Universidad de Valladolid

Escuela de Ingeniería Informática

Trabajo Fin de Grado

Grado en Ingeniería Informática
(Mención en Computación)

APRENDIZAJE AUTOMÁTICO DE MODELOS PARA LA PREDICCIÓN DE RESULTADOS DE PARTIDOS DE LA NBA

Autor:
D. Álvaro Berrío Galindo

Tutores:
Dr. José Belarmino Pulido Junquera
Dr. Pedro César Álvarez Esteban

Índice

Agradecimientos

La realización de este proyecto ha supuesto un gran esfuerzo, hay mucha gente detrás de la que no me puedo olvidar y que han aportado mucho tiempo y trabajo para que sea lo más completo posible.

En primer lugar, quiero agradecer a Belarmino Pulido su disposición para ayudarme, proponer nuevos enfoques y corregirme cuando lo ha creído oportuno. En la misma línea, debo dar las gracias a Pedro César Álvarez por sus correcciones y apoyo durante el trabajo, así a como a Hristo por sus ideas sobre cómo llevar a cabo el análisis.

Además de éstos, también quiero agradecer a Hector Saénz y Javier Martínez por el trabajo complementario realizado, sin el que no podría haber llevado a cabo mi proyecto. Por último, quiero dar las gracias a mis compañeros y amigos de la carrera por el apoyo mutuo y los consejos que nos aportamos frecuentemente.

Resumen

Este trabajo trata sobre la predicción de los resultados de partidos propios de la NBA. Se estudia la forma en la que se obtienen los datos adecuados a partir de las fuentes de datos utilizadas. Una vez se tienen unos datos con un formato con el que trabajar, se plantean técnicas de aprendizaje automático (Regresión Lineal Múltiple, Regresión Logística, SVM, Perceptrón Multicapa, Random Forest, K-Vecinos, Naive Bayes) para llevar a cabo el objetivo, las cuales se van refinando según los resultados y los ajustes y modificaciones de los datos. En el apartado de los modelos de predicción, se realizan tres experimentos diferentes teniendo en cuenta distintos períodos de tiempo y formas en las que se obtienen los datos.

Además de ésto, se implementa una herramienta mediante la que un usuario puede acceder a ciertos modelos ya guardados e interactuar con ella planteando ciertos partidos concretos para los que predecir el resultado. Esta herramienta aporta una idea del funcionamiento de cada modelo almacenado.

Abstract

This project is about the prediction of the result of NBA games. It is studied the way that the data is obtained from the different data sources. Once the data is correctly formatted, there are several machine learning techniques explained (Multiple Linear Regression, Logistic Regression, SVM, Multilayer Perceptron, Random Forest, K-Nearest Neighbours, Naive Bayes). These are improved during the work in order to get a better performance. Moreover, the data is updated to include new variables or delete others with less importance. There can be seen three different experiments in the section of the predictive models.

In addition to this, a tool is implemented for a user to interact with it and discover the performance of certain predictive models previously stored. A user can input a game to get the predictions out of it and get an idea of how the models work.

1 Introducción

El mundo de la tecnología avanza a pasos agigantados, y más aún en las últimas décadas. Esto ha permitido adentrarse en campos en los que, hasta el momento, todo se realizaba de una manera manual o más rudimentaria. Si se profundiza aún más, el ámbito de la computación es uno de los campos en los que se han conseguido progresos determinantes para la evolución de las sociedades actuales. Gracias a este paso hacia adelante en las tecnologías usadas en computación, se han desarrollado nuevas técnicas de análisis de datos y se han llevado a cabo tareas imposibles de realizar con las herramientas de las que se disponía.

El deporte ha sufrido un cambio drástico en los últimos años debido a la introducción de estas técnicas avanzadas que tienen como objetivo hacer hincapié en la información que se puede obtener de un partido, una carrera o cualquier competición, y obtener así datos que se desconocían o patrones escondidos. Esto ha hecho que los diferentes equipos en cada disciplina se especialicen mucho más consiguiendo que las diferencias entre ellos sean menores. Es por esta razón que cada vez más equipos tienen un departamento dedicado solamente al análisis de datos y realizan inversiones mayores en este aspecto. Sin ir más lejos, el equipo de fútbol de la ciudad de Valladolid cuenta con uno de éstos.

El ejemplo más popular de que incluir un estudio de las estadísticas puede ser muy beneficioso es el caso de Billy Beane, gerente del equipo de béisbol Oakland Athletics. Este hombre introdujo técnicas adelantadas a la época a la hora de contratar nuevos jugadores, realizar alineaciones o las tácticas que debían llevar a cabo en cada fase de un partido. Hizo de un equipo humilde un equipo ganador capaz de conseguir 20 victorias consecutivas, récord absoluto en el campeonato estadounidense de béisbol, la MLB. A pesar de la notable mejora del equipo, perdieron en la primera ronda de la post-temporada pero marcaron una nueva forma de entender el deporte del béisbol. Este caso fue plasmado en un libro llamado **Moneyball**, llevado posteriormente al cine con un título homónimo.

Así como en el béisbol, se ha adoptado esta forma de trabajar tanto en el baloncesto, como el fútbol, deportes de motor o casi cualquier otra disciplina. En el caso de deportes en equipo como el primero mencionado, se pueden conseguir modelos capaces de predecir lesiones [10] o determinar la influencia que tiene cada jugador en el desarrollo de un partido o una temporada [9]. La mayoría de los equipos cuenta ya con instrumentos que pueden usar los jugadores durante entrenamientos y partidos capaces de proporcionar información acerca de la distancia recorrida o las pulsaciones, datos que serán usados luego por los analistas. Los diferentes análisis pueden ayudar también a los directores deportivos a la hora de tomar decisiones en la planificación de la plantilla, contratando jugadores que puedan aportar lo que el equipo necesite. También los entrenadores pueden adaptar el juego del equipo a las características de los integrantes de la plantilla. En deportes de motor, las estadísticas pueden ayudar a un equipo a saber cúal es el mejor momento para realizar una parada en boxes minimizando el tiempo que se pierde, por ejemplo.

Pero el análisis de datos en el deporte no se limita al ámbito de los propias entidades interesadas. Muchos intentan sacar beneficio económico. Aquí es donde entran las casas de apuestas, las cuales consiguen obtener una suma importante de dinero realizando modelos de predicción de resultados deportivos y estableciendo unas cuotas adecuadas para cada partido, carrera o la competición correspondiente. En la página oficial de *betfair* [3] se pueden encontrar algunas explicaciones acerca de los modelos predictivos que se usan en esa casa de apuestas, los cuales han sido creados por expertos.

Lo que se va a tratar en este trabajo es el análisis de datos aplicado al baloncesto, concretamente a la liga de baloncesto norteamericana, la **NBA** (National Basketball Association). La idea es intentar predecir los resultados de los encuentros que se vayan a disputar en las próximas fechas así como descubrir patrones en los datos que puedan aportar nueva información sobre los distintos equipos.

1.1 Funcionamiento NBA

La NBA es la competición deportiva que más dinero recauda y una de las más vistas en todo el mundo. En España, país en el que el *deporte rey* es el fútbol, es la tercera competición más seguida, después del fútbol y el tenis, según un estudio realizado por la empresa Nielsen Sports [15].

Actualmente, la NBA cuenta con la participación de 30 franquicias distribuidas por el terreno norteamericano. Las franquicias equivalen a los equipos, pero reciben este nombre por el modelo ecónomico que adoptan. A menudo se dan ventas de franquicias que conllevan que éstas se trasladen a otra ciudad adoptando otro nombre. Por ejemplo, los *Golden State Warriors* nacieron en Philadelphia con el nombre de *Philadelphia Warriors*. Tras varios años, se trasladó la sede a San Francisco cambiando el nombre a *San Francisco Warriors*, posteriormente se adoptó el nombre actual. Este hecho debe ser tenido en cuenta a la hora de hallar los enfrentamientos directos entre franquicias.

La competición consta de dos conferencias principales, según la zona geográfica: la conferencia este y la conferencia oeste. Ambas conferencias contienen a su vez tres agrupaciones diferentes que incluyen cinco franquicias cada una [8]. Estas agrupaciones son conocidas como **divisiones**, se puede ver su distribución en la Figura 1.

- **Conferencia oeste**:
    1. Noroeste: Portland Trail Blazers, Utah Jazz, Denver Nuggets, Oklahoma City Thunder y Minnesota Timberwolves.
    2. Pacífica: Sacramento Kings, Golden State Warriors, Los Angeles Lakers, Los Angeles Clippers y Phoenix Suns.
    3. Suroeste: Memphis Grizzlies, Dallas Mavericks, San Antonio Spurs, Houston Rockets y New Orleans Pelicans.
- **Conferencia este**:

1. Atlántica: Boston Celtics, New York Knicks, Brooklyn Nets, Philadelphia 76ers y Toronto Raptors (Canadá).
2. Central: Milwaukee Bucks, Detroit Pistons, Chicago Bulls, Cleveland Cavaliers e Indiana Pacers.
3. Sureste: Washington Wizards, Charlotte Hornets, Atlanta Hawks, Orlando Magic y Miami Heat.

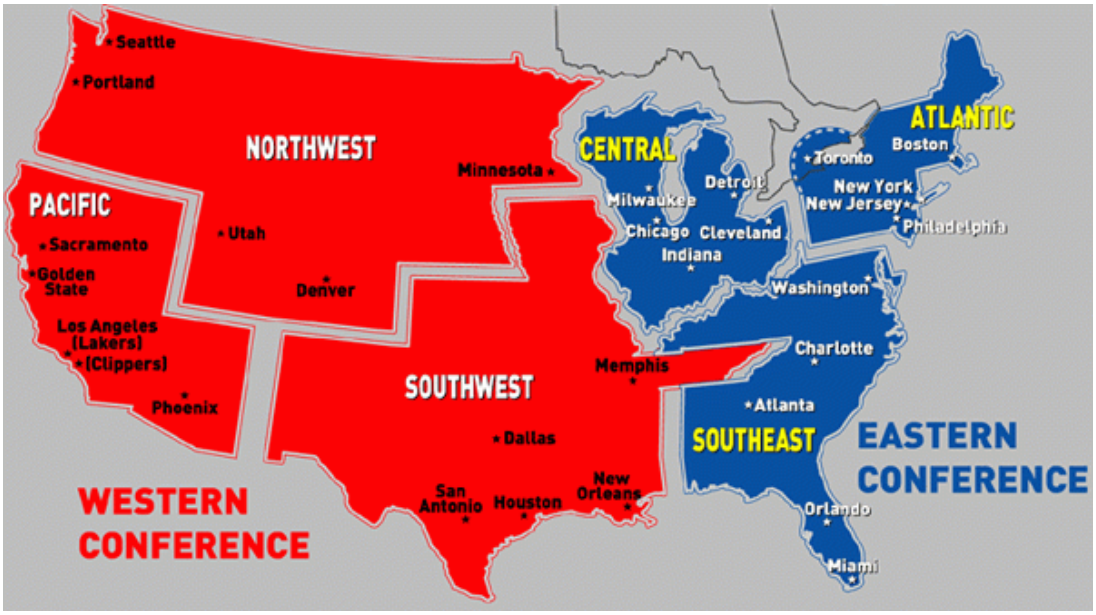

Figura 1: Conferencias y divisiones NBA

Fuente: nbaexplained.wordpress.com

Los partidos tienen una duración de 48 minutos distribuidos en cuatro tiempos, los cuartos, de 12 minutos cada uno. Si al final del tiempo reglamentario, el resultado es un empate hay prórrogas sucesivas de 5 minutos hasta deshacerlo. Cada equipo tiene 5 jugadores en la cancha. Las plantillas, salvo ocasiones excepcionales, tienen un límite de 15 jugadores, de los cuales solo 12 pueden participar en el partido.

Cada una de las franquicias disputa un total de 82 partidos en la temporada regular. Juegan 4 encuentros contra las franquicias de su misma división, otros 3 ó 4 contra las del resto de divisiones de la misma conferencia y 2 partidos contra cada franquicia de la otra conferencia. Pero no todas las franquicias de una misma división tiene un mismo calendario, lo cual puede ser un aspecto determinante a la hora de realizar las predicciones.

En Febrero finaliza el mercado de fichajes, por lo que los equipos solo podrán contratar a jugadores sin equipo, los Agentes Libres. Este es un factor que se puede tener en cuenta ya que, al poderse realizar fichajes tanto tiempo después de haber comenzado la liga, algunos jugadores que eran determinantes en un equipo se pueden marchar a otro.

Pasada esta fecha sigue la temporada regular hasta Abril. Se disputan entonces los **playoffs** [7]. Consiste en un pequeño torneo que disputan los 8 mejores equipos de cada conferencia. A la hora de elegir qué equipos han sido los mejores de cada conferencia, se los escoge según el porcentaje de victorias obtenido en la temporada regular, asegurando la participación en el playoff de los campeones de cada división. En la primera ronda, los cuartos de final, juegan el 1º contra el 8º, 2º contra 7º, 3º contra 6º y 4º contra 5º de manera que la final de cada conferencia se dispute entre los mejores equipos de ella. Se puede ver la distribución de los *playoffs* en la Figura 2. Cada eliminatoria se realizará al mejor de 7 enfrentamientos, de manera que el que consiga 4 victorias pasará de ronda. Los enfrentamientos siguen el esquema 2-2-1-1-1, es decir, los 2 primeros encuentros se disputan en la cancha del equipo que tiene la **ventaja de local** y los 2 siguientes en la cancha del otro equipo. Si fuesen necesarios más partidos, se disputa uno en cada una de las dos localizaciones hasta que uno de las dos franquicias consiga ganar 4 enfrentamientos. La ventaja de local se otorga según la actuación de cada equipo durante la temporada, primando la posición en la clasificación. Si hay empate se miran criterios como los enfrentamientos entre los dos equipos o si uno de ellos fue el campeón de su división. Los equipos que ganen la final de cada conferencia se enfrentarán, al mejor de 7 partidos y con los mismos criterios para decidir qué franquicia tiene la ventaja de local, en la final de la NBA de manera que el vencedor de esta eliminatoria será el campeón de la NBA de esa temporada.

Figura 2: Playoffs de la NBA

Fuente: safestbettingsites.com

1.2 Objetivos

Este trabajo se encuadra dentro de un proyecto común que tiene como objetivo general llevar a cabo un análisis de datos relativos a la competición de la NBA. Se pueden diferenciar tres trabajos diferentes.

El primero de ellos tiene que ver con la extracción y almacén de los datos. Para ello es necesaria la colaboración de todos los integrantes del proyecto para saber qué datos son los necesarios para llevar a cabo las tareas posteriores. Se han usado los servicios de **Cloudera** con **Hive** para llevar a cabo el almacén de los datos en un *data lake*. Para extraerlos de aquí y poder manipularlos basta con realizar una consulta SQL. Esta tarea ha sido realizada por **Héctor Saénz Niño**, estudiante de Ingeniería Informática.

El siguiente proyecto tiene que ver con los índices y métricas que se usan en esta competición para comparar jugadores y equipos y descubrir cuáles de ellos son los más precisos. Además se pretende crear nuevos índices relacionados con esta temática que tengan un mayor poder de predicción. El encargado del proyecto es **Javier Martínez**, estudiante del Grado de Estadística.

Por último, se encuentra el proyecto presente, cuyo objetivo principal es la predicción de resultados de partidos de la NBA. Para ello, los trabajos anteriormente citados serán cruciales dado que es necesario realizar un correcto tratamiento de los datos y muchas de las métricas aportadas pueden servir como predictores. Además, se pretende crear una herramienta en la que un usuario pueda seleccionar un modelo de entre una serie de ellos para conocer el funcionamiento de este y poder hallar la predicción de algunos partidos. Se intentará que este proyecto sirva para que futuros trabajos tengan un punto de inicio y puedan mejorar los resultados aquí obtenidos.

2 Planificación inicial del proyecto

Éste es un proyecto de investigación a pesar de que también se creará una herramienta software, aunque como objetivo secundario. Como proyecto de investigación, una tarea importante antes de comenzar a realizar un proyecto de este tipo consiste en la planificación del mismo. Una buena planificación supone un mejor aprovechamiento del tiempo y los recursos, y es una manera de estimar el tiempo que va a llevar el trabajo. Normalmente no se cumplen las estimaciones realizadas durante la planificación del proyecto de forma exacta, pero puede servir para hacerse una idea y organizar el tiempo de la forma más eficiente posible.

Una forma recurrente de realizar la estimación del tiempo dedicado a cada tarea es mediante un diagrama de Gantt. Ésta es una herramienta que permite conocer el tiempo empleado en cada tarea y subtarea y las relaciones y dependencias entre cada una. Estos diagramas se realizan antes de comenzar el propio proyecto. El diagrama de Gantt de este proyecto se puede observar en la Figura 3.

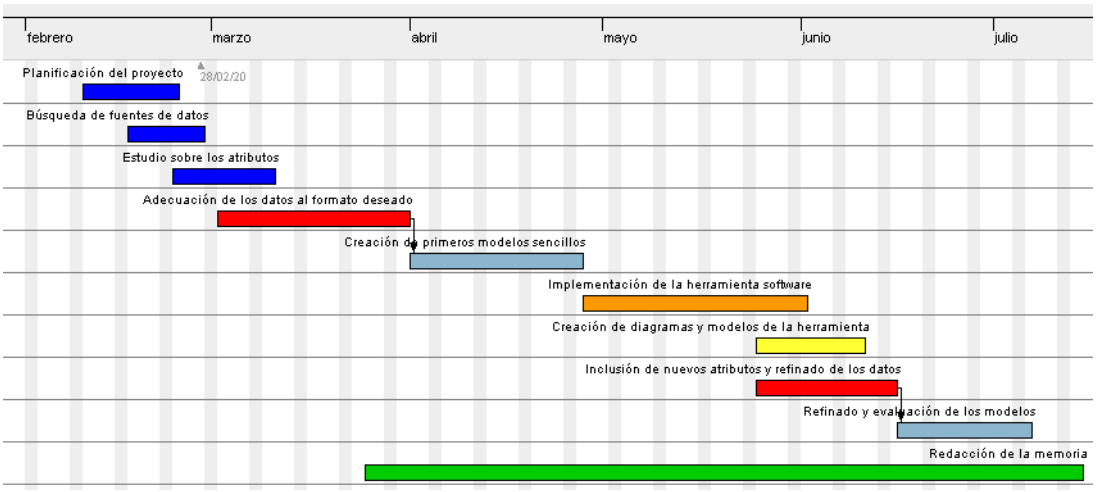

Figura 3: Diagrama de Gantt

Se han clasificado las tareas por colores según el tipo de cada una de ellas, para hacer una división más clara.

- Las azules se refieren a tareas propias del análisis del proyecto, para descubrir qué y cómo se quiere hacer y a partir de qué datos. Por orden cronológico estas tareas serán:

  1. *Planificación del proyecto*
  2. *Búsqueda de fuentes de datos*
  3. *Estudio de los atributos*

- Una vez se tengan las fuentes de datos, será necesario darles el formato deseado y escoger los atributos que se quieran, estas tareas tienen un color rojo. Las tareas serán:

1. *Adecuación de los datos al formato deseado*
2. *Inclusión de nuevos atributos y refinado de los datos*

- Cuando se posean los datos en el formato adecuado para realizar los modelos, se implementarán éstos y se estudiará cada uno de ellos. Estas tareas aparecen en color gris y serán:

  1. *Creación de primeros modelos*
  2. *Refinado y evaluación de los modelos*

- La tarea que aparece en el diagrama con color naranja se refiere a la *Implementación de la herramienta software*, que será usada por un usuario.

- Relacionada con la tarea anterior, será necesario realizar todos los diagramas que expliquen el funcionamiento de la herramienta, esta tarea será *Creación de diagramas y modelos de la herramienta.*

- Por último, todo lo realizado durante el proyecto habrá que plasmarlo en la memoria del mismo realizando las explicaciones pertinentes, *Redacción de la memoria.*

.

Este es el esquema que se pretende seguir durante el desarrollo del proyecto, clasificando las tareas a realizar para la consecución de los objetivos impuestos al principio. Para la realización del Trabajo de Fin de Grado se otorgan 300 horas, las cuales habrá que distribuir de la mejor forma posible entre todas las tareas explicadas.

3 Análisis - CRISP-DM

3.1 Introducción

En este apartado se pretende explicar la manera en que se ha trabajado para intentar alcanzar los objetivos. A la hora de realizar el proyecto se ha seguido el modelo CRISP-DM [17] (Cross Industry Standard Process for Data Mining). Este modelo proporciona una forma estándar de llevar a cabo las tareas que conciernen a un proyecto de minería de datos. Surgió como forma de realizar proyectos de minería de datos independientes del sector en el que se trabaja y del software utilizado. Tiene como objetivos conseguir trabajos de una manera menos costosa, más rápida, mecánica y con resultados de mayor confianza. Es un proceso útil para la planificación, documentación y comunicación de un proyecto.

En un proceso CRISP se encuentran 4 niveles fundamentales organizados de una manera jerárquica desde un carácter general a lo más específico.

- **Fases**: describen las partes principales en las que se divide un proyecto de estas características. Se componen de *tareas generales*.
- **Tareas generales**: este nivel pretende tener un carácter general de manera que se cubran todas las posibles situaciones que se puedan dar en un proyecto de minería de datos. Estas tareas deben ser completas, no depender de otras, y estables, de manera que se puedan adaptar a futuros desarrollos.
- **Tareas especializadas**: en este nivel se describe cómo se deben realizar las acciones que componen el nivel de las tareas generales en situaciones determinadas. Son tareas propias del proyecto en cuestión, no tienen un carácter general.
- **Instancia de proceso**: es el nivel más bajo, en este se mantiene un registro de las acciones y resultados obtenidos en el proyecto. Se organizan en función de las tareas de los niveles superiores

El desarrollo ideal de las fases y tareas propias del modelo CRISP implica que éstas se lleven a cabo siguiendo un orden, como una secuencia de eventos. Pero en la realidad el orden en el que se realizan las tareas varía e, incluso, muchas veces es necesario volver sobre las tareas ya realizadas para implementar ciertas modificaciones. Se observa la jerarquía de los diferentes niveles en la Figura 4.

En cuanto al ciclo de vida de un proyecto de minería de datos, se han establecido 6 fases diferenciadas, las cuales se pueden observar en la Figura 5. Las líneas entre fases indican que existe una dependencia directa entre ambas. Generalmente, se sigue este orden pero no es una regla estricta. El círculo exterior indica el ciclo del proyecto de manera que se dará por finalizado cuando se consiga el despliegue de la herramienta, software, aplicación, etcétera.

- **Comprensión del proyecto**: en esta fase se pretende definir los objetivos del proyecto así como los requisitos que deberá cumplir una vez acabado y las dificultades a las que se harán frente durante el desarrollo.

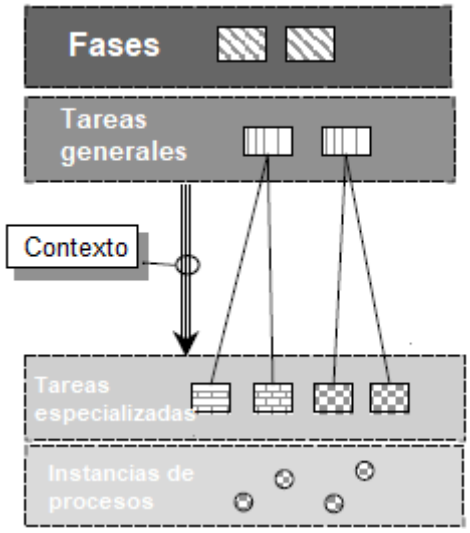

Figura 4: Niveles de la metodología CRISP-DM

Fuente: *Adaptación de CRISP-DM: Towards a standard process model for data mining* [17]

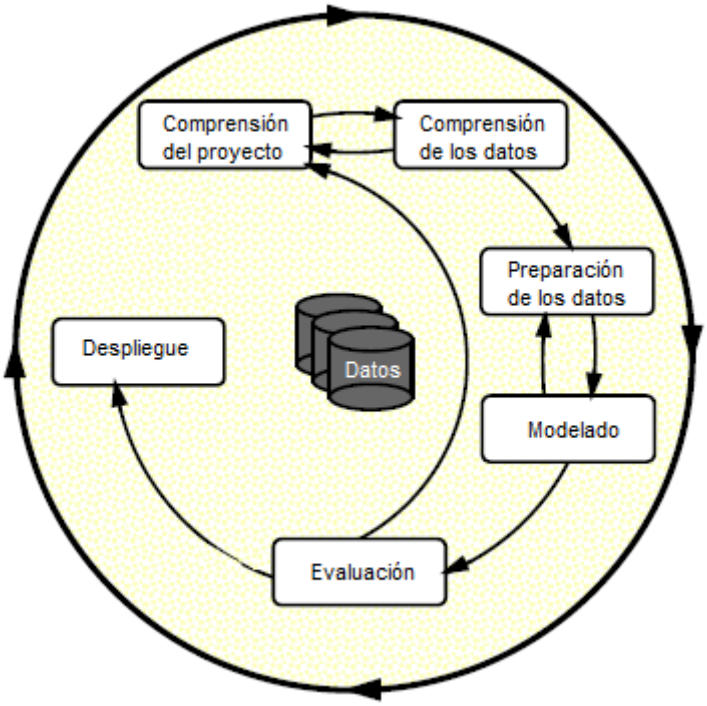

Figura 5: Fases de la metodología CRISP-DM

Fuente: *Adaptación de CRISP-DM: Towards a standard process model for data mining* [17]

- **Comprensión de los datos**: esta fase consiste en la búsqueda de unas primeras ideas a partir de los datos para ir especificando qué datos concretos se quieren obtener y, en general, familiarizarse con los datos.

- **Preparación de los datos**: aquí se pretende obtener el conjunto de datos con el que se va a trabajar en las tareas posteriores. Es una fase a la que se vuelve en varias ocasiones con el objetivo de conseguir el mejor conjunto de datos posible. Se llevan a cabo tareas de limpieza, detección de outliers, selección de atributos o transformación de los datos, entre otros.

- **Modelado**: en esta fase se seleccionan los algoritmos y técnicas de aprendizaje que se quieren usar y se lleva a cabo un proceso mediante el cual se consigan los parámetros más cercanos al óptimo posible. Como se ve en la Figura 5, existe una gran dependencia entre la preparación de los datos y el modelado ya que a la hora de encontrar el mejor modelo se llevan a cabo muchas modificaciones de los datos.

- **Evaluación**: se pretende hacer mayor hincapié en aquel o aquellos modelos que han resultado más positivos. Se buscan posibles errores revisando los pasos realizados para poder pasar a la fase definitiva. Además, se intenta refinar el modelo para obtener mejores resultados aún.

