

|  |
| --- |
| **Universidad Internacional de La Rioja**  **Escuela Superior de Ingeniería y Tecnología**  **Máster Universitario en Análisis y Visualización de Datos Masivos** |
|  |
| Estudio de nuevas variables para el cálculo de goles esperado en un partido  de futbol. |

**Trabajo Fin de Máster**

**Tipo de trabajo:** Técnicas estadísticas y de inteligencia artificial; y su aplicación para el análisis masivo de datos

**Presentado por:** Ruiz Guevara, Sergio

**Director/a:** Díaz Benito, César Orlando

**Resumen**

**Nota:**

**Palabras Clave:**

**Abstract**

**Keywords:**

**Índice de contenidos**

[1. Introducción 8](#_Toc82281237)

[1.1 Justificación 8](#_Toc82281238)

[1.2 Planteamiento del trabajo 9](#_Toc82281239)

[1.3 Estructura de la memoria 9](#_Toc82281240)

[2. Contexto y estado del arte 11](#_Toc82281241)

[2.1 Contexto 11](#_Toc82281242)

[2.2 Estado del arte 14](#_Toc82281243)

[3. Objetivos concretos 19](#_Toc82281244)

[3.1. Objetivo general 19](#_Toc82281245)

[3.2. Objetivos específicos 19](#_Toc82281246)

[4. Metodología del trabajo 20](#_Toc82281247)

[4.1. Obtención de los datos 20](#_Toc82281248)

[4.1.1. StatsBomb Open-Data 21](#_Toc82281249)

[4.1.2. WyScout Soccer match event dataset 22](#_Toc82281250)

[4.2. Preparación del *dataset* 23](#_Toc82281251)

[4.2.1. Importación 23](#_Toc82281252)

[4.2.2. Creación de los *datasets* de tiros 24](#_Toc82281253)

[4.2.3. Combinación de *datasets* 26](#_Toc82281254)

[4.3. *Dataset* utilizado 29](#_Toc82281255)

[4.4 Diseño del modelo 32](#_Toc82281256)

[4.4.1 Métricas utilizadas 32](#_Toc82281257)

[4.4.2 Regresión Logística 35](#_Toc82281258)

[4.4.3 LightGBM 39](#_Toc82281259)

[4.4.4 XGBoost 43](#_Toc82281260)

[4.4.5 Random Forest 44](#_Toc82281261)

[5. Desarrollo específico de la contribución 46](#_Toc82281262)

[5.1 Análisis de las nuevas variables 46](#_Toc82281263)

[5.2 Análisis de los modelos obtenidos 46](#_Toc82281264)

[5.3 Comparación con otros modelos 46](#_Toc82281265)

[6. Conclusiones y trabajo futuro 47](#_Toc82281266)

[6.1. Conclusiones 47](#_Toc82281267)

[6.2. Líneas de trabajo futuro 47](#_Toc82281268)

[7. Bibliografía 48](#_Toc82281269)

[Anexos 58](#_Toc82281270)

**Índice de tablas**

[Tabla 1: Partidos y temporadas de cada competición en los datos de StatsBomb Open-Data. 21](#_Toc82281230)

[Tabla 2: Partidos y temporadas de cada competición en los datos de WyScout Soccer match event dataset. 22](#_Toc82281231)

[Tabla 3: Ejemplo de creación de variables dummies a partir de una variable categórica (Body\_type). 37](#_Toc82281232)

[Tabla 4: Lanzamientos, goles y proporción de los datos de entrenamiento y de test para cada dataset del modelo de regresión logística. 37](#_Toc82281233)

[Tabla 5: Lanzamientos, goles y proporción de los datos de entrenamiento y de test para cada dataset del modelo de LightGBM. 42](#_Toc82281234)

**Índice de figuras**

**No se encuentran elementos de tabla de ilustraciones.**

**Glosario**

TFM Trabajo de Fin de Master

xG *Expected goals* o goles esperados

df *Dataframe*

CV *Cross-validation* o validación cruzada

# 1. Introducción

El presente Trabajo de Fin de Master (TFM) corresponde a la titulación del Máster Universitario en Análisis y Visualización de Datos Masivos de la Escuela Superior de Ingeniería y Tecnología (ESIT) de la Universidad Internacional de La Rioja (UNIR). Este TFM, así como todo el Máster ha sido realizado de manera online.

## 1.1 Justificación

En la actualidad palabras como *big data*, inteligencia artificial o análisis de datos son cada vez más usadas en nuestro día a día. Si bien no siempre se hace un uso correcto está claro que son conceptos cada vez más conocidos y usados en todo aquello que nos rodea. Las empresas desde hace años están usando estas tecnologías para mejorar sus negocios. En el mundo del deporte no es diferente.

La estadística avanzada es muy usada en el mundo del deporte profesional tanto por entrenadores como ojeadores o periodistas para entender mejor que ha ocurrido en un partido o a lo largo de una temporada. Si bien los deportes americanos fueron los pioneros en este sentido pocos años después el mundo del futbol también empezó a usar esta tecnología. En la faceta deportiva se usa para mejorar el rendimiento del equipo como para encontrar jugadores para contratar (Mena Camino, 2021), así como evitar lesiones, aunque su uso dentro de los clubes también sirve para mejorar los ingresos provenientes de los fans o la optimización de operaciones internas como los dispositivos de seguridad o el uso de los recursos materiales (De Torres, 2021). También lo usan agentes externos como son las casas de apuestas para decidir sus cuotas o los medios de comunicación para representar aquello que ocurre en el terreno de juego (Pérez, 2017).

Dentro de la estadística avanzada los goles esperados (xG del inglés *expected goals*) son una de las medidas existentes más populares. Esta variable le da a cada disparo realizado un porcentaje de probabilidad de ser gol. El porcentaje depende de distintas variables como son la posición, la distancia o la parte del cuerpo usada. Si estos porcentajes de cada disparo son sumados a lo largo de un partido se obtiene el resultado esperado en ese momento.

Este trabajo busca encontrar nuevas variables a esta medida para darle una mayor precisión. En este trabajo se estudian variables que están relacionadas con el factor mental, se estudia si el marcador, el minuto y si se juega de local o visitante entre otras variables son fundamentales a la hora de conseguir marcar y si estas variables se pueden trasladar al modelo de manera numérica. También se estudia si la repetición de acciones parecidas afecta, es decir, si la segunda vez que se realiza un disparo de características parecidas se ve alterado por el anterior y de qué manera.

La obtención de un mejor modelo de cálculo puede ser de gran ayuda para mejorar el análisis de los datos por parte de los distintos analistas en el mundo del futbol, así como para que estos puedan predecir mejor futuros encuentros y mejorar así su rendimiento.

## 1.2 Planteamiento del trabajo

Como se ha explicado anteriormente este trabajo busca obtener nuevas variables para el cálculo de los xG. En base a los datos que se tienen de un encuentro se usa como base el cálculo actual que usan las herramientas de análisis de datos.

A partir de ese modelo inicial se añaden variables al cálculo para comprobar si existe correlación con lograr marcas y si mejora el modelo previo. En caso afirmativo se añade al modelo y se le añade una nueva variable. Una vez se hayan probado todas las variables posibles el modelo resultante es comparado con el inicial para hacer estudiar qué nivel de mejora se ha obtenido.

Para extraer el cálculo de nuevas variables y comparar el nuevo modelo con el actual se ha utilizado el lenguaje de programación Python.

## 1.3 Estructura de la memoria

Esta memoria se estructura en seis capítulos de la siguiente forma:

En el primer capítulo se realiza una introducción del trabajo explicando que motivos justifican el mismo, así como el planteamiento realizado.

En el segundo capítulo se explica la situación actual en cuanto a la estadística avanzada en el mundo del futbol y especialmente en el caso de los goles esperados en un partido a partir de la información recogida. Se muestran estudios previos realizados y la historia de esta rama hasta la actualidad para comprender mejor el contexto del trabajo.

En el tercer capítulo se explican los distintos objetivos de este trabajo, así como la metodología usada para lograrlos.

Posteriormente en el cuarto se presenta el desarrollo de la contribución realizada en el trabajo. Se explica los cálculos realizados, y los algoritmos creados en base a estos cálculos y los resultados obtenidos.

En el capítulo número cinco se muestran las conclusiones del trabajo y resumiendo el trabajo realizado, que objetivos han sido logrados satisfactoriamente y que posibles futuras líneas de trabajo han aparecido al finalizar este TFM.

Por último, en el capítulo seis aparece la bibliografía usada durante todo el trabajo.

# 2. Contexto y estado del arte

En este apartado se explican tanto la historia del análisis estadístico en el mundo del futbol así como los estudios sobre la probabilidad de marcar gol durante un partido para así poder poner en contexto el trabajo. También se comentan los estudios más recientes relacionados con los goles esperados para conocer el estado actual del arte.

## 2.1 Contexto

El análisis estadístico en el mundo del futbol comienza en la década de los años 50 con Charles Reep, un contable de las fuerzas aéreas británicas que empezó a anotar todas las jugadas ofensivas de los partidos a los que atendía (Weiss, 2020) . A partir de sus anotaciones, sus vivencias y el análisis realizado en más de 600 partidos y 4 copas del mundo durante 15 años terminó escribiendo el primer *paper* para una revista científica de estadística con su escrito “Skill and chance in association football” (Reep et al., 1971) donde se explicaba estadísticamente como las opciones de anotar eran mayores cuanto menor fuese el número de pases realizados (Williams, 2020) (Friends of Tracking, 2020). Esta idea era contraria a la filosofía futbolística del momento donde el espectador prefería ver gran cantidad de pases y regates en vez de un futbol directo por lo que la opinión pública británica en un primer momento estuvo en contra de sus teorías y nunca obtuvo ningún reconocimiento. Una década más tarde, entre finales de los setenta y principios de los ochenta, el tiempo le dio la razón y el futbol inglés pasó a ser el dominador a nivel europeo debido a su futbol directo y de pocos pases, un futbol que ha sido el característico del país hasta hace pocos años.

Este estudio fue el primero en crear un modelo probabilístico, pero todavía estaba lejos de lo que hoy se conoce como goles esperados. No fue hasta 1997 cuando el propio Charles Reep junto a Richard Pollard, a partir del registro de todos los disparos del Mundial de Futbol de 1986, publicaron “Measuring the effectiveness of playing strategies at soccer” (Pollard & Reep, 1997) cuando se publicó el primer modelo con diversas variables usando una regresión lineal para predecir la probabilidad de anotar un gol mediante un disparo. Este modelo fue el inicio de todos los estudios posteriores y es considerado el precursor de los goles esperados pese a haber cierta discusión puesto que el termino goles esperados fue usado por primera vez en 1993 en el artículo “The Effect of an Artificial Pitch Surface on Home Team Performance in Football (Soccer)” (Barnett & Hilditch, 1993) aunque su estudio se centraba en si jugar en campos de hierba artificial daba ventaja al equipo local.

Según el modelo la probabilidad de anotar dependía de algunas variables que todavía se usan actualmente como son la distancia horizontal entre la posición del disparo y el punto de penalti, el ángulo entre el lugar donde se produce el disparo y el palo más cercano, la distancia del defensa más cercano o si era una jugada a balón parado (córner, falta) o una jugada abierta. Otras variantes que más adelante se dejaron de usar y que este primer modelo si tenía en cuenta era la posición en que se obtenía el balón en un primer momento, el número de veces que el futbolista tocaba el balón antes de disparar.

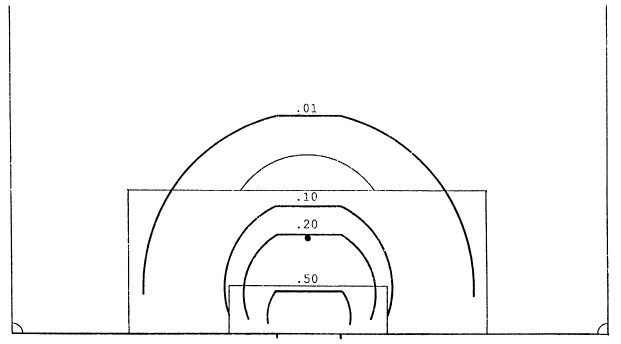


Ilustración 1: Contornos de probabilidad de anotar en tiros lanzados en juego abierto y con el defensa más cercano a menos de una yarda. (Pollard & Reep, 1997)

Con la entrada en siglo XXI hubo una gran mejora tecnológica para las empresas de captación de datos deportivos gracias al *machine learning* y a la captura óptica de datos espaciales que dejaron atrás la captación manual de los datos, estas mejoras hicieron crecer la industria de las estadísticas deportivas convirtiéndolo en un negocio multimillonario. Al factor tecnológico también se le unió el aumento de la popularidad del análisis estadístico en el mundo del deporte (principalmente en Estados Unidos) gracias a la novela “Moneyball: The Art of Winning an Unfair Game” (Lewis, 2004) que posteriormente llegó a la gran pantalla con la película “Moneyball” (Miller, 2011) que explica la historia real de como un equipo de béisbol norteamericano mejoró sus resultados al empezar a basar sus contrataciones en métodos estadísticos. Gracias a todo ello el número de investigaciones académicas sobre métricas avanzadas creció exponencialmente. (Williams, 2020)

En 2004 Richard Pollard usando como base su publicación de 1997 publicó “Applications of logistic regression to shots at goal in association football: calculation of shot probabilities, quantification of factors and player/team” donde hizo el mismo estudio para los disparos registrados en el Mundial de Futbol de 2002 donde investigó 12 posibles variables de las cuales determinó que solo 5 factores era significativos: la distancia a la portería, el ángulo desde la portería, si el jugador que efectuó el tiro estaba al menos a 1m del defensor más cercano, si el tiro fue precedido inmediatamente por una centro o no y el número de jugadores de campo entre el lanzador y el arco. El autor esperaba que la posición en la que se ubicaba el portero a la hora del lanzamiento fuera otra variable, pero tuvo que descartar tal hipótesis.

