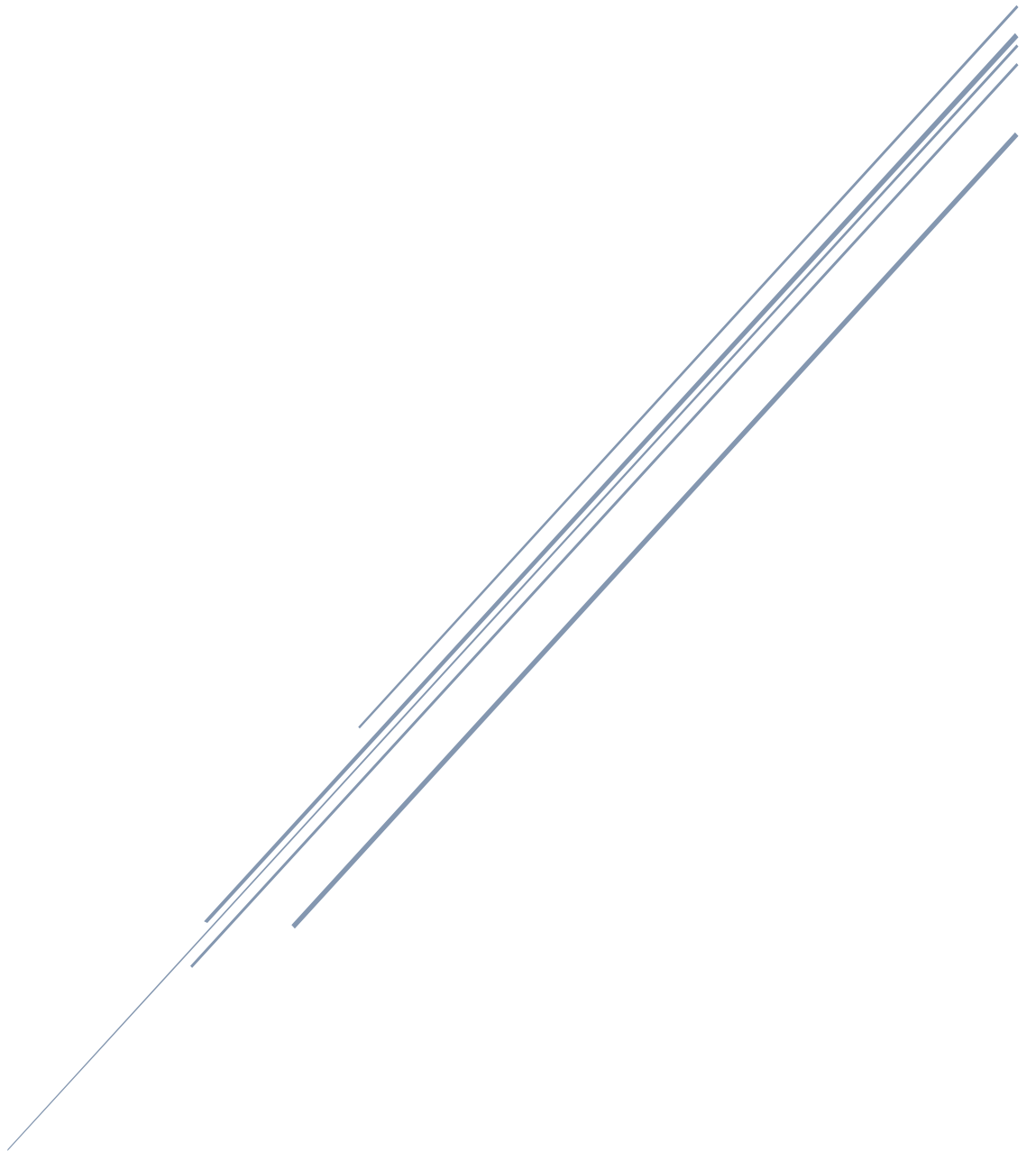


# MONEY FIGHTER

FRANCISCO RODRIGUEZ LOZANO



KSCHOOL  
DATA SCIENCE

# INTRODUCCIÓN

---

## ARTES MARCIALES MIXTAS

Las artes marciales mixtas (MMA) son una combinación de técnicas de distintos deportes y artes marciales como el boxeo, karate, jiu jitsu brasileño, lucha libre, muay thai, etc. Es el deporte de combate de más rápido crecimiento de los últimos años y está atrayendo a un gran número de aficionados a nivel mundial.

Inicialmente, se caracterizaba por sus escasas reglas y tenía el objetivo de demostrar qué disciplina de combate era superior. Más tarde, los peleadores empezaron a adoptar técnicas de otras artes marciales e incorporarlas a su estilo y se empezó a promover la seguridad en las peleas aplicando nuevas reglas que y profesionalizando el deporte.

Al ser un deporte con una variedad técnica tan amplia, existen diferentes estilos de pelea. Un peleador más centrado en el *striking* tendrá unas mejores estadísticas en golpeo, pero suelen ser peores en estadísticas de grappling y jiu jitsu como derribos y sumisiones. Estas cualidades pueden ser algo diferencial al aplicar la estrategia para una pelea y a su vez en el resultado.

Actualmente, las MMA son un deporte y un negocio gestionado por compañías como Ultimate Fighting Championship (UFC), Bellator o One Championship. UFC es la mayor empresa de MMA albergando los mejores peleadores y produciendo eventos por todo el mundo.