- **Despliegue**: esta fase consiste principalmente en la presentación de los resultados. Normalmente se basa simplemente en un informe explicando gráficas o soluciones a los problemas planteados en el proyecto. También puede referirse al desarrollo de un software o una herramienta, como en este caso.

3.2 Comprensión del proyecto

Como ya se comentó en la Sección 1.2, el objetivo principal es llevar a cabo un análisis de datos referidos a partidos de la NBA para intentar comparar diferentes modelos y predecir los resultados de encuentros que estén por disputar y de algunos que ya se han disputado, y así poder comparar las predicciones con el resultado real.

Más concretamente, se busca la creación de una herramienta mediante la cual un usuario pueda elegir el modelo de aprendizaje de entre una serie de ellos para calcular cómo de bien se comporta con los datos utilizados. Además, se podrá introducir el identificador de dos equipos rivales y una fecha para el encuentro de manera que se devuelve la probabilidad de victoria de cada equipo.

La idea es hacer una pequeña herramienta simple, rápida y fácil de usar de la que se puedan nutrir trabajos de años próximos. Así, se tendrá una base de la que partir y se podrán alcanzar otros objetivos ahora más lejanos. La prioridad del proyecto será conseguir los mejores modelos posibles con las limitaciones que se poseen, y poder plasmarlo en esta herramienta.

3.3 Comprensión de los datos

Dado que se realizará un análisis sobre datos referidos a la NBA, es necesario hacer una búsqueda de las posibles fuentes de datos que existen actualmente en Internet. La página principal de la NBA (stats.nba.com) contiene un apartado dedicado exclusivamente a las estadísticas producidas en los encuentros así como en temporadas pasadas, ya sea según jugadores o según equipos. Es seguramente la página web más completa pero la manera de extraer los datos no es la más cómoda.

Por esta razón, se exploran otras fuentes de datos. La siguiente página que ofrece una cantidad muy grande datos es basketball-reference.com. Aporta información de muchas temporadas, datos según los jugadores y los equipos y otros muchos datos que pueden ser de utilidad. Las estadísticas son, en general, iguales a las de la página de la NBA por lo que es muy probable que se nutra de ahí. Debido a su mayor facilidad de extracción, será ésta la página web que se usará principalmente como fuente. Existe otra fuente de datos, que es espn.com/nba, la cual sirve como complemento a la información que se vaya a utilizar y para comprobar que ésta sea correcta.

La manera en que se extraen los datos y se almacenan está detallada en el trabajo de fin de grado de Héctor Saénz. Ha creado una herramienta para extraer los datos y poder almacenarlos en una base de datos implementada en Cloudera mediante Hive, un *data lake*. De ahí es de dónde se obtienen los datos en este proyecto mediante el uso de consultas SQL.

3.4 Preparación de los datos

El conjunto de datos principal del que se va a partir es el *box score* de cada partido. Estos son tablas que reflejan las variables principales que se dan en un partido y que se obtendrán a partir de los archivos almacenados en el *data lake* implementado en el proyecto complementario a éste. Incluye las características de cada uno de los jugadores de los dos equipos así como la información total del equipo. Se puede ver cómo es un *box score* en la Figura 6. La información presente en un *box score* está explicada en el Anexo 7.1. Todo esto para cada jugador y la suma total para el equipo. Además, se incluye un índice conocido como +/- (más/menos) para cada jugador, que indica la diferencia de puntos que se logró con el jugador en cuestión en el campo. La gran mayoría de esta información no es directamente significativa a la hora de predecir un partido, lo que realmente importa es la evolución a lo largo de la temporada y la construcción de características más avanzadas y con mayor poder de predicción a partir de ellas.

3.4.1 Extracción y transformación de los datos

A partir de los *box score* tradicionales almacenados en el *data lake* se construyen una serie de estadísticas más avanzadas, las cuales ayudan de una manera más determinante a la hora de predecir futuros resultados deportivos. Para la construcción de las métricas que se van a explicar en este apartado se ha tenido en cuenta el análisis lle-

New York Knicks (21-45)

	Basic Box Score Stats																		
Starters	**MP**	**FG**	**FGA**	**FG%**	**3P**	**3PA**	**3P%**	**FT**	**FTA**	**FT%**	**ORB**	**DRB**	**TRB**	**AST**	**STL**	**BLK**	**TOV**	**PF**	**PTS**	**+/-**
Maurice Harkless	40:53	6	13	.462	0	4	.000	2	3	.667	1	5	6	6	1	0	1	3	14	+10
RJ Barrett	39:23	9	14	.643	2	5	.400	6	6	1.000	0	5	5	4	2	1	2	1	26	+3
Julius Randle	38:23	13	22	.591	3	6	.500	4	4	1.000	0	11	11	3	1	1	3	5	33	+8
Elfrid Payton	34:37	4	8	.500	0	0		3	4	.750	0	5	5	12	4	1	5	1	11	+11
Taj Gibson	19:36	4	6	.667	1	2	.500	0	0		0	3	3	1	2	3	0	2	9	+7
Reserves	**MP**	**FG**	**FGA**	**FG%**	**3P**	**3PA**	**3P%**	**FT**	**FTA**	**FT%**	**ORB**	**DRB**	**TRB**	**AST**	**STL**	**BLK**	**TOV**	**PF**	**PTS**	**+/-**
Mitchell Robinson	26:10	7	7	1.000	0	0		2	7	.286	4	2	6	0	1	2	0	3	16	+5
Frank Ntilikina	23:16	1	7	.143	1	3	.333	5	6	.833	1	3	4	2	1	0	0	4	8	-10
Bobby Portis	16:58	3	7	.429	1	2	.500	0	0		0	2	2	2	0	1	1	1	7	-6
Kevin Knox	16:21	3	7	.429	3	3	1.000	3	4	.750	0	3	3	0	2	0	0	2	12	-4
Wayne Ellington	9:23	0	3	.000	0	3	.000	0	0		0	2	2	2	0	0	0	1	0	+1
Dennis Smith	Did Not Play																			
Reggie Bullock	Did Not Play																			
Damyean Dotson	Did Not Play																			
Team Totals	**265**	**50**	**94**	**.532**	**11**	**28**	**.393**	**25**	**34**	**.735**	**6**	**41**	**47**	**32**	**14**	**9**	**12**	**23**	**136**	

Figura 6: Box Score tradicional

Fuente: basketball-reference.com

vado a cabo por Javier Martínez en su TFG.

En primer lugar, se crean los denominados *four factors* para cada equipo. Estos fueron ideados por *Dean Oliver*, un exjugador y actualmente entrenador de baloncesto. Creó otras muchas métricas que se pueden estudiar en su libro *Basketball On Paper* [12], algunas se verán más adelante. La gran mayoría de éstas están explicadas en el glosario de basketball-reference.com [1], donde se incluye la forma de calcularlas. Los cuatro factores son un conjunto de estadísticas que ayudan a resumir el comportamiento de ambos equipos durante un partido. Estos cuatro factores tienen que ver con los *tiros*, *pérdidas*, *rebotes* y *tiros libres*. Dean Oliver estableció que los pesos para cada uno de estos factores debían ser del 40 %, 25 %, 20 % y 15 %, respectivamente.

1. **Tiros**: en este apartado se halla el factor de *effective Field Goal Percentage*, el cual es una estadística que ajusta el porcentaje de tiros de campo teniendo en cuenta que los triples suman más puntos.

$$eFG\,\% = \frac{FG + 0{,}5 \times 3P}{FGA}$$

2. **Pérdidas**: respecto a las pérdidas de balón, se creó el factor *Turnover Percentage*. Es una estimación del número de pérdidas por cada 100 jugadas dentro del partido.

$$TOV\,\% = \frac{100 \times TOV}{FGA + 0{,}44 \times FTA + TOV}$$

3. **Rebotes**: en este caso se plantea el porcentaje de rebotes ofensivos, *Offensive Rebound Percentage*, que determina la relación entre los rebotes ofensivos del equipo en cuestión y los rebotes defensivos del oponente.

$$ORB\,\% = \frac{ORB}{ORB + OPP.DRB}$$

También se puede considerar el porcentaje de rebotes defensivos, *Defensive Rebound Percentage*, el cual determina la relación contraria.

$$DRB\,\% = \frac{DRB}{DRB + OPP.ORB}$$

4. **Tiros libres**: por último, se considera el factor relacionado con los tiros libres, *Free Throw Factor*. Es una medida aproximada de la importancia de los tiros libres para un equipo durante el transcurso de un partido.

$$FT\_FACTOR = \frac{FT}{FGA}$$

Si además de hallar el porcentaje de rebotes ofensivos se halla también el porcentaje de rebotes defensivos se tendrían 5 factores. Asimismo, se hallarán estos 5 factores tanto para el equipo local como para el visitante. Para un partido concreto, el ratio de rebotes ofensivos de un equipo y el ratio de rebotes defensivos del equipo contrario suman 1. Se hallarán ambos para cada equipo ya que lo importante es la evolución durante la temporada.

Además de estos factores, en la página de *basketball-reference*, también se considera importante el rating ofensivo ($ORtg$) [1]. Este estadístico sirve para expresar la actuación ofensiva del equipo, estimando el número de puntos conseguidos por cada 100 posesiones. También se ha tenido en cuenta el rating defensivo ($DRtg$), el cual se halla utilizando los puntos concedidos en vez de los conseguidos por lo que, en un partido concreto, el rating ofensivo de un equipo es igual al defensivo del equipo contrario, pero los ratings ofensivos y defensivos de una temporada completa no son iguales. Estos índices fueron también ideados por Dean Oliver.

$$ORtg = 100 \times \frac{PTS}{POSS}$$

Para calcular las estadísticas anteriores es necesario conocer el número de posesiones ($POSS$) de las que dispuso el equipo en un partido. La página que se utiliza como fuente, *basketball-reference*, no aporta esta información, por lo que es necesario estimarla. Para ello se utiliza la siguiente fórmula, la cual tiene en cuenta diferentes factores como los tiros realizados, los rebotes cogidos o los balones perdidos. Además, se estudian todas estas características tanto para el equipo local como para el visitante, promediando el de los dos para así obtener el número de posesiones en el partido. Por lo tanto, la estimación del número de posesiones es igual para ambos equipos en un

partido. Al tratarse de una estimación es difícil proporcionar el número de posesiones exacto. El número de posesiones es un estadístico importante ya que, como se ha visto anteriormente, es necesario para calcular los ratings ofensivos y defensivos. También sirve para calcular otras estadísticas y para arrojar una idea del ritmo del partido. Esta información se ha obtenido a través de nbastuffer.com [16].

$$\begin{aligned} POSS =& 0{,}5 \times ((FGA + 0{,}4 \times FTA - 1{,}07 \times \frac{ORB}{ORB + OPP.DRB} \times (FGA - FG) \\ &+ TOV) + (OPP.FGA + 0{,}4 \times OPP.FTA - 1{,}07 \times \\ &\frac{OPP.ORB}{OPP.ORB + DRB} \times (OPP.FGA - OPP.FG) + OPP.TOV)) \end{aligned}$$

Lo negativo de la métrica anterior es que no tiene en cuenta la duración del partido y puede dar una falsa idea del desarrollo del partido. Algunos partidos duran más por las prórrogas pero esto no se tiene en cuenta. Por eso, se calcula el *PACE*, el cual pondera según el número de minutos jugados en un partido. De nuevo, se ha utilizado la página web nbastuffer.com [16] como fuente de esta información.

$$PACE = 48 * \frac{POSS + OPP.POSS}{2 * \frac{MP}{5}}$$

Para cada enfrentamiento, el *porcentaje de partidos ganados* (W-L %) [1] previos durante la temporada también se considera importante, así como el porcentaje de victorias locales que tiene el equipo que jugará como local, además del porcentaje de victorias como visitante que posee el equipo visitante. Así se tiene una idea general de las actuaciones de un equipo durante una temporada y se puede conocer de una manera más concreta si el hecho de jugar en casa o no tiene algún efecto en el equipo.

Otro estadístico muy simple pero con mucha importancia es la diferencia de puntos (*Points Differential*) por partido de cada equipo. Aporta una idea acerca de la diferencia entre el balance ofensivo y defensivo de un equipo. Además de la diferencia de puntos, también se guardará la cantidad de puntos conseguida por el equipo contrario en cada partido para conocer el potencial defensivo del equipo en cuestión.

$$PointsDifferential = PTS - OPP.PTS$$

A partir de los puntos conseguidos por cada equipo durante la temporada y de los puntos concedidos se pueden establecer algunas métricas que intentan predecir el resultado de los partidos. Una de ellas es el Pythagorean Winning Percentage, donde PTS son los puntos conseguidos y $PTS.ALLOWED$ son los puntos concedidos. El exponente usado puede variar según la fuente, en la página nbastuffer.com usan 16.5.

$$PythagoreanWin\,\% = \frac{PTS^{16,5}}{PTS^{16,5} + PTS.ALLOWED^{16,5}}$$

La siguiente métrica incluida es *Projected Winning Percentage* (PW %). Esta sirve como estimación de la probabilidad de ganar un partido en función de la diferencia de puntos acumulada en partidos previos, la cual también se ha incluido en las tablas. Este índice se ha obtenido de la página web nbastuffer.com [16]. Los valores de este índice son muy similares a los del anterior.

$$ProjectedWin\,\% = \frac{(PointsDifferential \times 2{,}7) + 41}{82}$$

A partir de la métrica anterior se crea el índice *Expected Winning Percentage* [1], la cual obtiene una estimación de victoria del equipo A sobre el equipo B a partir del porcentaje de victoria proyectado de cada uno de ellos. En el caso realizado en este trabajo, el equipo A es el local mientras que el B es el visitante. Esta métrica también se encuentra en nbastuffer.com [16].

$$ExpectedWin(A)\,\% = \frac{PW\,\%(A) \times (1 - PW\,\%(B))}{(PW\,\%(A) \times (1 - PW\,\%(B))) + (PW\,\%(B) \times (1 - PW\,\%(A)))}$$

También se trabaja con características menos técnicas, que tienen que ver más con apartados físicos y psicológicos. Aquí se encuentran el número de días de descanso de cada equipo entre partidos (*rest*) y la racha de victorias (*streak*) en los últimos 5 partidos, es decir, de los pasados 5 encuentros, cuántas victorias cosechó el equipo. También se tendrá en cuenta qué equipo juega como local y cuál es visitante.

Otro apartado que puede ser muy significativo es el histórico de enfrentamientos de dos equipos (*head-to-head*). Por ello, para cada franquicia se registra el número de partidos disputados con el resto de franquicias y cuántos de ellos ha ganado. Esto puede ayudar a recuperar características más subjetivas del juego como las que se pueden dar en un derbi, dónde el equipo que parte con las mejores estadísticas no tiene por qué ser favorito.

3.4.2 Cálculo de las características avanzadas

Como se mencionó previamente, se parte de los archivos almacenados en el *data lake* dentro de la carpeta *Raw*, que contiene información acerca de los partidos, pero tiene el inconveniente de que, además de los *box score*, también tiene información que no es relevante para este trabajo, como las estadísticas por cuartos. En este proyecto se utiliza la información que aportan los *box score* que están presentes en esos archivos, los cuales contienen las características ya mencionadas para cada jugador y también

para el equipo completo, en cada partido, como se mostraba en la Figura 6.

La idea es hallar, para cada partido, las estadísticas acumuladas por cada equipo hasta ese momento para luego conseguir estadísticas avanzadas. En los conjuntos de datos proporcionados se tienen datos de partidos de toda la temporada, pero lo que interesa para predecir es solo la temporada regular por lo que se hace un filtro en función de la fecha en que comenzó el playoff en cada temporada. Se hace esto ya que la actuación de los equipos durante los playoffs suele ser distinta a la del resto de la temporada, perdiendo capacidad predictiva. Para llevar a cabo la adecuación de los datos se realizan ciertos pasos.

1. Los archivos con los *box scores* presentes en la base de datos tienen también información por cuartos y las estadísticas por jugadores, pero en este proyecto sólo interesan las estadísticas de los partidos completos conseguidas por los equipos al completo. Por lo tanto, la primera tarea para proporcionar el formato deseado a los datos es seleccionar la información correspondiente a los *box scores* de los totales de los equipos, ignorando la información individual de cada jugador y la de por cuartos.

   En esta fase, además de las variables presentes en el *box score* tradicional, se añaden la fecha (*date*) a partir del identificador del partido (*game_id*) y la temporada del partido (*season*) así como si el equipo consiguió la victoria (*victory*) y era local (*home*), los puntos concedidos (*pt_allowed*), la diferencia de puntos (*pt_diff*) y las posesiones (*poss*). También se añade la franquicia a la que pertenece el equipo (*franchise*) ya que algunas cambian de ciudad y los equipos se modifican. Esto servirá para calcular los enfrentamientos directos entre las franquicias, en vez de entre equipos. Además, se añaden algunas estadísticas propias del oponente en cada partido que serán de utilidad más adelante para calcular alguno de los cuatro factores.

   También se añade en esta fase del proceso dos columnas por cada franquicia actual para indicar los partidos que se van jugando y ganando contra ellos para, posteriormente, acumular esta información y poder obtener el ratio de partidos ganados contra cada franquicia. En la Tabla 1 se muestra el formato que tienen estas columnas. El equipo *MIN* ha disputado un partido contra *GSW*, como se ve en la variable *GSWg*, y lo ha ganado, según se ve en el 1 que aparece en *GSWw*.

season	**team_id**	**game_id**	**...**	**DETg**	**GSWg**	**HOUg**	**...**	**DETw**	**GSWw**	**HOUw**	**...**
2005	MIN	200504130MIN	...	0	1	0	...	0	1	0	...

Tabla 1: Enfrentamientos directos contra el resto de franquicias

   La información obtenida en el transcurso de esta fase se almacenará en el *data lake* como una tabla intermedia para agilizar el proceso a la hora de introducir nuevos partidos en la base de datos. Dado que hasta el momento se ha obtenido la información de cada partido según el equipo, esta tabla se almacena

como *team_totals*. En la Tabla 2 se muestran tres observaciones aleatorias de este conjunto de datos. Además de estas columnas, también se tiene la información referente a los enfrentamientos directos con el resto de franquicias, explicada ya en la Tabla 1.

season	**team_id**	**game_id**	**date**	**home**	**ast**	**blk**	**drb**	**fg**
2001	ATL	200010310ATL	2000-10-31	True	14	2	18	30
2003	MIN	200303040SEA	2003-03-04	False	22	3	27	34
2005	MIN	200504130MIN	2005-04-13	True	32	8	36	50

fg3	**fg3a**	**fga**	**ft**	**fta**	**orb**	**pf**	**pts**	**stl**	**tov**	**trb**	**mp**	**franchise**
6	15	81	16	21	11	31	82	9	13	29	240.0	ATL
3	9	72	12	16	9	19	83	9	14	36	240.0	MIN
6	10	88	10	13	5	18	116	4	14	41	240.0	MIN

pt_allowed	**drb_opp**	**orb_opp**	**opp**	**fr_opp**	**victory**	**pt_diff**	**poss**
106	37	7	CHH	NOP	False	-24	88.402588
92	26	7	SEA	OKC	False	-9	83.066403
100	24	13	GSW	GSW	True	16	96.014011

Tabla 2: Información por partido y equipo: *Team totals*

2. A continuación se desea obtener las estadísticas de cada equipo previas al partido en cuestión. Para ello, se acumula la información de los partidos anteriores, por lo que muchas de las variables del primer partido de un equipo en una temporada están a 0, ya que no ha habido partidos antes de éste en esa temporada. A partir de estas estadísticas acumuladas se calculan la mayor parte de las características avanzadas comentadas en el apartado anterior: *four factors* (*eFG, tovR, orbR, drbR, ftR*), *rating ofensivo* y *defensivo* (*ort* y *drt*) y *pythagoras winning percentaje* (*pytha*) y *projected winning percentage* (*proj_win*).

   En esta etapa se añaden también las variables referidas a la racha de victorias de los últimos 5 partidos (*streak*), al número de días de descanso entre partidos (*rest*) y el porcentaje de partidos ganados hasta el momento, tanto el total (*win_pct_total*), como el local (*win_pct_home*) y el visitante (*win_pct_away*).

   Además, en esta fase se rellenan las columnas correspondientes al número de partidos jugados contra cada equipo y cuantos de ellos se ganaron para calcular la variable *head_to_head*, que indica el porcentaje de partidos que el equipo local ha ganado a ese oponente en concreto. Una vez calculado el ratio de victorias en los enfrentamientos directos previos, se eliminan las variables relacionadas con el resto de franquicias mostradas en la Tabla 1 así como aquellas propias del *box score* original que ya no tienen uso (asistencias, tapones, robos, etcétera).

   Al archivo resultante de estas operaciones se le llama *Before game*, se muestra en la Tabla 3. Se aprecia que las características se van sumando, por eso, variables

como la racha de los partidos no presentan un valor hasta el sexto partido ya que contiene la racha de 5 partidos previos.

season	team_id	game_id	date	home	pt_allowed	victory	pt_diff
2007	GSW	200611010GSW	2006-11-01	True	0.000000	False	0.000000
2007	GSW	200611030GSW	2006-11-03	True	110.000000	True	-12.000000
2007	GSW	200611040UTA	2006-11-04	False	99.500000	False	0.500000
2007	GSW	200611060DAL	2006-11-06	False	101.666667	True	-7.666667
2007	GSW	200611070NOK	2006-11-07	False	102.250000	False	-5.000000

poss	rest	streak	win_pct_total	win_pct_home	win_pct_away	ort	drt
0.000000	NaN	NaN	0.000000	0.0	0.0	0.000000	0.000000
98.165755	2.0	NaN	0.000000	0.0	0.0	99.831148	112.055370
94.746204	1.0	NaN	0.500000	0.5	0.0	105.545125	105.017400
93.899760	2.0	NaN	0.333333	0.5	0.0	100.106752	108.271487
93.381284	1.0	NaN	0.500000	0.5	0.5	104.142925	109.497317

pace	eFG	ftR	tovR	orbR	drbR	pytha	proj_win	head_to_head
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.250000
98.165755	0.432927	0.329268	0.090514	0.250000	0.641509	0.129435	0.104878	0.416667
94.746204	0.477987	0.301887	0.115676	0.272727	0.663043	0.520665	0.516463	0.347826
93.899760	0.461207	0.293103	0.133325	0.259259	0.646154	0.215231	0.247561	0.208333
93.379663	0.485246	0.304918	0.140792	0.255814	0.666667	0.304229	0.335366	0.266667

Tabla 3: Información acumulada por partido y equipo: *Before game*

Una vez llegados a este punto, se almacenan en el *data lake* las características de cada equipo en el momento actual, una vez acabado el último encuentro disputado. Se deja la fecha en la que se produjo este partido para poder calcular los días de descanso más adelante. Así, se podrán crear predicciones para partidos según los equipos que participen en él. Esta tabla se almacena como *current*. Se puede ver en la Tabla 4, se tiene la información actual de cada equipo en la última temporada. Además, se incluyen las columnas con la situación histórica respecto al resto de equipos en la que se reflejan el número de partidos jugados contra cada franquicia así como las victorias cosechadas en esos encuentros, Tabla 5.

3. Hasta el momento se tiene un conjunto de datos con una observación por cada partido y equipo, es decir, dos entradas para cada partido. La siguiente fase del proceso consiste en unir la actuación de los dos equipos que se enfrentan en cada partido de manera que el conjunto de datos contenga como observaciones las características de cada partido. El resultado de esta etapa consta de prácticamente las mismas variables que el apartado anterior pero distinguiendo entre el equipo local y el visitante. Además, se incluye el porcentaje esperado de victoria (*expected*) del equipo local creado a partir de los porcentajes proyectados de victoria de cada equipo. Este conjunto de datos que se almacenará en el data lake con el nombre de *games* en referencia a las estadísticas que se almacenan de ambos equipos participantes en un partido.

Todos estos pasos se pueden resumir en la Figura 7, la cual refleja el flujo que realizan los datos desde que se extraen hasta que tiene el formato deseado para poder

season	team_id	game_id	date	home	ast	blk	drb	fg	fg3
2020	BOS	NaN	2020-03-10	NaN	1459	358	2259	2634	794
2020	GSW	NaN	2020-03-10	NaN	1663	299	2138	2510	678
2020	WAS	NaN	2020-03-10	NaN	1621	273	2024	2682	788

fg3a	fga	ft	fta	orb	pf	pts	stl	tov	trb	pt_allowed	victory	pt_diff
2190	5735	1170	1461	683	1368	7232	529	832	2942	6837	NaN	395
2032	5730	1214	1511	647	1304	6912	534	930	2785	7478	NaN	-566
2117	5821	1247	1584	644	1445	7399	519	860	2668	7658	NaN	-259

poss	rest	streak	win_pct_total	win_pct_home	win_pct_away	ort
6359.312498	NaN	2.0	0.671875	0.718750	0.625000	113.722985
6522.504728	NaN	2.0	0.230769	0.235294	0.225806	105.971560
6572.057843	NaN	2.0	0.375000	0.500000	0.250000	112.582697

drt	pace	eFG	ftR	tovR	orbR	drbR	head_to_head
98.563642	0.528509	0.204010	0.115398	0.238644	0.775223	NaN	107.511622
99.550039	0.497208	0.211867	0.126965	0.215236	0.763844	NaN	114.649208
102.191197	0.528432	0.214224	0.116563	0.220096	0.748521	NaN	116.523624

Tabla 4: Situación actual de los equipos

game_id	ATLg	BOSg	...	UTAg	WASg	ATLw	BOSw	...	UTAw	WASw
BOS	72	NaN	...	39	73	43	0	...	24	43
GSW	38	39	...	74	39	24	20	...	33	25
WAS	77	73	...	39	NaN	38	30	...	13	0

Tabla 5: Histórico de enfrentamientos entre equipos

construir los modelos de predicción.