Si bien durante los últimos años ha habido un gran número de investigaciones en este campo la más importante sin duda ha sido la de Sam Green. Sam Green es un analista de datos de Opta, una de las principales empresas de estadísticas deportivas, y es conocido por ser el creador de la formulación moderna de los xG (Green, 2012). Su modelo trata de dar un % de acierto a cada disparo a partir de una regresión logística donde las variables de entrada son las siguientes (Gregory, 2017):

* Tipo de jugada (juego abierto, tiro libre directo, tiro de esquina, saque de banda, rebote, solo contra el portero…)
* Tipo de asistencia (balón largo, centro, pase corto…)
* Parte del cuerpo usada (cabeza, pierna u otra)
* Distancia a portería
* Ángulo visible de la portería
* Gran oportunidad (variable subjetiva)

El ejemplo más básico es el lanzamiento de penalti que siempre es contabilizado como 0,76xG. Al sumar los % de cada disparo durante un partido, una sucesión de partidos o toda una temporada se puede obtener el número de xG que ha obtenido un jugador o un equipo durante el periodo elegido.

Para crear el modelo la empresa Opta usó los datos de 300.000 disparos y, si bien el modelo no es público por motivos empresariales, se conoce que es redefinido cada cierto tiempo con los nuevos disparos que han sido recopilados.

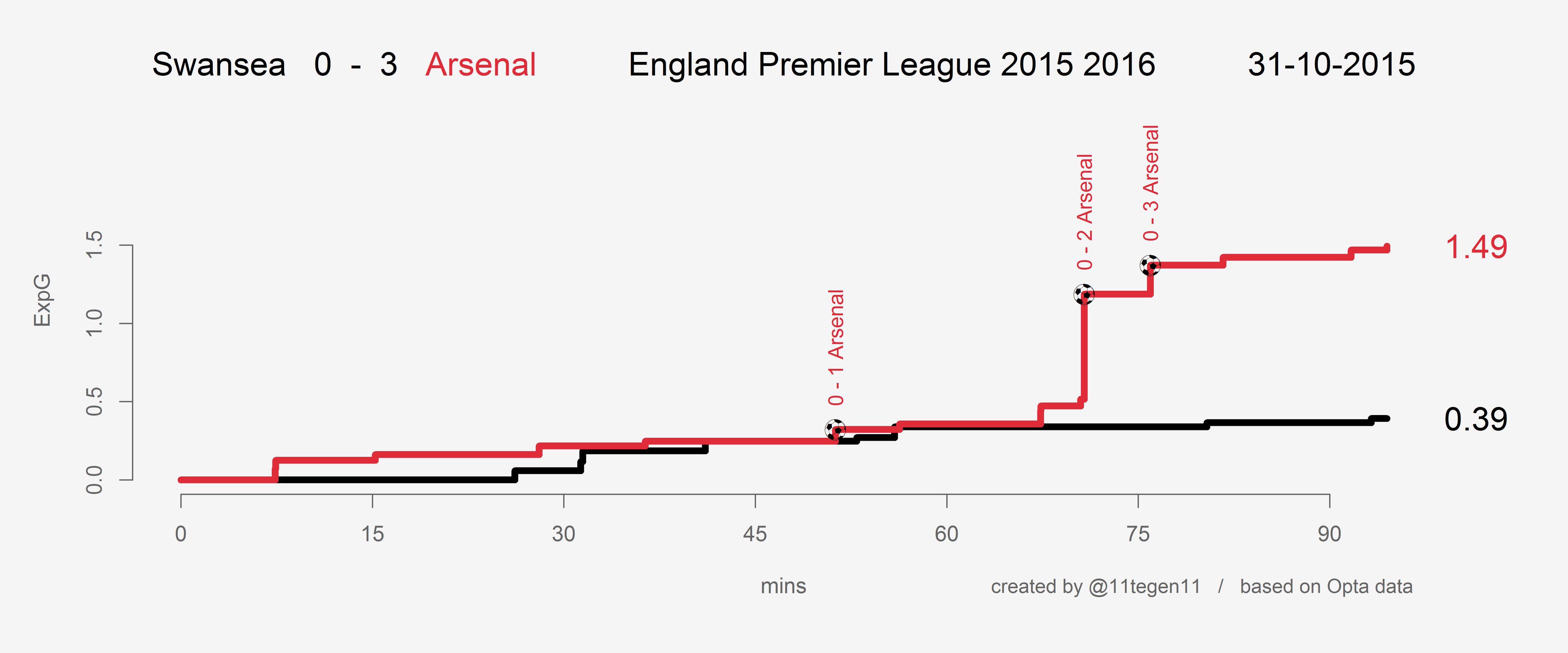


Ilustración 2: Gráfico de la acumulación de los xG durante un partido de futbol.(11tegen11, 2015)

Durante los últimos 10 años se han generado una gran cantidad de modelos usando como base la idea de Sam Green y haciendo algunas modificaciones en las variables usadas. (Brener, 2017) Cada empresa de métricas deportivas tiene su modelo e incluso existen modelos *open-source*, cabe destacar en la mayoría de los casos las diferencias entre ellas no son grandes.(Martinez Arastrey, 2018)

En agosto de 2017 la cadena británica BBC utilizó por primera vez los xG para su resumen estadístico de los partidos de la liga de futbol inglesa en su programa “Match of the Day” haciendo crecer todavía más su popularidad y llegando a ser conocido por el público general.(Coronis, 2021) Actualmente la mayoría de equipos de futbol profesional tienen su equipo de analistas de datos e incluso algunos de ellos han hecho grandes inversiones en el sector.(Andersen, 2021)

## 2.2 Estado del arte

Como se ha comentado previamente en la actualidad existen una gran cantidad de modelos para calcular los xG. Algunos de ellos han sido comparados entre sí (Mackay, 2017) y se pueden apreciar ciertas diferencias. Una de las principales es el uso de la variable *Big Chance* (o Gran Oportunidad) que utiliza la empresa Opta y que añade precisión al modelo. En este aspecto hay ciertas dudas sobre si usarla al ser una variable subjetiva basada en la opinión de la persona que recoge los datos y la decisión de contabilizar un disparo como gran oportunidad puede ser influenciada por el propio resultado del disparo. En la Ilustración 3 se puede comprobar como todos los disparos con más de 0,3xG fueron considerados grandes ocasiones.

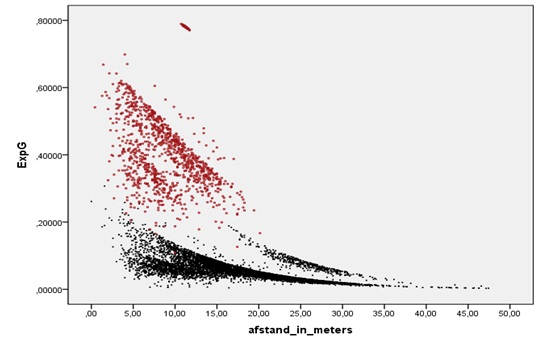


Ilustración 3: Relación entre xG y distancia a porteria diferenciando por Big Chance o no.(Mullenberg, 2016)

Para no depender de una variable subjetiva como es la de Big Chance y gracias a la mejora en la captura de datos que ha habido en los últimos años la empresa SportsBomb ha elaborado un modelo propio a partir de capturar la posición de todos los futbolistas en el momento del disparo, así como si están en movimiento, parados o en el suelo. De esta manera se puede evaluar cuantos futbolistas hay entre el jugador que dispara y la portería otorgando un alto xG si no hay ningún rival en la portería pese a disparar desde lejos y un bajo xG si hay una gran cantidad de futbolistas para bloquear el disparo pese a que sea este más cercano. En la Ilustración 4 se puede comprobar la diferencia entre modelos.



Ilustración 4: Comparativa del mismo disparo entre el modelo de xG de UnderStat (0,93xG) con el de SportsBomb (0,29xG).(Goodman, 2018b)

Los creadores del modelo explican que no modifica excesivamente las predicciones al analizar grandes cantidades de disparos, pero es útil para corregir casos atípicos, dejar de usar variables sesgadas y mejorar la precisión en muestras pequeñas como pueden ser un único partido. Mas allá de la medida explicita de los xG este nuevo modelo también ayuda a analizar otras métricas del juego como son las relacionadas con la buena colocación de la defensa o del portero en los momentos del disparo.

La propia empresa SportsBomb posteriormente ha añadido otra variable al cálculo de los xG, en este caso se trata de la altura en la que se realiza el disparo o *Shot Impact Height* (SIH). Hay que destacar que la empresa ya tenia incorporado en su modelo una variable para saber si el disparo venia precedido por un pase a ras de suelo o un pase alto por lo que cierta medida ya se contemplaba anteriormente, pero al tener un mayor abanico de opciones el nuevo modelo da mayor valor a disparar un balón que va a ras de suelo que antes mientras que los xG se ven reducidos en casos más atípicos donde el futbolista debe hacer un movimiento antinatural para golpear y que requieren de una mayor técnica como se muestra en la Ilustración 5.(Knutson, 2020)

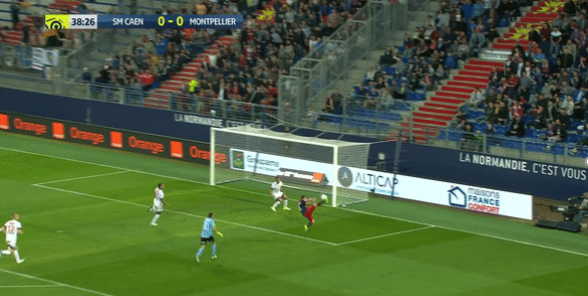


Ilustración 5: Ejemplo de disparo complicado por la altura del balón al golpear. Los xG se reducen en el nuevo modelo de 0,65xG a 0,35xG.(Knutson, 2020)

Otros tipos de modelos creados han tenido en cuenta variables como la calidad específica del futbolista en cuestión, su calidad ofensiva en el caso del atacante y defensiva por parte de la defensa o el portero. Ejemplos de modelos que usan la calidad de los futbolistas son “Improving the estimation of outcome probabilities of football matches using in-game information” (Noordman, 2019) y “Expected Goals in Soccer: Explaining Match Results using Predictive Analytic” (Eggels, 2016). En ambos casos se usaron como valores para determinar las habilidades de los futbolistas los atributos de los jugadores en el videojuego FIFA de Electronic Arts.

Si bien en ambos trabajos se explica que esta variable mejora la precisión del modelo cabe destacar tres aspectos: por un lado, igual que ocurría con la variable Big Chance en este caso se usa una variable subjetiva para medir la habilidad de los futbolistas, no todo el mundo estará de acuerdo en que X jugador sea mejor que Y disparando o parando disparos. Por otro, al tener en cuenta la habilidad del futbolista se elimina poder analizar los futbolistas entre sí puesto que para un mismo disparo para la misma ocasión dos futbolistas acumularán distinto grado de xG. La propia medida de xG debería ser la que permita hacer el análisis de que futbolista es mejor o peor y para ello los disparos deben evaluarse de la misma manera para todos. El tercer aspecto es que para futbolistas que no aparecen en el videojuego no se tendrá conocimiento de esa variable.

En este último trabajo también se usó el marcador como variable, es decir, si el equipo del lanzador iba ganando, perdiendo o empatando en ese momento. Esta variable es interesante puesto que el marcador puede afectar psicológicamente a ambos equipos, el trabajo, pero, no explica de qué manera afecta a los xG esta variable y para jugadas parecidas en que escenarios se marcan más goles. También en “Un nuovo modello di Expected Goals” (Giacobbe, 2016) se muestra un modelo donde también se tiene en cuenta el resultado para calcular los xG, el autor muestra gráficamente como para una misma distancia la probabilidad de marcar es más alta cuando se tiene una ventaja de más de 2 goles que cuando el resultado esta empatada y en caso de perder por 2 goles o más las probabilidades son aún más bajas. El mismo modelo analiza el factor campo y muestra como para disparos igual de lejanos la probabilidad de marcar es mayor en el caso de los equipos locales respecto de los visitantes tal y como se aprecia en la Ilustración 6.