## ESTADO DEL ARTE

El problema de clasificación binaria (0/1) es uno de los más comunes en el ámbito de la ciencia de datos y el aprendizaje automático. El estado del arte actual en este tipo de problemas incluye la utilización de modelos de aprendizaje supervisado, como árboles de decisión, random forests, redes neuronales, entre otros. Además, se ha experimentado con técnicas de aprendizaje no supervisado, como el clustering, para abordar problemas de clasificación binaria. Estas técnicas buscan identificar patrones o grupos en los datos sin tener una etiqueta previa para cada muestra.

En resumen, el estado del arte en problemas de clasificación binaria incluye la utilización de diferentes modelos de aprendizaje supervisado y no supervisado, así como técnicas de preprocesamiento de datos y selección de características. La elección del método óptimo depende del problema específico y de los requisitos del proyecto.

En cuanto a predicción de resultados en deportes de combate, se ha estudiado la literatura existente y aparecen una gran cuantía de artículos y documentos relacionados, pero se centran en la predicción de lesiones, no en la predicción del ganador.

## OBJETIVOS

Los deportes de contacto, y sobre todo las MMA, son un deporte muy variado con una gran diversidad de estilos de pelea, técnicas y estrategias muy diferentes, dando lugar a

resultados, a veces, sorprendentes. Sin embargo, el objetivo de este proyecto es determinar la influencia del estilo del peleador y su rival en el resultado de una pelea usando las estadísticas de golpeo y grappling.

Los objetivos de este proyecto son los siguientes:

- Crear un modelo de predicción del ganador de una pelea de MMA que supere la probabilidad aleatoria inicial del 50%
- Evaluar si es posible elaborar un modelo de predicción del resultado de una pelea de MMA teniendo en cuenta tan solo las cualidades técnicas y físicas de ambos peleadores

## METODOLOGIA

---

### LIBRERIAS UTILIZADAS

1. **Pandas:** Es una biblioteca de código abierto que permite la manipulación y análisis de datos de forma rápida y eficiente en Python. Proporciona estructuras de datos de alto nivel como DataFrames y Series, que facilitan el trabajo con datos tabulares y estadísticos.
2. **Numpy:** Es una biblioteca para el cálculo numérico en Python. Ofrece una amplia gama de funciones matemáticas y de álgebra lineal para trabajar con arrays multidimensionales. Es muy útil para tareas de cálculo científico y técnico.
3. **Matplotlib:** Es una biblioteca de gráficos en Python que permite crear una amplia variedad de gráficos y visualizaciones de datos, incluyendo gráficos de barras, líneas, scatterplots, histogramas, entre otros.
4. **Seaborn:** Es una biblioteca de visualización de datos basada en Matplotlib que proporciona una amplia gama de gráficos estadísticos y estéticamente atractivos. Permite crear gráficos complejos de forma fácil y rápida.
5. **Streamlit:** Es una biblioteca de Python para crear aplicaciones web interactivas. Permite crear aplicaciones web con una interfaz de usuario amigable y altamente interactiva para visualizar y explorar datos.
6. **Sklearn:** Es una biblioteca de aprendizaje automático de código abierto en Python que incluye una amplia gama de algoritmos de aprendizaje automático, incluyendo regresión, clasificación, clustering, entre otros.
7. **Tensorflow Keras:** Es una biblioteca de inteligencia artificial y aprendizaje profundo que permite crear y entrenar modelos de deep learning de forma fácil y eficiente. Se basa en Tensorflow, una biblioteca de código abierto para computación numérica en tiempo real.
8. **Time:** Es una biblioteca que proporciona funciones y estructuras para trabajar con tiempos y fechas en Python. Permite realizar tareas como medir el tiempo de ejecución de un programa, convertir entre diferentes formatos de fecha y hora, entre otros.

9. **Requests:** Es una biblioteca de Python que permite realizar solicitudes HTTP de forma fácil y rápida. Se utiliza comúnmente para acceder a información desde páginas web y APIs.
10. **BeautifulSoup:** es una librería en Python para la extracción y manipulación de datos de páginas web, útil para el web scraping.

## OBTENCION DE LOS DATOS

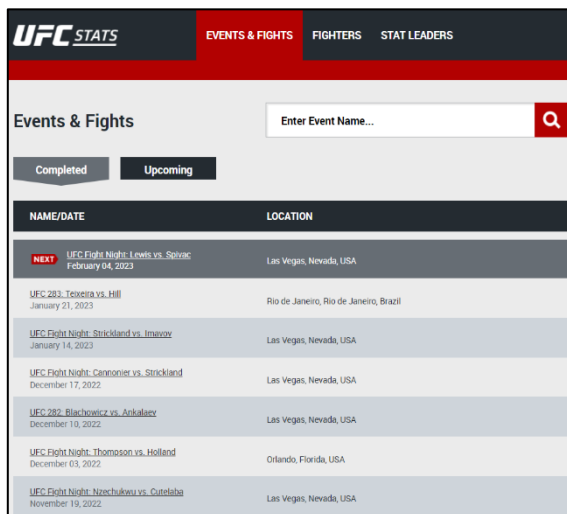
Para llevar a cabo el proyecto se han adquirido los datos mediante Web Scraping. El Web Scraping es una técnica que consiste en extraer automáticamente información de sitios web a través de programas o scripts.

En este caso se utilizó Python, concretamente las librerías BeautifulSoup y Pandas para extraer los datos de las páginas web. Primero, identificamos las fuentes relevantes de información en línea relacionadas con nuestro tema de estudio, en este caso las páginas oficiales de estadísticas de la UFC. Luego, desarrollamos scripts para extraer la información desde las páginas web y almacenarla en un formato estructurado (CSV).

