Cómo encontrar el mejor jugador para tu Equipo de Fútbol

CABA, Argentina. Diciembre 2024

Abstract

En la última década, el análisis deportivo ha evolucionado hacia una perspectiva cada vez más matemática y sofisticada. Aplicaciones como el uso de análisis espacial en Basketball (Goldsberry, 2012) y la investigación estadística del Brentford con Smartodds son ejemplos claros de la tendencia creciente en este campo. El béisbol, por mucho tiempo el deporte preferido para la analítica, ha experimentado una profunda transformación con la implementación de Sabermetrics (Baumer, 2015). La introducción de herramientas analíticas avanzadas ha producido resultados positivos para muchos equipos, lo que resalta el valor de estudiar métricas específicas dentro de cada deporte.

Este desarrollo se centra en el fútbol, un deporte en el cual los análisis previos se han concentrado, en su mayoría, en predecir resultados de partidos y mejorar el rendimiento de los equipos. Sin embargo, este trabajo propone un enfoque diferente al analizar el impacto de los jugadores sobre la posesión de balón y los disparos del equipo desde una perspectiva probabilística.

A partir de la métrica PSL propuesta en el paper *Soccer Networks* (Huang et al., n.d.) planteamos un proceso para comparar el impacto que tienen los jugadores sobre la performance del equipo. Logramos formular una metodología para estudiar la distribución de la performance de un equipo. Luego, proponemos una serie de métodos y métricas para comparar el rendimiento de dos formaciones de jugadores. Además, desarrollamos una forma de representación vectorial (Embeddings) de los jugadores, llamada Player2Vec, un modelo de Machine Learning también basado sobre el modelo de redes de jugadores planteado en el mismo paper del PSL. Esto último permite desarrollar modelos predictivos sobre el rendimiento de los jugadores en un equipo. Nuestro modelo final logra predecir la performance de los jugadores un 58.99% mejor que asumir las distribuciones previas como *priors*.

Palabras Clave: Fútbol, Análisis de Datos, Machine Learning, Redes de Jugadores, Embeddings , Expected Goals, Cadenas de Markov

Contents

1 Introducción

A diferencia de otros deportes como el béisbol o el basketball, el fútbol ha sido tradicionalmente menos propenso a la aplicación de técnicas avanzadas de análisis de datos y aprendizaje automático. Sin embargo, en los últimos años ha habido un crecimiento significativo en el uso de herramientas analíticas para evaluar el rendimiento de los jugadores y los equipos.

En la última década el análisis del fútbol ha evolucionado hacia una perspectiva cada vez más matemática y sofisticada. El desarrollo que más impacto tuvo sin dudas es el de la métrica de Expected Goals (**xG**) (Green, 2012), que permiten evaluar la calidad de las oportunidades de gol de un equipo. El uso de **xG** en el análisis de partidos y jugadores ha permitido una mayor capacidad predictiva y una mejor comprensión del rendimiento de los equipos. La industria que mas potenció este cambio fue la de las apuestas deportivas, que comenzó a utilizar modelos predictivos para estimar las probabilidades de los partidos. La aparición de empresas como StatsBomb y Opta Sports son claros ejemplos de como la analítica de datos ha crecido en importancia en la industria del fútbol. Tanto es asi que el Arsenal y el Brentford de la Premier League poseen sus propias empresas de analítica de datos; StatDNA y Smartodds (Tippett, 2019, p. 37).

El trabajo en desarrollo *Soccer Networks* (Huang et al., n.d.) propone un modelo de red de jugadores para calcular la probabilidad de disparar al arco antes de perder el balón (**PSL**), una métrica poca estudiada. En el paper se demuestra que el **PSL** tiene una alta correlación con el rendimiento del equipo y una gran importancia al nivel del **xG**.

Este trabajo profundiza en el análisis de la métrica **PSL** y propone un análisis probabilístico sobre las componentes del modelo de redes de jugadores y su injerencia en el rendimiento de los jugadores y consecuentemente del equipo. Proponemos una metodología para comparar el rendimiento de jugadores y formaciones de jugadores en base a la métrica **PSL**. Finalmente, desarrollamos un modelo de representación vectorial de los jugadores, llamado **Player2Vec**, para poder utilizarlo en modelos predictivos sobre el rendimiento de los jugadores.

2 Definición del problema

A partir de la pregunta de la investigación, se plantea el problema de encontrar el jugador ideal para un equipo de fútbol. En un comienzo nos encontramos planteando cómo definir la *performance* de un jugador y cómo compararla con otros jugadores. Surgió la necesidad de encontrar una métrica para evaluar el impacto de un jugador en el rendimiento de un equipo y cómo definir estos agentes. Además es necesario poder representar concretamente a un Jugador J de forma vectorial para poder utilizarlo en modelos predictivos.

2.1 PSL como métrica de Performance

En el paper en proceso *Soccer Networks* (Huang et al., n.d.) se plantea la descomposición del Gol Esperado (xG) como:

$$xG(A) = P(A) \cdot PSL(A) \cdot SA(A)$$

Donde A es el equipo, $P(A)$ es el número de posesiones del balón, $PSL(A)$ es la probabilidad de patear al arco antes de perder el balón y $SA(A)$ es la probabilidad de que un disparo al arco se convierta en gol. A diferencia de la posesión del balón y la probabilidad de convertir un disparo en gol, $PSL(A)$ no es una métrica comúnmente utilizada en el análisis de fútbol ni existen modelos que la calculen. El paper *Soccer Networks* plantea un modelo de red de jugadores que permite calcular $PSL(A)$ para cada equipo.

2.2 Modelo de Red de Jugadores

Utilizando Cadenas de Markov de Tiempo Continuo (CTMC) se puede calcular la probabilidad de que un equipo pierda el balón antes de patear al arco. En este modelo de red de jugadores se plantea un modelo de 14 estados: 11 jugadores ($J_1 \dots J_{11}$), Ganancia, Pérdida y Disparo.