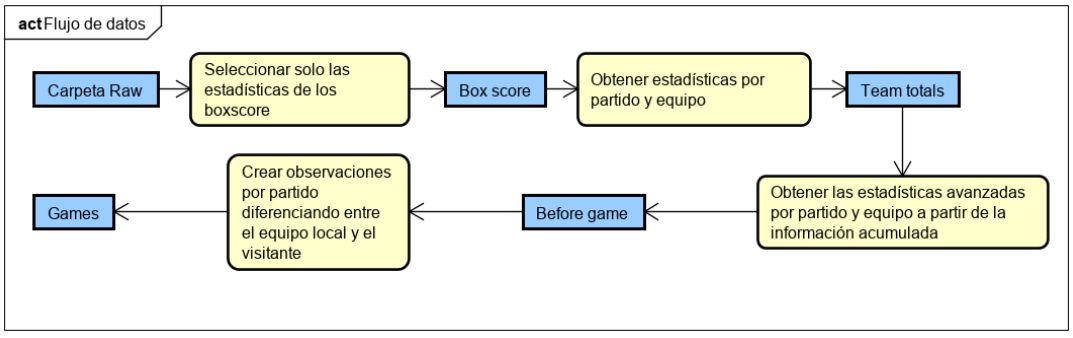

Figura 7: Flujo de datos

También es de mencionar que será necesario almacenar un conjunto de datos referente a los nombres que han tomado las franquicias a lo largo de la historia, para así poder obtener los enfrentamientos directos no sólo contra la franquicia actual, sino contra todas las formas que ha tomado esa franquicia. Se puede ver un ejemplo en la

Tabla 6, en la columna izquierda está el nombre actual de la franquicia (ATL), mientras que en la derecha se pueden encontrar todos los nombres que ha ido tomando.

franchise	team_id
ATL	ATL
ATL	STL
ATL	MLH
ATL	TRI

Tabla 6: Tabla con el histórico de nombres de las franquicias

3.4.3 Introducción de nuevos partidos

La idea de implementar el *data lake* es disponer de toda la información necesaria y realizar las actualizaciones pertinentes sin demasiada complicación. La información de la que se parte es la tabla de los archivos de *Raw*, la cual se actualizará cada vez que haya resultados de nuevos partidos. Pero también será necesario actualizar la información referida a las estadísticas avanzadas.

A partir del conjunto de datos almacenado previamente, *team_totals*, se llevará a cabo la actualización de las estadísticas avanzadas. Primero será necesario proporcionar el formato presente en el archivo de *team_totals* a los nuevos encuentros incluidos en la base de datos, esto se realiza aplicando las tareas pertenecientes al primer paso expuesto anteriormente en la Sección 3.4.2, para así tener las estadísticas cosechadas por un equipo en un partido. Una vez formateados los nuevos encuentros, se crea un conjunto de datos mediante la unión de estos y las observaciones que ya se tenían de *team_totals*.

En este punto se actualiza también la tabla intermedia *team_totals*. A partir de aquí se aplican los procedimientos explicados en el segundo y el tercer paso del apartado anterior (3.4.2). De nuevo, una vez actualizadas las características avanzadas de cada encuentro (*games*), habrá que actualizar la tabla presente en el *data lake* así como la tabla que contiene las características avanzadas de cada equipo en el momento actual de la temporada (*current*). Estos pasos están resumidos en el diagrama de la Figura 8.

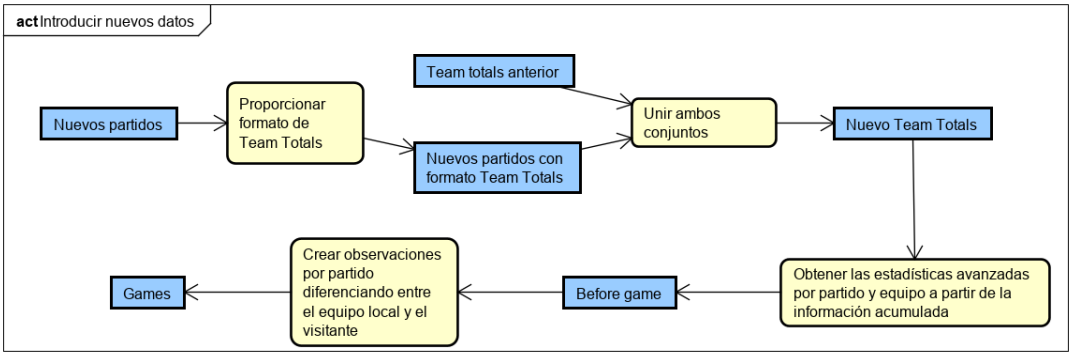

Figura 8: Pasos para la introducción de nuevos partidos

3.4.4 Creación de predicciones

Como ya se ha dicho, los modelos funcionan con conjuntos de datos con el formato de *games*. Cuando se quiera conocer la predicción que de un partido que todavía no se ha disputado se debe proporcionar el identificador de la franquicia local, el de la franquicia visitante y la fecha en la que se disputará el encuentro, por defecto se coge el día actual.

Partiendo de la tabla de *current*, la cual contiene información sobre las estadísticas actuales de todos los equipos, se crea una sola fila con ambos equipos con las columnas presentes en *games*. Se calculan los días de descanso de cada equipo desde su último partido así como el porcentaje de partidos entre ambos equipos ganados por el local (*head_to_head*) y el porcentaje de victoria esperado (*expected*). Con este formato y los modelos creados se podrán inferir predicciones para el resultado del partido.

Si el partido tiene una fecha anterior al día actual, se buscará el partido de entre el conjunto de entrenamiento, que corresponderá al conjunto de partidos de la temporada actual 2019/2020, y se devolverá la predicción calculada para ese encuentro durante el proceso del cálculo de la tasa de acierto.

3.4.5 Exploración de los datos

Una tarea fundamental en todo proceso de minería de datos consiste en explorar los atributos de los datos en busca de outliers o valores vacíos así como obtener algunas ideas principales que puedan ser de ayuda en futuras tareas referidas a la creación de modelos de predicción.

Como se vio anteriormente, existen algunos valores vacíos, `NAN`, correspondientes a la racha de victorias en los 5 últimos partidos. Los primeros 5 partidos de cada temporada no tendrán tantos precedentes, por lo que estarán sin ningún valor. Esto no es un problema ya que los primeros partidos de cada temporada no se escogerán porque se necesita tener una mínima evolución en los datos que reflejen la realidad de cada franquicia durante la temporada.

Para comprobar que todas las estadísticas estén según lo esperado se va a realizar un análisis descriptivo de cada variable según la temporada. Así se podrán encontrar posibles errores en los datos y ver una evolución de las principales características que se han tenido en cuenta. Las tablas en las que se va a representar estos datos cuentan con la temporada, el número de repeticiones, la media y desviación estándar y los cuartiles.

Se cuenta con información correspondiente a los encuentros desde la temporada 68/69, pero solo se van a utilizar las estadísticas desde el año 2000 ya que la forma de jugar varía con el tiempo y la distribución de franquicias varía. Es muy probable que en la búsqueda de los mejores modelos se reduzca incluso más el conjunto de datos en cuanto a temporadas.

La variable correspondiente al conteo de observaciones presenta un valor de 2378 en las primeras 4 temporadas ya que solo había 29 equipos a diferencia de las siguientes,

en las que participan 30. Después es 2460 en el resto de temporadas salvo la 2011/2012, en la que se declaró una huelga y se retrasó la liga, la 2012/2013, en la que se anuló un partido (hay dos observaciones menos porque se cuenta cada partido para cada equipo) y en la actual (2019/2020) ya que todavía no se ha finalizado la liga. Salvo algunas excepciones que se explicarán en su momento, éstos son los valores que se repiten en todos los atributos. En las últimas temporadas, este valor se obtiene a partir de los 82 partidos que juega cada franquicia en una temporada y, dado que son 30 las que participan en la liga, hay 2460 observaciones en el conjunto de datos. Antes el calendario era diferente y el número de partidos que se jugaban no era el mismo.

Primero se van a analizar los cuatro factores que, como ya se explicó antes, son 5 realmente. En la Tabla 7 se encuentran estadísticas descriptivas acerca del ratio de tiros efectivos (eFG). El valor correspondiente al porcentaje efectivo de tiros de campo ha ido subiendo en los últimos años ligeramente. Esto puede indicar que se ha mejorado la precisión a la hora de encestar tiros de campo o, más concretamente, ha aumentado el ratio de tiros de 2 y/o de 3. Se podrá confirmar más adelante. En cuanto a los cuantiles, no se aprecia ningún valor anormal.

Temporada	**Conteo**	**Media**	**Desv. estándar**	**Min**	**25 %**	**50 %**	**75 %**	**Max**
00/01	2378	0.474	0.064	0.278	0.432	0.473	0.516	0.699
01/02	2378	0.478	0.062	0.272	0.438	0.477	0.519	0.726
02/03	2378	0.475	0.063	0.275	0.432	0.474	0.516	0.699
03/04	2378	0.472	0.063	0.274	0.427	0.470	0.515	0.694
04/05	2460	0.483	0.064	0.263	0.440	0.482	0.526	0.715
05/06	2460	0.491	0.064	0.287	0.448	0.487	0.534	0.721
06/07	2460	0.497	0.064	0.296	0.454	0.494	0.539	0.732
07/08	2460	0.498	0.066	0.288	0.455	0.494	0.540	0.729
08/09	2460	0.501	0.063	0.308	0.456	0.500	0.544	0.747
09/10	2460	0.502	0.066	0.285	0.457	0.500	0.545	0.766
10/11	2460	0.500	0.064	0.305	0.456	0.500	0.542	0.736
11/12	1980	0.488	0.065	0.267	0.444	0.487	0.530	0.700
12/13	2458	0.498	0.066	0.305	0.453	0.494	0.542	0.712
13/14	2460	0.503	0.065	0.285	0.458	0.500	0.545	0.737
14/15	2460	0.497	0.064	0.313	0.456	0.495	0.538	0.756
15/16	2460	0.504	0.064	0.286	0.459	0.500	0.547	0.750
16/17	2460	0.515	0.065	0.312	0.470	0.512	0.559	0.759
17/18	2460	0.522	0.065	0.312	0.478	0.522	0.565	0.750
18/19	2460	0.525	0.063	0.320	0.483	0.523	0.567	0.739
19/20	1942	0.529	0.066	0.335	0.483	0.527	0.575	0.758

Tabla 7: Porcentaje efectivo de tiros de campo (eFG)

Otro de los factores es el porcentaje de pérdidas de balón que comete un equipo por cada 100 jugadas ($tovR$). Se muestran estadísticas sobre este en la Tabla 8. El conteo es correcto y, a pesar de que ha habido un ligero descenso desde las primeras temporadas estudiadas, no se aprecia ninguna diferencia significativa, se ha mantenido más o menos constante con pequeñas oscilaciones. De cada 100 jugadas, un equipo

pierde algo más de 12 balones.

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.136	0.037	0.029	0.111	0.134	0.160	0.278
01/02	2378	0.131	0.036	0.028	0.106	0.129	0.153	0.250
02/03	2378	0.135	0.037	0.030	0.109	0.132	0.158	0.296
03/04	2378	0.136	0.036	0.039	0.110	0.135	0.160	0.273
04/05	2460	0.131	0.035	0.032	0.106	0.130	0.154	0.275
05/06	2460	0.132	0.035	0.019	0.107	0.131	0.154	0.255
06/07	2460	0.137	0.036	0.039	0.112	0.136	0.161	0.255
07/08	2460	0.128	0.035	0.028	0.103	0.126	0.150	0.258
08/09	2460	0.127	0.035	0.020	0.103	0.127	0.149	0.259
09/10	2460	0.128	0.034	0.027	0.104	0.126	0.151	0.268
10/11	2460	0.129	0.034	0.029	0.105	0.128	0.151	0.270
11/12	1980	0.132	0.035	0.028	0.109	0.131	0.156	0.263
12/13	2458	0.132	0.034	0.018	0.108	0.131	0.154	0.267
13/14	2460	0.131	0.034	0.030	0.107	0.130	0.153	0.276
14/15	2460	0.128	0.036	0.030	0.102	0.125	0.152	0.254
15/16	2460	0.127	0.035	0.031	0.102	0.125	0.149	0.266
16/17	2460	0.123	0.034	0.021	0.099	0.121	0.145	0.238
17/18	2460	0.125	0.033	0.028	0.102	0.124	0.147	0.246
18/19	2460	0.120	0.033	0.021	0.097	0.118	0.141	0.248
19/20	1942	0.123	0.035	0.026	0.098	0.122	0.145	0.278

Tabla 8: Porcentaje de pérdidas por 100 jugadas ($tovR$)

Los siguientes factores que se han tratado son los referidos a los rebotes. En la Tabla 9 se muestra el ratio de rebotes ofensivos ($orbR$) por partido mientras que en la Tabla 10 se muestran los defensivos ($drbR$). Aparentemente no hay valores extraños en cuanto al conteo, media y desviación típica, se ha mantenido constante a lo largo de las temporadas. Evidentemente, la suma de los ratios ofensivos y defensivos dan 1 ya que esto compone la totalidad de los rebotes. Se aprecia un ligero descenso en el factor de rebotes ofensivos y, consecuentemente, un aumento en el factor referido a los rebotes defensivos. Destaca que el valor mínimo de rebotes ofensivos es 0 en las temporadas 2001/2002, 2017/2018 y 2019/2020. Puede parecer un error pero tanto San Antonio Spurs contra Utah Jazz el 23 de enero de 2002, como Phoenix Suns el 5 de diciembre de 2017 contra los Toronto Raptors, como Oklahoma City Thunder el 22 de enero de 2020 contra Orlando Magic no consiguieron ningún rebote en ataque. Lógicamente, aparece un 1 como máximo ratio de rebotes defensivos en esas mismas temporadas correspondientes a los partidos mencionados.

El último factor de Dean Oliver es el referente a los tiros libres frente a los tiros de campo (ftR). Se sitúa en torno a 0.2 en las últimas temporadas, lo que indica que por cada 100 tiros de campo que realiza un equipo, tira unos 20 tiros libres. Sin embargo, en las primeras temporadas estudiadas este valor era un tanto superior, ha ido disminuyendo paulatinamente, es decir, los tiros libres gozan de menor importancia que antes en el desarrollo de los partidos. Esto puede deberse a que, como se verá a

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.280	0.076	0.026	0.229	0.278	0.333	0.561
01/02	2378	0.286	0.079	0.000	0.233	0.286	0.341	0.543
02/03	2378	0.282	0.078	0.054	0.231	0.281	0.333	0.542
03/04	2378	0.284	0.080	0.033	0.229	0.283	0.333	0.524
04/05	2460	0.285	0.076	0.061	0.233	0.283	0.333	0.567
05/06	2460	0.271	0.082	0.031	0.213	0.268	0.326	0.636
06/07	2460	0.269	0.079	0.048	0.212	0.267	0.318	0.523
07/08	2460	0.264	0.076	0.049	0.211	0.262	0.314	0.588
08/09	2460	0.266	0.077	0.044	0.213	0.263	0.317	0.545
09/10	2460	0.260	0.076	0.031	0.209	0.259	0.311	0.556
10/11	2460	0.261	0.078	0.047	0.206	0.261	0.314	0.512
11/12	1980	0.267	0.078	0.022	0.213	0.265	0.325	0.489
12/13	2458	0.263	0.078	0.028	0.209	0.260	0.314	0.556
13/14	2460	0.253	0.075	0.024	0.200	0.250	0.302	0.541
14/15	2460	0.249	0.074	0.026	0.200	0.250	0.298	0.507
15/16	2460	0.236	0.075	0.030	0.182	0.233	0.288	0.509
16/17	2460	0.231	0.074	0.020	0.179	0.229	0.280	0.525
17/18	2460	0.221	0.072	0.000	0.171	0.218	0.268	0.469
18/19	2460	0.227	0.071	0.029	0.178	0.222	0.275	0.471
19/20	1942	0.224	0.070	0.000	0.175	0.222	0.271	0.463

Tabla 9: Porcentaje de rebotes ofensivos ($orbR$)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.720	0.076	0.439	0.667	0.722	0.771	0.974
01/02	2378	0.714	0.079	0.457	0.659	0.714	0.767	1.000
02/03	2378	0.718	0.078	0.458	0.667	0.719	0.769	0.946
03/04	2378	0.716	0.080	0.476	0.667	0.717	0.771	0.967
04/05	2460	0.715	0.076	0.433	0.667	0.717	0.767	0.939
05/06	2460	0.729	0.082	0.364	0.674	0.732	0.787	0.969
06/07	2460	0.731	0.079	0.477	0.682	0.733	0.788	0.952
07/08	2460	0.736	0.076	0.412	0.686	0.738	0.789	0.951
08/09	2460	0.734	0.077	0.455	0.683	0.737	0.787	0.956
09/10	2460	0.740	0.076	0.444	0.689	0.741	0.791	0.969
10/11	2460	0.739	0.078	0.488	0.686	0.739	0.794	0.953
11/12	1980	0.733	0.078	0.511	0.675	0.735	0.787	0.978
12/13	2458	0.737	0.078	0.444	0.686	0.740	0.791	0.972
13/14	2460	0.747	0.075	0.459	0.698	0.750	0.800	0.976
14/15	2460	0.751	0.074	0.493	0.702	0.750	0.800	0.974
15/16	2460	0.764	0.075	0.491	0.712	0.767	0.818	0.970
16/17	2460	0.769	0.074	0.475	0.720	0.771	0.821	0.980
17/18	2460	0.779	0.072	0.531	0.732	0.782	0.829	1.000
18/19	2460	0.773	0.071	0.529	0.725	0.778	0.822	0.971
19/20	1942	0.776	0.070	0.537	0.729	0.778	0.825	1.000

Tabla 10: Porcentaje de rebotes defensivos ($drbR$)

continuación, los tiros de 3 han aumentado, reduciendo el número de faltas y, por lo tanto, de tiros libres. No se observa ningún valor anómalo en la tabla.

Temporada	**Conteo**	**Media**	**Desv. estándar**	**Min**	**25 %**	**50 %**	**75 %**	**Max**
00/01	2378	0.235	0.088	0.034	0.171	0.225	0.286	0.706
01/02	2378	0.224	0.086	0.024	0.165	0.213	0.275	0.655
02/03	2378	0.233	0.087	0.026	0.171	0.222	0.287	0.635
03/04	2378	0.232	0.088	0.012	0.170	0.219	0.282	0.702
04/05	2460	0.250	0.091	0.022	0.185	0.241	0.308	0.688
05/06	2460	0.253	0.095	0.031	0.186	0.244	0.308	0.692
06/07	2460	0.250	0.092	0.022	0.187	0.241	0.304	0.667
07/08	2460	0.235	0.085	0.032	0.175	0.227	0.288	0.592
08/09	2460	0.239	0.087	0.038	0.176	0.231	0.291	0.721
09/10	2460	0.231	0.082	0.035	0.174	0.224	0.279	0.623
10/11	2460	0.232	0.085	0.033	0.172	0.225	0.282	0.672
11/12	1980	0.211	0.080	0.012	0.152	0.205	0.260	0.551
12/13	2458	0.207	0.078	0.024	0.150	0.200	0.256	0.587
13/14	2460	0.218	0.082	0.011	0.160	0.210	0.266	0.650
14/15	2460	0.208	0.079	0.010	0.150	0.200	0.253	0.581
15/16	2460	0.212	0.079	0.011	0.156	0.204	0.258	0.642
16/17	2460	0.212	0.078	0.025	0.156	0.204	0.260	0.500
17/18	2460	0.196	0.074	0.011	0.143	0.188	0.241	0.528
18/19	2460	0.201	0.075	0.022	0.148	0.195	0.247	0.638
19/20	1942	0.201	0.076	0.030	0.146	0.193	0.247	0.547

Tabla 11: Factor de tiros libres (ftR)

Las siguientes variables que se analizan son las referidas a los ratings de los equipos en cada partido. El rating ofensivo ($ORtg$) a lo largo de la temporada es igual al defensivo ($DRtg$) ya que indica puntos por posesiones y los puntos que consigue un equipo los recibe el otro, además de que en un partido el número de posesiones estimado es igual para los dos equipos, por lo que solo se muestra el primero de ellos. Se puede ver en la Tabla 12. Se observa que ha ido aumentando ligeramente en los últimos años pero no se encuentran valores extraños. El hecho de que aumente quiere decir que los equipos anotan más puntos en menos posesiones. Si a esto se le suma el incremento de las posesiones en un partido, lo cual se observa en la Tabla 13, se intuye que a lo largo de las últimas temporadas ha aumentado el número de puntos anotados en cada partido. De nuevo, no se encuentra ningún indicio de que la tabla tenga valores incorrectos, no hay ningún error en el conteo de las observaciones o valores extremos.

La variable *pace* es una estimación de las posesiones cada 48 minutos, que es el tiempo reglamentario de un partido. Se confirma lo que se intuía antes, que el ritmo de los partidos se ha incrementado considerablemente en los últimos años aumentando en casi 10 posesiones por cada 48 minutos desde la temporada 2000/2001.

Las siguiente tabla se refiere a la diferencia de puntos (Tabla 15). En esta se ve que la media es 0 ya que los puntos que ha conseguido un equipo los ha concedido el

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	103.810	11.325	66.967	96.181	103.902	111.840	137.807
01/02	2378	105.334	11.466	68.282	97.399	105.567	113.290	147.129
02/03	2378	104.383	11.570	64.412	96.801	104.500	112.257	141.030
03/04	2378	103.749	11.422	67.299	96.161	103.838	111.406	148.780
04/05	2460	106.876	11.389	68.065	99.237	107.119	114.507	140.642
05/06	2460	107.011	11.369	68.564	99.274	106.946	115.026	144.999
06/07	2460	107.195	11.352	66.848	99.568	107.460	115.111	148.937
07/08	2460	108.172	11.858	64.379	100.208	107.971	116.100	154.533
08/09	2460	108.994	11.413	69.375	101.384	109.242	116.619	154.407
09/10	2460	108.391	11.372	69.115	100.390	108.245	116.023	152.683
10/11	2460	108.066	11.183	64.099	100.504	108.112	115.623	147.554
11/12	1980	105.383	11.581	62.418	97.484	105.591	112.990	141.705
12/13	2458	106.605	11.453	66.052	99.028	106.346	114.207	145.369
13/14	2460	107.382	10.985	69.263	99.730	107.508	114.662	146.917
14/15	2460	106.446	11.380	70.311	98.713	106.554	113.797	155.625
15/16	2460	107.136	11.209	68.211	99.357	106.796	114.536	144.339
16/17	2460	109.528	11.423	71.813	101.916	109.442	117.322	147.233
17/18	2460	109.324	11.213	75.206	101.863	109.554	116.861	147.969
18/19	2460	111.070	11.300	68.517	103.731	111.027	118.540	152.434
19/20	1942	111.220	11.307	76.913	103.805	111.305	118.434	154.016

Tabla 12: Rating ofensivo (*ort*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	91.369	5.553	75.515	87.457	91.176	94.690	120.761
01/02	2378	90.675	5.312	76.127	87.038	89.986	94.065	118.530
02/03	2378	91.088	5.447	76.382	87.398	90.736	94.373	113.911
03/04	2378	89.998	5.708	73.036	86.190	89.840	93.465	113.328
04/05	2460	90.965	5.805	75.369	86.930	90.632	94.464	113.190
05/06	2460	90.664	5.922	74.213	86.563	90.182	94.236	128.536
06/07	2460	92.119	6.216	77.376	87.863	91.463	95.802	125.021
07/08	2460	92.370	6.354	77.213	87.969	91.711	96.423	127.318
08/09	2460	91.702	6.256	76.142	87.416	91.165	95.688	119.331
09/10	2460	92.698	5.520	77.259	88.928	92.253	96.128	118.673
10/11	2460	92.124	5.752	76.149	88.044	91.730	95.732	117.674
11/12	1980	91.378	5.323	77.619	87.974	90.755	94.367	121.286
12/13	2458	92.103	5.527	76.063	88.413	91.762	95.068	123.410
13/14	2460	94.097	5.989	77.217	89.917	93.580	97.275	119.249
14/15	2460	94.010	5.560	78.310	90.252	93.596	97.293	119.999
15/16	2460	95.891	5.565	81.099	92.047	95.663	99.274	130.645
16/17	2460	96.455	5.683	81.982	92.565	95.986	99.975	129.759
17/18	2460	97.293	5.334	83.426	93.655	96.897	100.313	121.313
18/19	2460	100.159	5.772	85.288	96.475	99.738	103.404	142.547
19/20	1942	100.258	5.637	86.758	96.380	99.766	103.762	123.610

Tabla 13: Posesiones (*poss*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	90.620	4.926	75.515	87.073	90.607	93.954	109.613
01/02	2378	90.057	4.730	76.127	86.900	89.601	93.196	105.809
02/03	2378	90.353	4.868	76.382	87.046	90.222	93.359	106.933
03/04	2378	89.370	5.152	73.030	85.836	89.431	92.843	104.401
04/05	2460	90.272	5.273	75.369	86.661	90.128	93.752	106.696
05/06	2460	89.885	5.093	73.837	86.318	89.636	93.226	109.559
06/07	2460	91.303	5.489	77.371	87.465	91.060	94.713	110.927
07/08	2460	91.806	5.975	77.213	87.654	91.305	95.572	117.879
08/09	2460	91.055	5.697	75.774	87.121	90.700	94.735	109.975
09/10	2460	92.077	5.177	77.259	88.533	91.656	95.418	109.971
10/11	2460	91.410	5.072	76.149	87.740	91.194	94.776	108.271
11/12	1980	90.657	4.628	77.608	87.583	90.265	93.750	105.001
12/13	2458	91.406	4.825	76.063	88.093	91.351	94.483	110.095
13/14	2460	93.351	5.319	76.582	89.723	93.115	96.491	112.710
14/15	2460	93.258	4.657	78.310	90.019	93.250	96.569	107.133
15/16	2460	95.188	4.974	79.269	91.757	95.197	98.539	111.253
16/17	2460	95.810	5.000	81.982	92.323	95.514	99.268	111.796
17/18	2460	96.734	4.783	83.426	93.419	96.551	99.807	117.175
18/19	2460	99.484	4.938	85.288	96.218	99.464	102.776	114.616
19/20	1942	99.567	5.112	86.758	95.994	99.294	102.836	116.057

Tabla 14: Pace

otro, entonces la diferencia total va a ser 0. En cuanto al resto de características, no se aprecia nada anómalo.

Las Tablas 16 y 17 muestran la evolución de las métricas usadas para elaborar un porcentaje de victoria. Los valores están en torno a 0.5 ya que al estar basado en la diferencia de puntos, es lógico que el porcentaje de victoria que tenga un equipo sea opuesto al del contrintante. Se llama porcentaje pero no sigue las leyes de la probabilidad en el caso de *proj_win* ya que existen valores negativos y mayores que 1, como se ve en los valores mínimos y máximos.

La Tabla 18, referida al descanso entre partidos, muestra un conteo ligeramente diferente. En esta ocasión, para la última temporada es de 1912, siendo para el resto de temporadas desde la temporada 2004/2005 de 2430, exceptuando las dos temporadas con casos especiales. El resto de las temporadas, del año 2000 al 2004, esta variable se repite 2349 veces. Estos valores vienen de lo comentado anteriormente, al primer partido de la temporada no le precede ningún otro y no se ha establecido ningún descanso, se ha dejado como `NA`. Esto no supone un problema porque los primeros partidos no se toman en los modelos para así poder ver algun patrón en los datos con la evolución de la temporada.