## WEB SCRAPING

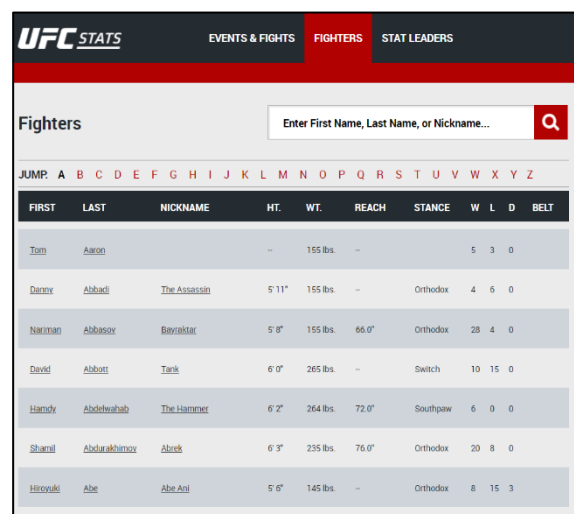
La página utilizada para obtener las estadísticas de los peleadores y del histórico de peleas ha sido [www.ufcstats.com](http://www.ufcstats.com).

A

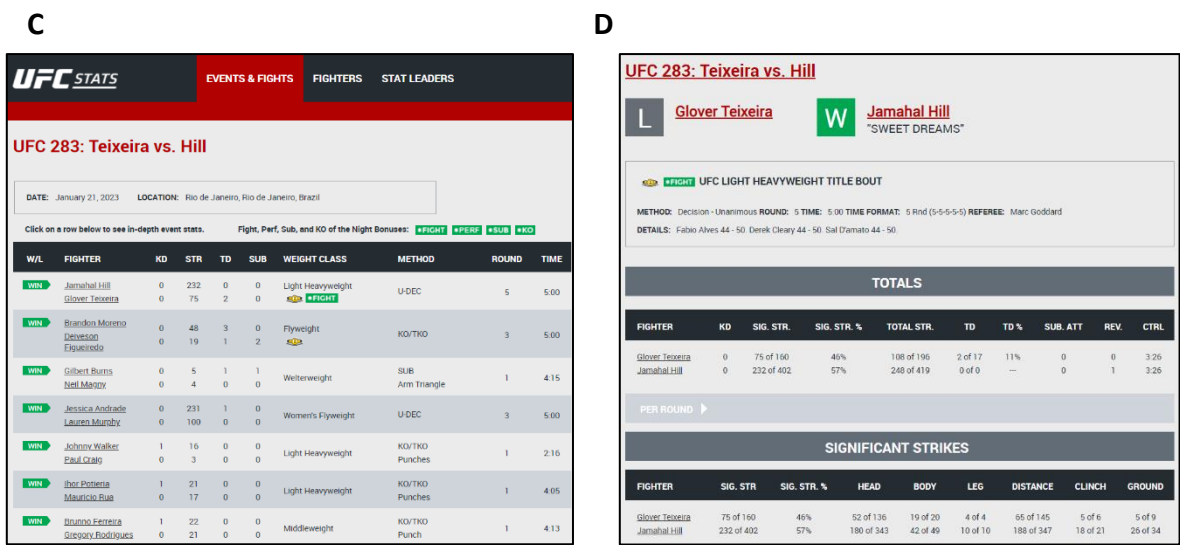


NAME/DATE	LOCATION
<b>NEXT</b> UFC Fight Night: Lewis vs. Solov February 04, 2023	Las Vegas, Nevada, USA
UFC 283: Teixeira vs. Hill January 21, 2023	Rio de Janeiro, Rio de Janeiro, Brazil
UFC Fight Night: Strickland vs. Imavov January 14, 2023	Las Vegas, Nevada, USA
UFC Fight Night: Cannonier vs. Strickland December 17, 2022	Las Vegas, Nevada, USA
UFC 282: Blachowicz vs. Ankalaev December 10, 2022	Las Vegas, Nevada, USA
UFC Fight Night: Thompson vs. Holland December 03, 2022	Orlando, Florida, USA
UFC Fight Night: Nzechukwu vs. Outelaba November 19, 2022	Las Vegas, Nevada, USA

B



FIRST	LAST	NICKNAME	HT.	WT.	REACH	STANCE	W	L	D	BELT
Tom	Aaron		5' 11"	155 lbs	70"	Orthodox	5	3	0	
Danny	Abduli	The Assassin	5' 11"	155 lbs	70"	Orthodox	4	6	0	
Nikolai	Ablesov	Bayraktar	5' 8"	155 lbs	66.0"	Orthodox	28	4	0	
David	Abbot	Tank	6' 0"	265 lbs	70"	Switch	10	15	0	
Handy	Abdelhakim	The Hammer	6' 2"	264 lbs	72.0"	Southpaw	6	0	0	
Shamil	Abdulkhaimov	Abrek	6' 3"	235 lbs	76.0"	Orthodox	20	8	0	
Hiroaki	Abe	Abe Ani	5' 6"	145 lbs	70"	Orthodox	8	15	3	



**Fig 1: UFC STATS. Página oficial de UFC. A** Página con el listado de todos los eventos realizados por UFC en su historia. **B** Página con el listado de los peleadores (activos e inactivos) que han pasado por UFC. **C** Ejemplo ultimo evento UFC y las estadísticas generales de cada pelea del evento. **D** Ejemplo de pagina de estadísticas específicas de cada pelea.

En primer lugar, se obtuvo toda la información general tanto de los eventos como de los peleadores y los links a los datos específicos de cada pelea de cada evento y las estadísticas concretas de cada pelador (Fig 1A y 1B).