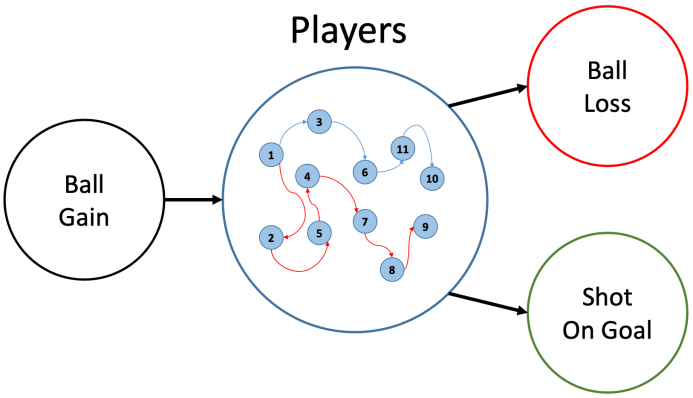


Figure 1: Modelo de Red de Jugadores

El grafo presentado en la figura 1 representa el modelo de red de jugadores. Cada nodo representa un estado y cada arista representa una transición entre estados. El nodo verde representa el estado de disparo al arco, el rojo la pérdida del balón, el negro la ganancia del balón por parte del equipo y los azules a los jugadores. Los ejes entre los nodos se representan con una matriz de adyacencia R donde cada valor $r(U, V)$ representa el ratio de transición entre los estados U y V.

$$R = \begin{pmatrix} 0 & r(G, J_1) & \dots & r(G, J_{11}) & 0 & 0 \\ 0 & 0 & \dots & r(J_1, J_{11}) & r(J_1, L) & r(J_1, S) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & r(J_{11}, J_1) & \dots & 0 & r(J_{11}, L) & r(J_{11}, S) \\ 0 & 0 & \dots & 0 & 1 & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 \end{pmatrix}$$

Los ratios de transición posibles se calculan de la siguiente manera:

$$r(G, J_i) = \frac{\text{Ganancias de } J_i}{\text{Tiempo Jugado por } J_i}$$
$$r(J_i, S) = \frac{\text{Disparos al arco de } J_i}{\text{Tiempo Jugado por } J_i}$$
$$r(J_i, S) = \frac{\text{Disparos al arco de } J_i}{\text{Tiempo Jugado por } J_i}$$
$$r(J_i, J_j) = \frac{\text{Pases de } J_i \text{ al jugador } J_j}{\text{Tiempo jugado entre } J_i \text{ y } J_j}$$

A partir de R, la matriz de ratio de acción sobre tiempo jugado (ganancias, pases, disparos o pérdidas), se puede obtener la matriz de transición de estados Q al normalizar sus filas.

Para cada par de estados U y V se define $q(U, V) = \frac{r(U,V)}{\sum_{i=1}^{14} r(U,i)}$

$$Q = \begin{pmatrix} 0 & q(G, J_1) & \dots & q(G, J_{11}) & 0 & 0 \\ 0 & 0 & \dots & q(J_1, J_{11}) & q(J_1, L) & q(J_1, S) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & q(J_{11}, J_1) & \dots & 0 & q(J_{11}, L) & q(J_{11}, S) \\ 0 & 0 & \dots & 0 & 1 & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 \end{pmatrix}$$

Finalmente a partir de la matriz de probabilidades de transición Q se puede calcular $PSL(A)$ como:

$$PSL(A) = [1, 0, ..., 0] \cdot (I - T)^{-1} \cdot X \cdot [0, 1]^T$$

Siendo T las probabilidades de transición de los estados transitorios, X las probabilidades de transición de los estados transitorios a los estados absorbentes e I la matriz identidad (Ross, 2019).

A partir de este modelo en el paper *Soccer Networks* se evaluó para una temporada de la Premier League (EPL 2012/13) (*Opta Data from Stats Perform*, n.d.) la diferencia entre los PSL de cada equipo y luego de forma empírica se demuestra como el $PSL(A)$

tiene alta correlación positiva con el rendimiento del equipo por sobre el contrincante. Finalmente hallamos una métrica significativa de rendimiento de un equipo en la métrica PSL. Sin embargo, da a lugar a la investigación de cómo se puede aplicar esta métrica a nivel de jugador y cómo se puede comparar el rendimiento de jugadores en distintos equipos.

Para evaluar el impacto de un jugador J se debe conocer la probabilidad de transición entre J y los otros 13 estados (10 jugadores, Ganancia, Pérdida y Disparo), o bien lograr estimar la probabilidad de transición entre J y los otros 13 estados.

En este trabajo se propone un método probabilístico bayesiano para hallar la Distribución del PSL dada la distribución de probabilidades de transición entre cada uno de los 11 jugadores y los otros 13 estados.

RESUMEN DISTRIBUCIONES DE PSL

2.3 Comparación de Distribuciones de PSL

En la siguiente sección postulamos una serie de métodos y métricas para comparar distribuciones de PSL de dos formaciones. En orden creciente de complejidad y rigurosidad, proponemos:

1. Comparación de Momentos Estadísticos
2. Dominancia Probabilística
3. Dominancia Estocástica

Para explicar la comparación de distribuciones de PSL, se propone un ejemplo de dos formaciones de 11 jugadores distintas, en una formación L_{MC} se encuentran 10 jugadores del equipo Manchester City (MCI) + Sergio Agüero delantero del mismo equipo y en la otra L_{MC}^{Giroud} los mismos 10 jugadores del MCI + Olivier Giroud delantero del equipo Arsenal.

Se realizó el proceso de Monte Carlo para estimar la distribución del PSL de cada formación a partir de 1000 simulaciones. Luego en la figura 2 se puede observar las funciones de densidad de probabilidad aproximadas de las distribuciones del PSL de las formaciones $\hat{f}_{PSL}^{1000}(L_{MC})$ y $\hat{f}_{PSL}^{1000}(L_{MC}^{\text{Giroud}})$.

2.3.1 Comparación de Momentos Estadísticos

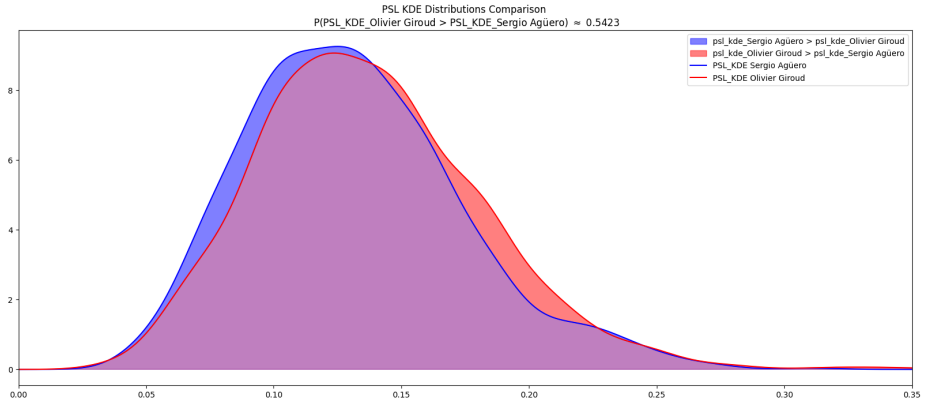


Figure 2: Ejemplo de dos distribuciones de PSL de dos formaciones distintas

Una posible comparación entre las distribuciones de PSL de dos formaciones es "a ojo" observando las funciones de densidad de probabilidad. En este caso puntual se puede observar como el equipo con Agüero tiene una distribución de PSL más desplazada a izquierda que el equipo con Giroud.