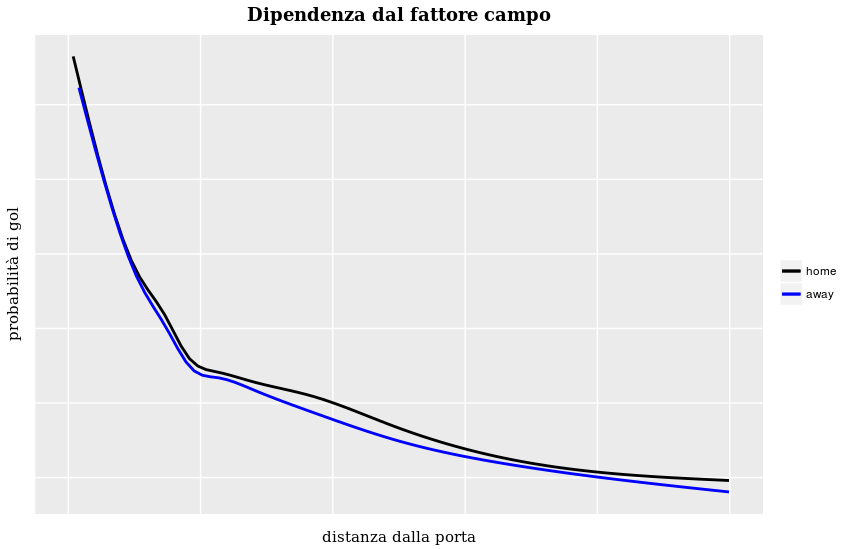


Ilustración 6: Diferencia entre la probabilidad de marcar si eres local o visitante para cada distancia a puerta.(Giacobbe, 2016)

Es interesante destacar que también se han realizado modelos de xG a partir de redes neuronales como son los descritos en la entrada de blog “Using Neural Networks to calculate Expected Goals” (Blum, 2017) y las tesis “Applying Machine Learning Methods to Predict the Outcome of Shots in Football” (Hedar, 2020) y “Creating a Model for Expected Goals in Football using Qualitative Player Information” (Madrero, 2020) donde muestran conseguir una mayor precisión en los datos que con un modelo generado a partir de la regresión logística aunque en ambos casos la diferencia no es especialmente significativa y se pierde cierta transparencia en el proceso que no ocurre con la regresión logística. En cuanto a si es mejor utilizar redes neuronales u otros modelos de machine learning como pueden ser Random Forest o XGBoost David Sumpter, *Data Scientist* del equipo de futbol sueco Hammarby IF, dice lo siguiente: “Soy escéptico si alguno de ellos (los modelos que no son la regresión logística) son necesarios para este problema. Es mejor utilizar la regresión lineal con las características seleccionadas correctamente” (Friends of Tracking, 2020).

La métrica de los xG tiene en cuenta toda la información posible en el momento previo al disparo y sirve para analizar a los atacantes. Similar a esta métrica, pero para analizar la habilidad de los porteros se ha creado la métrica xGOT (expected goals on target) que solo analiza aquellos disparos que van entre los tres palos (terminan en parada del portero o son gol) y que tiene en cuenta información sobre el propio disparo. Las variables que se tienen en cuenta en este caso son la trayectoria, la velocidad del disparo y las coordenadas de la portería donde fue el disparo. Un disparo pese a tener un xG bajo por la situación en la que se dispara puede tener un alto xGOT si la ejecución es buena, tal como muestra la Ilustración 7. (Goodman, 2018a; Whitmore, 2021)

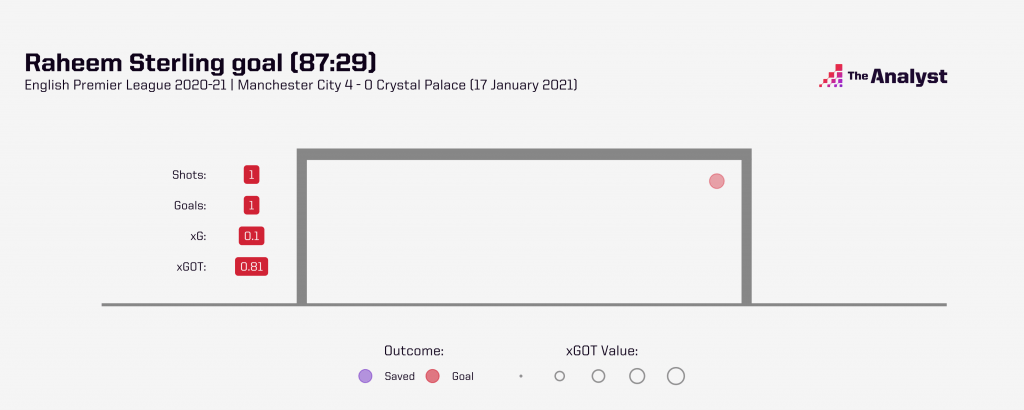
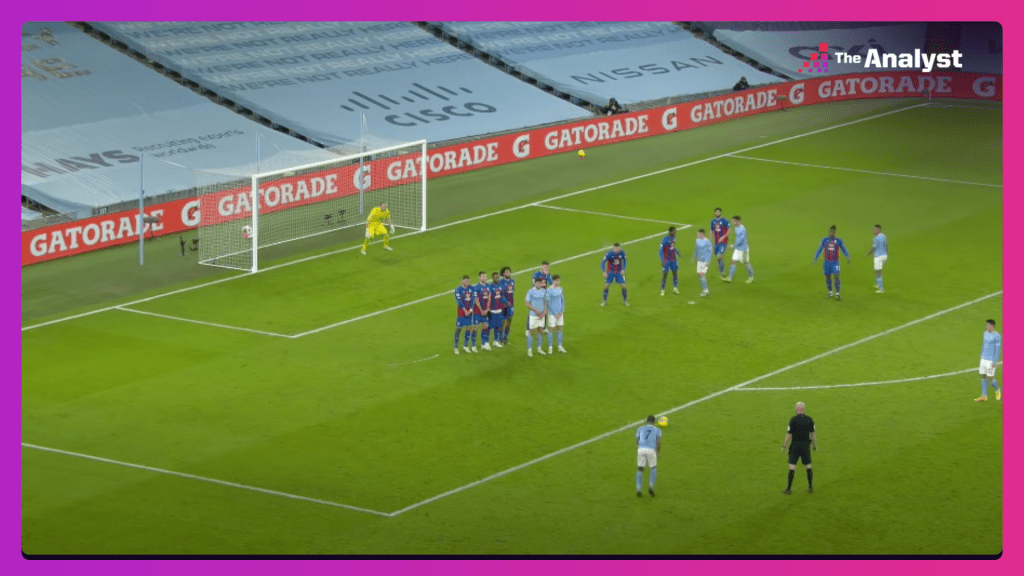


Ilustración 7: Ejemplo de disparo con bajo xG (0,1 xG) y alto xGOT (0,81 xGOT) al ser un gran disparo muy bien colocado pese a la dificultad.

# 3. Objetivos concretos

## 3.1. Objetivo general

El objetivo de este TFM consiste en generar un modelo para el cálculo de los xG en el futbol que añada nuevas variables con el objetivo de obtener una mayor precisión. Este nuevo modelo debe ser válido para poder obtener el dato tanto al finalizar el partido como en tiempo real.

## 3.2. Objetivos específicos

1. Estudiar y conocer el actual estado del arte en el campo del estudio probabilístico de un disparo. Como han evolucionado los modelos probabilísticos durante los años y cuáles son los últimos estudios en este campo.
2. Obtener todos los datos posibles relacionados con un disparo en un partido de futbol, así como las circunstancias en las que ocurre el disparo.
3. Analizar y decidir qué variables son útiles para los cálculos a realizar.
4. Desarrollar una metodología para comprobar mediante cálculos si las variables escogidas mejoran el modelo probabilístico existente.
5. Evaluar el modelo final obtenido con el inicial y comparar en qué grado se ha mejorado el modelo. Evaluar posibles inconvenientes de este nuevo modelo.

# 4. Metodología del trabajo

En esta sección se especifica la metodología empleada para lograr un modelo de xG propio con nuevas variables, el análisis de las nuevas variables y el del propio modelo. La estructura es la siguiente:

* **Obtención de los datos:** Se explica el origen de los datos utilizados en el trabajo, el formato de origen de los datos y los motivos por los que han sido utilizados estos datos.
* **Preparación del *dataset*:** Se detalla los pasos realizados para transformar los datos de origen en el *dataset* utilizado para realizar el modelo.
* **Dataset utilizado:** Se detalla los datos utilizados finalmente y se comparan con los utilizados en otros estudios.
* **Diseño del modelo:** Se expone las técnicas utilizadas para generar el modelo, así como los parámetros utilizados.

El código para implementar la metodología explicada a continuación ha sido escrito en el lenguaje de programación Python 3 mediante Jupyter Notebook. Los principales paquetes utilizados han sido Numpy (Harris et al., 2020; Oliphant, 2006), Pandas (McKinney, 2010; Reback et al., 2020), Matplotlib (Hunter, 2007) , SeaBorn (Waskom, 2021), MPLSoccer (Durgapal & Rowlinson, 2021), SciKit-Learn (Buitinck et al., 2013; Pedregosa et al., 2011), SiciKit-Optimize (Scikit-Optimize, 2016), LightGBM (Ke et al., 2017), XGBoost (Chen & Guestrin, 2016) y Shap (Lundberg & Lee, 2017). El código puede encontrase en GitHub[[1]](#footnote-1).

## 4.1. Obtención de los datos

Tras realizar una búsqueda de *datasets* públicos solo se ha encontrado dos fuentes *open-data* que proporcionen una cantidad de información suficiente para generar un modelo de xG: una de la empresa StatsBomb y otra de WyScout. El resto de las bases de datos futbolísticas solo aportaban datos generales de los partidos como el resultado, el número de disparos, la posesión, etc.

También se ha estudiado la posibilidad de pagar a alguno de los proveedores de datos existentes para obtener una mucho mayor cantidad de los datos, pero el coste era en todos los casos muy elevado puesto que son productos orientados a profesionales del sector (equipos, ojeadores, jugadores…).

### 4.1.1. StatsBomb Open-Data

Los datos provenientes de la empresa StatsBomb han sido obtenidos a partir de repositorio de Github abierto para uso académico y personal (Lawrence et al., 2018/2021). Este repositorio aporta datos en formato JSON (JavaScript Object Notation) de todos los eventos ocurridos en los distintos partidos que hay en el repositorio (ver Tabla 1) así como datos sobre los partidos, las competiciones y las alineaciones de estos.

En los datos sobre eventos se tiene todos los pases, disparos, paradas, faltas, recepciones de balón, córneres… con las coordenadas donde ha ocurrido, el jugador que lo ha realizado, el momento del partido en el cual ha ocurrido, así como información específica sobre cada tipo de evento, por ejemplo, en caso de pase que tipo de pase ha sido. Además, permite relacionar datos entre sí por ejemplo un pasé completado con la recepción de balón del jugador que lo recibe.

En el caso de los disparos también se tiene información sobre el sistema Freeze Frame creado por StatsBomb (StatsBomb, 2021) que a partir de una imagen en el momento del disparo conoce la posición de todos los futbolistas en dicho momento. De esta manera se puede tener una mayor información como el número de jugadores entre el balón y la portería o la cercanía de los rivales.

En total a partir de StatsBombs Open-Data se obtiene un total de:

* 889 Partidos.
* 7 Competiciones.
* 3.198.449 Eventos.

Que provienen de las distintas competiciones y temporadas:

|  |  |  |  |
| --- | --- | --- | --- |
| Competición | Partidos | Temp. | Detalle |
| La Liga | 485 | 16 | Primera división española masculina |
| FA Women's Super League | 195 | 3 | Primera división inglesa femenina |
| FIFA World Cup | 64 | 1 | Trofeo de naciones mundial masculino |
| Women’s World Cup | 52 | 1 | Trofeo de naciones mundial femenino |
| NWSL | 36 | 1 | Primera división estadounidense femenina |
| Premier League | 33 | 1 | Primera división inglesa masculina |
| UEFA Champions League | 14 | 1 | Competición europea de clubes masculina |

Tabla 1: Partidos y temporadas de cada competición en los datos de StatsBomb Open-Data.

### 4.1.2. WyScout Soccer match event dataset

Por su parte los datos de WyScout provienen del repositorio de Figshare abierto también para uso personal y académico (Pappalardo et al., 2019; Pappalardo & Massuco, 2019). En este caso el repositorio ofrece datos de entrenadores, árbitros, jugadores, equipos, competiciones, partidos y eventos. Al igual que con los de StatsBomb los datos están en ficheros JSON.

