	0.130
Pases recibidos /90	0.159
Pases largos recibidos/90	0.067
Pases hacia adelante/90	0.000
Precisión pases hacia adelante, %	0.027
Pases cortos / medios /90	0.140
Precisión pases cortos / medios, %	0.027
Pases largos/90	-0.071
Precisión pases largos, %	0.049
xA/90	0.107
Asistencias/90	0.102
Desmarques/90	0.092
Jugadas claves/90	0.126
Pases en el último tercio/90	0.044
Precisión pases en el último tercio, %	0.127
Pases al área de penalti/90	0.102
Pases hacía el área pequeña, %	0.085
Pases en profundidad/90	0.073
Precisión pases en profundidad, %	0.025
Pases progresivos/90	-0.007