En un enfoque más numérico, se puede realizar una comparación por momentos de las distribuciones de PSL de dos formaciones. Se propone comparar la media y la varianza de las distribuciones $\hat{f}^{1000}_{PSL}(L_{MC})$ y $\hat{f}^{1000}_{PSL}(L^{\text{Giroud}}_{MC})$ ya que el método de Monte Carlo nos permite obtener una muestra significativa de las distribuciones. Al no ser distribuciones normales, la skewness y la kurtosis nos proveen información adicional sobre la forma de la distribución.

Table 1: Comparación de momentos de $\hat{f}^{1000}_{PSL}(L_{MC})$ y $\hat{f}^{1000}_{PSL}(L^{\text{Giroud}}_{MC})$

Jugador	Media	Varianza	Desvío Estándar	Skewness	Kurtosis
Aguero	0.130861	0.041160	0.001694	0.554998	0.362611
Giroud	0.134403	0.043310	0.001876	0.580404	0.405658

Para este caso de ejemplo, se observa que la media y la varianza de las distribuciones de PSL de la formación L_{MC} y L^{Giroud}_{MC} son similares, aunque mayores en la formación con Giroud. Además, el tercer momento (skewness) nos confirma lo observado "a ojo" en las funciones de densidad de probabilidad, la distribución de PSL de la formación con Agüero es más sesgada a la izquierda que la de la formación con Giroud. Por último el cuarto momento (kurtosis) nos indica que la $\hat{f}^{1000}_{PSL}(L^{\text{Giroud}}_{MC})$ tiene colas más pesadas que la $\hat{f}^{1000}_{PSL}(L_{MC})$.

2.3.2 Dominancia Probabilística

Otra forma de comparar las distribuciones de PSL de dos formaciones es a través de la dominancia probabilística.

En este caso, se puede calcular la probabilidad de que una muestra aleatoria de una distribución sea mayor que una muestra aleatoria de la otra distribución. De esta forma podemos tomar samples de las distribuciones $\hat{f}^{1000}_{PSL}(L_{MC})$ y $\hat{f}^{1000}_{PSL}(L^{\text{Giroud}}_{MC})$ y calcular la probabilidad de que un sample de la formación con Giroud sea mayor que un sample de la formación con Agüero.

Sean $X_{L_{MC}} \sim \hat{f}^{1000}_{PSL}(L_{MC})$ y $X_{L^{\text{Giroud}}_{MC}} \sim \hat{f}^{1000}_{PSL}(L^{\text{Giroud}}_{MC})$ las variables aleatorias que se distribuyen según las distribuciones de PSL de las formaciones L_{MC} y L^{Giroud}_{MC} respectivamente. Luego para evaluar si la formación con Giroud tiene dominancia probabilística sobre la formación con Agüero, se puede calcular la probabilidad $P(X_{L^{\text{Giroud}}_{MC}} > X_{L_{MC}})$.

El algoritmo para calcular la dominancia probabilística es el siguiente:

Input: Distribuciones de PSL $\hat{f}^{1000}_{PSL}(L)$ y $\hat{f}^{1000}_{PSL}(L')$
Output: Probabilidad de que un sample de PSL de la formación L sea mayor que un sample de PSL de la formación con L'

$N \leftarrow 1000$;
$M \leftarrow 0$;
for $i = 1$ **to** N **do**
    $PSL \leftarrow$ Muestrear de $\hat{f}^{1000}_{PSL}(L)$;
    $PSL' \leftarrow$ Muestrear de $\hat{f}^{1000}_{PSL}(L')$;
    if $PSL' > PSL$ **then**
        $M \leftarrow M + 1$;
    end
end
$P \leftarrow \frac{M}{N}$;

Algorithm 1: Dominancia Probabilística

Para el caso de ejemplo, se obtuvo que la probabilidad de que un sample de PSL de la formación con Giroud sea mayor que un sample de PSL de la formación con Agüero es $P(X_{L^{\text{Giroud}}_{MC}} > X_{L_{MC}}) \approx 0.5423$. De esta forma podemos concluir que la formación con Giroud tiene dominancia probabilística sobre la formación con Agüero.

2.3.3 Comparación de CDFs de las distribuciones de PSL

Otra forma de comparar las distribuciones de PSL de dos formaciones es a través de las funciones de distribución acumulada (CDF). Llamemos $\hat{F}^{N}_{PSL}(L)$ a la función de distribución acumulada de PSL obtenida a partir de N simulaciones del proceso de Monte Carlo para la formación L.

En la siguiente figura se observa la comparación de las CDFs de las distribuciones de PSL de las formaciones L_{MC} y L^{Giroud}_{MC}.

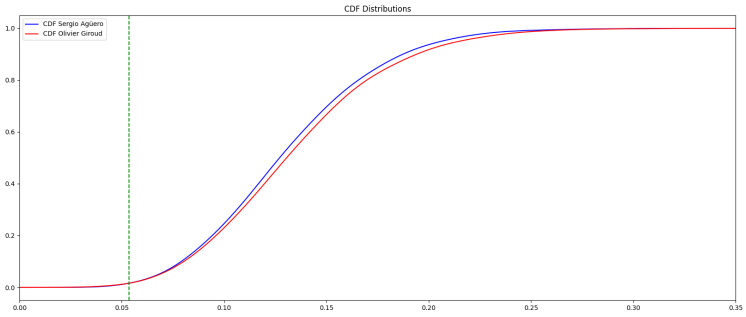


Figure 3: Comparación de CDFs de las distribuciones de PSL de las formaciones L_{MC} y L_{MC}^{Giroud}

Nuevamente "a ojo" se puede analizar la relación entre las distribuciones $\hat{F}_{PSL}^{1000}(L_{MC})$ y $\hat{F}_{PSL}^{1000}(L_{MC}^{\text{Giroud}})$, en este caso podemos ver como la CDF de la formación con Agüero es menor a la de la formación con Giroud en la mayoría de los puntos, lo que indica que la formación con Agüero tiene un PSL menor que la formación con Giroud en la mayoría de los casos.