Las tres últimas tablas corresponden al número de puntos (Tabla 19), de triples conseguidos por partido (Tabla 20) y la racha de encuentros ganados de entre los últimos 5 (Tabla 21). Las dos primeras ayudan a esclarecer por qué razón ha aumentado el factor referido a los tiros de campo. Además de que los puntos por partido han

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.0	12.963	-45.0	-9.0	0.0	9.0	45.0
01/02	2378	0.0	13.489	-53.0	-9.0	0.0	9.0	53.0
02/03	2378	0.0	13.276	-52.0	-9.0	0.0	9.0	52.0
03/04	2378	0.0	12.558	-47.0	-9.0	0.0	9.0	47.0
04/05	2460	0.0	12.653	-40.0	-9.0	0.0	9.0	40.0
05/06	2460	0.0	12.812	-45.0	-9.0	0.0	9.0	45.0
06/07	2460	0.0	13.150	-50.0	-9.0	0.0	9.0	50.0
07/08	2460	0.0	14.046	-52.0	-10.0	0.0	10.0	52.0
08/09	2460	0.0	13.452	-48.0	-9.0	0.0	9.0	48.0
09/10	2460	0.0	13.512	-50.0	-9.0	0.0	9.0	50.0
10/11	2460	0.0	13.057	-55.0	-9.0	0.0	9.0	55.0
11/12	1980	0.0	13.726	-44.0	-9.0	0.0	9.0	44.0
12/13	2458	0.0	13.557	-45.0	-9.0	0.0	9.0	45.0
13/14	2460	0.0	13.502	-48.0	-9.0	0.0	9.0	48.0
14/15	2460	0.0	13.682	-53.0	-9.0	0.0	9.0	53.0
15/16	2460	0.0	13.602	-51.0	-9.0	0.0	9.0	51.0
16/17	2460	0.0	14.041	-49.0	-9.0	0.0	9.0	49.0
17/18	2460	0.0	13.814	-61.0	-9.0	0.0	9.0	61.0
18/19	2460	0.0	14.659	-56.0	-9.0	0.0	9.0	56.0
19/20	1942	0.0	14.384	-49.0	-9.0	0.0	9.0	49.0

Tabla 15: Diferencia de puntos (*pt_diff*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.5	0.336	0	0.18	0.5	0.82	1
01/02	2378	0.5	0.336	0	0.182	0.5	0.818	1
02/03	2378	0.5	0.336	0	0.181	0.5	0.819	1
03/04	2378	0.5	0.337	0	0.172	0.5	0.828	1
04/05	2460	0.5	0.329	0.001	0.186	0.5	0.814	0.999
05/06	2460	0.5	0.33	0.001	0.184	0.5	0.816	0.999
06/07	2460	0.5	0.33	0	0.185	0.5	0.815	1
07/08	2460	0.5	0.342	0	0.164	0.5	0.836	1
08/09	2460	0.5	0.333	0	0.181	0.5	0.819	1
09/10	2460	0.5	0.335	0	0.179	0.5	0.821	1
10/11	2460	0.5	0.329	0	0.19	0.5	0.81	1
11/12	1980	0.5	0.34	0	0.17	0.5	0.83	1
12/13	2458	0.5	0.336	0	0.177	0.5	0.823	1
13/14	2460	0.5	0.331	0.001	0.183	0.5	0.817	0.999
14/15	2460	0.5	0.335	0	0.179	0.5	0.821	1
15/16	2460	0.5	0.332	0	0.181	0.5	0.819	1
16/17	2460	0.5	0.329	0	0.194	0.5	0.806	1
17/18	2460	0.5	0.325	0	0.201	0.5	0.799	1
18/19	2460	0.5	0.326	0	0.201	0.5	0.799	1
19/20	1942	0.5	0.322	0.001	0.203	0.5	0.797	0.999

Tabla 16: Porcentaje de victorias de Pitágoras (*pytha*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	0.5	0.427	-0.982	0.204	0.5	0.796	1.982
01/02	2378	0.5	0.444	-1.245	0.204	0.5	0.796	2.245
02/03	2378	0.5	0.437	-1.212	0.204	0.5	0.796	2.212
03/04	2378	0.5	0.413	-1.048	0.204	0.5	0.796	2.048
04/05	2460	0.5	0.417	-0.817	0.204	0.5	0.796	1.817
05/06	2460	0.5	0.422	-0.982	0.204	0.5	0.796	1.982
06/07	2460	0.5	0.433	-1.146	0.204	0.5	0.796	2.146
07/08	2460	0.5	0.462	-1.212	0.171	0.5	0.829	2.212
08/09	2460	0.5	0.443	-1.080	0.204	0.5	0.796	2.080
09/10	2460	0.5	0.445	-1.146	0.204	0.5	0.796	2.146
10/11	2460	0.5	0.430	-1.311	0.204	0.5	0.796	2.311
11/12	1980	0.5	0.452	-0.949	0.204	0.5	0.796	1.949
12/13	2458	0.5	0.446	-0.982	0.204	0.5	0.796	1.982
13/14	2460	0.5	0.445	-1.080	0.204	0.5	0.796	2.080
14/15	2460	0.5	0.451	-1.245	0.204	0.5	0.796	2.245
15/16	2460	0.5	0.448	-1.179	0.204	0.5	0.796	2.179
16/17	2460	0.5	0.462	-1.113	0.204	0.5	0.796	2.113
17/18	2460	0.5	0.455	-1.509	0.204	0.5	0.796	2.509
18/19	2460	0.5	0.483	-1.344	0.204	0.5	0.796	2.344
19/20	1942	0.5	0.474	-1.113	0.204	0.5	0.796	2.113

Tabla 17: Porcentaje proyectado de victoria (*proj_win*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2349	2.082	0.932	1.0	2.0	2.0	2.0	8.0
01/02	2349	2.083	0.967	1.0	1.0	2.0	2.0	8.0
02/03	2349	2.075	0.978	1.0	1.0	2.0	2.0	8.0
03/04	2349	2.074	0.954	1.0	1.0	2.0	2.0	9.0
04/05	2430	2.074	0.964	1.0	1.0	2.0	2.0	8.0
05/06	2430	2.077	0.923	1.0	2.0	2.0	2.0	8.0
06/07	2430	2.074	0.946	1.0	1.0	2.0	2.0	8.0
07/08	2430	2.070	0.933	1.0	2.0	2.0	2.0	7.0
08/09	2430	2.075	0.912	1.0	2.0	2.0	2.0	7.0
09/10	2430	2.075	0.945	1.0	2.0	2.0	2.0	8.0
10/11	2430	2.076	0.954	1.0	2.0	2.0	2.0	9.0
11/12	1950	1.879	0.870	1.0	1.0	2.0	2.0	8.0
12/13	2428	2.072	0.928	1.0	2.0	2.0	2.0	7.0
13/14	2430	2.077	0.942	1.0	2.0	2.0	2.0	7.0
14/15	2430	2.075	1.098	1.0	2.0	2.0	2.0	11.0
15/16	2430	2.076	1.035	1.0	2.0	2.0	2.0	10.0
16/17	2430	2.074	0.993	1.0	2.0	2.0	2.0	9.0
17/18	2430	2.157	1.025	1.0	2.0	2.0	2.0	10.0
18/19	2430	2.157	0.986	1.0	2.0	2.0	2.0	10.0
19/20	1912	2.180	1.080	1.0	2.0	2.0	2.0	10.0

Tabla 18: Descanso entre partidos (*rest*)

aumentado en las últimas temporadas, el intento de tiros de 3 se ha incrementado en más del 147 % desde la temporada 2000/2001 hasta la actual. Esto indica claramente que ha habido una modificación en el modo de juego en este aspecto confiando más en los tiros lejanos. En cuanto a la tabla de la racha de victorias, el conteo no es como en el resto de variables ya que esta comienza a tener valores a partir del sexto encuentro, cuando cada equipo ya ha disputado 5 partidos. Como se comentó anteriormente, esto no supone un problema porque se escogen encuentros de más adelante de la temporada. Se observa que la media se sitúa en torno al 2.5 y los valores mínimos y máximos son 0 y 5, indicando que no se ha ganado ningún partido de los últimos 5 y que se han ganados los 5, respectivamente.

Cabe destacar que cada atributo tiene un dominio diferente, lo que puede hacer que se proporcione más importancia a alguno de estos atributos que realmente no merece. Por esta razón, se ha llevado a cabo una **normalización** de los datos restando la media y dividiendo entre la desviación típica de manera que todas las variables consideradas partan con un mismo peso. En futuros análisis se probará también a realizar un escalado de los datos para hacer los datos compatibles con algunas técnicas de aprendizaje. Además, tareas futuras como el análisis de componentes principales pueden proporcionar un funcionamiento más adecuado cuando los datos están normalizados.

Temporada	**Conteo**	**Media**	**Desv. estándar**	**Min**	**25 %**	**50 %**	**75 %**	**Max**
00/01	2378	94.810	11.490	56.0	87.0	95.0	103.0	130.0
01/02	2378	95.476	11.529	56.0	88.0	95.0	103.0	141.0
02/03	2378	95.081	11.987	53.0	87.0	95.0	103.0	140.0
03/04	2378	93.397	12.064	56.0	85.0	93.0	101.0	138.0
04/05	2460	97.199	11.877	60.0	89.0	97.0	105.0	139.0
05/06	2460	97.009	11.970	60.0	89.0	97.0	105.0	152.0
06/07	2460	98.739	12.382	62.0	90.0	98.0	107.0	161.0
07/08	2460	99.923	12.924	54.0	91.0	100.0	109.0	168.0
08/09	2460	99.951	12.528	63.0	91.0	99.0	108.0	154.0
09/10	2460	100.447	11.905	59.0	92.0	100.0	108.0	152.0
10/11	2460	99.550	12.002	56.0	91.0	99.0	108.0	144.0
11/12	1980	96.260	11.635	56.0	89.0	96.0	104.0	149.0
12/13	2458	98.138	11.631	58.0	90.0	98.0	106.0	141.0
13/14	2460	101.009	11.861	66.0	93.0	101.0	109.0	145.0
14/15	2460	100.014	11.757	65.0	92.0	100.0	107.0	144.0
15/16	2460	102.672	11.741	68.0	95.0	103.0	111.0	147.0
16/17	2460	105.591	12.149	64.0	97.0	106.0	114.0	149.0
17/18	2460	106.333	12.065	69.0	98.0	106.0	114.0	149.0
18/19	2460	111.209	12.651	68.0	103.0	111.0	120.0	168.0
19/20	1942	111.446	12.410	73.0	103.0	111.0	120.0	159.0

Tabla 19: Puntos por partido (*pts*)

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2378	13.708	4.959	1.0	10.0	13.0	17.0	31.0
01/02	2378	14.749	5.689	1.0	10.0	14.0	18.0	40.0
02/03	2378	14.681	5.947	1.0	10.0	14.0	18.0	38.0
03/04	2378	14.925	5.494	2.0	11.0	14.0	18.0	41.0
04/05	2460	15.751	6.252	1.0	11.0	15.0	20.0	39.0
05/06	2460	15.981	5.971	1.0	12.0	16.0	20.0	44.0
06/07	2460	16.940	6.010	1.0	13.0	17.0	21.0	46.0
07/08	2460	18.107	6.045	2.0	14.0	18.0	22.0	45.0
08/09	2460	18.123	5.832	3.0	14.0	18.0	22.0	39.0
09/10	2460	18.139	5.766	4.0	14.0	18.0	22.0	47.0
10/11	2460	18.013	5.746	3.0	14.0	18.0	22.0	41.0
11/12	1980	18.381	5.790	3.0	14.0	18.0	22.0	43.0
12/13	2458	19.962	5.896	5.0	16.0	20.0	24.0	46.0
13/14	2460	21.534	5.981	5.0	17.0	21.0	25.0	42.0
14/15	2460	22.413	6.490	4.0	18.0	22.0	27.0	46.0
15/16	2460	24.082	6.665	4.0	19.0	24.0	29.0	47.0
16/17	2460	27.001	6.680	7.0	22.0	27.0	31.0	61.0
17/18	2460	29.000	6.687	11.0	24.0	29.0	33.0	59.0
18/19	2460	32.009	7.310	12.0	27.0	32.0	37.0	70.0
19/20	1942	33.938	6.936	14.0	29.0	34.0	38.0	60.0

Tabla 20: Tiros de 3

Temporada	Conteo	Media	Desv. estándar	Min	25 %	50 %	75 %	Max
00/01	2262	2.502	1.286	0.0	2.0	3.0	3.0	5.0
01/02	2262	2.500	1.274	0.0	2.0	3.0	3.0	5.0
02/03	2262	2.497	1.243	0.0	2.0	2.5	3.0	5.0
03/04	2262	2.497	1.263	0.0	2.0	2.0	3.0	5.0
04/05	2340	2.499	1.318	0.0	2.0	3.0	3.0	5.0
05/06	2340	2.500	1.264	0.0	2.0	2.0	3.0	5.0
06/07	2340	2.499	1.280	0.0	2.0	2.0	3.0	5.0
07/08	2340	2.499	1.316	0.0	2.0	3.0	3.0	5.0
08/09	2340	2.502	1.299	0.0	2.0	3.0	3.0	5.0
09/10	2340	2.501	1.317	0.0	2.0	3.0	3.0	5.0
10/11	2340	2.501	1.325	0.0	2.0	3.0	3.0	5.0
11/12	1860	2.508	1.273	0.0	2.0	3.0	3.0	5.0
12/13	2338	2.497	1.318	0.0	2.0	3.0	3.0	5.0
13/14	2340	2.499	1.290	0.0	2.0	3.0	3.0	5.0
14/15	2340	2.501	1.352	0.0	1.0	3.0	3.0	5.0
15/16	2340	2.504	1.278	0.0	2.0	2.5	3.0	5.0
16/17	2340	2.500	1.234	0.0	2.0	2.0	3.0	5.0
17/18	2340	2.500	1.310	0.0	2.0	3.0	3.0	5.0
18/19	2340	2.501	1.252	0.0	2.0	3.0	3.0	5.0
19/20	1822	2.497	1.310	0.0	2.0	3.0	3.0	5.0

Tabla 21: Racha de victorias en los últimos 5 partidos (*streak5*)

3.4.6 Selección de atributos

Cuando se consigue tener el conjunto de datos en el formato deseado, ya se puede empezar a probar diferentes algoritmos de aprendizaje para ver algunos resultados, pero se plantea el problema de que se cuenta con un número muy elevado de atributos. Será necesario llevar a cabo alguna técnica que reduzca el espacio de variables que se usan con el objetivo de eliminar el ruido que puedan aportar algunas variables inservibles o el hecho de que algunas de ellas estén muy correlacionadas entre sí.

Manual

En muchas ocasiones, la mejor manera de seleccionar los atributos relevantes en un problema de minería de datos es la selección manual. Se basa en el criterio de la persona encargada de llevar a cabo el proceso y en su conocimiento del problema y las variables involucradas. En el caso que se trata en este proyecto, se puede intuir que aquellas métricas creadas por analistas y entrenadores pueden tener mayor importancia.

Una posible combinación de variables incluye el porcentaje proyectado de victoria de cada equipo, el porcentaje de partidos ganados total así como el local del equipo local y el porcentaje como vistante del equipo visitante. También pueden influir los ratings ofensivos y defensivos además de alguno de los factores ideados por Dean Oliver. Por último, puede venir bien considerar la racha de partidos ganados de los últimos 5 de cada equipo para introducir un factor que muestre la forma del equipo.

Correlaciones

Una de las formas de elegir los posibles atributos es ver cuáles de ellas presentan una mayor correlación en valor absoluto con la variable respuesta *victory*. Estas serán previsiblemente las que aportan mayor información a la hora de llevar a cabo predicciones. Cuando se usa la correlación como métrica para la selección de atributos es necesario seleccionar aquellos que tienen relación con la variable respuesta pero que no estén intercorreladas entre sí ya que esto proporciona redundancia a los modelos. En la Tabla 22 se muestra la correlación entre todas las variables y la variable respuesta indicando aquellas con valores más grandes.

Como se ha mencionado previamente, es necesario quedarse con aquellas variables que no presenten correlación entre ellas. Para ello, se presenta la Figura 9 en la que se muestra la matriz de correlaciones de aquellas variables que presentan una correlación superior a 0.1 con la variable respuesta (marcadas en amarillo en la Tabla 22). En el estudio de los modelos se irá afinando en las variables concretas a elegir, pero de la imagen se pueden sacar algunas conclusiones. Por ejemplo, las variables referidas al porcentaje de victorias de Pitágoras y el porcentaje proyectado de victoria (*h_pyhta* y *a_pyhta*, *h_proj_win* y *a_proj_win*) están muy correladas entre sí y con las variables de las diferencias de puntos (*h_pt_diff* y *a_pt_diff*). Es lógico ya que la probabilidad se halla a partir de la diferencia de puntos, razón por la cual puede ser recomendable eliminar del conjunto porcentaje proyectado de victoria y la diferencia de puntos.

Variable	Correlación	Variable	Correlación
a_drbR	-0.033134	h_drbR	0.004703
a_drt	0.032535	h_drt	-0.055551
a_eFG	-0.076790	h_eFG	0.047605
a_ftR	-0.032828	h_ftR	0.036996
a_orbR	0.016823	h_orbR	0.007734
a_ort	-0.068733	h_ort	0.048428
a_pace	0.007611	h_pace	0.006620
a_poss	-0.011713	h_poss	-0.016345
a_pytha	-0.219779	h_pytha	0.235837
a_proj_win	-0.211204	h_proj_win	0.226269
a_pt_allowed	0.027731	h_pt_allowed	-0.058396
a_pt_diff	-0.217557	h_pt_diff	0.241176
a_rest	-0.025607	h_rest	0.007451
a_streak	-0.159736	h_streak	0.182469
a_tovR	0.031254	h_tovR	-0.063891
a_win_pct_away	-0.176796	h_win_pct_home	0.189109
a_win_pct_total	-0.204128	h_win_pct_total	0.216830
expected	0.049691	head_to_head	0.123105

Tabla 22: Correlación con la variable *victory*

Análisis de Componentes Principales

Otra manera popular de reducir el espacio de características en un trabajo de este tipo consiste en realizar un *análisis de componentes principales* (PCA) [5]. La idea detrás de esta técnica es conseguir un número reducido de atributos haciendo mínima la pérdida de información, creando nuevos ejes de coordenadas. Primero será necesario normalizar el conjunto de datos de manera que cada atributo tenga media 0 y varianza 1. Cada una de estas nuevas variables explica un porcentaje de inercia del conjunto de datos original, habrá que quedarse con aquellas que contengan una porción de inercia suficiente. Se define la inercia como la suma de las distancias al cuadrado de las observaciones al centro de gravedad, situado en el 0 por la normalización. Para conseguir las componentes principales se hallan los autovectores de la matriz de covarianzas del conjunto de datos normalizado. Estos autovectores proporcionan la proyección sobre los nuevos ejes así que para conocer los puntos modificados se realiza una combinación lineal de estos junto con los atributos anteriores. La inercia explicada por cada nueva coordenada viene dada por los autovalores correspondientes, la suma de todos ellos es la inercia total. Una forma muy utilizada de elegir el número de componentes es el 'método del codo', consistente en buscar el momento en el que el gráfico con la inercia explicada por cada una de ellas comienza a descender.

En la Figura 10 se muestra la evolución de la varianza explicada según la componente. Se aprecia que la cuarta componente presenta una bajada de la varianza

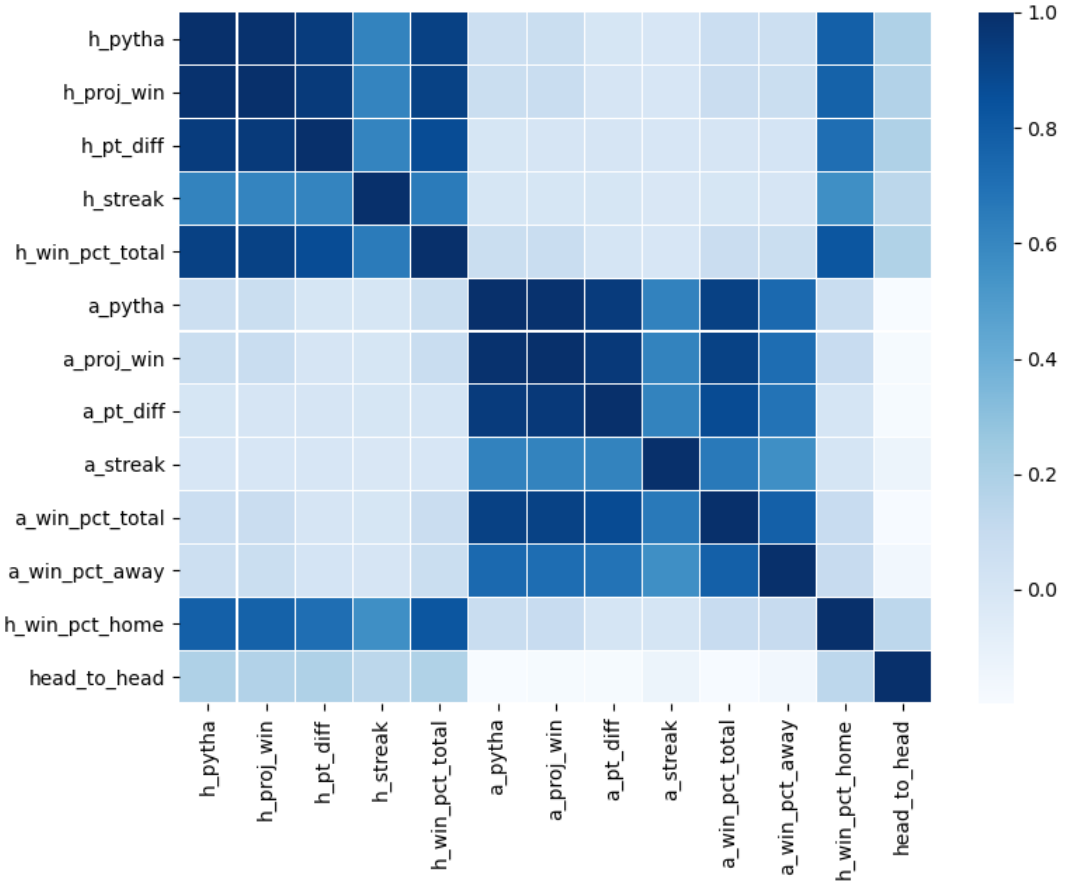

Figura 9: Correlación entre las variables

explicada muy grande respecto a la tercera. Además la quinta componente explica más o menos la misma que la cuarta, por lo tanto con tres componentes en este caso puede valer. Pero este tipo de criterio podría valer si en cada ejecución se viese la inercia que aporta cada componente, ya que cada ejecución va a ser diferente debido a la cantidad de variables que se usen y el número de temporadas. Se ha optado por elegir los componentes que aporten un 70 % de la inercia total, más adelante, en la Sección 3.5.2 se explica el por qué.

Este tipo de análisis y el de las correlaciones no son excluyentes. Esto significa que se pueden obtener primero las variables que mayor correlación tengan con la variable respuesta y eliminar aquellas intercorreladas, y luego aplicar el análisis de componentes principales sobres ellas.

Métodos envoltorio (Wrapper)

Existen una serie de técnicas en el ámbito de la minería de datos referidas a la selección de atributos. Dos de las más populares son los métodos *filtro* y los *envoltorio*.

- Los primeros tratan de reducir el espacio de características mediante la elección

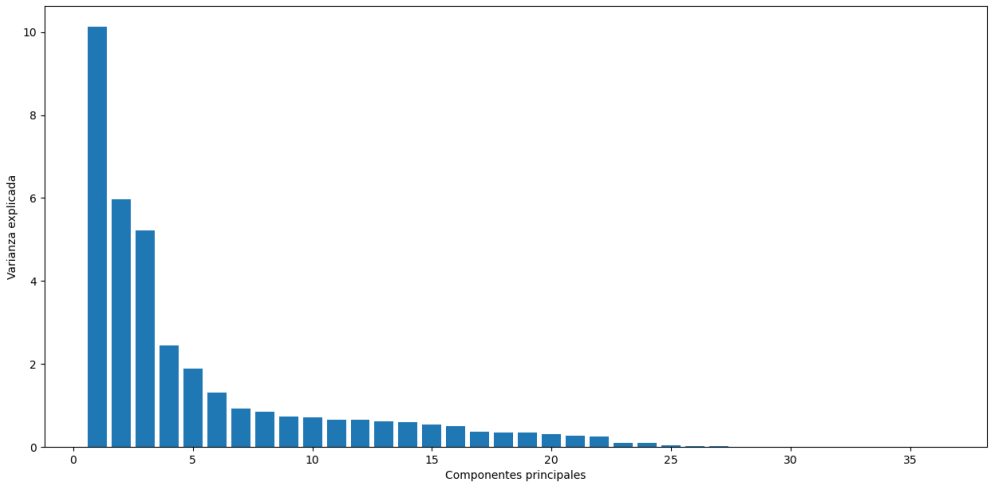

Figura 10: Análisis de Componentes Principales

de ciertos atributos basándose en medidas de distancias, correlaciones u otras informaciones referidas a los datos. El método de la correlación del que se ha hablado anteriormente entra dentro de este grupo. Los métodos filtro son sencillos pero se tiende a seleccionar un número demasiado elevado de atributos.

- Los métodos envoltorio utilizan el propio método de aprendizaje que se usará posteriormente en la tarea de clasificación para hallar las variables que más información aportan y mejor se comportan con ese método de aprendizaje. Estos métodos proporcionan resultados mejores para el algoritmo correspondiente aunque tienen un coste computacional mayor.

3.5 Modelado

En esta etapa se trata de encontrar aquellos modelos que consigan tasas de acierto mayores a la hora de clasificar cada partido como victoria local o no, dado por la variable *victory*. En fases posteriores se seleccionan los mejores de estos para poder refinarlos y mejorar su funcionamiento. Se explican ligeramente las diferentes técnicas de aprendizaje utilizadas así como la forma de proceder a la hora de elegir los conjuntos de entrenamiento y test, y cómo se eligen los parámetros que más convienen a cada algoritmo.