Los datos sobre eventos son muy parecidos a los de StatsBomb si bien tiene algo menos de detalles sobre estos y no tiene la posición del resto de jugadores durante ningún evento más allá de las coordenadas del futbolista que realiza la acción.

Aunque los datos de StatBomb tienen más información sobre los disparos los datos de WyScout también han sido usados puesto que el nivel de información es suficiente y la cantidad de datos es incluso mayor (ver Tabla 2) por lo que se ha escogido combinar ambos *datasets* para así tener una cantidad mayor de datos pese a que no todos los disparos tendrán la misma cantidad de información.

En este caso WyScout Soccer match event dataset aporta datos de:

* 1941 Partidos.
* 7 Competiciones.
* 3.251.294 Eventos.

Que provienen de las distintas competiciones y temporadas:

|  |  |  |  |
| --- | --- | --- | --- |
| Competición | Partidos | Temp. | Detalle |
| Ligue 1 | 380 | 1 | Primera división francesa masculina |
| Premier League | 380 | 3 | Primera división inglesa masculina |
| Serie A | 380 | 1 | Primera división italiana masculina |
| La Liga | 380 | 1 | Primera división española masculina |
| Bundesliga | 306 | 1 | Primera división alemana masculina |
| FIFA World Cup | 64 | 1 | Trofeo de naciones mundial masculino |
| European Championship | 51 | 1 | Trofeo de naciones europeas masculino |

Tabla 2: Partidos y temporadas de cada competición en los datos de WyScout Soccer match event dataset.

## 4.2. Preparación del *dataset*

Para la preparación del *dataset* final a partir de los datos originales explicados anteriormente se ha aprovechado el código creado por Andrew Rowlinson en su TFM (Rowlinson, 2020b) y al que se puede acceder a partir de su repositorio de Github (Rowlinson, 2020a) como base puesto que los datos de origen son los mismos salvo que en su caso hay unos pocos partidos menos provenientes del repositorio de StatsBomb.

Si bien la mayor parte del código usado en este trabajo para la preparación del dataset ha sido el mismo al de Andrew Rowlinson (Rowlinson, 2020a) se han hecho algunas modificaciones, ya sea por funciones que han quedado obsoletas en nuevas versiones de Python o para añadir nuevos parámetros para ser estudiados y utilizados en la creación de este modelo de xG.

Los pasos seguidos son los siguientes:

### 4.2.1. Importación

Los datos del repositorio de StatsBomb son descargados de manera manual y se convierten en archivos Parquet mediante código para así comprimirlos y poder trabajar posteriormente con ellos de una manera más rápida y eficiente. Al importarlos se obtienen los siguientes *dataframes (df)*:

* df\_competition: 37 entradas
* df\_match: 879 entradas
* df\_lineup: 26794 entradas
* df\_event: 3198449 entradas
* df\_freeze: 277829 entradas
* df\_tactic: 36817 entradas
* df\_related: 6219794 entradas

En el caso del repositorio de WyScout los datos son descargados desde el propio código y posteriormente también se convierten en archivos Parquet. En este caso los *dfs* obtenidos son los siguientes:

* df\_coach: 208 entradas
* df\_player: 3603 entradas
* df\_team: 142 entradas
* df\_competition: 7 entradas
* df\_match: 1941 entradas
* df\_formation: 74098 entradas
* df\_substitution: 11097 entradas
* df\_event: 3251294 entradas

Además, en el caso de WyScout se hacen algunas modificaciones en los nombres de los equipos y las competiciones para así estar escritas de la misma forma que lo están en los datos de StatsBomb. También se comprueba que hay 100 partidos que aparecen en ambos repositorios por lo que se eliminan todos los datos de esos partidos del repositorio de WyScout al tener una menor información de estos.

### 4.2.2. Creación de los *datasets* de tiros

Une vez importados los datos de una manera óptima para trabajarlos el siguiente paso es crear los *datasets* con únicamente los tiros y toda la información posible sobre ellos. En el código original se realiza la modificación de los *df* de eventos originales tanto de WyScout como de StatsBomb para poder conocer las acciones previas a cada disparo y relacionarlas con el mismo. Por ejemplo, si un disparo dice que viene precedido por un pase se busca toda la información sobre ese pase para así añadirla al disparo (altura del pase, tipo de pase…). Se tiene en cuenta la diferencia de tiempo entre eventos para, por ejemplo, poder determinar si un disparo viene de un contraataque o si se puede seguir considerando que la jugada es un córner o una falta si no ha pasado suficiente tiempo desde que se sacó.

Otra característica añadida es conocer si el disparo se realiza con su mejor pierna, en el caso de WyScout se consigue a partir de la información sobre cada jugador que hay en *df\_player* mientras que en el caso de StatsBomb, que no tiene información específica sobre la pierna de cada jugador, se obtiene calculando que pierna es la más usada por cada futbolista en los distintos eventos del *dataset*.

A nivel posicional se estandarizan las coordenadas de los eventos para que se considere que todos los estadios son iguales (en realidad cada estadio tiene unas medidas distintas dentro de unos límites reglamentarios). También se añaden variables como el ángulo respecto al centro de la portería (*middle\_angle*), el ángulo de portería visible, la distancia hasta la portería (*distance\_to\_goal*) y variables que son creadas a partir de estas como *distance\_visible\_angle* y *log\_distance\_to\_goal* (ver Ilustración 1).

Interfaz de usuario gráfica, Aplicación, Gráfico de líneas

Descripción generada automáticamente

Ilustración 8: Cálculo de las distancias y ángulos. (Rowlinson, 2020b)

Además de todas las variables creadas con el código original para este trabajo también se ha añadido nuevas. Entre ellas se ha añadido una columna para saber si el equipo que realiza el disparo es local o visitante. Anteriormente se ha mostrado como el factor campo es importante (ver Ilustración 6) y por ello se quiere añadir al modelo para tenerlo en cuenta y ver qué importancia tiene dentro del modelo.

También se ha añadido el número de jugadores que hay en el campo en el momento del disparo tanto en el equipo que ataca (*players*) como en el equipo rival (*players\_rival*). Ambas variables nos pueden mostrar si los futbolistas logran una mayor confianza al estar con ventaja de futbolistas o si ocurre cuando se está en una situación de desventaja. También una mayor eficiencia con más futbolistas puede ser debido a una peor defensa rival por falta de jugadores en el momento del disparo o durante la creación de la jugada. Para añadir estas variables las expulsiones que aparecen en el *df* de eventos teniendo en cuenta si el futbolista expulsado estaba en el campo en ese momento (puede ser expulsado un jugador que está en el banquillo).

Sumado a las variables ya expuestas también se ha añadido el tipo de competición (*competition\_type*) para diferenciar los partidos de liga de los de torneos con eliminatorias (Mundial, UEFA Champions League…) y ver si hay diferencias a nivel de xG y la jornada en la que se realiza el partido (*match\_week*) para conocer si afecta el momento de la temporada a la hora de la efectividad de los disparos ya sea debido a un menor cansancio al inicio de la competición, un mejor momento de forma en una parte intermedia o un mayor cansancio y una presión mayor en los últimos partidos de la competición.

### 4.2.3. Combinación de *datasets*

Una vez están preparados los *datasets* por separado se unen y se trata de hacer coincidir los nombres de los equipos y de los jugadores a partir del paquete de Python fuzzymatcher (Linacre, 2017/2017) para tener así una única ID para ambos campos. También se añaden las variables obtenidas mediante el *df* de Freeze Features en los disparos de StatsBomb.

Además, en nuestro caso se ha incluido la variable Match Moment (Momento del partido) en la que se separa el minuto del partido en grupos de 15 minutos (del minuto 0 al 15, del 15 al 30…). Con esta variable se quiere estudiar cómo afecta el momento en el que se encuentra el partido en la efectividad de los goles. Por ejemplo, si es posible que se marquen más goles de larga distancia en los primeros momentos por la falta de ritmo del portero o si la efectividad es mayor o menor en los últimos minutos cuando hay mayor presión para modificar el resultado antes de que acabe el partido.

Otras variables añadidas han sido el número de disparo que es cada uno de ellos durante el partido tanto para el equipo (*shot\_number*) como para el futbolista que lo realiza (*shot\_player\_number*). Estas variables quieren estudiar si hay alguna mejora en los disparos a medida que el equipo o el futbolista realiza un mayor número de ellos o por el contrario hay menos probabilidad de marcar ya sea por cansancio o mejor conocimiento de la defensa o del portero.

También se han incluido otras dos variables como las descritas anteriormente, pero separando en distintas zonas del campo escogidas como “similares” y llamadas Shot Zones (zonas de disparo). Una de las variables cuenta cuantos disparos lleva un jugador en cada zona específica (*shot\_zone\_player\_number*) y otra cuenta los disparos de todo el equipo en esa zona (*shot\_zone\_number*). Estas dos variables quieren profundizar más acerca de la mejora o el empeoramiento de la calidad de los disparos al realizar un mayor número, pero con un mayor detalle al separar por zonas donde los disparos pueden ser considerados “similares”. Estas zonas son las siguientes (ver Ilustración 9):

* **Zona 1:** Disparos cercanos pero escorados.
* **Zona 2:** Disparos centrados y desde muy poca distancia, en boca de gol.
* **Zona 3:** Disparos centrados y desde dentro del área.
* **Zona 4:** Disparos de media distancia y laterales.
* **Zona 5:** Disparos desde la frontal del área.
* **Zona 6:** Disparo centrados y de media distancia.
* **Zona 7:** Disparos lejanos.

Gráfico, Gráfico de embudo

Descripción generada automáticamente

1

1

4

7

6

5

4

3

2

Ilustración 9: Mapa de las distintas zonas generadas.

También se han unificado las jornadas en distintos grupos (*competition\_part*) para hacer las diferenciaciones del momento de la temporada que se ha comentado en el punto anterior. En el caso de torneo con eliminatorias se han separado la fase de grupos de las propias eliminatorias y en el caso de las ligas nacionales se han dividido en inicio, mitad y final de la temporada.

Por último, también se ha creado un *dataset* con 1000 disparos ficticios realizados dentro del área en zonas donde se tiene menos de 100 disparos. A estos tiros se les coloca un 0% de probabilidad si están pegados a la línea de fondo y un 4,1% en el resto de tiros al ser la probabilidad existente en los tiros reales lanzados desde esas zonas (Rowlinson, 2020b). Este *dataset* es útil para incluir aquellos tiros que no realizan los futbolistas en la vida real puesto que son muy complicados y así incluir tiros en zonas donde el *dataset* original tiene muy pocos o ningún tiro (Sumpter, 2020). Junto a ello se han identificado y eliminado los *outlier*, es decir, zonas donde el número de disparos es menor de 20 y el porcentaje de acierto es mayor del 8% (Rowlinson, 2020b). Las zonas donde los disparos han sido eliminados se muestra en la Ilustración 10. En total han sido eliminados 228 disparos.

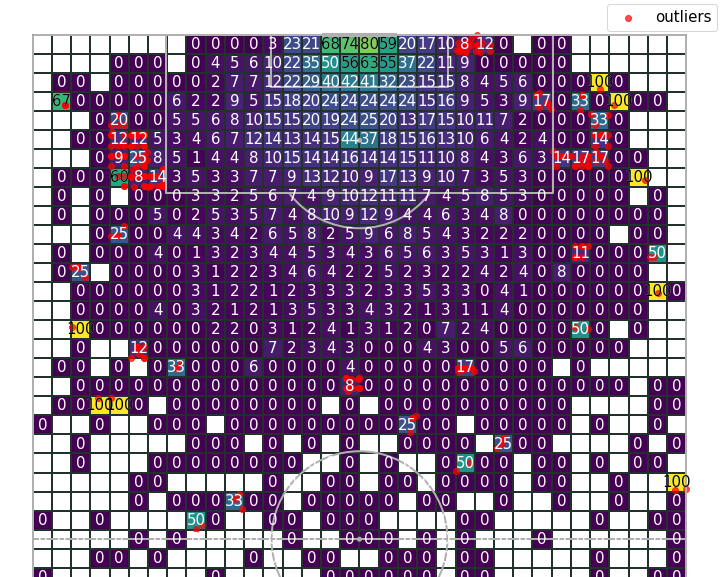


Ilustración 10: Porcentaje de acierto de los disparos en cada zona del campo e identificación de outliers con los datos de StatsBomb y WyScout.