Posteriormente, se extrajo la información de cada pelea de los links de eventos recopilado previamente y las estadísticas concretas de cada pelea (Fig 1C y 1D) y se unieron a los datos del evento. Así se consiguió, por un lado, un dataset de todas las peleas de UFC y sus estadísticas (Fig 4), y por otro, un dataset de las cualidades técnicas y físicas de cada peleador de la compañía (Fig 3).

	First	Last	Nickname	Ht.	Wt.	Reach	Stance	W	L	D	Belt	enlaces	Height	Weight	Reach	STANCE	DOB	SLpM	Str. Acc.	SAPM	Str. Def	TD Avg.	TD Acc.	TD Def.	Sub. Avg.
0	Tom	Aaron		NaN	--	155 lbs.	--	NaN	5.0	3.0	0.0	NaN	<a href="http://ufcstats.com/fighter-details/93fe7332d1...">http://ufcstats.com/fighter-details/93fe7332d1...</a>	--	155 lbs.	--	Jul 13, 1978	0.00	0%	0.00	0%	0.00	0%	0%	0.0
1	Danny	Abbad	The Assassin	5' 11"	155 lbs.	--	Orthodox	4.0	6.0	0.0	NaN	<a href="http://ufcstats.com/fighter-details/15df64c02b...">http://ufcstats.com/fighter-details/15df64c02b...</a>	5' 11"	155 lbs.	--	Orthodox	Jul 03, 1983	3.29	38%	4.41	57%	0.00	0%	77%	0.0
2	Nariman	Abbasov	Bayraktar	5' 8"	155 lbs.	66.0"	Orthodox	28.0	4.0	0.0	NaN	<a href="http://ufcstats.com/fighter-details/59a9d6dac6...">http://ufcstats.com/fighter-details/59a9d6dac6...</a>	5' 8"	155 lbs.	66"	Orthodox	Feb 01, 1994	3.00	20%	5.67	46%	0.00	0%	66%	0.0
3	David	Abbott	Tank	6' 0"	265 lbs.	--	Switch	10.0	15.0	0.0	NaN	<a href="http://ufcstats.com/fighter-details/b361180739...">http://ufcstats.com/fighter-details/b361180739...</a>	6' 0"	265 lbs.	--	Switch	--	1.35	30%	3.55	38%	1.07	33%	66%	0.0
4	Hamdy	Abdelwahab	The Hammer	6' 2"	264 lbs.	72.0"	Southpaw	6.0	0.0	0.0	NaN	<a href="http://ufcstats.com/fighter-details/3329d692ae...">http://ufcstats.com/fighter-details/3329d692ae...</a>	6' 2"	264 lbs.	72"	Southpaw	Jan 22, 1993	3.87	52%	3.13	59%	3.00	75%	0%	0.0

**Fig 3. Dataset peleadores (fighters.csv).** Datos crudos de peleadores y sus estadísticas actuales.

W/L	Fighter	Kd	Str	Sub	Weight class	Method	Round	Time	date_fight	link	Sig. str.	Total str.	Td	Sub. att	Rev.	Ctrl	
0	win	Jared Cannonier Sean Strickland	0 0	141 152	0 0	Middleweight	S-DEC	5.0	5:00	UFC Fight Night: Cannonier vs. Strickland Dec...	<a href="http://ufcstats.com/fight-details/4ea48bf2407b...">http://ufcstats.com/fight-details/4ea48bf2407b...</a>	141 of 310 152 of 400	141 of 310 157 of 410	0 of 1	0 0	0 0	0:00 0:48
1	win	Arman Tsarukyan Damir Ismagulov	0 0	34 36	0 0	Lightweight	U-DEC	3.0	5:00	UFC Fight Night: Cannonier vs. Strickland Dec...	<a href="http://ufcstats.com/fight-details/ce7744d45a69...">http://ufcstats.com/fight-details/ce7744d45a69...</a>	34 of 81 81 of 86	50 of 105 51 of 106	7 of 21 0 of 0	0 0	0 0	9:25 0:06
2	win	Amir Albazi Alessandro Costa	2 0	37 17	0 0	Flyweight	KO/TKO Punch	3.0	2:13	UFC Fight Night: Cannonier vs. Strickland Dec...	<a href="http://ufcstats.com/fight-details/d95eeba3de7e...">http://ufcstats.com/fight-details/d95eeba3de7e...</a>	37 of 81 17 of 71	62 of 110 38 of 107	1 of 4 0 of 0	0 0	0 0	5:54 0:00
3	win	Alex Caceres Julian Erosa	1 0	16 10	0 0	Featherweight	KO/TKO Kick	1.0	3:04	UFC Fight Night: Cannonier vs. Strickland Dec...	<a href="http://ufcstats.com/fight-details/9c1a2ac64c98...">http://ufcstats.com/fight-details/9c1a2ac64c98...</a>	16 of 30 10 of 22	16 of 30 10 of 22	0 of 0 0 of 0	0 0	0 0	0:05 0:00
4	win	Drew Dober Bobby Green	1 0	34 73	0 0	Lightweight	KO/TKO Punch	2.0	2:45	UFC Fight Night: Cannonier vs. Strickland Dec...	<a href="http://ufcstats.com/fight-details/c335f44e4f35...">http://ufcstats.com/fight-details/c335f44e4f35...</a>	34 of 121 73 of 142	34 of 121 75 of 145	0 of 1 0 of 1	0 0	0 0	0:07 0:11

**Fig 4. Dataset peleas (bouts\_UFC.csv).** Datos crudos de las peleas y las estadísticas principales.

Una vez completada la adquisición de los datos se obtuvo un dataset de peleas (BOUTS) y uno de peleadores (FIGHTERS). BOUTS contiene un total de 6914 peleas y FIGHTERS contiene las estadísticas de un total de 3915 peleadores.