3.5.1 Técnicas de aprendizaje

Se van a crear diferentes modelos a partir de ciertos algoritmos de clasificación y regresión. Se utilizan los siguientes:

- **Regresión lineal múltiple**: la regresión intenta explicar el comportamiento de una variable dependiente, la variable *respuesta* (Y) a partir de variables independientes, los regresores $(X_1, X_2, ..., X_p)$. El hecho de que sea regresión multiple

implica que hay más de un regresor. Evidentemente, la función que relaciona la variable respuesta con los regresores es lineal.

$$Y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + ... + \beta_p + *X_p + \epsilon$$

El parámetro β indica el coeficiente de cada uno de los regresores. Si éste toma el valor 1 quiere decir que con un aumento de una unidad de la variable regresora asociada, también aumenta una unidad la variable explicativa. Para estimar estos coeficientes normalmente se hace uso del método de mínimos cuadrados.

$$\hat{\beta} = (X^t X)^{-1} X^t Y$$

Este tipo de regresión se usa en el análisis llevado a cabo en el proyecto tratado en [13].

- **Regresión logística**: este tipo de regresión tiene de característico que intenta predecir una variable respuesta que es categórica, especialmente binaria, o una variable entre 0 y 1 que representa una probabilidad. Sirve para modelar la probabilidad de un evento en función de otros, dados por las variables regresoras. En vez de usar una función lineal, hace uso de una función **logística**.

$$\pi(x) = \frac{e^{\beta_0 + \beta_1 * x_1 + ... + \beta_p * x_p}}{e^{\beta_0 + \beta_1 * x_1 + ... + \beta_p * x_p} + 1}$$

El término π se refiere a la probabilidad de pertenencia a la clase, es decir, en este problema una probabilidad de 1 quiere decir que el equipo local gana mientras que si es próxima a 0, el equipo local pierde. Por esto, se establece como umbral el 0.5 y, si la probabilidad hallada es superior a éste, esa observación se clasifica como *victoria*, de lo contrario será una *derrota*. Al ser una variable binaria, tiene mucho sentido aplicar este tipo de regresión al problema que se trata. El paquete de Python *sklearn* del que provienen todos los algoritmos usados ofrece la posibilidad de modificar ciertos parámetros de la gran mayoría de los algoritmos con el objetivo de mejorar los resultados. En el caso de la regresión logística se van a modificar dos de ellos.

  1. **Solver**: tiene que ver con el algoritmo usado para la optimización.
  2. **C**: relacionado con la regularización a la hora de optimizar la función correspondiente. Valores más pequeños indican una regularización mayor. El tipo de regularización depende del *solver* utilizado. La regularización ayuda a evitar el sobreajuste de los modelos a los conjuntos de entrenamiento.

- **Support Vector Machine (SVM)** [6]: éste es un algoritmo que puede ser usado tanto para tareas de regresión como de clasificación. El objetivo es encontrar un hiperplano que separe de la forma más cercana al óptimo las diferentes clases existentes en el conjunto de datos. Se llaman vectores soporte a los puntos que se encuentran más cerca de este hiperplano, los cuales ayudan a construirlo. La idea fundamental de este algoritmo es maximizar el margen que se encuentra entre los vectores soporte y el propio hiperplano.

Este algoritmo se comporta muy bien cuando los datos son linealmente separables. Cuando no se da el caso, es necesario transformar los datos de forma implícita para corregir esta situación, llevando a cabo transformaciones no lineales llegando a aumentar virtualmente las dimensiones de los datos. Para ello se usan funciones **kernel**. Las funciones que ofrece el paquete de Python utilizado para esta tarea son:

1. **Lineal**:
$$K(x, x') = x \cdot x'$$
2. **Polinómico**:
$$K(x, x') = (x \cdot x' + c)^d$$
d y c son hiperparámetros que habrá que modificar para averiguar cuál es la mejor combinación
3. **Función de base radial**:
$$K(x, x') = exp(-\gamma||x - x'||^2)$$
γ controla el comportamiento del kernel, cuando es pequeño se comporta como uno lineal
4. **Sigmoide**:
$$K(x, x') = tanh(\gamma(x \cdot c') + r)$$

Otro de los parámetros que hay que configurar es el de regularización, *C*. Este sirve para indicar cómo de restrictiva debe ser la regularización durante la optimización. También se encuentran reseñas positivas acerca de este tipo de algoritmo para el análisis de datos referidos a baloncesto en [13].

- **Perceptrón Multicapa** [11]: es una red neuronal artificial formada por múltiples capas, cada una de ellas contiene un número de neuronas. Las capas pueden ser de **entrada**, mediante las que se introducen los datos en la red, **ocultas**, formada por neuronas que trasladan la información de entrada realizando alguna modificación hasta la última capa, la de **salida**, a través de la que se devuelve el resultado de todo el procesamiento que ha llevado a cabo la red neuronal. Cada una de las neuronas de una capa está conectada con varias o todas las neuronas de la siguiente capa, estas conexiones tienen unos pesos.

  Este tipo de red neuronal presenta dos fases: la *propagación hacia adelante* y la *propagación hacia atrás*. En la primera se traslada la información de la capa de entrada a través de las capas ocultas hasta la capa de salida. Se lleva a cabo mediante combinaciones lineales de los pesos de cada conexión y los valores que presentan las neuronas. La fase de propagación hacia atrás sirve para actualizar los pesos de las conexiones haciendo uso del error hallado en la última capa.

  Cada neurona recibe el resultado de la combinación lineal de los pesos y neuronas de la capa anterior con las que está conectada. A este resultado se le aplica una *función de activación*, la cual indica si esta neurona se activa o no, es decir,

devuelve un valor lógico verdadero o falso. Las posibles funciones de activación que ofrece el paquete de **sklearn** de Python son:

1. **Identidad**
$$f(x) = x$$
2. **Logística**
$$f(x) = \frac{1}{1 + e^{-x}}$$
3. **Tangente hiperbólica**
$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$
4. **Rectificador**
$$f(x) = max(0, x)$$

El objetivo del perceptrón multicapa es encontrar la función que mejor se ajuste a los datos reales, tarea que realiza minimizando una función de error. Por ello se lleva a cabo el proceso de propagación hacia atrás tratando de hacer mínimo el error. Existen diferentes formas de optimizar esta función. De nuevo, estas son las que permite Python:

1. **LBFGS**: es un método quasi-newtoniano, lo que indica que aproxima la matriz Hessiana de una función, en este caso, la función de pérdida. La matriz Hessiana está compuesta por las derivadas segundas de la función respecto a todas las combinaciones de dos variables de la misma.
2. **Descenso estócastico de gradiente**: es un método de optimización que utiliza los gradientes para la actualización de los pesos. En el descenso del gradiente, se utilizan todas las observaciones del conjunto de entrenamiento para actualizar los pesos en cada iteración. Sin embargo, en el descenso estócastico del gradiente, se usa una sola observación del conjunto de entrenamiento en cada iteración del método.
3. **Adam**: es una extensión del descenso estocástico de gradiente. A diferencia de éste, la tasa de aprendizaje se modifica en el transcurso del proceso.

Otro de los parámetros a estudiar dentro de una red neuronal es α, el parámetro de regularización, el cual se encarga de penalizar los pesos otorgados a las grandes magnitudes durante la optimización de la función de pérdida para evitar así un sobreajuste.

Por último, es necesario configurar el número de capas ocultas que tendrá la red y el número de neuronas que tendrá cada una de ellas.

- **Random Forest**: este es un método de *ensemble* de clasificadores, esto es, una combinación de clasificadores. Se encuadra dentro de los *ensembles* de *bagging*, cuya característica principal es que todos los clasificadores base son el mismo algoritmo. En este caso, el algoritmo es el de *árboles de decisión*. A cada árbol se le otorga un conjunto de entrenamiento y en cada nodo se eligen al azar un

conjunto de variables con las que entrenar el modelo. Finalmente, se halla la respuesta promediando la de todos los árboles. En [13], el random forest es uno de los métodos que mejores resultados obtiene a la hora de realizar predicciones de partidos.

En la implementación se prueban diferentes combinaciones para los parámetros correspondientes a:

1. **Número de estimadores**: indica el número de árboles aleatorios que se usan.
2. **Criterio**: indica la forma de evaluar cómo de bien funciona un árbol. Puede ser *Gini* o *Entropía*.
3. **Máximo de características**: sirve para elegir el número de atributos considerados en cada árbol. Se va a considerar siempre el logaritmo en base 2 del número total de atributos.

- **Vecinos más cercanos (KNN)**: este es un algoritmo de clasificación supervisada, aunque también se usa para la regresión. Su funcionamiento se basa en clasificar cada observación según la clase a la que pertenecen las observaciones más cercanas a ella. Hay dos parámetros cruciales en el funcionamiento de este algoritmo. El primero de ellos es el número de vecinos que se van a considerar a la hora de llevar a cabo la clasificación. El otro parámetro de interés es la métrica usada para calcular la distancia entre observaciones. En Python, la distancia por defecto es la euclídea pero hay más alternativas:

  1. Euclídea
  $$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_p - y_p)^2}$$
  2. Minkowski
  $$d(x, y) = \sqrt{(x_1 - y_1)^P + (x_2 - y_2)^P + ... + (x_p - y_p)^P}$$
  P es un parámetro a decidir.
  3. Manhattan
  $$d(x, y) = |x_1 - y_1| + |x_2 - y_2| + ... + |x_p - y_p|$$
  4. Mahalanobis
  $$d(x, y) = (x - y)' \times \Sigma^{-1} \times (x - y)$$
  Σ es la matriz de covarianza

  En este proyecto se considera la distancia euclídea ya que es la más usada y, en muchas ocasiones, no modifica los resultados obtenidos por otra métrica de distancia.

- **Naive Bayes** [2]: también existen algoritmos basados en análisis bayesiano, en vez de frecuentista, que es lo habitual. Mediante el teorema de Bayes se puede hallar la probabilidad a posteriori de una hipótesis dados unos datos observados. Si las variables son discretas se utiliza la siguiente ecuación para el teorema de Bayes para conocer la probabilidad a posteriori:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Sin embargo, cuando las variables son continuas es necesario conocer la distribución de probabilidad de las mismas y se hallará la distribución a posteriori. Cuando se tienen varias hipótesis, se puede hallar el *Factor Bayes* como forma de comparar ambas. Esto es algo que puede equivaler al p-valor en el análisis frecuentista. Por lo tanto, el teorema de Bayes proporciona un método lógico que usa observaciones nuevas así como probabilidades a priori de las hipótesis a demostrar.

A partir de esta teoría se crea el clasificador *Naive Bayes*. El aprendizaje de este algoritmo se basa en la creación de un modelo probabilístico bayesiano que asigna una probabilidad de pertenencia a una clase para cada instancia. El clasificador *Naive Bayes* asume la independencia de los atributos, cosa que se suele violar en la mayoría de los casos. En esta ocasión se utiliza lo siguiente:

$$P(y_j|\mathbf{x}) = \frac{\prod_{k=1}^{p} P(x_k|y_j) \times P(y_j)}{P(\mathbf{x})}$$

Donde y es la clase y $\mathbf{x}$ se refiere a los atributos. Dado que se otorgará a cada observación la clase para la que se presente la mayor probabilidad a posteriori, basta con quedarse con los valores que proporcionen un valor mayor para el numerador de la ecuación mostrada.

El paquete de *sklearn* de Python permite la modificación de un parámetro α, para el suavizado de las distribuciones, y un parámetro lógico para aprender o no las probabilidades a priori, de no elegirse esta opción, establece una distribución uniforme.

Para un mejor funcionamiento de los métodos explicados es necesario llevar a cabo un proceso de estandarización de las entradas. El método *Naive Bayes* no funciona con entradas negativas, por lo que se lleva a cabo un escalado para obtener las entradas con valores ente 0 y 1. Para el resto de algoritmos de clasificación, se usa la normalización mediante la sustracción de la media y la división entre la desviación estándar.

3.5.2 Creación de modelos

A la hora de crear los modelos, además de los diferentes métodos de aprendizaje explicados en el apartado anterior, se van a tener en cuenta diversos factores de los que

ya se ha hablado ligeramente en este proyecto.

El primero de todos es el número de observaciones que se van a tener en cuenta. Como se ha visto en la exploración de los datos, ha habido una evolución en la forma de jugar primando el apartado ofensivo, en especial, los tiros de 3 puntos. Por esta razón se van a utilizar, por una parte, los datos desde la temporada 2000/2001, por otra, los *box score* desde la temporada 2004/2005 porque aún había 29 equipos desde el 2000 hasta el 2004 y, por último, las últimas 5 temporadas ya que la forma de jugar puede no haberse visto tan modificada en tan poco tiempo. Además, no se van a utilizar todos los partidos de cada temporada, solo aquellos desde Noviembre, ya que la temporada comienza a mediados de Octubre y se requiere de una pequeña evolución en los datos de la temporada para que todos los equipos hayan disputado un mínimo de 5 partidos y poder conocer su racha en ellos. Cambiando la forma de realizar los experimentos, también se va a probar a utilizar los datos de la temporada actual, la 2019/2020, para predecir los resultados de los partidos restantes en esa misma temporada.

Otro de los aspectos fundamentales a tener en cuenta es el conjunto de atributos que se utilizan para entrenar los modelos e intentar predecir el resultado de un partido. Ya se han comentado en el apartado 3.4.6 cuáles son los distintos conjuntos, se prueban todos ellos con el resto de configuraciones. Siguiendo con este aspecto, también se tendrá en cuenta si se lleva a cabo o no un análisis de componentes principales, el cual modifica el espacio de características con el que se trabaja. El método de *wrapper* puede llevarse a cabo realizando un proceso de validación cruzada para intentar conseguir el número óptimo de variables. Dado que este proceso necesita de muchos recursos, solo se va a usar con las regresiones. Para el SVM y el Random Forest se usará el *wrapper* manteniendo fijo el número de atributos en 10. El resto de algoritmos no funcionan por incompatibilidades de los paquetes. El método *wrapper* que se utiliza consiste en la eliminación recursiva de atributos, es decir, se comienza con todos ellos y se van quitando para conseguir el conjunto de atributos que mejores resultados proporcionan.

También se tiene que estudiar qué parámetros de cada método de aprendizaje son los que proporcionan un mejor funcionamiento del mismo. Para la elección de éstos se lleva a cabo un proceso de validación cruzada. Éste consiste en dividir el conjunto de datos en tantos bloques como se especifique, en este caso, se han utilizado 10. Es un proceso iterativo de manera que en cada una de estas iteraciones se cogen todos los bloques menos uno de ellos para entrenar sirviendo el bloque sobrante como test. Se realizan tantas iteraciones como bloques hay para que cada uno de ellos pueda servir de test en alguna de ellas. Para la búsqueda de los parámetros óptimos se crean plantillas con las diferentes combinaciones de todos los parámetros ejecutando cada una de ellas mediante validación cruzada de manera que la que menor tasa de error proporcione será la elegida.

El número de componentes a seleccionar durante el análisis en componentes principales no se ha incluido entre los parámetros a elegir de los modelos en el proceso de la validación cruzada ya que esto puede aumentar el tiempo de ejecución de una forma muy grande. En cambio, se ha probado con algunos modelos en su versión estándar, es

decir, manteniendo todos los atributos y con todas las temporadas como entrenamiento, y se ha llegado a la conclusión de que 70 % es el mejor valor en cuanto a cantidad de inercia explicada por las componentes principales.

Siguiendo con el tema de las diferentes configuraciones de los parámetros, todos los métodos implementados constan de diversos parámetros configurables, excepto la regresión lineal. Estos parámetros ya se han comentado en el apartado anterior, en el siguiente apartado se muestran los valores que se han planteado para cada método de aprendizaje y los mejores resultados obtenidos por cada uno de ellos.

3.5.3 Primer experimento

En este primer experimento se utilizarán como entrenamiento los datos de las tres posibilidades de conjuntos de datos, es decir, desde la 2000/2001, desde la 2004/2005, y desde la 2014/2015, exceptuando la temporada 2019/2020, que se dejará para el test. Sobre estos conjuntos se llevará a cabo el proceso de validación cruzada para conocer los hiperparámetros óptimos de cada uno. Una vez se tengan éstos, se utilizará los conjuntos de entrenamiento mencionados para entrenar el clasificador y se hallará la tasa de acierto sobre el conjunto de test, que será la última temporada, la 2019/2020. Hay que evitar que los modelos sobreajusten al conjunto de entrenamiento, es decir, se ajusten demasiado a este conjunto y no tengan capacidad para generalizar.

En los modelos de este experimento se reflejan los parámetros utilizados para conocer el funcionamiento óptimo de las técnicas de aprendizaje.

- **Regresión lineal**
  Como se comentó previamente, en la *regresión lineal* no se ha modificado ningún parámetro, Python ofrece la inclusión o no de un intercepto, cosa que no supone un cambio muy significativo en los resultados, y la normalización, tarea que ya se ha realizado previamente.

  - Selección de atributos: manual con PCA
  - Temporadas utilizadas: desde la 2014/2015
  - Tasa de acierto: 0.6667

- **Regresión logística**
  El siguiente método de aprendizaje usado es la *regresión logística*. En esta ocasión, sí hay parámetros que pueden ser modificados. Son el parámetro C, el inverso del parámetro de regularización, y el algoritmo usado para optimizar la función, *solver*. En la Tabla 23 se muestran los valores utilizados para cada uno de estos parámetros.

  - Selección de atributos: manual con PCA
  - Temporadas utilizadas: desde la 2000/2001

C	0.001	0.1	0.5	1	
Solver	*newton-cg*	*lbfgs*	*liblinear*	*sag*	*saga*

Tabla 23: Parámetros de la regresión logística

– Tasa de acierto: 0.67

En este caso se consigue una tasa de acierto para el entrenamiento similar a la de test por lo que se descarta el sobreajuste. En cuanto a los parámetros, C es 0.001 y el solver usado es el *liblinear*.

- **Support Vector Machine**
  Para la creación de modelos utilizando la máquina de vectores soporte como algoritmo, es necesario configurar los parámetros dados en la Tabla 24. Estos corresponden al parámetro de regularización y al kernel usado durante la ejecución del algoritmo.

C	0.00001	0.001	0.1	1	2
Kernel	*linear*	*poly*	*rbf*	*sigmoid*	

Tabla 24: Parámetros de SVM

– Selección de atributos: manual con PCA
– Temporadas utilizadas: desde la temporada 2004/2005
– Tasa de acierto:0.6667

Se han conseguido los mejores resultados cuando C es 2 y el kernel usado es el *linear*. La tasa de acierto obtenida durante el entrenamiento ha sido muy ligeramente superior a la de test, del orden de 0.67, por lo que no se considera que ha habido sobreajuste.

- **Perceptrón multicapa**
  Las redes neuronales tienen una gran cantidad de parámetros modificables. En este proyecto se han tenido en cuenta el parámetro α, el cual es el parámetro de regularización; la función *activation*, la función de activación en cada neurona de las capas ocultas; el *solver*, la forma de optimizar los pesos; y el número de capas ocultas y neuronas que tiene cada una. En la siguiente Tabla, la 25, se encuentran los valores probados para cada parámetro.

α	0.001	0.1	1	2
Activación	*identity*	*logistic*	*tanh*	*relu*
Solver	*lbfgs*	*sgd*	*adam*	
Capas ocultas	(100,)	(30,5)		

Tabla 25: Parámetros del perceptrón multicapa

– Selección de atributos: manual con PCA

- Temporadas utilizadas: desde la 2004/2005
- Tasa de acierto: 0.6711

Estos resultados se han dado haciendo uso del perceptrón multicapa con $\alpha = 2$, activación *relu*, solver *adam* y estructura de capas (30,5), es decir, la primera capa oculta con 30 neuronas y la segunda con 5. Cabe mencionar que éste es un algoritmo no determinista, lo que supone que ejecuciones del mismo modelo puede reportar resultados ligeramente diferentes.

- **Random Forest**
  El siguiente método de aprendizaje usado es el *random forest*. En esta ocasión, se determinan el número de árboles aleatorios a usar y el criterio por el que se mide la calidad de los resultados obtenidos por cada árbol. Otro parámetro importante es el número de atributos tratados en cada árbol, éste se ha mantenido fijo en todas las combinaciones como log_2 del número total de características introducidas. Los valores para el número de árboles y el criterio son los de la Tabla 26.

Número de árboles	50	100	150
Criterio	*gini*	*entropía*	

Tabla 26: Parámetros del Random Forest

  - Selección de atributos: todos los atributos
  - Temporadas utilizadas: desde la 2014/2015
  - Tasa de acierto: 0.6633

  La tasa de acierto durante el entrenamiento es muy similar a la conseguida sobre el test. El número de árboles que mejores resultados ha proporcionado es 150, mientras que el criterio usado ha sido el de *gini*.

- **K-Vecinos Más Cercanos**
  Para este algoritmo, sólo se ha probado un parámetro, el referido al número de vecinos que se tienen en cuenta. Se ha evaluado el método para un número de vecinos desde 16 hasta 46 de 2 en 2.

  - Selección de atributos: manual
  - Temporadas utilizadas: desde la 2014/2015
  - Tasa de acierto: 0.6589

  En este caso la tasa de acierto durante el entrenamiento es de 0.6523, por lo que no existe sobreajuste ya que es muy similar a la de test. Este resultado se ha proporcionado usando 42 vecinos.

α	0	1	10	100
fit_prior	False	True		

Tabla 27: Parámetros del Naive Bayes

- **Naive Bayes**
  Por último, se ha hecho uso del clasificador Naive Bayes. Para éste se prueban valores diferentes para el parámetro de suavizado de las distribuciones y para aprender o no la distribución a priori.

  - Selección de atributos: método envoltorio (10 atributos)
  - Temporadas utilizadas: desde la 2004/2005
  - Tasa de acierto: 0.6511

  La tasa de acierto durante la validación cruzada es del mismo orden que 0.6511. Los parámetros que mejores resultados han dado han sido $\alpha = 100$ y $fit\_prior = False$.

Hay que destacar que la mayoría de estos clasificadores tienen más parámetros que modificar, pero con el objetivo de agilizar el proceso de la validación cruzada se han tenido en cuenta sólo aquellos que se consideran más determinantes en el comportamiento de cada algoritmo.

Se aprecia que la elección manual de los atributos a participar en el modelado es la que mejores resultados reporta en general. Además, el análisis en componentes principales también suele mejorar ligeramente la tasa de acierto conseguida.

En cuanto a las plantillas de parámetros elegidas para el proceso de la validación cruzada, se han tenido en cuenta la prueba previa de los modelos con algunos parámetros concretos, que han aportado una idea del orden que deben tener éstos.

3.5.4 Experimento con los datos hallados según local o visitante

En este experimento se han modificado ligeramente los pasos para la creación de las estadísticas avanzadas que se utilizan para la realizacion de los modelos. En esta ocasión, se tiene en cuenta si el equipo es local o visitante para la acumulación de los valores mencionada anteriormente. Por ejemplo, un equipo ha jugado 3 partidos, los 2 primeros de local y el siguiente de visitante, en los que consiguió 90, 91 y 92 posesiones, respectivamente. Si el siguiente partido que juega este equipo lo hace de local, llevará 90.5 posesiones por partido, si lo hace de visitante llevará 92 por partido. La forma anterior cogía todos los partidos anteriores, sin hacer distinción entre local y visitante en este aspecto.

La manera en que se realiza el experimento es igual que en el apartado anterior, tomando las mismas temporadas como entrenamiento y test y realizando un proceso

de validación cruzada igual, probando los mismos parámetros.

Los resultados obtenidos en este experimento son ligeramente peores que en el anterior. Las mejores tasas de acierto se han conseguido con la regresión logística, el random forest y el perceptrón multicapa.

En el primer caso la tasa de acierto ha sido de 0.6589, se ha hecho uso de los datos desde la temporada 2014/2015 con los atributos seleccionados manualmente y sin hacer un análisis de componentes principales.

El random forest que mejor ha funcionado lo ha hecho con los datos desde la temporada 2000/2001 y con las columnas seleccionadas mediante la correlación. En este caso la tasa de acierto ha sido de 0.6611.

Por último, el perceptrón multicapa proporciona una tasa de acierto del 0.6578 usando los datos desde la temporada 2000/2001 y las variables seleccionadas mediante la correlación.

Estas tasas de acierto siguen siendo más pequeñas que las del anterior experimento pese a ser las mejores separando las características avanzadas según se juegue en casa o como visitante. En cuanto al resto de clasificadores, las tasas de acierto se sitúan en torno al 0.64 o 0.65.

El hecho de que estas tasas sean ligeramente peores que con los datos calculados usando el total de partidos puede deberse a que de esta manera no se capturan de igual forma las características de los equipos, no siendo tan importante las características conseguidas dividiendo según el lugar del partido, sino la evolución completa.

3.5.5 Experimento con los datos de una misma temporada

En este experimento se utilizan sólo los datos de la última temporada para crear los conjuntos de entrenamiento y test. Se realiza este experimento porque los otros no han reportado buenos resultados, así que es de preveer que la información más cercana a los partidos que se quieren predecir puede ser más determinante en los resultados.

Se va a trabajar con períodos de quince días. Primero se cogerán los primeros quince días de la temporada desde que todos los equipos hayan jugado ya 5 partidos, para así tener todas las estadísticas rellenas y no contar con `NAN` en los datos. Con esos quince días como entrenamiento, se comprobarán los modelos con los quince días siguientes de la temporada. El siguiente paso es aumentar el conjunto de entrenamiento añadiendo los quince días que antes han funcionado como test al entrenamiento. Ahora se tendrá como test los siguientes quince días. Este proceso se repite hasta que se hayan tomado todos los partidos de la temporada.

En este experimento se llegan a conseguir, en cierto tramos de la temporada, tasas de acierto muy superiores a las obtenidas en experimentos anteriores. Especialmente, los resultados son muy satisfactorios cuando se tiene una cantidad reducida de datos,

circunstancia que se da cuando se escogen las primeras semanas de la temporada.

- **Regresión lineal**
  Los resultados no son muy esclarecedores. Tomando como entrenamiento el primer período de quince días, se llega a conseguir una tasa de acierto de 0.76 sobre los siguientes quince días haciendo uso de los atributos seleccionados manualmente y un análisis de componentes principales. Sin embargo, en los siguientes períodos estos resultados bajan drásticamente. En los siguientes períodos los resultados se estabilizan ligeramente para, finalmente, mejorar ligeramente hasta el último período. En la Figura 11 se muestra la evolución de la tasa de acierto para los modelos que utiliza las variables halladas mediante el método envoltorio y de forma manual, ambas habiendo aplicado PCA.