## 4.3. *Dataset* utilizado

Finalmente se obtiene un *dataset* único con las siguientes variables (se obvia las variables relacionadas con ID’s y nombres de partidos, equipos y jugadores):

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre | Descripción | StatsBomb | WyScout |
| **dataset** | StatsBomb o WyScout | X | X |
| **competition\_gender** | Masculino o femenino | X | X |
| **competition\_type** | Liga o copa | X | X |
| **match\_week** | Jornada de la competición | X | X |
| **competition\_part** | Momento de la competición (Inicio, mitad, final, grupos o eliminatorias) | X | X |
| **H\_A\_column** | Equipo local o visitante | X | X |
| **minute** | Minuto de partido | X | X |
| **match\_moment** | Momento de partido (0-15min, 15-30min, 30-45min…) | X | X |
| x | Coordenada X del disparo en el campo. | X | X |
| y | Coordenada Y del disparo en el campo. | X | X |
| **visible\_angle** | Ángulo formado entre el lugar del disparo y los dos postes de la portería. | X | X |
| **middle\_angle** | Ángulo formado por la recta que va del centro de la portería al lugar del disparo y de la perpendicular a la portería. | X | X |
| **distance\_to\_goal** | Distancia entre el lugar del disparo y el centro de la portería. | X | X |
| **distance\_visible\_angle** | Distancia entre el lugar del disparo y el poste más cercano. | X | X |
| **log\_distance\_to\_goal** | Logaritmo de la distancia entre el lugar del disparo y el centro de la portería. | X | X |
| **shot\_type\_name** | Juego abierto, penalti, tiro libre (si ha ocurrido menos de 10 segundos después del lanzamiento), córner (si ha ocurrido menos de 10 segundos después del lanzamiento) o saque de banda (si ha ocurrido menos de 10 segundos después del saque). | X | X |
| **shot\_one\_on\_one** | Si el disparo se realiza en una situación de 1 contra 1. | X |  |
| **shot\_open\_goal** | Si el disparo se realiza sin ningún rival en el camino. | X |  |
| **under\_pressure** | Cuando el disparo empieza o termina simultáneamente a un evento de presión por parte del rival. (StatsBomb, 2019) | X |  |
| **counter\_attack** | Si el disparo se realiza durante un contraataque (robar el balón y atacar rápidamente). (StatsBomb, 2019; WyScout, 2018) | X | X |
| **fast\_break** | Si el disparo se realiza en un ataque rápido (robar el balón en tercio de campo propio y disparar en el último cuarto en menos de 25 segundos. (Rowlinson, 2020b) | X | X |
| **strong\_foot** | Si el disparo se ha realizado con la pierna buena/más utilizada. | X | X |
| **body\_part\_name** | Parte del cuerpo del disparo (pierna derecha, pierna izquierda u otro). | X | X |
| **shot\_zone** | Zona de disparo. Ver Ilustración 9. | X | X |
| **shot\_number** | Número de disparo del equipo en el partido. | X | X |
| **shot\_zone\_number** | Número de disparo del equipo en el partido y en una zona concreta. | X | X |
| **shot\_player\_number** | Número de disparo del futbolista en el partido. | X | X |
| **shot\_zone\_player\_number** | Número de disparo del futbolista en el partido y en una zona concreta. | X | X |
| **assist\_type** | Tipo de asistencia (pase, recuperación, despeje, jugada directa o rebote). | X | X |
| **pass\_end\_x** | La coordenada x de la asistencia de pase. | X |  |
| **pass\_end\_y** | La coordenada y de la asistencia de pase. | X |  |
| **carry\_length** | La distancia entre el lugar que se recibe el pase y el lugar donde se ha disparado. | X |  |
| **pass\_switch** | Si el pase previo fue un cambio de costado (el pase recorrió un 50% del campo horizontalmente). (StatsBomb, 2019) | X | X |
| **pass\_cross** | Si el pase previo fue un centro (un pase desde un costado y desde la parte final del campo). (StatsBomb, 2019) | X | X |
| **pass\_cut\_back** | Si el pase previo fue un pase atrás desde la parte final del campo y hacia el interior del área. (StatsBomb, 2019) | X | X |
| **pass\_height\_name** | Altura del pase previo. Alto si se recibe encima de los hombros (StatsBomb) o 1 metro encima o más alto (WyScout). En el resto de los casos se considera Bajo/A ras de suelo. | X | X |
| **pass\_technique\_name** | Técnica del pase previo (Pase en profundidad, córner lanzado recto, córner lanzado con efecto hacia la portería, córner lanzado con efecto hacia fuera de la portería u otro) | X | X |
| **smart\_pass** | Si el pase previo ha sido un pase inteligente, que ha pasado entre 2 o 3 rivales y ha generado una ventaja. (WyScout, 2018) |  | X |
| **area\_shot** | El área alrededor del jugador que dispara. Calculado como el área de un diagrama de Voronoi, es decir, el área donde el lanzador es el jugador más cercano a ese punto en el campo. (Rowlinson, 2020b) | X |  |
| **area\_goal** | El área alrededor del portero. Calculado como el área de un diagrama de Voronoi, es decir, el área donde el portero es el jugador más cercano a ese punto en el campo. (Rowlinson, 2020b) | X |  |
| **n\_angle** | Número de jugadores dentro del ángulo formado entre el lugar del disparo y los dos postes de la portería. (Rowlinson, 2020b) | X |  |
| **goalkeeper\_x** | Coordenada X del portero. | X |  |
| **goalkeeper\_y** | Coordenada Y del portero. | X |  |
| **players** | Jugadores en el campo del propio equipo. | X | X |
| **players\_rival** | Jugadores en el campo del equipo rival. | X | X |
| **goal** | Si el disparo ha sido gol. | X | X |

## 4.4 Diseño del modelo

En este problema se dispone de unos parámetros de entrada (variables posicionales, de acción previa, de tipo de acción, de contexto…) y una variable de salida (la variable binaria gol) lo que significa que se trata de un problema de aprendizaje supervisado donde se buscará la probabilidad (xG) de que la variable de salida sea 1 (es decir gol) mediante una técnica de clasificación.

Para obtener esta probabilidad se han utilizado distintos modelos de clasificación que han sido comparados para identificar cuál de ellos nos da un mejor resultado. A continuación, se hará una descripción de las métricas empleadas para comparar los modelos, una descripción de los distintos modelos, así como la metodología empleada en cada caso.

### 4.4.1 Métricas utilizadas

**Log-loss**

Log-loss (logaritmic loss, perdida logarítmica o entropía cruzada) es la métrica de clasificación más importante basada en probabilidades para un clasificador binario e indica como de cercana es la predicción del valor real correspondiente, cuanto menor sea la perdida logarítmica mayor será la verosimilitud del modelo. La pérdida logarítmica es la entropía cruzada entre la distribución de las predicciones y las etiquetas verdaderas. (Dembla, 2020; Godoy, 2019; Rowlinson, 2020b). La ecuación de la perdida logarítmica para un problema de clasificación binaria es la siguiente:

(Ec. 1)

Donde *N* es el número de observaciones en los datos, *yij* es un indicador binario de si la etiqueta *j* es la clasificación correcta para la observación *i* y *pij*es la probabilidad del modelo de la etiqueta *j* para la observación *i*. La pérdida logarítmica castiga en mayor medida las predicciones cuanto más alejadas están del valor de la clase (ver Ilustración 11).

Gráfico

Descripción generada automáticamente

Ilustración 11: Gráfico de la función log loss. (ML Glossary, s. f.)

**ROC-AUC**

La curva ROC (Receiver Operating Characteristics o Característica Operativa del Receptor) es una curva que traza la tasa de verdaderos positivos (TPR) contra la tasa de falsos positivos (FPR) en diferentes umbrales de clasificación (Narkhede, 2018). El calculo de cada tasa se realiza mediante las siguientes ecuaciones:

(Ec. 2)

(Ec. 3)

Siendo TP los verdaderos positivos, FN los falsos negativos, FP los falsos positivos y TN los verdaderos negativos. Se considera positivo aquella probabilidad superior a 0,5.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Ilustración 12: Tres tipos de curvas ROC con sus respectivas AUC. (Wikipedia, 2021)

Para estudiar qué modelo tiene mayor capacidad para distinguir entre clases se utiliza la ROC AUC (Area Under the ROC Curve o Area Bajo la curva ROC) que nos da un valor numérico de la curva de cada modelo. Cuanto mayor sea el valor ROC AUC mejor será el modelo (ver Ilustración 12).

**McFadden’s Pseudo R2**

La Pseudo R2 de McFadden compara nuestro modelo con un modelo nulo que da la misma predicción a todos los tiros (Rowlinson, 2020b). Viene definido como:

(Ec. 4)

Siendo *LC* el valor de verosimilitud (*likelihood* *value*) del modelo y *Lnull* el valor de verosimilitud de un modelo nulo. La verosimilitud es la multiplicación de todas las diferencias entre las predicciones y los valores reales. Al dar un valor muy pequeño se suele mostrar como el logaritmo de la verosimilitud (*log likelihood*) siendo su valor negativo el valor de la perdida logarítmica explicada anteriormente (Becker, 2018). Cuando mayor sea el valor mejor será el modelo.

**Puntuación de Brier**

La puntuación de Brier mide la diferencia cuadrática media entre la probabilidad predicha y el resultado real y cuanto más bajo sea su valor más precisa es la predicción (Scikit-Learn, s. f.-f). La ecuación para la calcular la puntuación de Brier es la siguiente:

(Ec. 5)

**Curva de calibración**

La calibración mide si las probabilidades predichas coinciden con la distribución esperada de cada clase. Cuanto mejor calibrado esté un modelo, más fiable será el pronóstico fuera de muestra. Un clasificador estará bien calibrado si de todos los tiros que han recibido una probabilidad cercana a 0,5 han sido gol la mitad de ellos. (Scikit-Learn, s. f.-a; Tucker, 2020)

Gráfico

Descripción generada automáticamente

Ilustración 13: Ejemplo de distintas curvas de calibración para distintos modelos. (Scikit-Learn, s. f.-a)

La curva de calibración (Ilustración 13) es la representación gráfica de la calibración del modelo en diferentes umbrales de clasificación. Mediante Scikit-Learn es posible visualizar la curva de manera uniforme (misma anchura para cada umbral) o por cuantiles (cada umbral tiene la misma cantidad de muestras y depende de la probabilidades predichas) (Scikit-Learn, s. f.-d).

### 4.4.2 Regresión Logística

La Regresión Logística Simple, desarrollada por David Cox en 1958, es un método de regresión que permite estimar la probabilidad de una variable cualitativa binaria en función de una variable cuantitativa. Una de las principales aplicaciones de la regresión logística es la de clasificación binaria, en el que las observaciones se clasifican en un grupo u otro dependiendo del valor que tome la variable empleada como predictor. En este caso se estimará la probabilidad de que un disparo sea gol o no gol en base a todas las variables disponibles. Es un método ampliamente utilizado desde la década de los 80 debido a las facilidades computacionales con que se cuenta desde entonces. (Alonso Fernández, 2006; Amat Rodrigo, 2016).

La función logística o sigmoide se expresa de la siguiente manera (Amat Rodrigo, 2016):

(Ec. 6)

Que al substituir x de la Ec. 1 por la función (- α - β1x1 - β2x2 - … - βpxp) se obtiene:

(Ec. 7)

Que es la probabilidad de que la variable Y sea 1 dado los distintos predictores x1, x2… Si esta probabilidad es mayor a 0,5 el clasificador lo catalogará como 1 (gol) y por el contrario si es menor lo catalogará como 0 (no gol) aunque para nuestro problema lo importante no es la catalogación si no la probabilidad en sí misma.

Nuestro objetivo, por lo tanto, es obtener los parámetros o pesos (α, β1, β2, … - βp) óptimos de cada variable para minimizar nuestro error. En el caso de la regresión logística se buscará minimizar la pérdida logarítmica (log loss).

El proceso para la generación del modelo de regresión logística ha sido el siguiente.

**Eliminación de variables**

El método de regresión logística no permite trabajar con datos faltantes por lo que aquellas columnas donde solo se tiene datos provenientes de una de las dos fuentes de datos deben ser eliminadas (la posición del portero, *smart pass*, jugadores en el ángulo entre el disparo y la portería…).

**Separación de *datasets***

Por el mismo motivo que el paso anterior es necesario separar el *dataset* en 3 *subdatasets*. El primero es para aquellos goles que provienen de una asistencia, puesto que tienen información sobre la asistencia recibida (técnica del pase, altura del pase…). El según *subdataset* es para los disparos que no han recibido una asistencia previa. Por último, el tercer *subdataset* es solo para los lanzamientos de penalti. En cada caso se han eliminado aquellas columnas que no tienen ninguna información sobre el disparo dejando de esta manera 3 *datasets* sin ningún dato en blanco.

**Creación de variables *dummies***

El método de regresión logística tampoco puede trabajar con datos categóricos como puede ser el tipo de competición o la parte del cuerpo utilizada. Pasar las variables categóricas a valores numéricos tampoco es una opción válida puesto que el modelo entenderá que hay una relación lineal entre las categorías cuando no es así. Es por eso que se han generado nuevas variables binarias para cada tipo de categoría de la variable inicial. Estas nuevas variables se conocen como *dummies* y funcionan tal como se muestra en la Tabla 3. (Langford, 2017; Li, 2017; Yadav, 2019)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Body\_type |  | Body\_type\_Right\_Foot | Body\_type\_Left\_Foot | Body\_type\_Other |
| Right Foot |  | 1 | 0 | 0 |
| Left Foot |  | 0 | 1 | 0 |
| Other |  | 0 | 0 | 1 |

Tabla 3: Ejemplo de creación de variables dummies a partir de una variable categórica (Body\_type).