#### DESCRIPCION VARIABLES

**Tabla 1: Descripción de los datos en bruto.**

Dataset	Variable	Descripción	Comentario
bouts.csv	W/L	Resultado de la pelea	Siempre el ganador es el primer peleador que aparece. Se eliminará esa variable.
bouts.csv	Fighter	Nombre y apellido de los peleadores enfrentados	Separar en dos
bouts.csv	Kd	Numero de Knock downs de cada peleador	Separar en dos
bouts.csv	Str	Numero de golpes conectados	Columna duplicada (Total str.). Se eliminará Str
bouts.csv	Sub	Intentos de sumisión	Columna duplicada (Sub att.). Se eliminará una de ellas.
bouts.csv	Weight Class	Categoría de peso en la que se ha pactado la pelea	
bouts.csv	Method	Como acaba la pelea: KO, Decisión, Sumisión	Simplificar tipos de finalización de la pelea a las tres principales
bouts.csv	Round	Nº de rounds de la pelea	
bouts.csv	Time	Tiempo transcurrido en el último asalto disputado	
bouts.csv	date_fight	Nombre del evento y fecha de realización	Simplificar solo a la fecha
bouts.csv	link	Link a las estadísticas de cada pelea para realizar comprobaciones del correcto funcionamiento del script de web scraping	Se eliminará

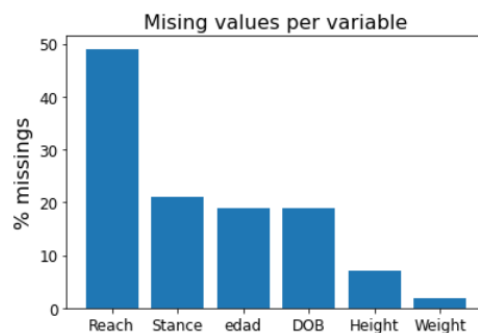
bouts.csv	Sig. str.	Golpes significativos conectados y totales de ambos peleadores	Separar en dos
bouts.csv	Total str.	Golpes conectados y totales de ambos peleadores	Separar en dos
bouts.csv	Sub. att.	Intentos de sumisión	Separar en dos
bouts.csv	Rev	NA	Separar en dos
bouts.csv	Ctrl.	Tiempo de control de la pelea de ambos peleadores	Separar en dos
fifghters.csv	Ht.	Altura (Pies y pulgadas)	Columna duplicada. Se eliminará una de ellas.
fifghters.csv	WL	Peso (libras)	Columna duplicada. Se eliminará una de ellas.
fifghters.csv	Stance	Tipo de guardia del peleador	Columna duplicada. Se eliminará una de ellas.
fifghters.csv	W	Nº peleas ganadas	
fifghters.csv	L	Nº peleas perdidas	
fifghters.csv	D	Nº peleas empatadas	
fifghters.csv	Belt	Indica si el peleador es el campeón de su division	Se eliminará, son muy pocos los campeones respecto al total de peleadores y peleas
fifghters.csv	enlace	Enlaces de las peleas para la revisión de la adquisición correcta de los datos	
fifghters.csv	Height	Altura (Pies y pulgadas)	
fifghters.csv	Weight	Peso (libras)	
fifghters.csv	Reach	Alcance del peleador. Distancia desde el extremo de un brazo y el otro. (Pulgadas)	
fifghters.csv	STANCE	Tipo de guardia del peleador.	
fifghters.csv	DOB	Fecha de nacimiento del peleador	
fifghters.csv	SLpM	Golpes significativos conectados por minuto	
fifghters.csv	Str. Acc.	Precisión los golpes significativos	
fifghters.csv	SAPM	Golpes significativos recibidos por minuto	
fifghters.csv	Str. Def	Porcentaje de golpes significativos evitados	
fifghters.csv	TD Avg.	Media de intentos de derribo completados por cada 15 minutos	
fifghters.csv	TD Acc.	Precisión en los derribos	
fifghters.csv	TD Def.	Porcentaje de derribos evitados	
fifghters.csv	Sub. Avg.	Media de sumisiones intentadas por cada 15 minutos	

## DATA CLEANING AND ANALYSIS

Una vez obtenidos los datos, se llevó a cabo la limpieza de los mismos. En primer lugar, se separaron los datos del primer y el segundo peleador del dataset BOUTS. También se eliminaron caracteres de unidades y se crearon nuevas columnas para separar el total, de los conectados de las estadísticas de pelea (por ejemplo, separa el número de golpes significativos conectados de los golpes totales). Se eliminaron variables duplicadas y se unificaron las unidades de medida de altura, alcance (cm) y tiempo (segundos o minutos).

A continuación, se limpiaron los datos del dataset FIGHTERS. Se ajustaron los tipos de datos de cada variable y se eliminaron caracteres de unidades. Además, se calculó la variable “Edad” con la columna DOB, y se llevó a cabo un análisis de las variables que usaremos como predictoras.

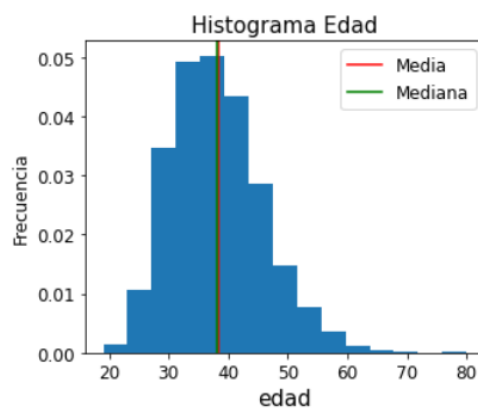
En este análisis, observamos que hay valores faltantes en varias columnas:



**Figura 5: Porcentaje de valores faltantes en cada variable.**

Como se observa en la figura 5, las columnas “Reach”, “Stance”, “edad”, “DOB” y “Height” son las que presentan un mayor porcentaje de valores faltantes.