2.3.4 Dominancia Estocástica

Más formalmente se puede evaluar la dominancia estocástica entre las CDFs $\hat{F}_{PSL}^{1000}(L_{MC})$ y $\hat{F}_{PSL}^{1000}(L_{MC}^{\text{Giroud}})$. La dominancia estocástica es una relación de orden entre dos funciones de distribución acumulada que indica si una distribución es mayor que la otra en todos los puntos.

Especificamente, podemos ver que a partir del umbral resaltado en verde en la figura 3 ($x = 0.05346757$), $\hat{F}_{PSL}^{1000}(L_{MC}^{\text{Giroud}})$ tiene **dominancia estocástica** *parcial* sobre $\hat{F}_{PSL}^{1000}(L_{MC})$ (Bawa, 1982; Vulcano, n.d.).

3 Player2Vec: Embeddings de Jugadores

Para poder representar a cada jugador de forma vectorial, se desarrolló el modelo de Player2Vec que permite obtener un embedding de cada jugador en un espacio de n dimensiones.

3.1 Definición

Player2Vec es un modelo para representar jugadores de fútbol en un espacio vectorial. Este modelo hace uso de Node2Vec, que es en sí una adaptación de Word2Vec, una técnica de NLP que permite representar palabras en un espacio vectorial (Grover & Leskovec, 2016; Mikolov et al., 2013).

3.2 Modelado de la EPL 2012/13 como Grafo

A partir de una formación de 11 (Lineup), para un equipo (Team), en un partido (Match), se construye el grafo de la red de jugadores. Llamemos a estos $G_{L,T,M}$ Grafo de Lineup.

Sean:

- $l \in L = \{0, 3\}$ las formaciones posibles (en la temporada 12/13 se permitían hasta 3 cambios de jugadores)
- $t \in T = \{\text{Local}, \text{Visitante}\}$ los equipos que jugaron el partido.
- $m \in M = \{1, 2, \ldots, 380\}$ los partidos de la temporada 12/13 de la EPL

$$
\begin{aligned}
G_{L,T,M} &= (V^{L,T,M}, E^{L,T,M}) \\
L &= \text{Número de Lineup del equipo en el partido} \\
T &= \text{Número de Equipo} \\
M &= \text{Número de Partido} \\
V^{L,T,M} &= \{\text{Gain}^{L,T,M}, J_1^{L,T,M}, J_2^{L,T,M}, \ldots, J_{11}^{L,T,M}, \text{Loss}^{L,T,M}, \text{Shot}^{L,T,M}\} \\
E^{L,T,M} &= \{(J_i^{L,T,M}, J_j^{L,T,M}, r(J_i^{L,T,M}, J_j^{L,T,M})) \mid i, j \in [1, 11]\} \\
&\quad \cup \{(\text{Gain}^{L,T,M}, J_i^{L,T,M}, r(\text{Gain}^{L,T,M}, J_i^{L,T,M})) \mid i \in [1, 11]\} \\
&\quad \cup \{(J_i^{L,T,M}, \text{Shot}^{L,T,M}, r(J_i^{L,T,M}, \text{Shot}^{L,T,M})) \mid i \in [1, 11]\} \\
&\quad \cup \{(J_i^{L,T,M}, \text{Loss}^{L,T,M}, r(J_i^{L,T,M}, \text{Loss}^{L,T,M})) \mid i \in [1, 11]\}
\end{aligned}
$$

Donde cada $J_i^{L,T,M} \mid i \in [1, 11]$ es un nodo que representa a un jugador en el lineup L del equipo T en el partido M. $Gain^{L,T,M}$ es el nodo que representa la ganancia del balón, $Loss^{L,T,M}$ la pérdida del balón y $Shot^{L,T,M}$ el disparo al arco en el lineup L del equipo T en el partido M.

Luego sean $J_i \mid i \in [0, 521]$ los jugadores reales de la temporada 2012/13 de la EPL

Se construye el grafo de la red de jugadores $G_{\text{EPL-12/13}}$ como la unión de todos los grafos de lineup $G^{L,T,M}$.

$$
\begin{aligned}
G_{\text{Full}} = (V, E) &= \bigcup_{L,T,M} G^{L,T,M} \\
V &= \{J_1, J_2, \ldots, J_{521}, Gain, Loss, Shot\} \\
&\cup \bigcup_{L,T,M} \{J_1^{L,T,M}, J_2^{L,T,M}, \ldots, J_{11}^{L,T,M}, G^{L,T,M}, L^{L,T,M}, S^{L,T,M}\} \\
E &= \bigcup_{L,T,M} E^{L,T,M} \\
&\cup \{(J_i, J_j^{L,T,M}, r(J_i, J_j^{L,T,M})) \mid i \in [0, 521], j \in [1, 11], L, T, M\} \\
&\cup \{(Gain, Gain^{L,T,M}, 1) \mid L, T, M\} \\
&\cup \{(Loss^{L,T,M}, Loss, 1) \mid L, T, M\} \\
&\cup \{(Shot^{L,T,M}, Shot, 1) \mid L, T, M\}
\end{aligned}
$$

El ratio de transición $r(J_i, J_i^{L,T,M})$ es el tiempo jugado por el Jugador J_i en el lineup L del equipo T en el partido M sobre el tiempo total jugado por el Jugador J_i

$$
r(J_i, J_i^{L,T,M}) = \frac{\text{Time Played}_{J_i^{L,T,M}}}{\text{Time Played}_{J_i}}
$$

La siguiente figura (4) es una visualización de una instancia de un Equipo en un Partido con sus lineups. En este caso el equipo hizo dos cambios en el partido (J_4 por J_{12} y J_2 por J_{13}). Se puede observar como los jugadores reales J_4 y J_{12} se encuentran representados por el mismo nodo $J_4^{L,T,M}$ y lo mismo para J_2 y J_{13} con $J_2^{L,T,M}$ para sus respectivos lineups. El resto de los nodos de jugadores reales mantienen su identidad en los grafos de lineups.