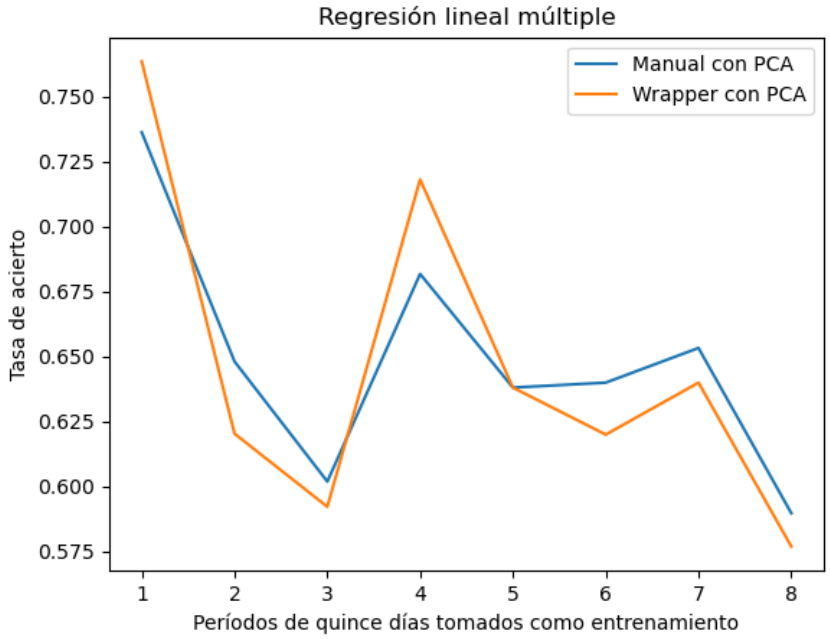

Figura 11: Tasa de acierto según el número de semanas escogidas como entrenamiento para los mejores modelos de regresión lineal múltiple

- **Regresión logística**
  Los resultados proporcionados por la regresión logística siguen la misma tónica que la regresión lineal múltiple. De nuevo, se consiguen tasas de acierto bastante superiores que en los experimentos anteriores, de entorno a 0.74 para el primer período seleccionado, pero los resultados bajan de nuevo rápidamente. A diferencia del anterior, sí se aprecia que los peores modelos no son tan malos como el caso de la regresión lineal ya que en ése se encontraban modelos con tasas de acierto por debajo del 0.5. Además, los mejores modelos no son los que tienen exactamente las mismas características ya que antes funcionaba bien el modelo con atributos seleccionados mediante *wrapper*, pero ahora no. En la Figura 12 se

ven los modelos con atributos seleccionados manualmente con PCA y mediante correlación sin PCA.

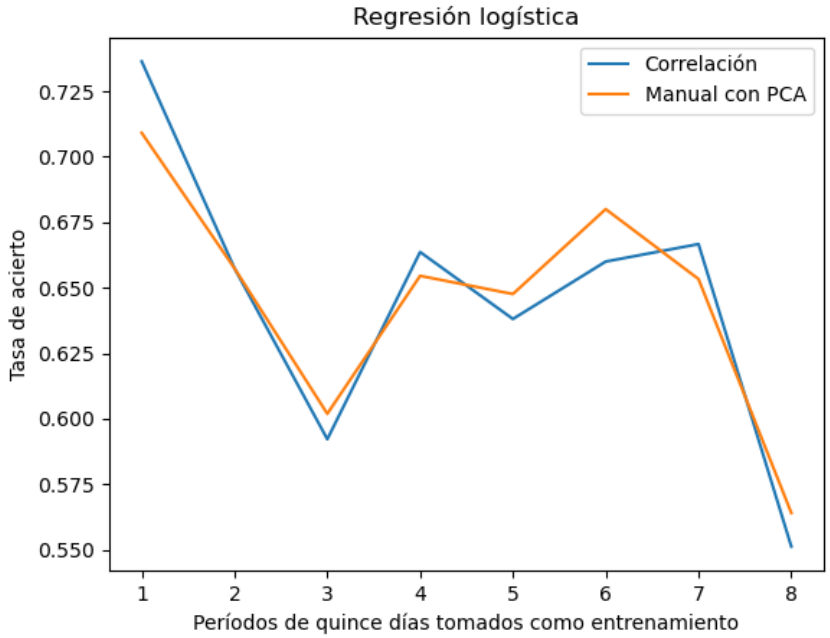

Figura 12: Tasa de acierto según el número de semanas escogidas como entrenamiento para los mejores modelos de regresión logística

- **Support Vector Machine**
  Con el SVM sucede algo similar a lo anterior, con la diferencia de que los resultados no son tan extremos como en los dos clasificadores anteriores. Los buenos resultados no son tan buenos, aunque sí bastante mejores que en los experimentos anteriores, y los malos resultados no son tan malos. En la Figura 13 se muestra la evolución del modelo utilizando los modelos con los atributos seleccionado manualmente y con el método *wrapper*, ambos realizando además un análisis en componentes principales.

- **Perceptrón multicapa**
  Los resultados mostrados por el perceptrón multicapa son de los mejores mostrados por los clasificadores. Aunque no llegue a las tasas de acierto más altas, como se han visto anteriormente, la mayoría de los modelos presenta unos resultados competitivos. Pasado el primero período, la tasa de acierto baja, aunque no tanto como en otros casos. En la Figura 14 se muestra la evolución de la tasa de acierto en función del número de períodos de quince días que se toman como entrenamiento para el modelo MLP en el que se escogen los atributos mediante el criterio de correlación, realizando además un análisis en componentes principales y el modelo con atributos seleccionados manualmente, también con PCA.

- **Random Forest**
  A diferencia de la mayoría de clasificadores probados en este trabajo, parece

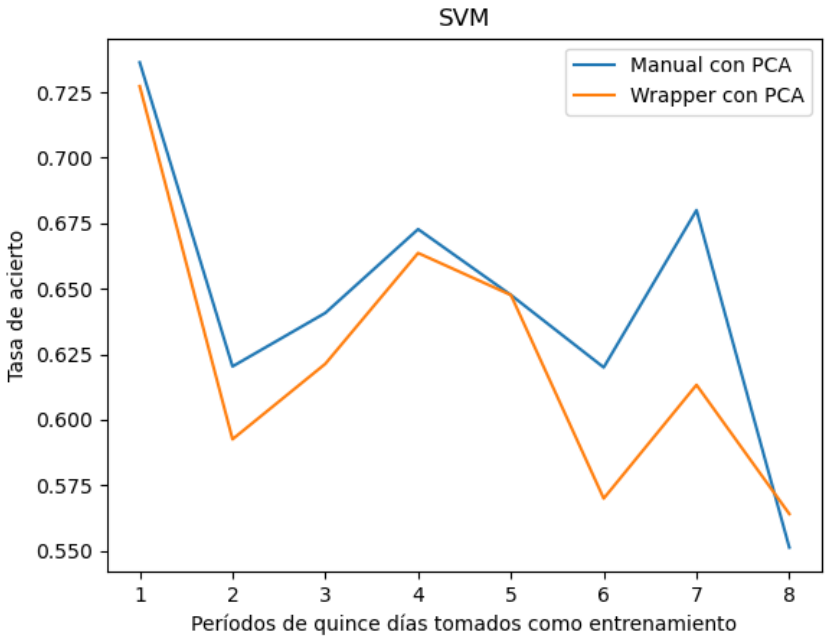

Figura 13: Tasa de acierto según el número de semanas escogidas como entrenamiento para los mejores modelos de SVM

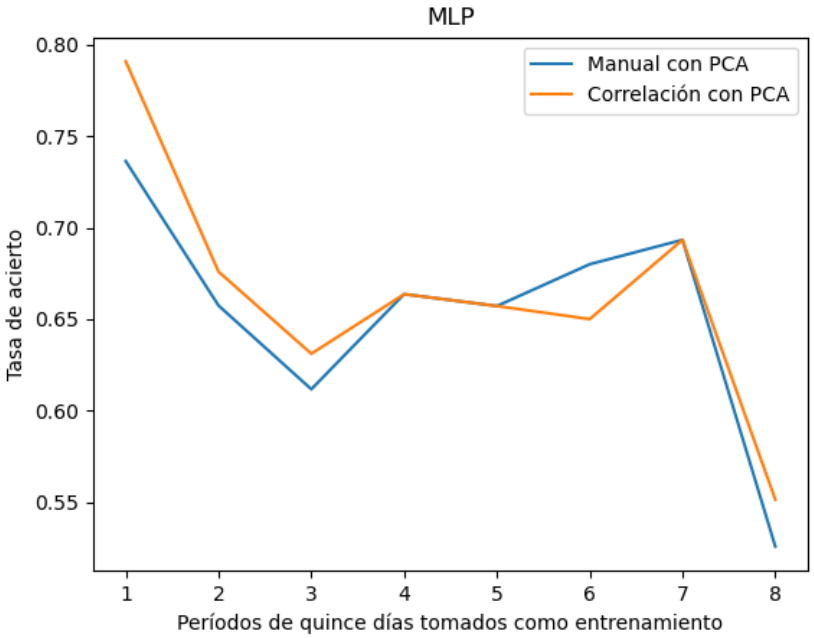

Figura 14: Tasa de acierto según el número de semanas escogidas como entrenamiento para el mejor modelo de MLP

que el análisis de componentes principales no funciona muy bien. El clasificador consigue tasas de acierto mayores de 0.75 utilizando el primer período como entrenamiento. Se puede apreciar en la Figura 15 la evolución de la tasa de acierto según el número de períodos de quince días tomados como entrenamiento para los modelos con todos los atributos y con los atributos seleccionados mediante el método envoltorio.

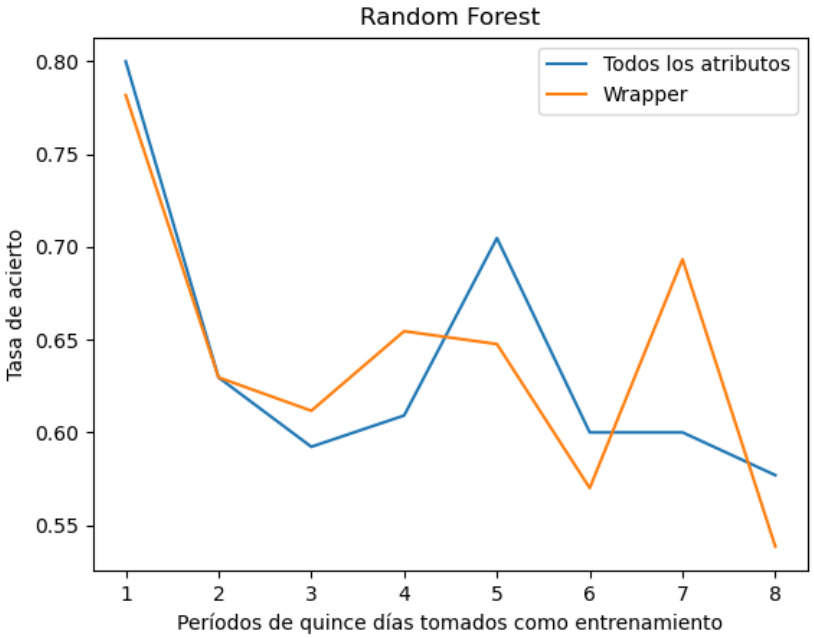

Figura 15: Tasa de acierto según el número de semanas escogidas como entrenamiento para el modelo con atributos seleccionados manualmente del Random Forest

- **K-Vecinos Más Cercanos**
  El clasificador de K-Vecinos, en experimentos anteriores, no era de los que mejor funcionaba. Ahora es capaz de conseguir tasas de acierto superiores al 0.76 en las primeras semanas de la temporada aunque, como el resto de algoritmos, vuelve a bajar la tasa de acierto. Funciona bastante mejor realizando un análisis en componentes principales. A continuación, se muestra en la Figura 16 la evolución del acierto para el modelo con todas los atributos y para el modelo con selección manual y PCA. A pesar de la bajada suele haber una recuperación en la tasa de acierto en las fases finales, sin tener en cuenta la última.

- **Naive Bayes**
  El Naive Bayes es, sin duda, el clasificador más estable de los probados en este experimento. Pese a que la mayor tasa de acierto es de 0.7, los peores modelos rara vez bajan de 0.6. Por ejemplo, en la Figura 17 se muestra la evolución de la tasa de acierto para el modelo con todos los atributos y para el modelo con los atributos seleccionados mediante la correlación.

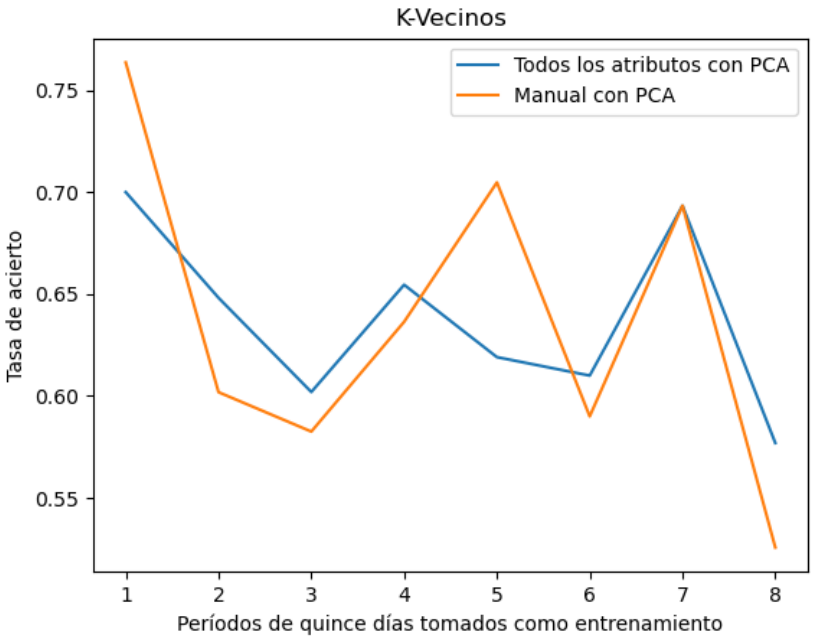

Figura 16: Tasa de acierto según el número de semanas escogidas como entrenamiento para algunos modelos de KNN

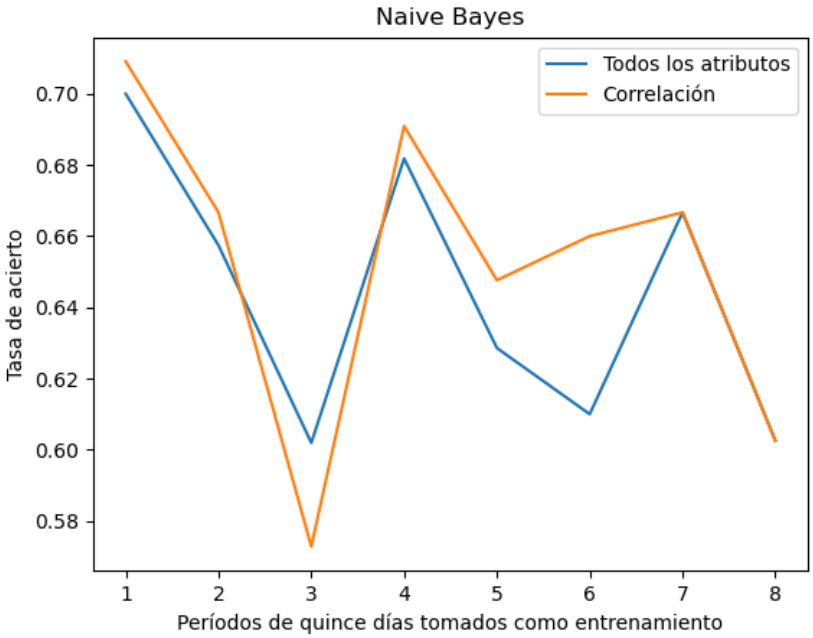

Figura 17: Tasa de acierto según el número de semanas escogidas como entrenamiento para algunos modelos de Naive Bayes

Todos los clasificadores siguen más o menos la misma evolución, cuando se utiliza solo el primer período como entrenamiento para hallar el funcionamiento del modelo sobre los siguientes quince días, se obtienen los mejores resultados. Sin embargo, a partir de este momento, la tasa de acierto suele descender hasta que se tienen 5 o 6 semanas de entrenamiento, donde se estabiliza ligeramente.

Se aprecia también que en el último tramo siempre hay un descenso drástico en la tasa de acierto. Esta fase corresponde al período justo después de haber disputado el partido de All-Star en el que participan los mejores jugadores de cada conferencia. Además, los jugadores que no participan en este partido tienen días de descanso, que pueden influir en el juego en los encuentros posteriores. Lo ideal para realizar un análisis sobre estos datos sería saltarse este período, ya que aporta mucho ruido.

La conclusión que se puede obtener de este apartado es que el pasado reciente tiene un mayor poder predictivo, como se puede ver en el primer período de estudio. El hecho de que al principio se consigan tan buenos resultados también puede deberse a la preparación de los equipos durante la pretemporada y la condición en la que llegan al principio de la temporada. Una vez pasado este período, la tasa de acierto baja, probablemente porque todos los equipos comienzan a ponerse en forma o porque se mezclan las tendencias de los equipos y se pierde capacidad de predicción.

3.6 Evaluación

En esta fase se pretende hacer hincapié en aquellos modelos que mejores resultados han mostrado para así poder profundizar más en ellos y conseguir resultados un poco mejores.

Dado que los resultados mostrados por el experimento llevado a cabo separando la adquisición de características avanzadas según la localización del partido (Sección 3.5.4) no son demasiado buenos, no se van a tener en cuenta esos modelos en este apartado.

3.6.1 Evaluación del primer experimento

Partiendo del primer experimento, se modificarán los hiperparámetros de los clasificadores ligeramente para los modelos que mejor funcionaron para ver si se puede afinar aún más en esos valores. También se probarán esos modelos sin normalizar ya que, en ocasiones, puede resultar contraproducente este procedimiento.

En el caso de la **regresión lineal múltiple**, el mejor modelo reportó una tasa de acierto de 0.6667. Si se realiza el mismo modelo con los datos sin estandarizar o con escalado en vez de normalización los resultados no son mejores que los anteriores, especialmente cuando no se realiza ninguna de las dos, en cuyo caso la tasa de acierto baja de 0.6.

Para la **regresión logística**, el mejor modelo consiguió un resultado de 0.67, similar al de la regreisón lineal. Éste se dio usando el *solver liblinear* y con $C = 0.001$

usando los datos desde la temporada 2000/2001. Por lo tanto, se van a probar valores de C en torno a ese valor y también a hacerlo sin estandarizar los datos. Se consigue llegar a 0.6722 al modificar el parámetro C a 0.002. Sin embargo, el hecho de no normalizar supone una tasa de acierto peor.

El modelo de **SVM** que mejor tasa de acierto proporciona es el realizado con los datos desde la temporada 2004/2005 con $C = 2$ y *kernel lineal*. Dado que 2 era el último valor que se probó en la validación cruzada, pueden existir valores mayores para este parámetro que consigan mejores resultados. No resulta ser el caso ya que un valor para el parámetro C de 1 y de 0.1 también reportan la misma tasa de acierto del 0.6667, aunque la tasa de acierto para valores más pequeños de C implica menor acierto durante el entrenamiento ya que este parámetro es el inverso de la constante de regularización del algoritmo, por lo que el sobreajuste es menor.

El **perceptrón multicapa** que mejor funcionó en el experimento tratado anteriormente reportó una tasa de aciero del 0.6711. El problema de este algoritmo es que no es determinista ya que la inicialización de los pesos de las relaciones entre las neuronas puede variar, por lo que otra ejecución de este modelo puede no conseguir ese mismo resultado. Aún así, se van a probar valores cercanos para el valor α y distintas configuraciones para las capas ocultas. Se consigue llegar a una tasa de error del 0.6733 usando como α el valor 3 y una capa oculta con 5 neuronas, una configuración más sencilla que la probada en el anterior experimento. Al intentar evitar la estandarización de los datos, los resultados son peores.

Para el clasificador **Random Forest** existe la posibilidad de mejorar los resultados aumentando el número de árboles aleatorios establecidos, ya que en el mejor modelo del experimento anterior se usaron 150, el valor más alto probado. Por lo tanto, existen valores mayores que éste que pueden mejorar el rendimiento. También se han tenido en cuenta valores más pequeños que 50, que era el valor más pequeño que se probó en la validación cruzada, pero los resultados son también peores que en el experimento original. Se ha probado también a limitar el número de hojas que puede tener un árbol, pero aún así los resultados obtenidos no han conseguido mejorar los que se hallaron anteriormente. Al igual que el perceptrón multicapa, el Random Forest no es determinista ya que los resultados se modifican en parte por los atributos seleccionados por cada árbol y qué subconjunto de datos va a parar cada uno de ellos.

Para el clasificador de **K-Vecinos** no existe mucha esperanza de mejora, ya que el único parámetro que se modifica es el número de vecinos que se tiene en cuenta y el valor óptimo fue de 42, un valor intermedio en la plantilla. Lo que se puede hacer es probar con los datos sin normalizar o escalados en vez de normalizados. Como era de esperar, no se mejoran los resultados obtenidos en el anterior experimento.

Como pasaba anteriormente, con el clasificador **Naive Bayes** se consiguen los mejores resultados para el valor límite probado para el parámetro α, que era 100. Por lo tanto, valores mayores de este parámetro pueden mejorar los resultados. Finalmente, no es así. Además, para este algoritmo no se puede cambiar la forma en que se tratan

los datos en cuanto a la estandarización, que actualmente se realiza un escalado entre 0 y 1, ya que no permite valores negativos.

3.6.2 Evaluación del experimento con datos una misma temporada

Se vio en el experimento con los datos de la última temporada que, a pesar de obtener muy buenos resultados para las primeras semanas de la temporada, rápidamente los resultados empeoraban de una forma drástica y no se conseguía una estabilidad en ellos. Sin embargo, con la mayoría de los clasificadores se intuía un ligero repunte a continuación, marcado por un último descenso en las últimas fechas de Febrero.

Se quiere comprobar si esto es circunstancial de esta temporada, es por la incapacidad de los modelos de lograr generalizar o cuál es la razón detrás de este suceso. Por ello, en este apartado se van a volver a ejecutar los modelos seleccionados en el apartado mencionado (Sección 3.5.5), pero esta vez utilizando los datos de la temporada 2018/2019 en vez de la 2019/2020 ya que esta última está incompleta.

Para la **regresión lineal múltiple**, en el apartado mencionado, los mejores resultados venían de la mano de los modelos con selección manual y con el método envoltorio, ambos usando PCA. La evolución en la tasa de acierto para la temporada 2018/2019 se puede apreciar en la Figura 18. La evolución es muy similar a la de la temporada 2019/2020. A partir del octavo período, parece que las predicciones mejoran bastante.

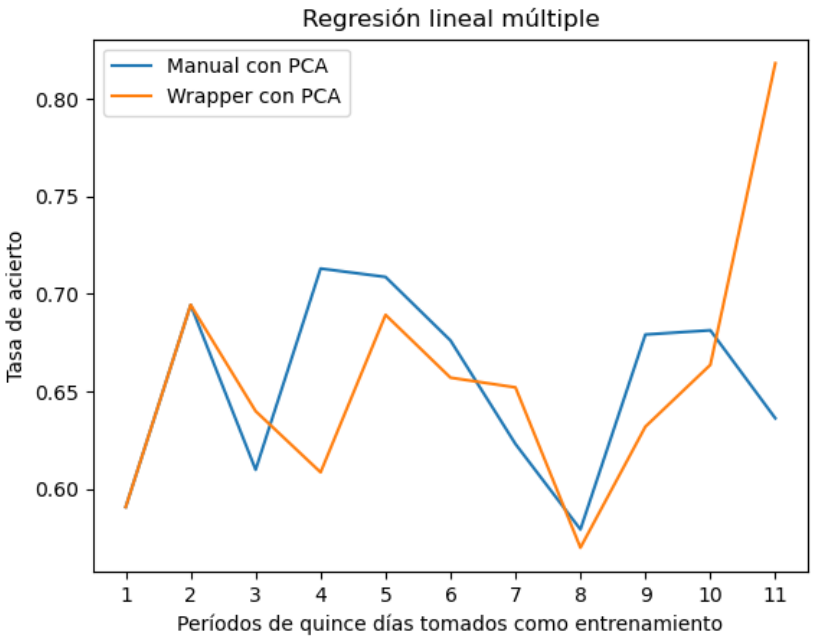

Figura 18: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de regresión lineal para la temporada 2018/2019

Los mejores modelos de **regresión logística** del experimento mencionado utilizaban los atributos hallados mediante correlación y los hallados de forma manual con análisis en componentes principales. Se han probado estos modelos para la temporada 2018/2019, se muestran los resultados en la Figura 19. El funcionamiento de este método sigue el mismo esquema que la regresión lineal, al principio se parte con resultados muy malos, pero oscilan mucho hasta llegar al cuarto o quinto período. A partir de aquí se produce un descenso para volver a subir en los últimos períodos, obviando el último de ellos.

Figura 19: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de regresión logística para la temporada 2018/2019

Para el clasificador **SVM** se seleccionaron los modelos que utilizaban los atributos seleccionados manualmente y con el método *wrapper*, ambos con PCA. En la Figura 20 se muestra la evolución de la tasa de acierto para la temporada 2018/2019. Los resultados de estos modelos oscilan mucho durante toda la temporada, no se puede sacar una conclusión importante salvo que se vuelve a repetir la bajada de la tasa de predicción en el período 8.

El siguiente método de clasificación probado fue el **perceptrón multicapa**, en este caso los atributos usados fueron los seleccionados manualmente por una parte, y los hallados mediante la correlación, por otra parte, ambos usando PCA. Para la temporada 2018/2019 se pueden ver los resultados en la Figura 21. De nuevo se ve mucha oscilación, pero parece que a partir del cuarto período se estabiliza bastante, salvando el caso del octavo período en el que siempre ocurre este suceso.