La variable “DOB” se eliminará y nos quedaremos con la variable “edad”. Para solucionar los valores faltantes de edad, se estudió la distribución de la variable:



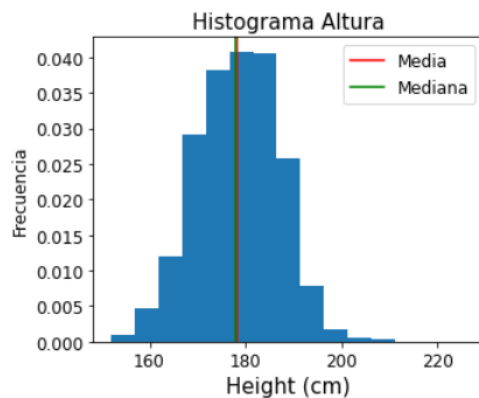
**Figura 6: Histograma de la variable “edad”.** Se representa la media y la mediana del total de peleadores como líneas verticales.



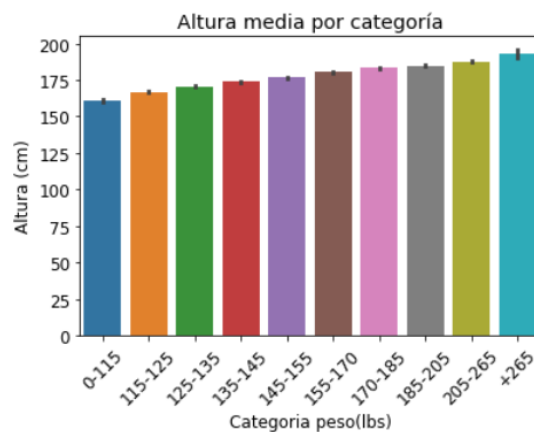
La variable “edad” muestra una distribución normal por lo que sustituiremos los valores por la media.

A continuación, se representó la distribución de la variable “Height” en ella también se observa una distribución normal. Sin embargo, se estudió también su variabilidad según el rango de peso en el que se encuentran:

**A**



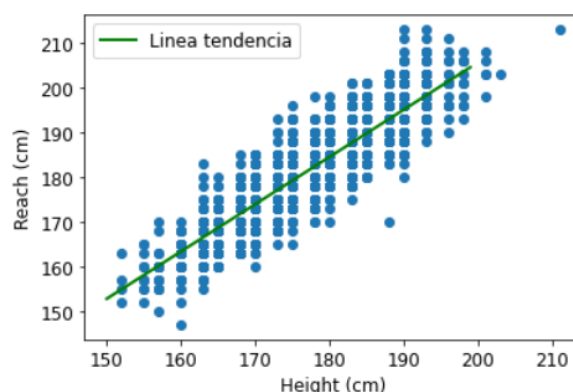
**B**



**Figura 7: Distribucion del avariable “Height”.** **A** Histograma de la altura de los peleadores. La media y la mediana se han representado com lineas verticales. **B** Gráfica de barras de la altura media de los peleadores según su categoria de peso.

Se observa que a pesos más bajos la altura es menor que a pesos altos por lo que una aproximación más exacta sería sustituir los valores faltantes por la media de la altura de su división de peso.

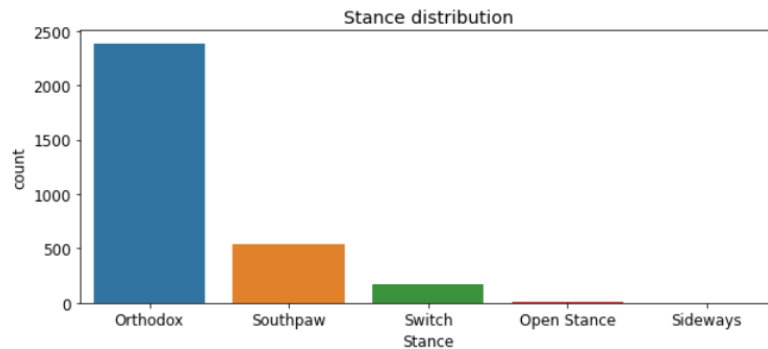
Por otro lado, se estudió si había una relación entre las variables “Height” y “Reach” y se observó que había una relación lineal entre ellas. Esto se apoya en que anatómicamente, la envergadura o alcance es directamente proporcional a la altura 1:1.



**Figura 8: Gráfica de dispersion entre las variables “Reach” y “Height”.** La relacion entre ambas variables se muestra mediante la linea de tendencia.

Se sustituyeron los valores faltantes de “Reach” por el mismo valor que la altura.

Por último, se representó un countplot de la variable “Stance” que representa la guardia del peleador:

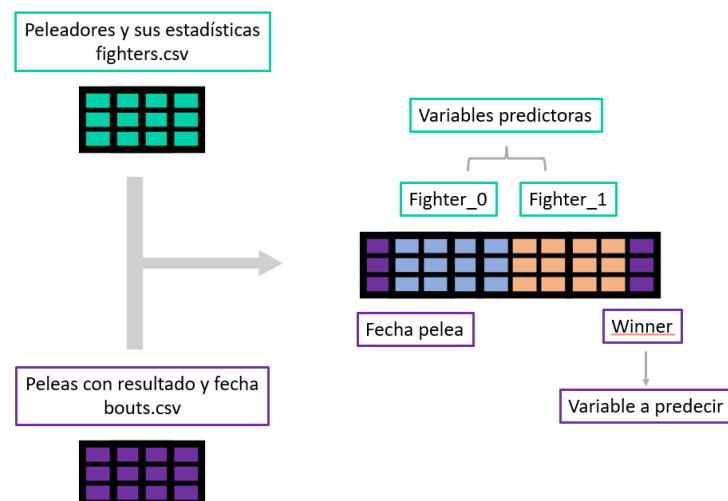


**Figura 8: Grafica de barras del total de peleadores con cada tipo de guardia.**

Se observó que la guardia “Orthodox” es la más habitual con diferencia al resto. Por ello, se sustituyeron los valores faltantes por el más frecuente.