**Separación de los *datasets* en datos de entrenamiento y datos de prueba**

Tanto en este modelo como en los siguientes se ha separado los distintos *datasets* utilizados en datos de entrenamiento y datos de prueba. El primer grupo de datos será el utilizado para entrenar el modelo mientras que el segundo se utilizará para comparar las predicciones del modelo con valores reales que no han sido utilizados durante el entrenamiento y así evaluar el modelo. En este trabajo se ha utilizado un 80% de los datos como datos de entrenamiento y el 20% restante como datos de prueba. En la se puede apreciar el número de lanzamientos y el número de goles que tiene cada *dataset* se ha pedido que la proporción de goles tanto en los datos de entrenamiento como en los de test sea igual a partir de un muestreo aleatorio estratificado (ver Tabla 4). (Rowlinson, 2020b)

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | Nº lanzamientos | Nº goles | % goles |
| Entrenamiento pases | 35198 | 3860 | 11,1% |
| Test pases | 8800 | 965 | 11,1% |
| Entrenamiento otros | 16406 | 1618 | 9,9% |
| Test otros | 4102 | 404 | 9,8% |
| Entrenamiento penaltis | 656 | 486 | 74,1% |
| Test penaltis | 165 | 122 | 73,9% |

Tabla 4: Lanzamientos, goles y proporción de los datos de entrenamiento y de test para cada dataset del modelo de regresión logística.

**Selección de variables y parámetros del modelo**

Tras separar el *dataset* el siguiente pasoes el uso del algoritmo RFE (Recursive Feature Elimination o Eliminación de características recursivas). Este algoritmo estudia cuales son las variables más importantes para el modelo que se desea entrenar (en este caso la regresión logística) y mantiene solo aquel número de variables que se le ha pedido mantener eliminado variables de una en una. (Li, 2017)

Una vez se ha mantenido únicamente el número de variables deseadas es utilizada la clase StandardScaler de SciKit-Learn para preprocesar los datos. Su función es estandarizar el conjunto de datos haciendo que todas las variables tengan una distribución normal con media cero y varianza unitaria. De esta forma ninguna variable dominará la función debido a una mayor varianza al resto. (Scikit-Learn, s. f.-b)

Después de estandarizar los valores se utiliza la clase GridSearchCV de SciKit-Learn. Esta clase ayuda a encontrar, de manera automatizada, los mejores hiperparámetros para ajustar el modelo de regresión lineal a partir de los datos de entrenamiento. Para conseguirlo aplica una técnica de validación cruzada (CV o *cross-validation*) donde se dividen los datos de entrenamiento en distintos pliegues (*folds*) de mismo tamaño y se crean distintos modelos secuencialmente. Cada modelo utiliza un pliegue como conjunto de validación y al resto como datos de entrenamiento. Finalmente se selecciona el modelo que ha logrado ofrecer un mejor resultado en la métrica que se haya pedido optimizar. En el caso concreto de este trabajo se ha buscado optimizar C (la fuerza de regulación inversa) con el objetivo de lograr la menor perdida logarítmica posible mediante una validación cruzada de 5 pliegues. (Krishni, 2018; Manna, 2020; Sharma, 2020)

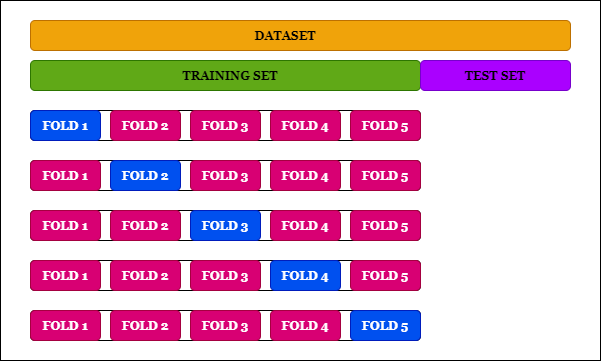


Ilustración 14: Ejemplo de CV con 5 pliegues.(Manna, 2020)

**Implementación del modelo y obtención de métricas**

Una vez se han seleccionado las variables a utilizar y se han obtenido los parámetros que optimizan el modelo este es implementado. Con el modelo implementado se obtienen las predicciones de cada disparo tanto para los datos de entrenamiento como los de prueba. Tanto los valores reales como las predicciones de los dos *datasets* de tiros que no han sido de penalti se unifican. Las predicciones obtenidas se comparan con los valores reales y se obtienen las distintas métricas para comparar los modelos de este trabajo entre sí y con otros trabajos.

A partir de las distintas métricas obtenidas se decide si hay que modificar el número de variables usadas en el modelo y por lo tanto se decide si volver al paso de selección de variables o se da ya por bueno el modelo obtenido.

### 4.4.3 LightGBM

El LightGBM es un *framework* de código abierto de refuerzo de gradientes (*gradient boosting*) desarrollado por Microsoft y lanzado en 2016. LightGBM utiliza algoritmos de aprendizaje basados ​​en árboles de decisión. Pese a ser un algoritmo relativamente nuevo cada vez es más usado al ser el más rápido, tener mayor precisión y utilizar menos memoria que otros algoritmos. Es especialmente útil para *datasets* grandes, para problemas con *datasets* pequeños puede provocar un sobreajuste. (Banerjee, 2020c; Gursky, 2020; Kasturi, 2019; Wikipedia, s. f.)

Los algoritmos de aprendizaje basados en árboles de decisión son uno de los mejores y más utilizados métodos tanto para problemas de categorización como de regresión. Tienen como ventaja respecto a los modelos lineales el hecho de mapear mucho mejor las relaciones no lineales y funcionan tanto para variables categóricas como continuas. Esta técnica divide la muestra en dos o más conjuntos homogéneos en función del valor de un atributo seleccionado (ganancia de información, Gini…). La función objetivo es representada como una serie de condiciones consecutivas: (Analytics Vidhya, 2016; Kurnia, 2021)

* Nodos: Atributos (Parte del cuerpo, Distancia a portería…).
* Arcos: Valores de los atributos (pierna derecha, pierna izquierda u otro para Parte del cuerpo).
* Hojas: Clases (Gol o No gol).
* Rama: Condiciones desde la raíz a la hoja unidas a través de conjunciones (AND) y entre ramas a través de disyunciones (OR).

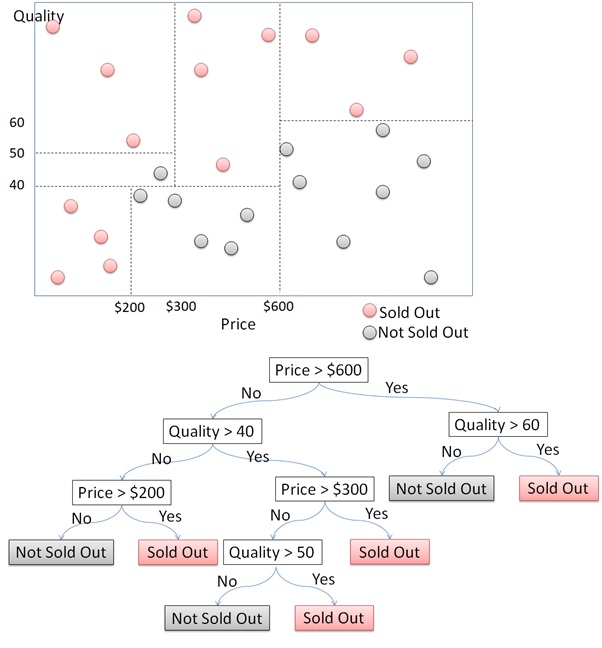


Ilustración 15: Ejemplo de árbol de decisión. (Kurnia, 2021)

En el caso de LightGBM se trata de un algoritmo que hace crecer el árbol verticalmente mientras que otros algoritmos de aprendizaje basados en árboles hacen crecer los árboles horizontalmente. Esto significa que LightGBM crece solo en una de las hojas, aquella con la máxima pérdida delta, y no crece en niveles (ver Ilustración 16). (Mandot, 2017)

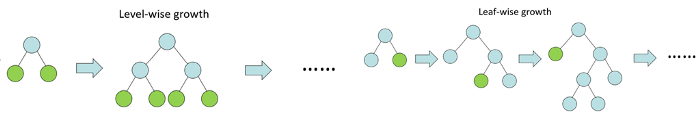


Ilustración 16: Ejemplo de árbol con crecimiento por niveles (izquierda) y por hojas (derecha). (Kasturi, 2019)

Si se hace crecer ambos arboles de manera total se termina obteniendo el mismo árbol, pero esto nunca ocurre (se realiza una parada temprana o una poda) y por lo tanto el orden de crecimiento nos hará obtener arboles distintos en ambos casos. Para decidir cómo se quiere que sea la construcción del árbol el algoritmo tiene más de 100 parámetros que pueden modificarse, los que se busca optimizar en nuestro TFM son (LightGBM, s. f.):

* **num\_leaves:** Máximo de hojas de árboles.
* **max\_depth:** Profundidad máxima del árbol.
* **min\_child\_samples:** Número mínimo de datos necesarios en una hoja.
* **reg\_alpha:** Término de regularización L1 en pesos.
* **reg\_lambda:** Término de regularización L2 en pesos.

El proceso para la generación del modelo de LightGBM ha sido el siguiente aprovechando el código utilizado por Andrew Rowlinson (Rowlinson, 2020a):

**Eliminación de variables**

En el caso del método LightGBM solo es necesario eliminar las columnas relacionadas con las ID’s y los nombres de jugadores, equipos, etc.

**Separación de *datasets***

Este método si permite tener variables con datos faltantes. Debido a ello solo se separa el *dataset* inicial en dos *datasets*; uno para lanzamientos de penalti y otro para el resto de los lanzamientos.

**Separación de los *datasets* en datos de entrenamiento y datos de prueba**

Mismo procedimiento que en el caso de la regresión logística. Cabe añadir que para este método se añade los disparos ficticios generados para añadir más información en los datos de entrenamiento y estos reducen algo la proporción de goles para el *dataset* de entrenamiento (ver Tabla 5).

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset | Nº lanzamientos | Nº goles | % goles |
| Entrenamiento no penaltis | 52604 | 5512 | 10,5% |
| Test no penaltis | 12902 | 1369 | 10,6% |
| Entrenamiento penaltis | 656 | 486 | 74,1% |
| Test penaltis | 165 | 122 | 73,9% |

Tabla 5: Lanzamientos, goles y proporción de los datos de entrenamiento y de test para cada dataset del modelo de LightGBM.

**Remplazo de las variables categóricas y booleanas**

Así como en el método anterior se utiliza la técnica de generar nuevas variables *dummies* para este modelo en el caso de las variables categóricas no hace falta dividirlas en distintas variables binomiales. Por eso en este modelo lo único que hay que hacer es convertir las variables categóricas y booleanas en variables numéricas.

**Selección de variables y parámetros del modelo**

En el caso de LightGBM no se elimina ninguna variable con información del disparo por lo que se utilizarán todas las de los *datasets* iniciales. En cuanto a los parámetros a optimizar explicados anteriormente se utiliza la optimización bayesiana de la clase BayesSearchCV de la librería Scikit-Optimize (o SkOpt). La optimización bayesiana de parámetros consiste en hacer un número de ajustes de parámetros fijo (el número de iteraciones elegidas, en este TFM 100) logrando que el algoritmo, mediante una búsqueda con validación cruzada de 5 pliegues, se vaya redirigiendo en cada iteración hacia las regiones de mayor interés, eligiendo únicamente los mejores candidatos y reduciendo el tiempo al no estudiar todas las combinaciones posibles.(Amat Rodrigo, 2020; Scikit-Optimize, s. f.)

**Implementación del modelo y obtención de métricas**

Antes de implementar el modelo mediante la clase LGBMClassifier se debe tener en cuenta que no es un clasificador bien calibrado como puede ser el de regresión logística. Por ello antes de implementar el modelo se utiliza la clase CalibratedClassifierCV de Scikit-Learn que calibra el clasificador mediante una validación cruzada (de 3 pliegues en este caso). También se especifica que el método de calibración será isotónico al funcionar mejor con cantidades de datos grandes. (Rowlinson, 2020b; Scikit-Learn, s. f.-c)

Una vez implementado el modelo se saca la importancia de cada variable en el modelo y las distintas métricas para compararlo con el resto.