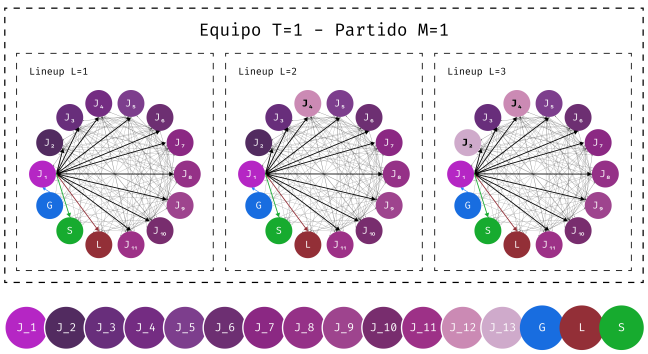


Figure 4: Grafo de Jugadores

El algoritmo en concreto para construir el grafo de la red de jugadores G_{Full} es el siguiente:

Input: Datos de eventos de partidos de la temporada 2012/13 de la EPL
Output: Grafo de la red de jugadores G_{Full}

$V \leftarrow \{J_1, J_2, \ldots, J_{521}, Gain, Loss, Shot\}$;
$E \leftarrow \emptyset$;
for *partido* M **do**
  for *lineup* L *del partido* M **do**
    for *jugador* J_i *en el lineup* L **do**
      $V \leftarrow V \cup \{J_i^{L,T,M}\}$;
      $E \leftarrow E \cup \{(J_i, J_i^{L,T,M}, r(J_i, J_i^{L,T,M}))\}$;
    end
    $V \leftarrow V \cup \{Gain^{L,T,M}, Loss^{L,T,M}, Shot^{L,T,M}\}$;
    $E \leftarrow E \cup$
     $\{(Gain, Gain^{L,T,M}, 1), (Loss^{L,T,M}, Loss, 1), (Shot^{L,T,M}, Shot, 1)\}$;
  end
end

Algorithm 2: Construcción del Grafo de Jugadores

3.3 Implementación

A partir de calcular las matrices de ratios $R^{L,T,M}$ para cada lineup L del equipo T en el partido M generamos el grafo dirigido $G^{L,T,M}$ haciendo uso de la librería `NetworkX` en Python para luego componerlos en G_{Full}, el grafo resultante contiene 37521 nodos y 47338 aristas.

Para obtener los embeddings de los jugadores, se utilizó la librería `node2vec` en Python, que implementa el algoritmo homónimo. Se configuró el modelo con una longitud de caminata de 16 nodos, 200 caminatas y un tamaño de ventana de 12 nodos. Se entrenaron 2 modelos de embeddings, uno con 64 dimensiones para utilizar en modelos de Deep Learning y otro con 3 dimensiones.

Para cada uno de los 37521 nodos se obtuvo un embedding, de los cuales nos quedamos solo con los 521 embeddings de los jugadores reales, estos finalmente son la representación vectorial de cada jugador en el espacio de embeddings.

En el caso de Player2Vec, los k random walks resultantes son una secuencia de jugadores y/o estados de juego en un partido de fútbol (Ganancia, Pérdida, Disparo). A modo ilustrativo los siguientes son posibles random walks obtenidos del grafo de la EPL 2012/13:

$$\text{Random Walk 1: Gain} \rightarrow \text{Gain}^{L,T,M} \rightarrow J_7^{L,T,M} \rightarrow \cdots \rightarrow \text{Shot}^{L,T,M} \rightarrow \text{Shot}$$

$$\text{Random Walk 2: } J_{93} \rightarrow J_{93}^{L,T,M} \rightarrow J_{15}^{L,T,M} \rightarrow J_{21}^{L,T,M} \rightarrow \text{Loss}^{L,T,M} \rightarrow \text{Loss}$$

$$\vdots$$

$$\text{Random Walk } k: \; J_{12} \rightarrow J_{12}^{L,T,M} \rightarrow J_{13}^{L,T,M} \rightarrow J_{33}^{L,T,M} \rightarrow \text{Shot}^{L,T,M} \rightarrow \text{Shot}$$

La cantidad de random walks k así como los otros hiperparametros del modelo de Node2Vec fueron seleccionados de forma empírica observando el resultado de los embeddings obtenidos.

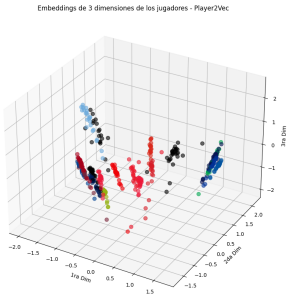
3.4 Visualización y Exploración de los Embeddings

Para comenzar a explorar el espacio vectorial generado por Player2Vec, se ajustó un modelo inicialmente a partir siguientes hiperparametros: Dimensión de embeddings: 3 - Longitud de caminata: 16 nodos - Número de caminatas: 200 - Tamaño de ventana: 12 nodos

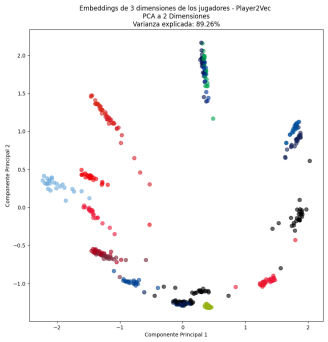
Se entrenó el modelo y se obtuvieron los embeddings de los 521 jugadores de la temporada 2012/13 de la EPL. La siguiente visualización en la figura 5.a muestra los embeddings de los jugadores en un espacio de 3 dimensiones, el color corresponde al equipo en el que juega el jugador. Para poder visualizar de forma más clara los embeddings de los jugadores, se realizó un Análisis de Componentes Principales (PCA) para reducir la dimensionalidad de los embeddings a 2 dimensiones. La visualización presente en la figura 5.b muestra los embeddings de los jugadores en un espacio de 2 dimensiones, el color corresponde al equipo en el que juega el jugador.

En la figura 5.b se observa cómo los jugadores de un mismo equipo se encuentran cercanos en el espacio vectorial, lo que indica que los embeddings resultantes de este modelo capturan las relaciones entre los jugadores de un mismo equipo. Además se observa como en este espacio las direcciones en las que se representan a los equipos divergen de forma clara, lo que indica que los embeddings capturan únicamente las diferencias entre los equipos y no las similitudes. Buscan cierta ortogonalidad entre los equipos que no logra existir en este espacio de 3 dimensiones.

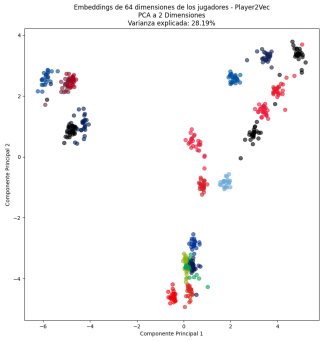
Para explotar aún más las relaciones a aprender por el modelo, se ajustó un segundo modelo con las siguientes características: Dimensión de embeddings: 64 - Longitud de caminata: 40 nodos - Número de caminatas: 500 - Tamaño de ventana: 30 nodos Luego para explorar los embeddings resultantes se realizó nuevamente un análisis de componentes principales para reducir la dimensionalidad de los embeddings a 2 dimensiones. Se puede observar en la figura 5.c como la direccionalidad de los equipos desaparece pero se mantienen las relaciones entre los jugadores de un mismo equipo.