El resto filas con valores faltantes se eliminaron ya que era un porcentaje bajo del total de peleadores. Además, se eliminaron los peleadores duplicados quedando con un total de 3842.

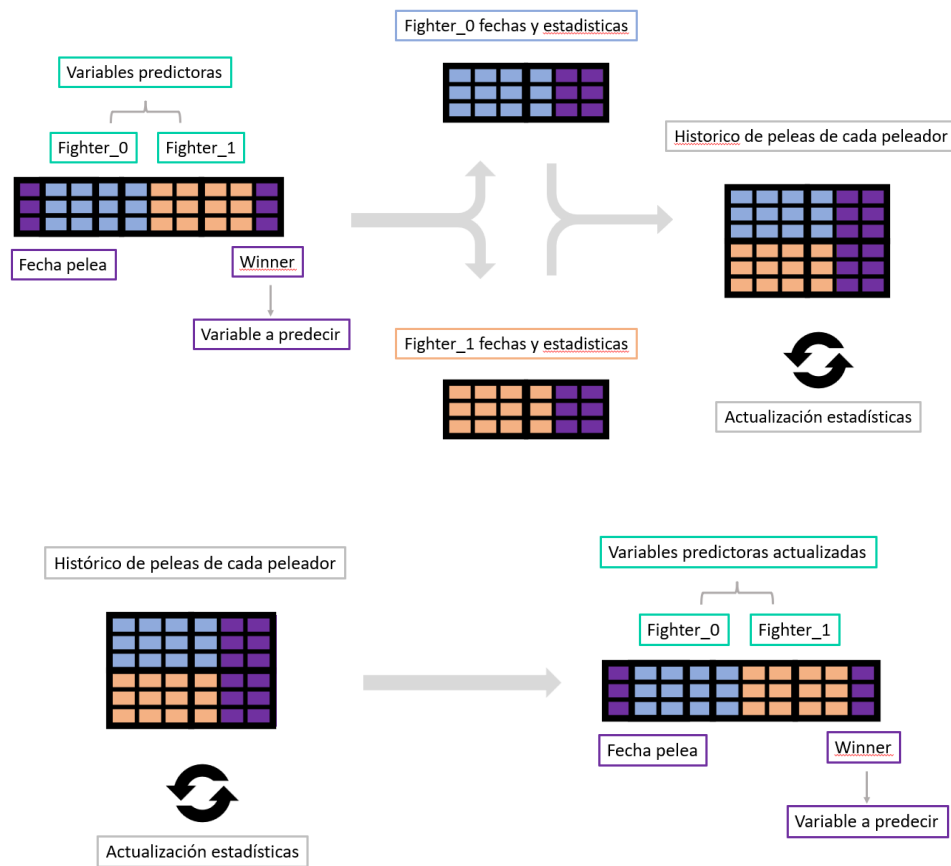
Por último, se combinaron ambos dataset consiguiendo tener en cada pelea las estadísticas de cada luchador y se eliminaron las variables que no vamos a utilizar (Figura 9).



**Figura 9: Diagrama de flujo de datos desde la adquisición a la unión en un único dataset.**

Una vez obtenido el dataset se crearon las variables “Racha” y “ufc\_bouts”, y se actualizaron las variables “edad”, “W” y “L” a la fecha de la pelea. Para ello, se hizo un

histórico de peleas de cada peleador uniendo las columnas \_0 y \_1. Con esto conseguimos unificar los datos en una única columna a los peleadores “locales” y “visitantes” para ordenar por fecha y poder calcular sus valores actualizados. A continuación, se devuelven estos datos a sus posiciones originales para volver al dataset inicial. Se muestra este proceso en la Figura 10.



**Figura 10: Diagrama de flujo de datos para la actualización de las variables por fecha de la pelea y su vuelta a la posición original.**

Finalmente se calculó la diferencia entre las variables del peleador 0 y del peleador 1 para obtener el dataset que se usaría para entrenar al modelo y evitar que el orden de los peleadores influya en la predicción (Tabla 2). Se obtuvo un dataset de 6707 peleas y 19 variables (Tabla 2). Se hizo una primera aproximación de la correlación de las variables predictoras y la variable a predecir: “winner” (Figura 11).

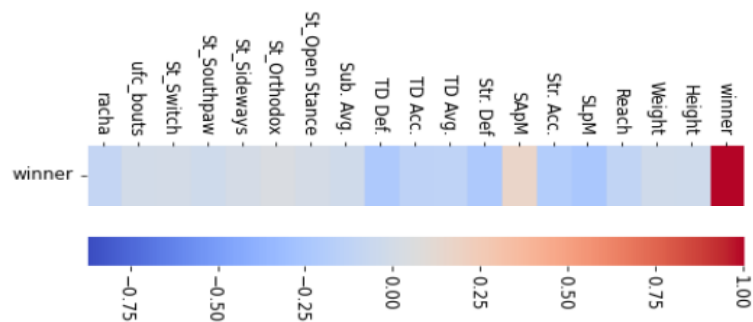


Figura 11: Correlación entre las variables predictoras y la variable “winner”.