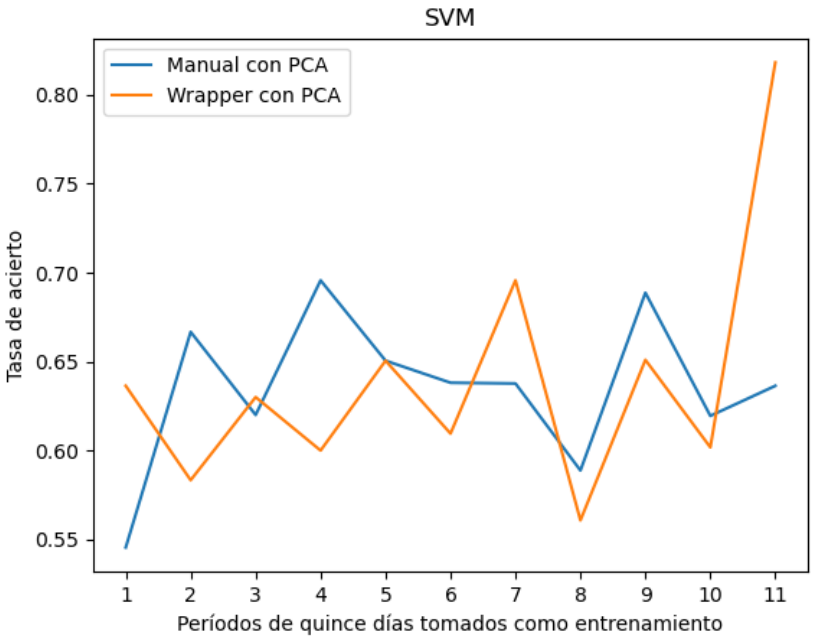

Figura 20: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de SVM para la temporada 2018/2019

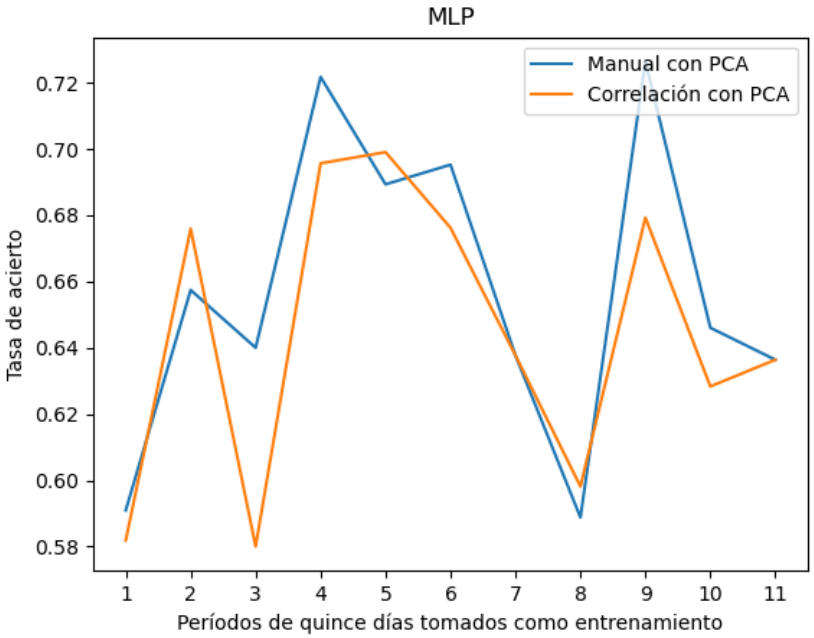

Figura 21: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de MLP para la temporada 2018/2019

Para el clasificador **Random Forest**, se usaban previamente los modelos con todos los atributos y con los atributos seleccionados mediante el método envoltorio (*wrapper*). Los resultados obtenidos en esta evaluación están presentes en la Figura 22. Hay oscilaciones en las tasas de acierto, pero a diferencia de otros clasificadores, la amplitud de éstas es menor. Se presenta la bajada de siempre en el período 8.

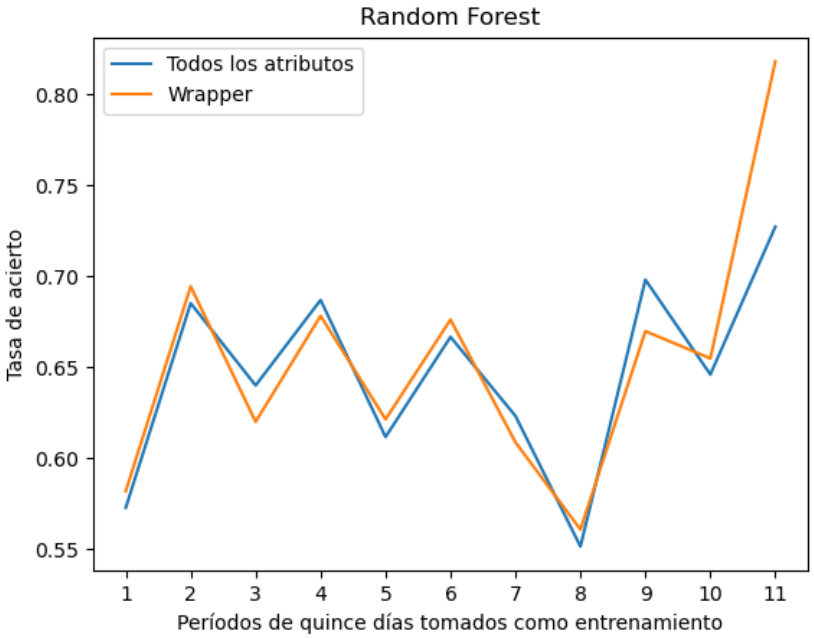

Figura 22: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de random forest para la temporada 2018/2019

El método de clasificación no supervisada de **K-Vecinos**, que se utilizó anteriormente, utilizaba todos los atributos y los atributos seleccionados manualmente, ambos modelos con análisis en componentes principales. Los datos de la temporada 2018/2019 para estos modelos aparecen en la Figura 23. Se aprecia que a partir del cuarto período las tasas de acierto se estabilizan, más o menos, exceptuando de nuevo el período octavo.

Por último, para el algoritmo **Naive Bayes** se usaron los modelos con todos los atributos y con atributos seleccionados mediante correlación. Se presentan los resultados en la Figura 24. La evolución es similar a la del resto de clasificadores, una mejora en la tasa de acierto desde el período 3, más o menos, para descender en el período 8.

Todos los clasificadores muestran evoluciones muy similares entre sí y respecto a los datos de la temporada 2019/2020, usados en la Sección 3.5.5. El período 8 supone que se han utilizado los primeros 8 conjuntos de quince días como entrenamiento, por lo tanto el conjunto de test usado se corresponde con los últimos días de Febrero y los primeros Marzo. Estos partidos son los acontecidos justo después del partido de las

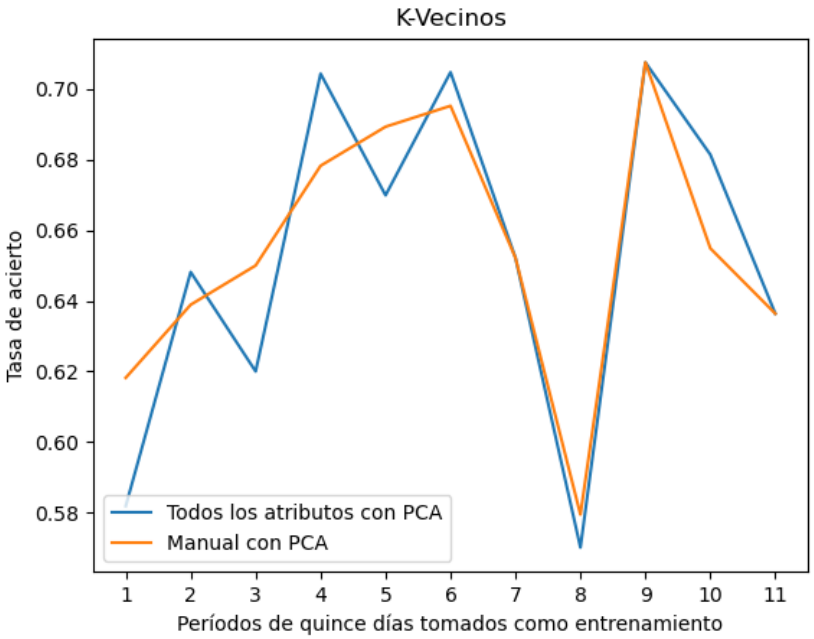

Figura 23: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de k-vecinos para la temporada 2018/2019

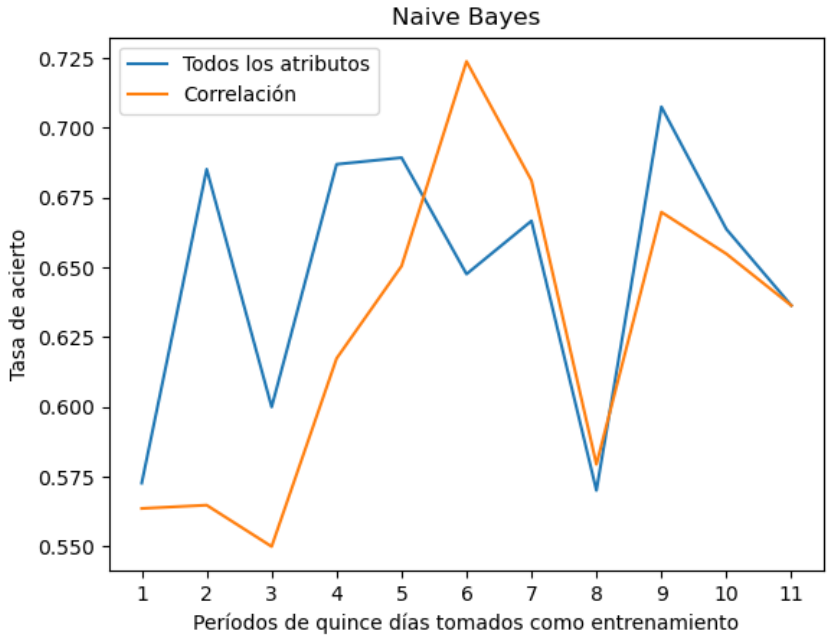

Figura 24: Tasa de acierto según el número de semanas escogidas como entrenamiento para modelos de Naive Bayes para la temporada 2018/2019

estrellas, además del fin del período de traspasos, cuando se suelen realizar algunos intercambios de jugadores en las plantillas. Ésto, asociado a la importancia que tienen los buenos jugadores en el desarrollo de los partidos puede hacer que las predicciones tengan unos resultados malos.

En el período 11 también suele haber un descenso, aunque a veces hay un aumento muy grande en la tasa de acierto. El conjunto de test de este período cuenta solo con 11 partidos, por lo que no se puede confiar en la información mostrada para este tramo.

Con relación a los modelos de la temporada 2019/2020, las tasas de acierto de cada modelo usado en la temporada 2018/2019 se sitúan en valores muy parecidos, llegando a conseguir tasas de más de 0.72 en algunos clasificadores en épocas intermedias de la temporada.

3.7 Despliegue

La forma en que se muestran los resultados obtenidos es mediante la creación de una herramienta con la que pueda interaccionar un usuario. Los detalles sobre el análisis y diseño de ésta se encuentran en el siguiente apartado, la Sección 4.

4 Ingeniería de software

En este apartado se detallan los diferentes aspectos que conciernen a la creación de la herramienta y los pasos realizados desde el punto de vista del software. Se explican los flujos de trabajo llevados a cabo, mediante los cuales se organizan las actividades fundamentales, y que proporcionan diagramas para entender el funcionamiento, además del propio código. Todo esta sección se ha realizado partiendo de [14] y [4].

4.1 Requisitos

Los requisitos proveen de una descripción de los servicios que debe ofrecer el sistema y las restricciones y limitaciones que cumplir. Idealmente, se debe llevar a cabo una elicitación de los requisitos de manera que se tenga una primera idea de lo que debe hacer el sistema para luego detallar más en la fase de la especifiación de requisitos. Cuando se finalice la implementación del sistema, se validarán los requisitos y se podrán cambiar, añadir o eliminar algunos para futuras producciones. En este proyecto se detallan los requisitos propios de la fase de especificación.

4.1.1 Requisitos funcionales

Los requisitos funcionales recogen qué servicios debe proporcionar el sistema, cómo se comporta bajo determinadas situaciones y cómo reacciona ante entradas concretas. En general, detallan qué es lo que debe hacer el sistema.

1. El sistema debe poder cargar los datos de la base de datos.
2. El sistema debe permitir la actualización de la base de datos de los partidos (*games*) pero no la de la información en bruto (*raw*).
3. El sistema debe permitir la elección y carga del modelo de predicción.
4. El sistema debe mostrar las características del modelo elegido.
5. El sistema debe permitir volver a entrenar el modelo seleccionado.
6. El sistema debe mostrar la tasa de acierto del modelo usado.
7. El sistema debe permitir la introducción de información acerca de un partido ya jugado para la predicción del resultado.
8. El sistema debe mostrar el equipo predicho como ganador en caso de que un usuario haya aportado información en este aspecto.

4.1.2 Requisitos no funcionales

Los requisitos no funcionales no tratan las tareas que debe realizar el sistema, sino cómo debe realizarlas. También se detallan las limitaciones y restricciones de las funciones ofrecidas por el sistema. Son requisitos en cuanto a tiempo dedicado, necesidades computaciones, seguridad, etcétera.

1. El sistema debe poder ser aprendido a usar en menos de 10 minutos.

2. El sistema debe ser fácil de manejar para un usuario inexperto.
3. El sistema debe trabajar con archivos en formato *csv*.
4. El sistema debe proporcionar tiempos de respuesta inferiores a 10 segundos en la carga de los modelos predictivos.
5. El sistema debe contar con una interfaz simple, clara y manipulable.
6. El sistema será implementado en Python.

4.2 Análisis

En esta etapa se pretende explicar las clases pertenecientes al sistema así como los diferentes actores que pueden participar. También se detallan los pasos realizados en cada caso de uso y las relaciones entre las clases previamente mencionadas.

4.2.1 Modelo de casos de uso

Los casos de uso definen las interacciones que tienen los actores con el sistema. También se detallan las relaciones entre los diferentes casos de uso, ya que algunos necesitan de la realización de otros para poder completarse. Un diagrama de casos de uso muestra toda esta información. En la Figura 25 se muestra el diagrama de este sistema.

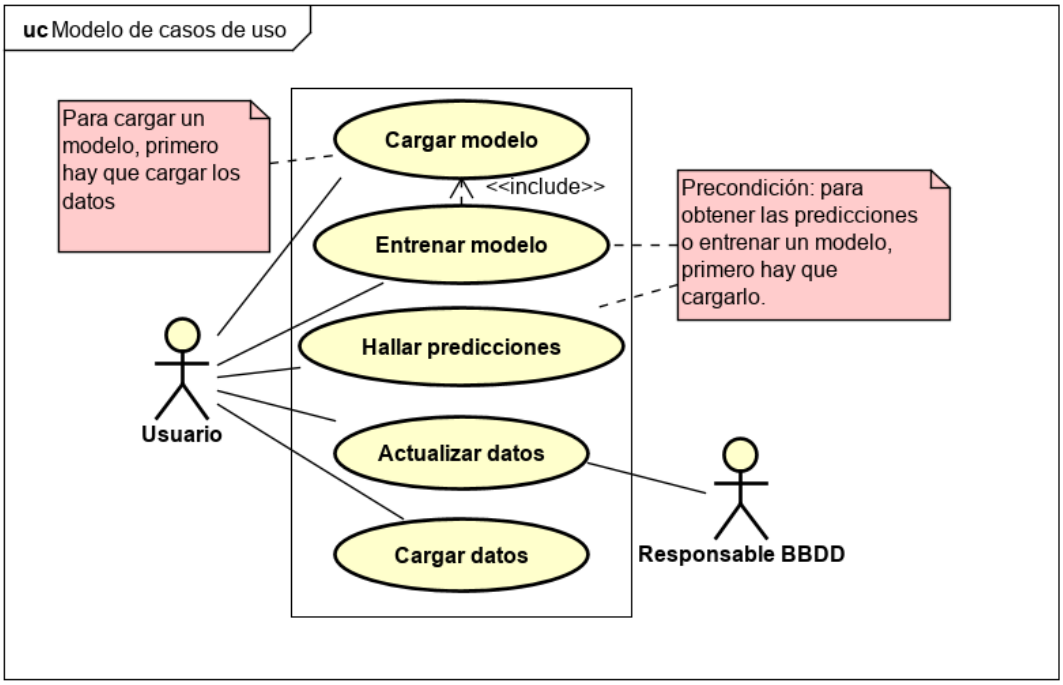

Figura 25: Diagrama de casos de uso

Como recordatorio y aclaración, a continuación se explican los 4 archivos con los que se trabaja en este proyecto presentes en la base de datos. Esta información ayuda a entender lo que se realiza en lo casos de uso que se explicarán posteriormente.

- *raw*: en este directorio se encuentra toda la información de los partidos por jugador y equipo. Además, se incluye información por cuartos. Es el archivo original del que se parte. De éste se obtienen los *box score* de cada partido y, posteriormente, el conjunto de datos de *team_totals*.

- *team_totals*: este archivo se usa para agilizar el proceso a la hora de incluir nuevos partidos en la base de datos. Contiene la información de los partidos según el equipo, por lo tanto tiene dos observaciones por partido disputado.

- *games*: este conjunto de datos contiene las estadísticas avanzadas utilizadas para realizar las predicciones. Contiene la información según el equipo local y visitante, por lo tanto, tiene una observación por partido.

- *current*: el último archivo del que se hace uso en este proyecto. Contiene la información relativa al momento actual de cada equipo en cuanto a las características avanzadas y la situación de los encuentros con el resto de franquicias. Este archivo se usa cuando se pide la predicción de un partido que todavía no se ha disputado.

De estos cuatro archivos se permite la modificación de los tres últimos. El primero de ellos lo modificará el encargado de obtener la información de la fuentes originales, tema ya tratado con anterioridad. También hay que mencionar que existe otro archivo llamado *franchises* que relaciona cada franquicia actual con el nombre que han tenido en el pasado para que el cálculo de los enfrentamientos directos sea más acertado. En cuanto a los casos de uso, son los siguientes:

- **Cargar los datos**: este caso de uso consiste simplemente en leer los datos de los partidos (*games*) de la base de datos y cargarlos en una variable del sistema.

- **Actualizar datos**: esta tarea compara la información presente en el archivo *games* con la de *raw*. Si hay algún partido que no ha sido tratado y del que no se han obtenido las estadísticas avanzadas con los pasos explicados en la Sección 3.4.2, se procederá a ello. La idea de esto es ir añadiendo los partidos que se jueguen cada día para después hallar las estadísticas avanzadas de éstos.

- **Cargar modelo**: en la herramienta se almacena un archivo por cada modelo que ha sido guardado, los que mejor han funcionado. El usuario puede seleccionar cuál de estos modelos usar para la predicción. Los modelos cargados ya están entrenados. Para realizar este caso de uso, es necesario haber realizado el caso de uso referido a la carga de los datos.

- **Entrenar modelo**: el modelo almacenado se vuelve a entrenar y calcular la tasa de acierto sobre el conjunto de test. Una vez hecho esto, se carga el modelo de nuevo.

- **Hallar predicciones**: cuando el usuario ha seleccionado un equipo local, otro visitante y la fecha en la que se disputó el partido, el sistema proporcionará el porcentaje de victoria de cada uno según el modelo que se esté utilizando. También mostrará el equipo que realmente venció en caso de proponorse un partido en el pasado. Sin embargo, si el usuario determina un partido que está por jugarse, se proporcionarán las probabilidades de vencer de cada equipo.

En el diagrama se ve que hay dos actores: un usuario que vaya a utilizar la herramienta para ver las predicciones de los modelos predictivos implementados y una persona que quiera gestionar los archivos de la base de datos que se utilizan. Este responsable de la base de datos puede actualizar la información de algunos de los archivos de la base de datos utilizados en este proyecto.

4.2.2 Modelo de clases

Las clases son las unidades principales del sistema. A través de ellas se realizan las tareas mencionadas en el apartado de los requisitos. Este modelo es estático, pues muestra la organización del diseño del sistema. Este modelo se representa mediante un diagrama de clases cuando se usa un paradigma de programación orientado a objetos, como es el caso. En estos diagramas se detallan las variables presentes en cada clase así como las operaciones que se pueden realizar en cada una de ellas. También se muestran las relaciones entre clases mediante líneas que las unen conformando asociaciones. En la Figura 26 se muestra el diagrama que representa las clases del sistema tratado en el proyecto.

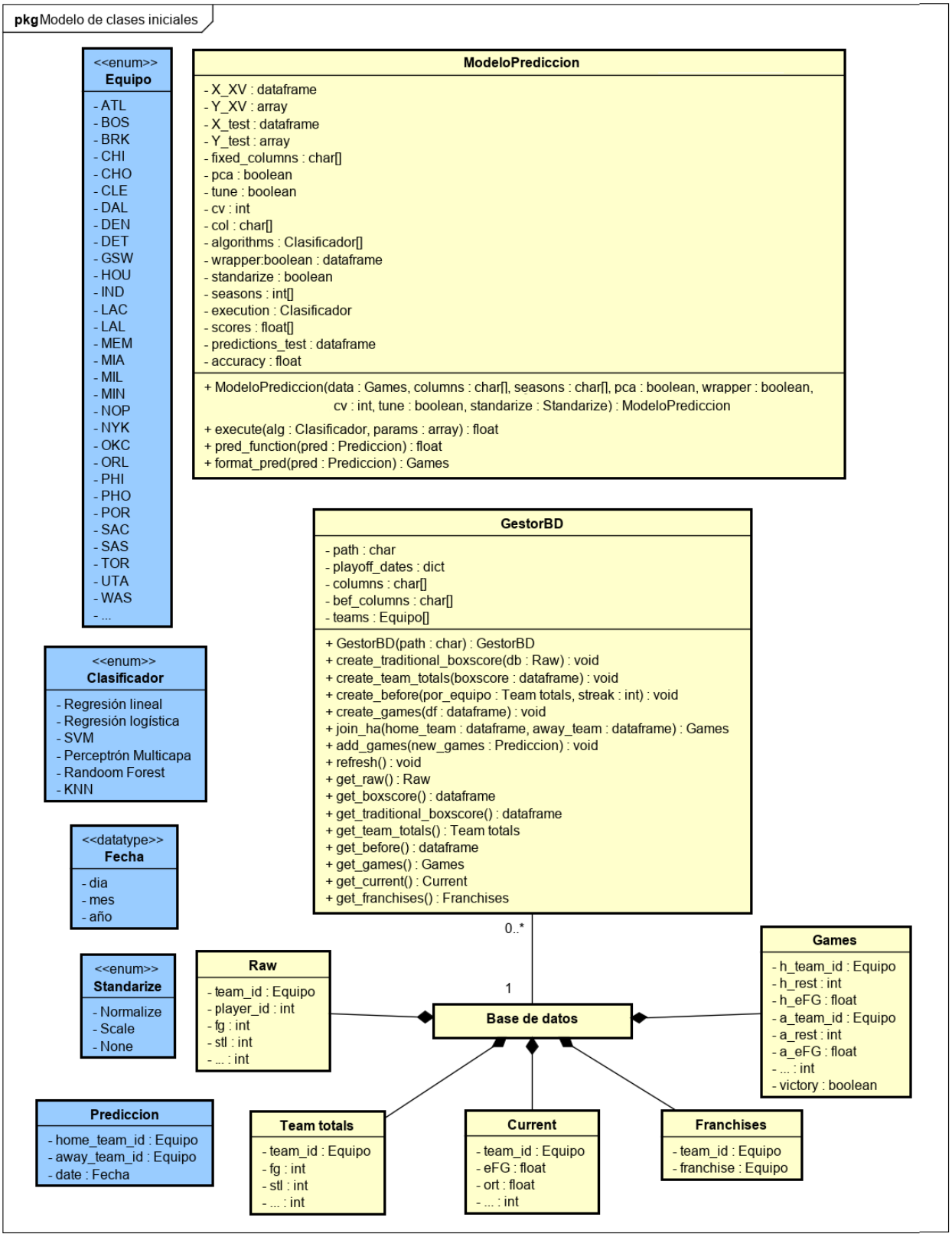

Figura 26: Diagrama de clases

Se aprecia que las dos clases principales son el gestor de la base de datos y la referida a los modelos predictivos. En la clase *GestorBD* se encuentran variables referidas a las columnas del *box score tradicional* y también otras variables que servirán en el procesado de la información. También hay una variable que contiene la abreviatura de todos los equipos para poder hallar los enfrentamientos directos entre franquicias. Entre las funciones se encuentran las correspondientes a la creación de todos los conjuntos de datos que se han mencionado previamente además de aquellas que sirven para leer de la base de datos los conjuntos requeridos y actualizar la información de estos, aunque por el momento solo se hace de manera local. La clase GestorBD está asociada a la clase Base de Datos. De esta base se podrán obtener los conjuntos deseados. Se aprecia que la base de datos está compuesta por los archivos ya mencionados. A pesar de que en la base de datos existen otros muchos archivos, en este proyecto solo se van a usar estos.

En cuanto a la clase *ModeloPrediccion*, se muestran los conjuntos que se crean para la validación cruzada y para el test a partir del conjunto *games*, que hay que pasar como argumento en la creación de un modelo. También hay algunas variables que son fijas que es necesario detallar para las tareas de predicción o del análisis en componentes principales.

Hay variables que se utilizan para saber si hay que realizar un análisis en componentes principales, selección de atributos mediante *wrapper*, o la validación cruzada para elegir los parámetros, detallado en la variable *tune*. Esta clase presenta la función usada para la ejecución del modelo creado, la cual requiere el clasificador usado y sus parámetros. Además, se ha implementado una función para hallar las predicciones de un partido, para la que hay que proporcionar un conjunto de datos con el formato *Predicción*, que contiene información de un partido del que se desea predecir el resultado. Como se ve, los partidos aportados por un usuario para su predicción tienen que tener el identificador del equipo local y visitante y la fecha del partido.

También se muestran en el diagrama de clases los tipos de clasificadores que se utilizan y las abreviaturas de todos los equipos así como la forma en que se define una fecha y cómo hay que pedir una predicción a un modelo. Se presentan además las distintas formas de tratado de los datos en cuanto a la estandarización.

4.3 Diseño

El diseño arquitectónico de la herramienta sirve para organizar el sistema de manera que se puedan cumplir los requisitos marcados. A la hora de diseñar la arquitectura hay que tener en cuenta aspectos como la fiabilidad, la eficiencia o la seguridad. También se muestran los pasos llevados a cabo en la realización de los casos de uso.

4.3.1 Arquitectura

En este proyecto se ha llevado a cabo una arquitectura *Modelo-Vista-Controlador*. Es una forma de separar la presentación de la interacción con el sistema. Se divide la lógica del sistema en tres componentes diferentes.

- **Modelo**: se encarga de la realización de las operaciones y de almacenar las variables del sistema. Sirve para realizar los procesos pertinentes en los datos y avisar de los cambios realizados para que la vista se modifique.
- **Vista**: esta componente se encarga de mostrar los datos a los usuarios. Refleja los valores de las variables del sistema y es la capa con la que interaccionan los usuarios.
- **Controlador**: sirve para manejar los eventos que un usuario provoca en la vista. Cuando el usuario interacciona con el sistema a través de la vista, es necesario llevar a cabo una serie de tareas. El controlador se encarga de comunicar al modelo las operaciones a realizar.