### 4.4.4 XGBoost

XGBoost (de *eXtreme Gradient Boosting*) es otro algoritmo de aprendizaje basado en arboles de decisión que utiliza un *framework* de refuerzo de gradientes al igual que lo es LightGBM. Fue desarrollado como un proyecto de investigación de la Universidad de Washigton y fue presentado en 2016 por Tianqi Chen and Carlos Guestrin (Chen & Guestrin, 2016). XGBoost construye árboles nuevos basándose en los errores de árboles de decisiones anteriores. Pese a que se han mejorado los tiempos de ejecución sigue siendo un algoritmo más lento que LightGBM pero actualmente sigue siendo muy utilizado al llevar más años disponible su versión estable y ser más conocido por los profesionales. El principal motivo de la diferencia en los tiempos de ejecución es el hecho que XGBoost crece por niveles y no por hojas. Pese a esta diferencia XGBoost tiene una precisión muy cercana a LightGBM y también funciona bien con grandes *datasets* y con *datasets* con gran cantidad de variables (Banerjee, 2020a, 2020b; Morde, 2019).

Al igual que en el algoritmo anterior, XGBoost tiene una gran cantidad de parámetros que se pueden modificar, en este trabajo se ha buscado optimizar los siguientes (XGBoost, s. f.):

* max\_depth: Profundidad máxima del árbol.
* min\_child\_weight: Suma mínima de peso de una instancia necesaria en una hoja.
* reg\_alpha: Término de regularización L1 en pesos.
* reg\_lambda: Término de regularización L2 en pesos.

La metodología empleada con el método XGBoost es la misma que para LightGBM.

También se realiza la optimización bayesiana de los parámetros y el calibrador del clasificador mediante una validación cruzada. La única diferencia recae en el uso de la clase XGBClassifier en este caso y el hecho de que algunos parámetros a optimizar son distintos.

Finalmente, como en el caso de LightGBM, se obtiene la importancia de cada variable, las distintas métricas y la curva de calibración del modelo.

### 4.4.5 Random Forest

Random Forest (o Bosque Aleatorio) es un método de aprendizaje supervisado para problemas de clasificación y regresión que funciona con la generación de varios árboles de decisión de manera que cada árbol depende de los valores de un vector aleatorio muestreado de forma independiente y con la misma distribución para todos los árboles del bosque (Breiman, 2001).

Estos árboles de decisión son entrenados con el método de empaquetado (o *bagging*) que trata de separar los datos de entrenamiento en distintos bloques y elaborar un árbol de decisión para cada uno. La idea general del método de empaquetado es que una combinación de modelos de aprendizaje aumenta el resultado general. El algoritmo hace la predicción a partir de una votación (para problemas de clasificación) o una media (para regresión) de la salida de los distintos árboles de decisión (ver Ilustración 17). De esta manera se logra que los errores de un modelo sean compensados por otro. Por tanto, cuanto mayor sea el número de árboles mayor precisión se obtendrá, así como también se reducirá el error de generalización. El algoritmo fue desarrollado por Leo Breiman en 2001 (Breiman, 2001; Mbaabu, 2020).

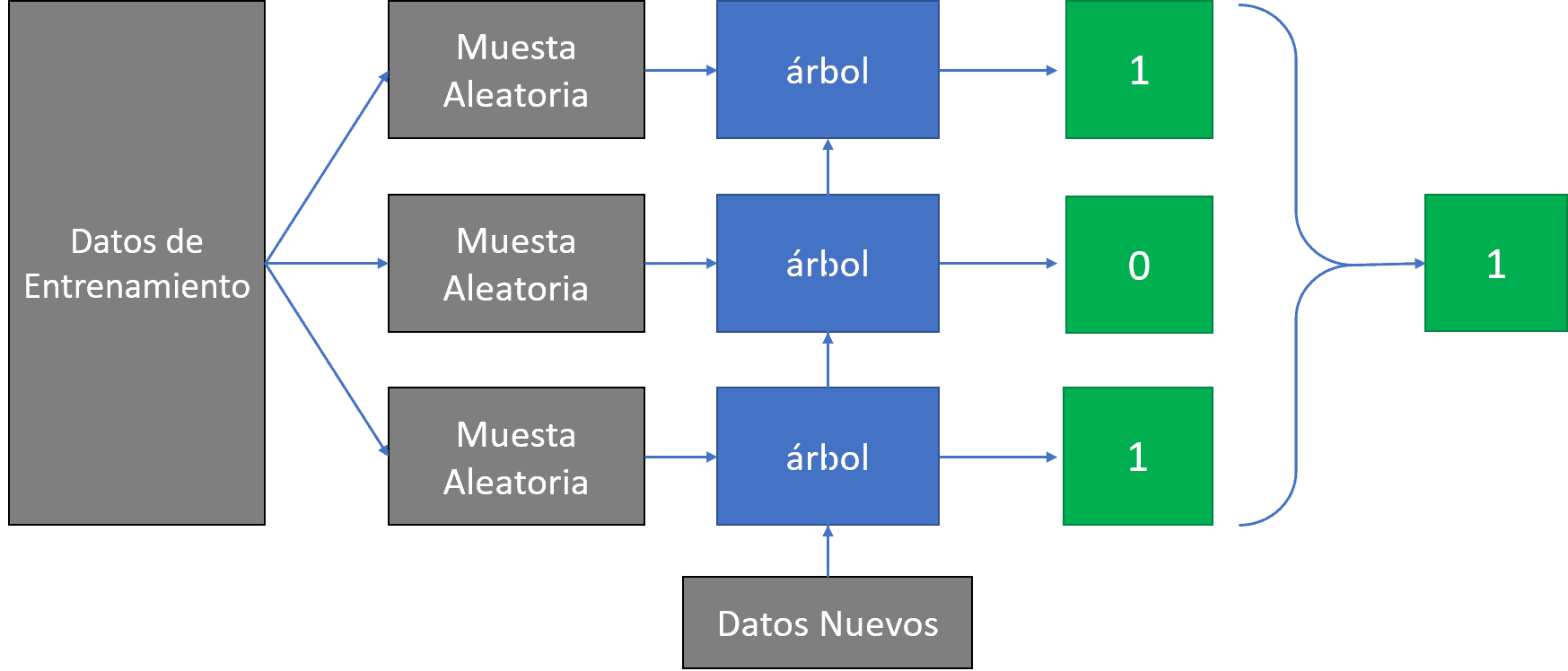


Ilustración 17: Esquema básico del método Random Forest.(Martinez Heras, 2019)

En el caso de Random Forest se ha buscado optimizar los siguientes parámetros (Scikit-Learn, s. f.-e):

* **n\_estimators:** Cantidad de árboles generados.
* **max\_features:** La cantidad de características que se deben considerar al buscar la mejor división.
* **min\_samples\_leaf:** El número mínimo de muestras necesarias que debe haber en un nodo final (hoja).
* **min\_samples\_split:** El número mínimo de muestras necesarias para dividir un nodo interno.
* **criterion:** La función para medir la calidad de una división. Los criterios admitidos son "gini" para la impureza de Gini y "entropía" para la ganancia de información.
* **max\_depth:** La profundidad máxima del árbol.

El proceso para la generación del modelo de Random Forest tiene etapas tanto de la Regresión Logística como de LightGBM y XGBoost.

Al igual que en el caso de la Regresión Logística es necesario dividir el *dataset* inicial de tiros que no son de penalti en dos *datasets*, uno para tiros provenientes de un pase y otro para el resto. El motivo es el mismo, aunque Random Forest puede trabajar con datos faltantes su manera de hacerlo es colocando el valor medio de la variable en las filas sin valor. Como en muchos casos donde faltan datos es debido a que no es posible tener ningún dato (si el gol no viene de una asistencia no es posible tener un valor del tipo de asistencia) este método para trabajar con datos faltantes no resulta positivo. (Breiman, 2001) También ese necesario convertir las variables categóricas en distintas variables *dummies* como ocurre en el caso de la Regresión Logística.

Una vez quedan preparados los dos *datasets* el procedimiento es idéntico al de LightGBM i XGBoost. En este caso se utiliza la clase RandomForestClassifier como clasificador y se utiliza la optimización bayesiana de los parámetros y el calibrador del clasificador mediante una validación cruzada.

Finalmente, al igual que en la Regresión Logística, se unifican los valores reales y las predicciones obtenidas en ambos *datasets* y se calculan las distintas métricas y la curva de calibración del modelo.

# 5. Desarrollo específico de la contribución

## 5.1 Análisis de las nuevas variables

## 5.2 Análisis de los modelos obtenidos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Regresión Logística | LightGBM | XGBoost | Random Forest |
| Log-loss | 0,2847 | 0,2854 | **0,2810** | 0,2858 |
| ROC AUC | 0,7817 | 0,7855 | **0,7912** | 0,7788 |
| McFadden’s R2 | 0,1584 | 0,1563 | **0,1694** | 0,1553 |
| Brier score | 0,0822 | 0,0818 | **0,0814** | 0,0824 |
| Tiempo | **00:00:04** | 00:30:13 | 1:03:48 | 01:33:46 |

## 5.3 Comparación con otros modelos

# 6. Conclusiones y trabajo futuro

## 6.1. Conclusiones

## 6.2. Líneas de trabajo futuro

# 7. Bibliografía

11tegen11. (2015, octubre 31). Arsenal pulled away in the 2nd half. Their defensive numbers look very impressive again. But what’s up with Swansea? Https://t.co/LKXVcvCaRT [Tweet]. *@11tegen11*. https://twitter.com/11tegen11/status/660510093365129216/photo/1

Alonso Fernández, A. M. (2006). *Introducción a la regresión logística*. http://halweb.uc3m.es/esp/Personal/personas/amalonso/esp/bstat-tema9.pdf

Amat Rodrigo, J. (2016, agosto). *Regresión logística simple y múltiple*. https://www.cienciadedatos.net/documentos/27\_regresion\_logistica\_simple\_y\_multiple

Amat Rodrigo, J. (2020, abril). *Optimización bayesiana de hiperparámetros*. https://www.cienciadedatos.net/documentos/62\_optimizacion\_bayesiana\_hiperparametros.html

Analytics Vidhya. (2016, abril 11). Tree Based Algorithms | Implementation In Python & R. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/

Andersen, K. (2021, marzo 13). Interview: The creator of OptaJoe and how xG came about. *A Word of Arsenal*. https://awordofarsenal.com/2021/03/13/interview-the-creator-of-optajoe-and-how-xg-came-about/

Banerjee. (2020a, diciembre 8). *XGBoost + k-fold CV + Feature Importance* [Kaggle]. https://kaggle.com/prashant111/xgboost-k-fold-cv-feature-importance

Banerjee, P. (2020b, julio 15). *A Guide on XGBoost hyperparameters tuning* [Kaggle]. https://kaggle.com/prashant111/a-guide-on-xgboost-hyperparameters-tuning

Banerjee, P. (2020c, julio 21). *LightGBM Classifier in Python*. Kaggle. https://kaggle.com/prashant111/lightgbm-classifier-in-python

Barnett, V., & Hilditch, S. (1993). The Effect of an Artificial Pitch Surface on Home Team Performance in Football (Soccer). *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, *156*(1), 39-50. https://doi.org/10.2307/2982859

Becker, D. (2018). *What is Log Loss?* Kaggle. https://kaggle.com/dansbecker/what-is-log-loss

Blum, J. (2017, octubre 27). Using Neural Networks to calculate Expected Goals. *Jon Blum*. https://jonblum.wordpress.com/2017/10/27/using-neural-networks-to-calculate-expected-goals/

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5-32. https://doi.org/10.1023/A:1010933404324

Brener, A. (2017). *Expected Goals Model* [Python]. https://github.com/andrebrener/expected\_goals

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., Vanderplas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. *arXiv:1309.0238 [cs]*, 108--122.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785-794. https://doi.org/10.1145/2939672.2939785

Coronis, A. (2021, mayo 3). Los “expected goals”, la métrica de moda del análisis big data. *Futbol Sapiens*. https://www.futbolsapiens.com/mas-sapiens/los-expected-goals-la-metrica-de-moda-del-analisis-big-data/

De Torres, A. (2021, enero). *¿Cómo funciona el Big Data en fútbol?* [Educativa]. ESIC. https://www.esic.edu/rethink/tecnologia/big-data-en-futbol

Dembla, G. (2020, noviembre 17). *Intuition behind Log-loss Score*. Medium. https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a

Durgapal, A., & Rowlinson, A. (2021). *Quick start—Mplsoccer 1.0.5 documentation*. mplsoccer. https://mplsoccer.readthedocs.io/en/latest/#

Eggels, H. P. H. (2016). *Expected Goals in Soccer: Explaining Match Results using Predictive Analytic* [Eindhoven University of Technology]. https://pure.tue.nl/ws/portalfiles/portal/46945853

Friends of Tracking. (2020, mayo 8). *The Ultimate Guide to Expected Goals*. https://www.youtube.com/watch?v=310\_eW0hUqQ&t=1101s

Giacobbe. (2016, septiembre 8). Un nuovo modello di Expected Goals. *L’Ultimo Uomo*. https://www.ultimouomo.com/un-nuovo-modello-di-expected-goals/