(a) Embeddings de Jugadores en 3D



(b) Embeddings de Jugadores en 3D - PCA a 2D



(c) Embeddings de Jugadores en 64D - PCA a 2D

Figure 5: Visualización de los Embeddings de Jugadores

3.5 Potencial de Player2Vec

Con el modelo planteado de Grafos de Lineups por Equipos y Partidos se puede representar no solo una temporada de una liga, como es nuestro caso, sino que se puede extender a múltiples temporadas y ligas. Esto permitiría poder comparar jugadores de distintas ligas y temporadas, y poder evaluar el rendimiento de un jugador en distintos contextos.

Otra cuestión considerada para expandir es además de tener un nodo general por jugador conectado a sus instancias en cada lineup, se podría tener un nodo que represente a un jugador en un equipo, de forma tal que el jugador real está conectado a su nodo “Jugador en Equipo” y este nodo a su vez conectado a “Jugador en Lineup de Partido de Equipo”. Esto permitiría poder evaluar el rendimiento de un jugador en un equipo en particular y como este se comporta en distintos contextos.

En el paper de *Soccer Networks* donde se plantea el PSL definen una serie de coeficientes h, a, ω, como la performance de un equipo al jugar de local, al jugar de visitante, y la performance ponderada de todos los otros equipos al jugar de visitante respectivamente. Se podrían escalar los ratios de transición entre jugadores y el estado de disparo al arco en función de estos coeficientes para obtener una mejor representación de la performance de un jugador en un partido en particular.

4 Modelo predictivo de Distribuciones de Ratios de Transición ($r(U, V)$)

El trabajo de Player2Vec nos permite obtener embeddings de jugadores que capturan las relaciones entre ellos en un espacio vectorial. A partir de estos embeddings, se propone un modelo predictivo de las distribuciones de $r(U, V)$.

4.1 Definición

Dado un jugador J_i, se obtiene su embedding $E(J_i)$ a partir del modelo de Player2Vec. Para este J_i, se busca predecir los estadísticos Media y Varianza de las distribuciones de ratios de transición de J_i ($r(\text{Gain}, J_i)$, $r(J_i, \text{Shot})$, $r(J_i, \text{Loss})$, $r(J_i, J_j)$, $r(J_j, J_i)$)

Sean μ_{Gain,J_i} y σ_{Gain,J_i} la media y la varianza de la distribución de $r(Gain, J_i)$ respectivamente. Análogamente para las otras distribuciones.

Asumiendo normalidad, podemos decir que $r(U, V) \dot{\sim} \mathcal{N}(\mu_{U,V}, \sigma_{U,V})$.

4.2 Modelo

El modelo planteado es una Red Neuronal (NN) de la forma de 1). La función de pérdida a minimizar es una ponderación de la Divergencia de Jensen-Shannon (JSD) entre la distribución real y la predicha para cada estadístico formulada en 2), donde $D_{KL}(p||q)$ es la divergencia de Kullback-Leibler entre las distribuciones p y q y $m = \frac{p+q}{2}$ formulada en 3).

$$NN(E(J_i)) = (\mu_{\text{Gain},J_i}, \sigma_{\text{Gain},J_i}, \mu_{J_i,\text{Shot}}, \sigma_{J_i,\text{Shot}}, \mu_{J_i,\text{Loss}}, \sigma_{J_i,\text{Loss}}, \mu_{J_i,J_j}, \sigma_{J_i,J_j}, \mu_{J_j,J_i}, \sigma_{J_j,J_i}) \tag{1}$$

$$\mathcal{L} = \frac{1}{5} \sum_{U \in \{Gain, J_i, J_j\}} \sum_{V \in \{J_i, Shot, Loss, J_j\} \setminus U} \text{JSD}(\mathcal{N}(\mu_{U,V}, \sigma_{U,V}) || \mathcal{N}(\mu_{U,V}^{pred}, \sigma_{U,V}^{pred} \tag{2}$$

$$JSD(p||q) = \frac{1}{2} D_{KL}(p||m) + \frac{1}{2} D_{KL}(q||m) \tag{3}$$

4.3 Datos

Para entrenar el modelo, se utilizó un dataset de eventos de partidos de la temporada 2012/13 de la EPL. Se separó la temporada en dos mitades (190 partidos cada una). Con la primera mitad, se construyó un grafo de la red de jugadores $G_{\text{First Half}}$ y se obtuvieron los embeddings de los jugadores con Player2Vec `(dimensions=64, window=30, num_walks=500, walk_length=40)`. Con la segunda mitad, se obtuvieron las distribuciones de ratios de transición de los jugadores.

Luego, de la segunda mitad de la temporada se obtuvieron las medias y varianzas de las distribuciones de ratios de transición de los jugadores. Un 80% de los datos se utilizó para entrenar el modelo y un 20% para test.

Especificamente los 521 jugadores de la temporada 2012/13 de la EPL se dividieron en 416 para entrenamiento y 105 para test.

Table 2: Datos de Entrenamiento y Test

Conjunto	Tamaño
Entrenamiento	416
Test	105
Total	521

4.4 Implementación

El modelo se implementó en PyTorch. Se utilizó una red neuronal con la siguiente arquitectura:

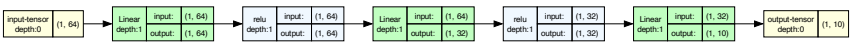


Figure 6: Arquitectura del Modelo Base

La función de pérdida utilizada fue la Divergencia de Jensen-Shannon (JSD) entre las distribuciones reales y predichas ponderando las 5 distribuciones a estimar. Se utilizó la implementación de Divergencia de Kullback-Leibler de PyTorch `nn.KLDivLoss` en la implementación del módulo `JSD` hallado en el foro de PyTorch(PyTorch Forums, 2022) y este luego se utilizó para la función de pérdida en el entrenamiento del modelo.

4.5 Entrenamiento

Para entrenar el modelo inicialmente, se utilizó el optimizador SGD (Stochastic Gradient Descent) con una tasa de aprendizaje (lr) de `0.005`, un momentum de `0.9` y un weight decay de `0.0005` para prevenir el sobreajuste.

Además, se utilizó un scheduler de tasa de aprendizaje (scheduler) con una estrategia de StepLR, que reduce la tasa de aprendizaje en un factor de `0.1` cada 100 épocas. El entrenamiento se llevó a cabo durante un máximo de 10,000 épocas, con un mecanismo de early stopping para detener el entrenamiento si no se observaban mejoras en el rendimiento del modelo en el conjunto de validación durante un número determinado de épocas consecutivas.