Los diferentes componentes deben estar comunicados entre sí de manera que el controlador es el encargado de elegir la vista que se quiere mostrar en cada momento, en este caso sólo hay una así que no se dará ese problema. Cuando se han realizado modificaciones por parte del modelo, éste debe notificarlas de manera que la vista puede realizar una consulta para conocer los nuevos valores y mostrarlos.

En el caso de la herramienta llevada a cabo en este proyecto solo se cuenta con una vista, la cual se muestra en la Figura 27.

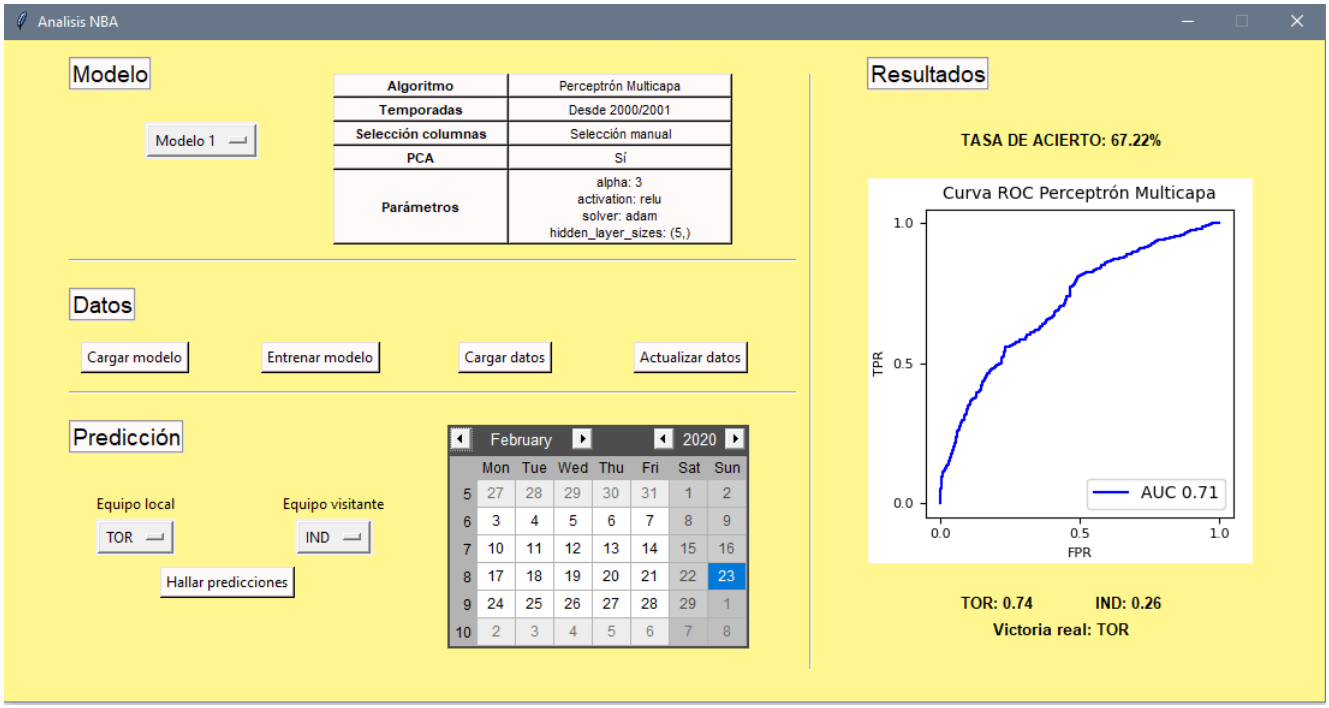

Figura 27: Vista del sistema

Se aprecia en la figura que existen cuatro apartados fundamentalmente, los cuales se explican a continuación.

1. **Modelo**: aquí se elige el modelo de entre los que han sido guardados. Además, se muestran las características de este modelo, desde las temporadas que se toman como entrenamiento hasta los parámetros del algoritmo de aprendizaje.

2. **Datos**: en esta sección se muestran los botones con los que interactuar para cargar los datos o actualizarlos, o también para cargar el modelo seleccionado o entrenarlo.

3. **Predicción**: este sector contiene etiquetas para el equipo local y para el visitante y un calendario para indicar la fecha del partido. Una vez que se tiene todo especificado, se puede pulsar sobre el botón que desencadenará el caso de uso *Hallar predicciones*.

4. **Resultados**: por último, se reflejan tanto la tasa de acierto del modelo predictivo seleccionado sobre el conjunto de datos de test, como una curva ROC sobre este mismo modelo. Además, si se ha especificado un partido a predecir, se mostrará en la parte inferior las probabilidades de victoria de cada equipo según el modelo predictivo, así como el equipo que realmente venció el encuentro, en el caso de solicitar un partido ya disputado.

A modo de aclaración, una curva ROC es un gráfico que muestra el ratio de *Verdaderos Positivos* frente al ratio de *Falsos Positivos*. Los primeros son las observaciones que, siendo verdaderas, fueran predichas como tal. Sin embargo, los falsos positivos son aquellas observaciones que, siendo falsas, fueron predichas como verdaderas. El área bajo la curva (AUC) es una estimación de lo bien que se comporta el modelo, en un caso ideal, este valor es 1.

4.3.2 Realización de los Casos de Uso

Este modelado sirve para mostrar el carácter dinámico del sistema. Ayuda a saber los pasos que se realizan en el tiempo de ejecución de la herramienta. En este caso se ha llevado a cabo un modelo *data-driven*, en el que el elemento principal son los datos. La disponibilidad de estos permite la realización de las tareas pertinentes. Estos modelos muestran la secuencia de acciones realizadas para procesar la información de entrada y obtener los resultados deseados. Para mostrar la realización de los casos de uso se han creado diagramas de secuencia en los que se detalla la interacción entre los actores y los elementos del sistema.

El primero de ellos es **Cargar los datos**. Para hacerlo simplemente se guarda en una variable del Modelo el conjunto de datos con el que se van a realizar las predicciones (*games*). La secuencia de pasos se ve en la Figura 28.

El diagrama de secuencia perteneciente al caso de uso **Actualizar los datos** está en la Figura 29. Para actualizar los datos, se recoge la fecha del último partido almacenado en los conjuntos de datos *Raw* y se compara con la fecha del último partido recogido en el conjunto de datos de *games*. Si no coincide, se procesa la información de los partidos no tratados para conseguir las estadísticas avanzadas y el resto de variables de *games*. Una vez se han hallado se actualizan los conjuntos de datos modificados:

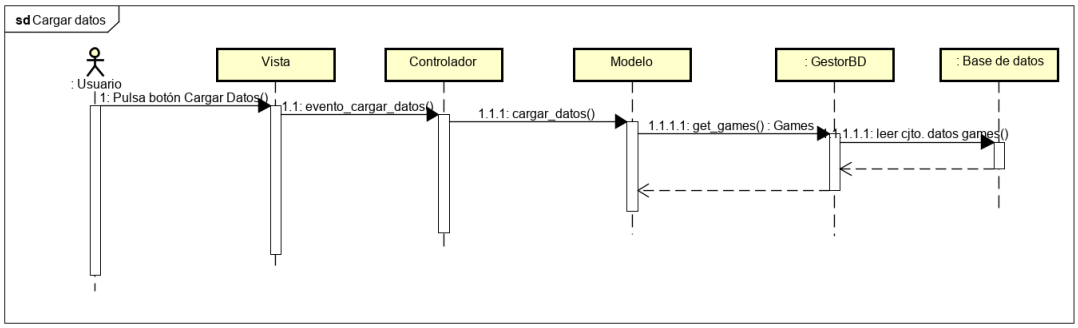

Figura 28: Diagrama de secuencia para *Cargar los datos*

games, *current* y *team_totals*.

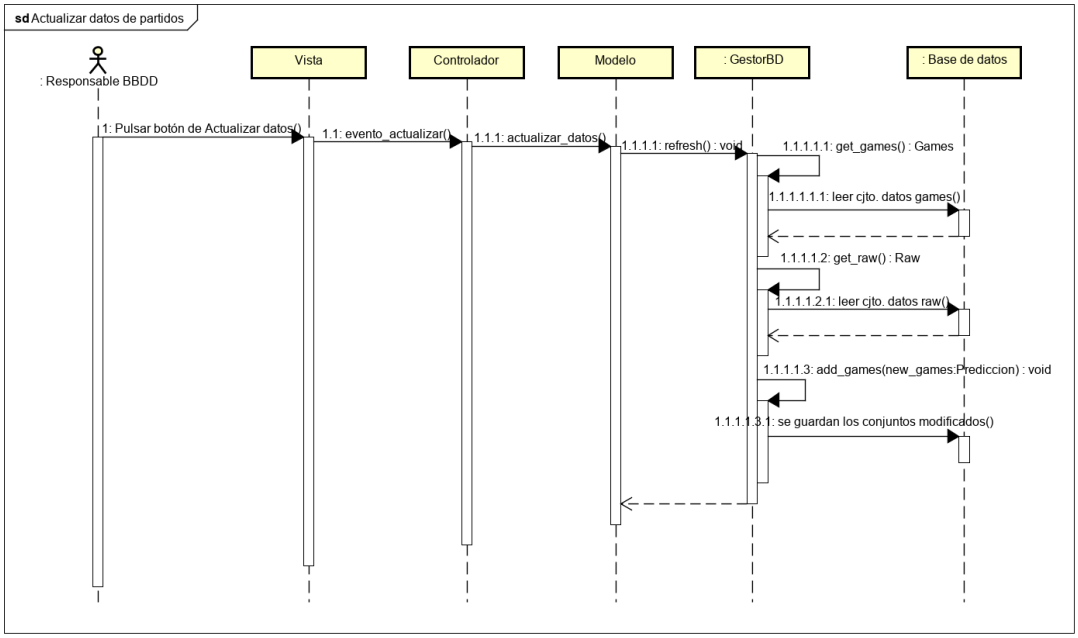

Figura 29: Diagrama de secuencia para *Actualizar los datos*

El siguiente caso de uso es **Cargar un modelo**, mostrado en la Figura 30. Este se realiza cuando el usuario ha cargado los datos y pulsa sobre el botón correspondiente a la carga del modelo seleccionado. Se asocia el modelo predictivo pertienente, guardado en el sistema, a una variable presente en Modelo.

En la Figura 31, se muestran los pasos necesarios para entrenar el modelo predictivo seleccionado. Lo que se hace es volver a crear el modelo predictivo con las mismas características que el anterior y entrenarlo para hallar finalmente la tasa de acierto con el conjunto de datos de test, que será la última temporada jugada.

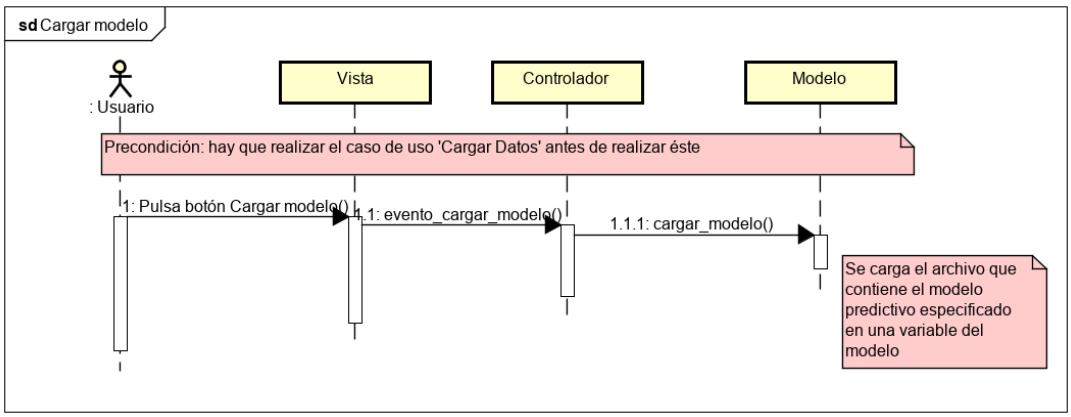

Figura 30: Diagrama de secuencia para *Cargar un modelo*

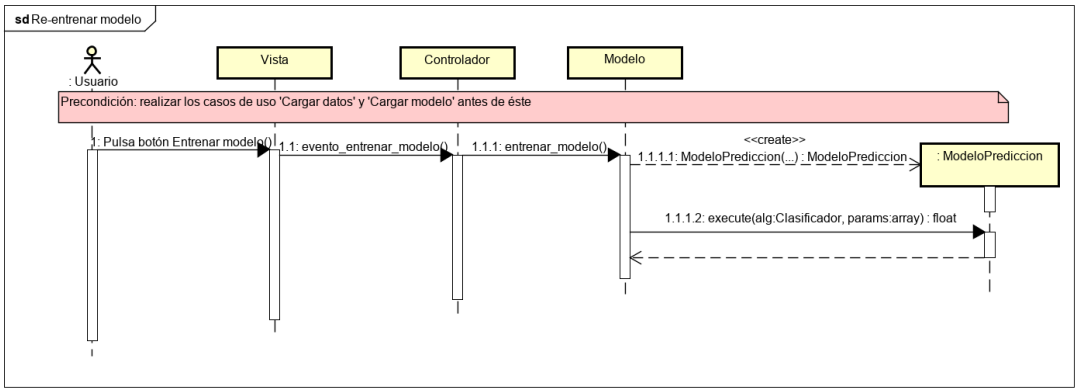

Figura 31: Diagrama de secuencia para *Entrenar el modelo*

Por último, queda la secuencia de pasos seguida en el caso de uso correspondiente a **Hallar predicciones**. En este caso, se comprobará que los datos introducidos por el usuario son correctos, que existe el partido especificado mediante el equipo local, visitante y la fecha. Cuando se tiene esto, se ejecutará la función *pred_function* de la clase correspondiente al modelo predictivo, y se mostrarán las probabilidades de victoria de cada equipo y el equipo que venció en la realidad, en caso de ser un partido ya disputado. Se tendrán en cuenta los partidos de la última temporada para predecir, sin tomar los de las primeras jornadas por las razones explicadas anteriormente. El diagrama de secuencia es el de la Figura 32.

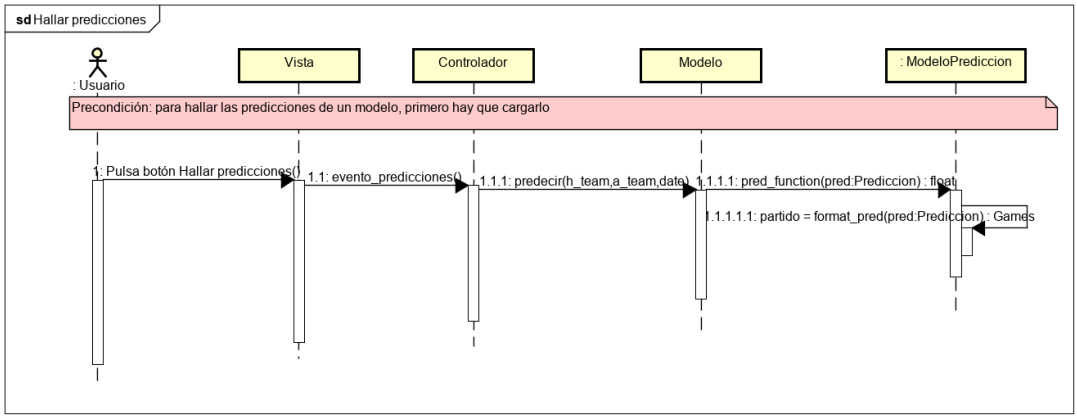

Figura 32: Diagrama de secuencia para *Hallar las predicciones*

5 Revisión de la planificación del proyecto

La planificación proporcionada en la Sección 2 ha sido seguida de la forma más realista posible durante el desarrollo del proyecto. Evidentemente, al tratarse de una estimación algunas tareas no se han seguido de la misma manera en las que estaban planteadas o no han requerido el tiempo indicado. En ocasiones, debido a ciertos inconvenientes no se han podido desarrollar cómo se pensaba, especialmente, en la parte referida a los modelos predictivos.

En principio, se pensaba realizar un solo experimento, tomando varias temporadas como entrenamiento y realizando el test sobre la última, pero por la falta de resultados buenos, se optó por realizar enfoques diferentes. El primero, basado en la adquisición de los datos diferenciando entre las actuaciones locales y las visitantes, y el otro tomando solo los datos de una única temporada. Además, se han ido añadiendo algunas técnicas de aprendizaje. Concretamente, el Naive Bayes se planteó al principio, pero por dificultades en la hora de implementarlo en el código se dejó para más adelante, cuando ya sí se consiguió introducir como una técnica más. Este algoritmo supuso la introducción de nuevas técnicas de estandarizado, ya que hasta entonces sólo se había planteado el uso de la normalización, pero con el Naive Bayes ha habido que introducir la posibilidad de escalar los datos.

Otro de los puntos donde no se ha estimado correctamente la duración y la organización ha sido la manipulación de los datos. Es una tarea que ha habido que revisar constantemente, por la introducción de nuevas variables o la forma en la que se querían obtener los datos para poder alimentar los modelos de predicción. Uno de los puntos analizados en este ámbito ha sido el hecho de mantener la diferencia de puntos total o por partido, como esta variable otras tantas.

En cuanto al tiempo dedicado a cada una de estas tareas, se ha llevado la duración de cada una de éstas a lo largo de los meses que ha durado el proyecto. Las tareas de color azul, referidas a la planificación y toma de contacto con el proyecto, han consumido unas 20 horas del proyecto. Las referidas a la manipulación de los datos han supuesto uno de los gruesos del trabajo, se han utilizado alrededor de 80 horas en la realización de éstas. Esta tarea, como se explica en el apartado de CRISP, ha sido llevada a cabo en diferentes períodos intentando optimizar el formato de los datos para los modelos predictivos. Para la implementación de estos modelos se han dedicado más de 100 horas, es uno de los puntos más importantes del trabajo, junto con la transformación de los datos. En las tareas relacionadas con la creación y documentación de la herramienta se han gastado unas 50 horas. Por último, en la redacción de la memoria se han utilizado otras 50 horas, de forma aproximada. Para la realización del trabajo se otorgaban 300 horas, las cuales se han cumplido.

6 Resultados y Conclusiones

Este proyecto sirve como referencia a futuros proyectos relacionados con este mismo tema. Se han propuesto muchos clasificadores diferentes, exponiendo los resultados de cada uno de ellos. Además, se han creado varios experimentos diferentes con el fin de esclarecer cuál de ellos puede ser el mejor enfoque para predecir los resultados de los partidos de la NBA y se han propuesto técnicas alternativas para la selección de las variables incluidas en los modelos de predicción.

Para la implementación de la herramienta software, se han guardado cuatro de los modelos estudiados en el primer experimento. Estos son los mejores modelos, ya mencionados, para las técnicas de aprendizaje del **perceptrón multicapa**, **regresión logística**, **random forest** y **regresión lineal**. No se han tenido en cuenta los modelos del segundo experimento porque los errores en la predicción eran mayores. Tampoco se cuenta en este apartado con los modelos del último experimento en el que se utilizan los datos de la última temporada, ya que las conclusiones son un tanto difusas y es difícil seleccionar un momento concreto de la temporada con el que quedarse para entrenar los modelos. Además, la capacidad para predecir partidos del final de la temporada si se escogen sólo las dos primeras semanas como entrenamiento será muy mala.

En general, los resultados de los modelos no son todo lo buenos que se esperaba, y es que si se predijese que todos los equipos locales ganan, se conseguiría una tasa de acierto del 59 %, aproximadamente, aunque la tasa de falsos positivos aumentaría bastante. Esto demuestra que tomar solamente las características relacionadas con el juego de los equipos puede resultar insuficiente a la hora de realizar un análisis de este tipo. Como se ha comentado, hay aspectos más profundos que no se han tenido en cuenta, como pueden ser los traspasos de jugadores a mitad de temporada o el hecho de que los equipos de la parte inferior de la tabla de clasificación busquen las mejores posiciones en el *draft*, pudiendo perder partidos a propósito.

En el último experimento se obtuvo como conclusión que datos más cercanos pueden suponer un mayor porcentaje de acierto, aunque a lo largo de una misma temporada existen muchas dinámicas diferentes en los equipos que hacen que los partidos sean muy difíciles de predecir.

En futuras ampliaciones de este trabajo se podrían incluir datos acerca de los jugadores que han participado en cada partido, o por lo menos el quinteto inicial o la convocatoria, de manera que se pueda predecir el resultado de un partido cuando ya se tienen las alineaciones. Así se podrá intuir qué jugadores aportan más a cada equipo, un factor que puede ayudar mucho ya que las actuaciones de los equipos de la NBA tienen una gran influencia por parte de las individualidades de los jugadores.

Además de la tasa de acierto, que sólo tiene en cuenta si un equipo ganó o no un encuentro, y las curvas ROC, que comparan las tasas de verdaderos positivos y de falsos positivos, se podrían haber tomado otras métricas para discernir si un modelo era bueno o no. Los modelos aportan la probabilidad de que gane cada equipo, se podría

haber usado esto con el conjunto de test hallando, por ejemplo, el error cuadrático medio, por tener más alternativas para hallar la bondad de ajuste de un modelo.

En general, creo que éste ha sido un proyecto muy completo y se ha trabajo de una forma adeacuada y ordenada. Aunque los resultados obtenidos no han cumplido las expectativas, se ha cumplido el objetivo principal, que era el análisis de los datos de la NBA y la realización de modelos predictivos para encuentros de esta competición. Además, se ha creado una herramienta con la poder interactuar. Sería necesario ahondar más en otros factores para conseguir mejores resultados y hacer de la herramienta un sistema más competitivo. Cabe destacar que artículos bibliográficos, como [13], que utilizan enfoques y técnicas similares, como las regresiones vistas en este proyecto, el SVM, el Naive Bayes y el Random Forest, aportan tasas de acierto similares a las de este proyecto.

Referencias

[1] *Glosario de baskteball-reference.* `https://www.basketball-reference.com/about/glossary.html`, Fecha de acceso: 16-04-2020.

[2] Berrar, Daniel: *Reference Module in Life Sciences*, capítulo Bayes' Theorem and Naive Bayes Classifier. Tokyo Institute of Technology, Enero 2018, ISBN 9780128096338.

[3] Betfair: *Betting models. data scientist. betfair predictions: The hub.* `https://www.betfair.com.au/hub/tools/models`.

[4] Crespo, Yania: *Introducción al diseño de software y arquitectura del software.* Planificación y Diseño de Sistemas Computacionales, Universidad de Valladolid: Escuela de Ingeniería Informática de Valladolid, Fecha de acceso: 15-06-2020.

[5] Fernández Temprano, Miguel Alejandro y Luis Ángel García Escudero: *Análisis en componentes principales*, Mayo 2020. Análisis de Datos/Análisis Multivariante, Universidad de Valladolid: Facultad de Ciencias.

[6] González, Carlos Alonso: *SVM: Máquinas de vectores soporte.* Técnicas de Aprendizaje Automático: Clasificación no lineal con modelos lineales, Universidad de Valladolid: Escuela de Ingeniería Informática de Valladolid, Fecha de acceso: 24-04-2020.

[7] Kirkwood, Chris: *NBA Playoffs Explained*, Marzo 2015. `https://nbaexplained.wordpress.com/2015/03/15/nba-playoffs-explained/`, Marzo, 2015, The NBA Explained, Fecha de acceso: 06-05-2020.

[8] Kirkwood, Chris: *NBA Standings Explained*, Marzo 2015. `https://nbaexplained.wordpress.com/2015/03/05/nba-standings-explained/`, Marzo, 2015, The NBA Explained, Fecha de acceso: 06-05-2020.

[9] Konefał, Marek, Pawel Chmura, Edward Kowalczuk, Marcin Andrzejewski y Jan Chmura: *The impact of players' motor skills on match performance in top german Bundesliga teams.* Trends in Sport Sciences, 4:185–190, Diciembre 2015. Proyecto: Identification and evolution physical, technical and tactical efficiency in modern soccer.

[10] Liu, Guangying, Hua Sun, Wanjian Bai, Hongmei Li, Zhigang Ren, Zhongde Zhang y Lingxia Yu: *A learning-based system for predicting sport injuries.* MATEC: Web of Conferences, Enero 2018.

[11] Marsland, Stephen: *Machine Learning: An Algorithmic Perspective.* Editorial Chapman & Hall/CRC by Taylor & Francis Group, segunda edición, 2014, ISBN 1466583282.

[12] Oliver, Dean: *Basketball on paper: rules and tools for performance analysis.* Potomac Books, Inc., 2011.

[13] Richardson, Lee, Daren Wang, Chi Zang y Xiaofeng Yu: *NBA predictions*. Artículo publicado en Carnegie Mellon Universitiy, 2014.

[14] Sommerville, Ian: *Software Engineering: Global Edition*. Editorial Pearson, 10ª edición, 2016.

[15] Sports, Nielsen: *Spain Sports Review*, Diciembre 2018. `https://nielsensports.com/wp-content/uploads/2015/02/Spain-Sports-Review-2018-Nielsen-Sports.pdf`, Fecha de acceso: 15-05-2020.

[16] Stuffer NBA: *Team evaluation metrics*, 2007. `https://www.nbastuffer.com/analytics-101/team-evaluation-metrics/`, Fecha de acceso: 29-04-2020.

[17] Wirth, R. y Jochen Hipp: *CRISP-DM: Towards a standard process model for data mining*. Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining, Enero 2000.

7 Anexos

7.1 *Box score*

MP	Minutos jugados	**ORB**	Rebotes ofensivos
FG	Tiros de campo encestados	**DRB**	Rebotes defensivos
FGA	Tiros de campo intentados	**TRB**	Rebotes totales
FG %	Porcentaje de tiros de campo encestados	**AST**	Asistencias
3P	Tiros de 3 puntos encestados	**STL**	Robos
3PA	Tiros de 3 puntos intentados	**BLK**	Tapones
3P %	Porcentaje de tiros de 3 puntos encestados	**TOV**	Pérdidas
FT	Tiros libres encestados	**PF**	Faltas personales
FTA	Tiros libres intentados	**PTS**	Puntos

7.2 Código y datos

Para la implementación del trabajo se ha utilizado el lenguaje Python, por sus muy variadas posibilidades para la creación de algoritmos predictivos y por su facilidad para la creación de una interfaz gráfica, para la que se ha usado el módulo *tkinter*. El proyecto está realizado mediante el paradigma de programación orientado a objetos. Se puede acceder al proyecto en el repositorio de drive del siguiente enlace, en el que se encuentran todos los datos con los que se trabaja, o en el de gitlab, enlace, dónde están solo los datos referidos a los partidos que usan los modelos predictivos. En ambos están la implementación en Python, una carpeta de datos y un archivo *astah* con los gráficos propios del apartado de Ingeniería de Software (Sección 4).