Godoy, D. (2019, febrero 7). *Understanding binary cross-entropy / log loss: A visual explanation*. Medium. https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a

Goodman. (2018a, noviembre 12). *A New Way to Measure Keepers’ Shot Stopping: Post-Shot Expected Goals*. StatsBomb. https://statsbomb.com/2018/11/a-new-way-to-measure-keepers-shot-stopping-post-shot-expected-goals/

Goodman, M. (2018b, mayo 18). The Dual Life of Expected Goals (Part 2). *StatsBomb*. https://statsbomb.com/2018/05/the-dual-life-of-expected-goals-part-2/

Green, S. (2012, abril 9). *Assessing The Performance of Premier League Goalscorers*. Stats Perform. https://www.statsperform.com/resource/assessing-the-performance-of-premier-league-goalscorers/

Gregory, S. (2017, enero 30). *Expected Goals in Context*. Stats Perform. https://www.statsperform.com/resource/expected-goals-in-context/

Gursky, J. (2020, marzo 30). *Boosting Showdown: Scikit-Learn vs XGBoost vs LightGBM vs CatBoost in Sentiment Classification*. Medium. https://towardsdatascience.com/boosting-showdown-scikit-learn-vs-xgboost-vs-lightgbm-vs-catboost-in-sentiment-classification-f7c7f46fd956

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357-362. https://doi.org/10.1038/s41586-020-2649-2

Hedar, S. (2020). *Applying Machine Learning Methods to Predict the Outcome of Shots in Football* [Uppsala University]. https://www.diva-portal.org/smash/get/diva2:1448482/FULLTEXT01.pdf

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science Engineering*, *9*(3), 90-95. https://doi.org/10.1109/MCSE.2007.55

Kasturi, S. N. (2019, julio 11). *LightGBM vs XGBOOST: Which algorithm win the race !!!* Medium. https://towardsdatascience.com/lightgbm-vs-xgboost-which-algorithm-win-the-race-1ff7dd4917d

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems*, 3149-3157.

Knutson, T. (2020, julio 31). StatsBomb Release Expected Goals with Shot Impact Height. *StatsBomb*. https://statsbomb.com/2020/07/statsbomb-release-expected-goals-with-shot-impact-height/

Krishni. (2018, diciembre 16). *K-Fold Cross Validation. Evaluating a Machine Learning model can… | by Krishni | DataDrivenInvestor*. Medium. https://medium.datadriveninvestor.com/k-fold-cross-validation-6b8518070833

Kurnia, R. (2021, abril 15). Tree-Based Machine Learning Algorithms | Compare and Contrast. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2021/04/distinguish-between-tree-based-machine-learning-algorithms/

Langford, R. (2017, marzo 24). *The Dummy’s Guide to Creating Dummy Variables*. Medium. https://towardsdatascience.com/the-dummys-guide-to-creating-dummy-variables-f21faddb1d40

Lawrence, T., Yorke, J., & haghanim. (2021). *StatsBomb Open Data*. StatsBomb. https://github.com/statsbomb/open-data (Original work published 2018)

Lewis, M. (2004). *Moneyball: The Art of Winning an Unfair Game*.

Li, S. (2017, septiembre 29). *Building A Logistic Regression in Python, Step by Step*. Medium. https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8

LightGBM. (s. f.). *lightgbm.LGBMClassifier—LightGBM 3.2.1.99 documentation*. LightGBM. Recuperado 9 de septiembre de 2021, de https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html#lightgbm.LGBMClassifier

Linacre, R. (2017). *RobinL/fuzzymatcher* [Python]. https://github.com/RobinL/fuzzymatcher (Original work published 2017)

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

Mackay, N. (2017, junio 19). How accurate are xG models II: the ‘Big Chance’ Dilemma. *Mackay Analytics*. https://mackayanalytics.nl/2017/06/19/how-accurate-are-xg-models-ii-the-big-chance-dilemma/

Madrero, P. (2020). *Creating a Model for Expected Goals in Football using Qualitative Player Information* [Universitat Politècnica de Catalunya]. https://upcommons.upc.edu/bitstream/handle/2117/328922/147841.pdf

Mandot, P. (2017, agosto 17). *What is LightGBM, How to implement it? How to fine tune the parameters?* Medium. https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc

Manna, S. (2020, marzo 20). K-Fold Cross Validation for Deep Learning using Keras. *The Owl*. https://medium.com/the-owl/k-fold-cross-validation-in-keras-3ec4a3a00538

Martinez Arastrey, G. (2018, mayo 22). *What are Expected Goals (xG)?* Sport Performance Analysis. https://www.sportperformanceanalysis.com/article/what-are-expected-goals-xg

Martinez Heras, J. (2019, junio 10). Random Forest (Bosque Aleatorio): Combinando árboles. *IArtificial.net*. https://www.iartificial.net/random-forest-bosque-aleatorio/

Mbaabu, O. (2020, diciembre 11). *Introduction to Random Forest in Machine Learning*. Section. https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. 56-61. https://doi.org/10.25080/Majora-92bf1922-00a

Mena Camino, L. (2021). *Buscando el recambio perfecto para Fernando*. Big Data International Campus. https://www.campusbigdata.com/difusion/Futuro\_Recambio\_Fernando.pdf

Miller, B. (2011). *Moneyball* [Drama]. https://www.filmaffinity.com/es/film974637.html

ML Glossary. (s. f.). *Loss Functions—ML Glossary documentation*. Recuperado 10 de septiembre de 2021, de https://ml-cheatsheet.readthedocs.io/en/latest/loss\_functions.html

Morde, V. (2019, abril 8). *XGBoost Algorithm: Long May She Reign!* Medium. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

Mullenberg, J. (2016, octubre 14). *Expected Goals: Wat is het en hoe berekenen we het? - Tussen de linies* [TussenDeLinies]. https://www.tussendelinies.nl/expected-goals-uitgelegd/

Narkhede, S. (2018, junio 26). *Understanding AUC - ROC Curve*. Medium. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Noordman, R. (2019). *Improving the estimation of outcome probabilities of football matches using in-game information* [Amsterdam School of Economics]. https://www.scisports.com/wp-content/uploads/2019/10/Noordman-Rogier-12366315-MSc-ETRICS.pdf. https://www.scisports.com/wp-content/uploads/2019/10/Noordman-Rogier-12366315-MSc-ETRICS.pdf

Oliphant, T. (2006). *Guide to NumPy*. Trelgol Publishing USA,. https://web.mit.edu/dvp/Public/numpybook.pdf

Pappalardo, L., Cintia, P., Rossi, A., Massucco, E., Ferragina, P., Pedreschi, D., & Giannotti, F. (2019). A public data set of spatio-temporal match events in soccer competitions. *Scientific Data*, *6*(1), 236. https://doi.org/10.1038/s41597-019-0247-7

Pappalardo, L., & Massuco, E. (2019). *Soccer match event dataset*. https://doi.org/10.6084/m9.figshare.c.4415000

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., & Cournapeau, D. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Pérez, D. (2017, junio 12). Big Data en el fútbol [Deportiva]. *Objetivo Analista*. https://objetivoanalista.com/big-data-futbol/

Pollard, R., & Reep, C. (1997). Measuring the Effectiveness of Playing Strategies at Soccer. *Journal of the Royal Statistical Society. Series D (The Statistician)*, *46*(4), 541-550.

Reback, J., McKinney, W., jbrockmendel, Bossche, J. V. den, Augspurger, T., Cloud, P., gfyoung, Sinhrks, Klein, A., Roeschke, M., Hawkins, S., Tratner, J., She, C., Ayd, W., Petersen, T., Garcia, M., Schendel, J., Hayden, A., MomIsBestFriend, … Mehyar, M. (2020). *pandas-dev/pandas: Pandas* (1.0.3) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.3715232

Reep, C., Pollard, R., & Benjamin, B. (1971). Skill and Chance in Ball Games. *Journal of the Royal Statistical Society. Series A (General)*, *134*(4), 623-629. https://doi.org/10.2307/2343657

Rowlinson, A. (2020a). *Expected-goals-thesis* [Jupyter Notebook]. https://github.com/andrewRowlinson/expected-goals-thesis

Rowlinson, A. (2020b). *Football Shot Quality: Visualizing the Quality of Soccer/ Football Shots* [Aalto University]. https://aaltodoc.aalto.fi/bitstream/handle/123456789/45953/master\_Rowlinson\_Andrew\_2020.pdf

Scikit-Learn. (s. f.-a). *1.16. Probability calibration* [Scikit-Learn]. Recuperado 9 de mayo de 2021, de https://scikit-learn.org/stable/modules/calibration.html

Scikit-Learn. (s. f.-b). *6.3. Preprocessing data—Scikit-learn 0.24.2 documentation*. Recuperado 9 de mayo de 2021, de https://scikit-learn.org/stable/modules/preprocessing.html

Scikit-Learn. (s. f.-c). *Sklearn.calibration.CalibratedClassifierCV — scikit-learn 0.24.2 documentation*. Scikit-Learn. Recuperado 9 de mayo de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.calibration.CalibratedClassifierCV.html

Scikit-Learn. (s. f.-d). *sklearn.calibration.calibration\_curve—Scikit-learn 0.24.2 documentation*. Scikit-Learn. Recuperado 10 de julio de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.calibration.calibration\_curve.html

Scikit-Learn. (s. f.-e). *sklearn.ensemble.RandomForestClassifier—Scikit-learn 0.24.2 documentation*. Scikit-Learn. Recuperado 11 de septiembre de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Scikit-Learn. (s. f.-f). *sklearn.metrics.brier\_score\_loss—Scikit-learn 0.24.2 documentation*. Scikit-Learn. Recuperado 10 de mayo de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.metrics.brier\_score\_loss.html

Scikit-Optimize. (s. f.). *Skopt.BayesSearchCV — scikit-optimize 0.8.1 documentation*. Recuperado 9 de mayo de 2021, de https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html

Scikit-Optimize. (2016). *Scikit-optimize: Sequential model-based optimization in Python—Scikit-optimize 0.8.1 documentation*. https://scikit-optimize.github.io/stable/

Sharma, M. (2020, marzo 20). *Grid Search for Hyperparameter Tuning*. Medium. https://towardsdatascience.com/grid-search-for-hyperparameter-tuning-9f63945e8fec

StatsBomb. (2019). *StatsBomb Open Events Structure and Data Specification v4.0.0*.

StatsBomb. (2021, marzo 4). *El Freeze Frame de StatsBomb y la cantidad de defensores entre balón y portería*. StatsBomb. http://statsbomb.com/es/2021/03/el-freeze-frame-de-statsbomb-y-los-defensores-entre-balon-y-porteria/

Sumpter, D. (2020, mayo 13). @903124S @andrew\_puopolo @the\_spearman The point of the fake data is two-fold. It allows you to include things you know that are impossible (put players never do because its impossible) and then you can push the non-linear terms to really understand how the probability of success is shaped. [Tweet]. *@Soccermatics*. https://twitter.com/Soccermatics/status/1260598182624575490

Tucker, B. (2020, febrero 15). *Random Forest is not a Calibrated Classifier*. Home. https://dataisblue.io/python/data\_science/2020/02/15/random-forest-is-not-calibrated.html

Waskom, M. (2021). seaborn: Statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Weiss, A. (2020, septiembre 7). *Charles Reep, la modernidad del pasado* [Deportiva]. La Media Inglesa. http://www.lamediainglesa.com/articulo/charles-reep-la-modernidad-del-pasado

Whitmore, J. (2021, marzo 24). What Are Expected Goals on Target (xGOT)? *The Analyst*. https://www.statsperform.com/resource/introducing-expected-goals-on-target-xgot/

Wikipedia. (s. f.). LightGBM - Wikipedia. En *Wikipedia, the free encyclopedia*. Recuperado 9 de mayo de 2021, de https://en.wikipedia.org/wiki/LightGBM

Wikipedia. (2021). Curva ROC. En *Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Curva\_ROC&oldid=138203905

Williams, A. (2020, abril 7). The roots of Expected Goals (xG) and its journey from «nerd nonsense» to the mainstream. *These Football Times*. https://thesefootballtimes.co/2020/04/08/the-roots-of-expected-goals-xg-and-its-journey-from-nerd-nonsense-to-the-mainstream/

WyScout. (2018, marzo 26). Wyscout main events description. *Wyscout FootballData*. https://footballdata.wyscout.com/events-manual/

XGBoost. (s. f.). *Python API Reference—Xgboost 1.5.0-dev documentation*. Recuperado 11 de septiembre de 2021, de https://xgboost.readthedocs.io/en/latest/python/python\_api.html

Yadav, D. (2019, diciembre 6). *Categorical encoding using Label-Encoding and One-Hot-Encoder*. Medium. https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd

# Anexos

1. https://github.com/SergiRuizUNIR/TFM [↑](#footnote-ref-1)