4.6 Resultados iniciales

El modelo base se entrenó con los hiperparámetros y arquitectura mencionados anteriormente, su entrenamiento finalizó en el epoch 2250.

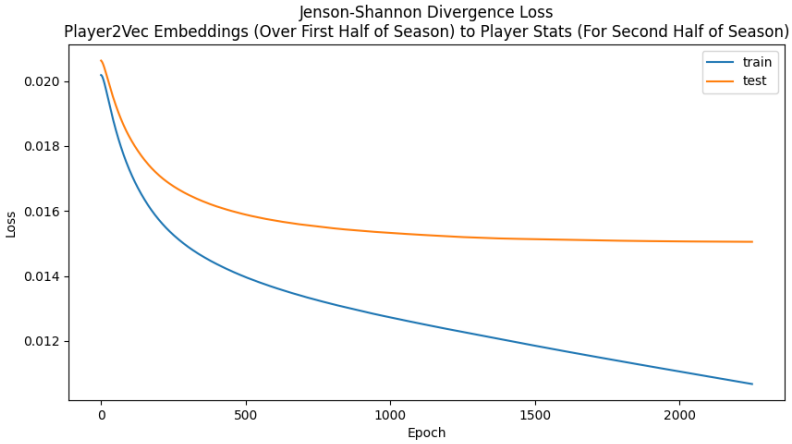


Figure 7: Resultados del Modelo Base

Table 3: Resultados del Modelo Base

Métrica	Entrenamiento	Test
Loss	0.01068	0.01505

El modelo base obtuvo un rendimiento aceptable en el conjunto de entrenamiento, con una pérdida de 0.01068, pero casi el 50% mas en el conjunto de test, con una pérdida de 0.01505. Esto puede ser un indicador de sobreajuste, por lo que se procedió a realizar un proceso de tuning de hiperparámetros y arquitectura con validación cruzada.

4.7 Tuning de Hiperparámetros y Arquitectura con Validación Cruzada

Para mejorar el rendimiento del modelo, se realizó un proceso de tuning de hiperparámetros y arquitectura con validación cruzada. Se evaluaron distintas combinaciones de hiper-

parámetros y arquitecturas de red neuronal, y se seleccionó la que mejor rendimiento presentó en el conjunto de validación.

Del 80% de los datos de entrenamiento, se separó un 20% para validación para hacer holdout CV.

Table 4: Split Datos de Entrenamiento, Validación y Test

Conjunto	**Tamaño**
Entrenamiento	332
Validación	84
Test	105
Total	521

Con la librería `Hyperopt` (Bergstra et al., 2015), se realizó una búsqueda de hiperparámetros bayesiana. Se evaluaron distintas combinaciones de hiperparámetros, y se seleccionó la que mejor rendimiento presentó en el conjunto de validación.

Además se implementó una clase de Red Neuronal paramétrica para poder explorar distintas arquitecturas de red neuronal. Se exploraron distintas combinaciones de cantidad de capas ocultas, tamaños de capas ocultas,activaciones y dropout.

4.8 Resultados del Tuning de Hiperparámetros

La búsqueda de hiperparámetros se realizó con 1000 iteraciones. La mejor combinación de hiperparámetros y arquitectura de red neuronal encontrada fue la siguiente: Tasa de aprendizaje: 0.01, Momentum: 0.9, Weight decay: 0.0001, Optimizador: `AdamW`, Scheduler: `CosineAnnealingLR` (T_max: 200), Arquitectura de red neuronal: (Activación: LeakyReLU - Dropout: 0.3 - Número de capas: 8 - Tamaños de capas ocultas: (1024, 256, 64, 256, 256, 16, 32, 512))

Figure 8: Arquitectura del Modelo Tuned

Table 5: Resultados del Modelo

	Train loss	Val loss	Test loss
0	0.004966	0.008916	0.014493

El modelo final seleccionado obtuvo un rendimiento aceptable en el conjunto de validación, con una pérdida de 0.008916, aún mejor que la pérdida de Train del modelo

base. Se procedió a evaluar el modelo en el conjunto de testeo y se obtuvo una pérdida de 0.014493.

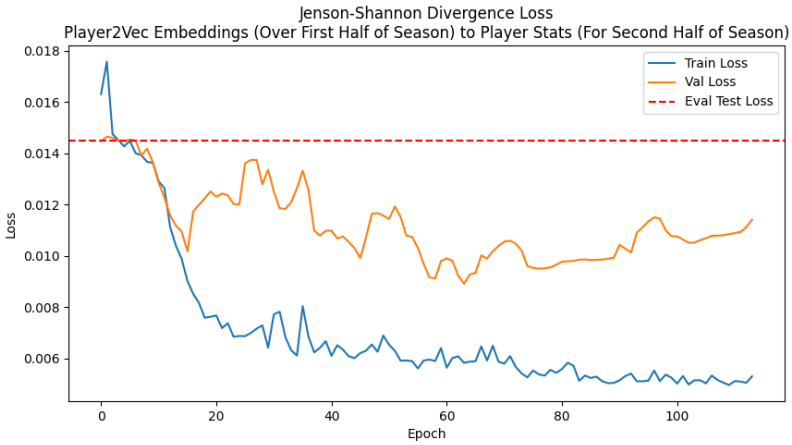


Figure 9: Resultados del Modelo Tuned

4.9 Hardware y Tiempos de Entrenamiento

El entrenamiento del modelo base se realizó en una máquina con las siguientes características: Macbook Pro M1 2020, Procesador: Apple M1, Memoria RAM: 16 GB, GPU: 8-core GPU, 16-core Neural Engine, OS: macOS Sequoia 15.0.1 (24A348),Python 3.10.14, PyTorch 2.2.2 MPS (Metal Performance Shaders)

El entrenamiento del modelo final tomó tan sólo 30 segundos a lo largo de los 115 epochs que duró el entrenamiento por el mecanismo de early stopping. La búsqueda de hiperparametros de 1000 iteraciones tomó alrededor de 7hs en la misma máquina.

4.10 Comparación contra *priors*

Para evaluar la capacidad predictiva de nuestro modelo, comparamos los resultados obtenidos con los *priors* de las distribuciones de ratios de transición de los jugadores. Los *priors* se obtuvieron a partir de las distribuciones de ratios de transición de los jugadores en la primera mitad de la temporada 2012/13 de la EPL.

Asumiendo que las distribuciones de ratios de transición de los jugadores en la segunda mitad de la temporada son similares a las de la primera mitad, evaluamos su capacidad predictiva con la misma función de pérdida JSD ponderada, obteniendo los siguientes resultados:

Table 6: Resultados del Modelo vs Priors

Priors	Modelo	Modelo Tuned
0.0353	0.01505	0.014493

Por lo que se puede observar, el modelo logra una mejora significativa en la capacidad predictiva de las distribuciones de ratios de transición de los jugadores en comparación con asumir igualdad de distribuciones entre las dos mitades de la temporada. En concreto, nuestro modelo logra reducir la pérdida en un $\sim 58.99\%$ en comparación con los *priors*.

5 Discusión

El presente trabajo presenta varias limitaciones y posibles áreas de mejora que deben ser consideradas para futuros desarrollos. En primer lugar, la cantidad de datos de entrenamiento disponibles es limitada, ya que se utilizó solo una temporada de la English Premier League (EPL). Extender el análisis a múltiples temporadas y ligas adicionales permitiría contar con un conjunto de datos más amplio y representativo. Esto no solo mejoraría la capacidad del modelo para generalizar, sino que también fortalecería su validez externa al ser probado en diferentes contextos competitivos y estilos de juego.

En cuanto a la validación del modelo, una metodología más robusta podría incluir el análisis del desempeño de los jugadores recomendados tras integrarse en nuevos equipos. Esto permitiría evaluar directamente si las estimaciones del modelo reflejan un impacto positivo en el rendimiento del equipo y del jugador, lo que fortalecería la confianza en las recomendaciones generadas.

Finalmente, es importante destacar que el modelo carece de ciertas variables contextuales clave. No se han incluido factores como el impacto del equipo rival, las estrategias del entrenador, las condiciones específicas de la fecha (por ejemplo, clima o nivel de fatiga acumulada), ni el efecto de la localía. La incorporación de estas variables podría enriquecer el análisis y permitir una comprensión más completa del entorno competitivo en el que se desarrolla el rendimiento del jugador.

En síntesis, aunque este trabajo ofrece un enfoque inicial prometedor, existe un amplio margen para mejorar la representatividad de los datos, la complejidad del modelo y la validez de las estimaciones a través de futuras extensiones y refinamientos.

6 Conclusiones

En este trabajo, se presentó un modelo predictivo de distribuciones de ratios de transición de jugadores de fútbol basado en embeddings de jugadores obtenidos con **Player2Vec**. El modelo fue entrenado y validado utilizando datos de eventos de partidos de la temporada 2012/13 de la EPL, y se evaluó su capacidad para predecir las distribuciones de ratios de transición de jugadores en un contexto de transferencias.

Se validó y presentó un caso de estudio de dos transferencias clave en las temporadas posteriores de la Premier League, demostrando la capacidad del modelo para evaluar el impacto de las transferencias en el rendimiento de los equipos. A través de la metodólogía propuesta, se encontró que el modelo es capaz de identificar transferencias que mejoran o empeoran el PSL de los equipos, proporcionando recomendaciones útiles para la toma de decisiones en el mercado de fichajes.

Los resultados obtenidos muestran que el modelo es capaz de predecir mejor que los *priors* las distribuciones de ratios de transición de los jugadores, lo que sugiere que puede ser una herramienta valiosa para la evaluación de transferencias en el fútbol profesional.

7 Referencias bibliográficas

Baumer, B. (2015). *Sabermetric revolution : Assessing the growth of analytics in baseball.* Univ Of Pennsylvania Pr.

Bawa, V. S. (1982). Stochastic dominance: A research bibliography. *Management Science*, *28*, 698–712. https://doi.org/10.1287/mnsc.28.6.698

Bergstra, J., Komer, B., Eliasmith, C., Yamins, D., & Cox, D. D. (2015). Hyperopt: A python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, *8*, 014008. https://doi.org/10.1088/1749-4699/8/1/014008

Brunetti, D., Ceria, S., Durán, G., Durán, M., Farall, A., Marucho, N., & Mislej, P. (2024). *Data science models for football scouting: The racing de santander case study*. 33rd European Conference on Operational Research. https://ic.fcen.uba.ar/uploads/files/Euro%202024%20-%20Data%20Science%20models%20for%20Football%20Scouting%20The%20Racing%20de%20Santander%20case%20study%20-%20REVISED.pdf

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 - KDD '16*, 785–794. https://doi.org/10.1145/2939672.2939785

Goldsberry, K. (2012). CourtVision : New visual and spatial analytics for the NBA. *Undefined*. https://www.semanticscholar.org/paper/CourtVision-%3A-New-Visual-and-Spatial-Analytics-for-Goldsberry/46e4a7271de62e9118625dec935c4aef1bc0ea74

Green, S. (2012). *Assessing the performance of premier league goalscorers*. Stats Perform. https://www.statsperform.com/resource/assessing-the-performance-of-premier-league-goalscorers/

Grover, A., & Leskovec, J. (2016). *node2vec: Scalable feature learning for networks*. arXiv.org. https://arxiv.org/abs/1607.00653

Huang, E., Segarra, S., Gallino, S., & Ribeiro, A. (n.d.). *How to find the right player for your soccer team?*

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*, 79–86. https://doi.org/10.1214/aoms/1177729694

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space*. arXiv.org. https://arxiv.org/abs/1301.3781

Opta data from stats perform. (n.d.). Stats Perform. https://www.statsperform.com/opta/

PyTorch Forums, A. N. K. -. (2022). *Jensen shannon divergence*. PyTorch Forums.

https://discuss.pytorch.org/t/jensen-shannon-divergence/2626/13
Rahimian, P., Van Haaren, J., & Toka, L. (2023). Towards maximizing expected possession outcome in soccer. *International Journal of Sports Science & Coaching*, 174795412311544. https://doi.org/10.1177/17479541231154494
Ross, S. M. (2019). *Introduction to probability models.* Academic Press.
Tippett, J. (2019). *The expected goals philosophy: A game-changing way of analysing football*. Independently Published.
Transfermarkt.com.ar. (2024a). *Danny welbeck - evolución del valor de mercado*. Transfermarkt.com.ar. https://www.transfermarkt.com.ar/danny-welbeck/marktwertverlauf/spieler/67063
Transfermarkt.com.ar. (2024b). *James milner - stats by club*. Transfermarkt.com. https://www.transfermarkt.com/james-milner/leistungsdatenverein/spieler/3333
Vulcano, G. (n.d.). *Decision under risk - module IV - NYU stern - master of science in business analytics*.