Tabla 2: Variables finales

Variable	Descripción	Comentario
<b>Height</b>	Diferencia altura peleadores	
<b>Weight</b>	Diferencia peso peleadores	
<b>Reach</b>	Diferencia alcance peleadores	
<b>SLpM</b>	Diferencia golpes significativos por minuto peleadores	
<b>Str. Acc</b>	Diferencia precisión striking peleadores	
<b>SAPM</b>	Diferencia de golpes significativos absorbidos por los peleadores.	
<b>Str. Def</b>	Diferencia en la defensa de striking de los peleadores. La defensa de striking se representa con el porcentaje de los golpes del adversario que no te han golpeado	
<b>TD Avg.</b>	Diferencia de la media de derribos conseguidos cada 15 minutos de los peleadores	
<b>TD Acc.</b>	Diferencia de la precisión de derribo de los peleadores	
<b>TD Def.</b>	Diferencia en la defensa de derribo de los peleadores. La defensa de derribos se representa con el porcentaje de intentos de derribo del adversario que no te han derribado	
<b>Sub. Avg</b>	Diferencia de la media intentos de sumision cada 15 minutos de los peleadores	
<b>St_Open Stance</b>	Representa si los peleadores usan esa guardia de pelea	-1 si solo el peleador 0 usa esa guardia 0 si ninguno o ambos usan esa guardia 1 si solo el peleador 1 usa esa guardia
<b>St_Orthodox</b>	Representa si los peleadores usan esa guardia de pelea	-1 si solo el peleador 0 usa esa guardia 0 si ninguno o ambos usan esa guardia

		1 si solo el peleador 1 usa esa guardia
<b>St_Sideways</b>	Representa si los peleadores usan esa guardia de pelea	-1 si solo el peleador 0 usa esa guardia 0 si ninguno o ambos usan esa guardia 1 si solo el peleador 1 usa esa guardia
<b>St_Southpaw</b>	Representa si los peleadores usan esa guardia de pelea	-1 si solo el peleador 0 usa esa guardia 0 si ninguno o ambos usan esa guardia 1 si solo el peleador 1 usa esa guardia
<b>St_Switch</b>	Representa si los peleadores usan esa guardia de pelea	-1 si solo el peleador 0 usa esa guardia 0 si ninguno o ambos usan esa guardia 1 si solo el peleador 1 usa esa guardia
<b>ufc_bouts</b>	Diferencia del número de peleas dentro de la ufc	La UFC es la mejor promotora de eventos de MMA. Los peleadores con una larga carrera en UFC suelen ser mejores.
<b>racha</b>	Representa la diferencia de la racha de ambos peleadores	La racha se calcula según cuantas peleas encadena con vitoria o derrota
<b>winner</b>	Variable a predecir: 0 → Ganador peleador 0 1 → Ganador peleador 1	

## MODELO PREDICTIVO Y METRICAS

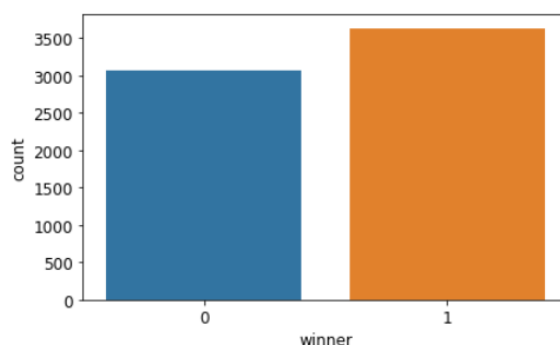
---

### NORMALIZACION

En primer lugar, se normalizaron los datos mediante StandardScaler de la librería sklearn para asegurar que todas las características tengan la misma importancia en el modelo y evitar que una característica con un rango de valores mucho mayor que otra característica tenga un peso desproporcionado en el modelo.

### VARIABLE WINNER DESBALANCEADA

Se realizó un primer modelo preliminar obteniendo una precisión del 70%. Sin embargo, en la matriz de confusión vemos una clara predisposición del modelo a predecir como ganador al peleador 1, por ello, estudiamos la distribución de la variable a predecir y observamos un ligero desbalanceo entre ambas clases que podría afectar a que el modelo prediga con mayor frecuencia como ganador al peleador 1.



**Figura 12: Countplot de la variable a predecir.**

Se utilizó SMOTE (Synthetic Minority Over-sampling Technique) que es una técnica de remuestreo supervisado utilizada para tratar la desbalanceo de clases. Para hacer esto, SMOTE identifica dos puntos de datos de la clase minoritaria cercanos entre sí y genera un nuevo punto de datos sintético en algún lugar entre ellos. Este nuevo punto de datos sintético representa una combinación de las características de los dos puntos de datos originales, y se agrega al conjunto de datos para equilibrar la proporción de clases. Este proceso se repite varias veces para generar un número suficiente de nuevos puntos de datos sintéticos y equilibrar la proporción de clases.

Con esto se consiguió balancear los datos y que se aproximaran los valores de las métricas de precisión y recall como se explica más adelante.

### MODELOS INICIALES

Se dividió el data set en train y test y se realizó una evaluación inicial de varios modelos de clasificación binaria para un conjunto de datos dado. Los modelos que se evaluaron incluyen Regresión Logística, Support Vector Classification (SVC), Random Forest y XGBoost. Además, también se probó un algoritmo no supervisado, KNeighbors, y una

red neuronal sencilla. Para evaluar los modelos, se utilizaron las métricas f1, matriz de confusión y curva ROC. Se utilizó también la precisión y el recall ya que en nuestro caso nos interesa que el porcentaje de acierto sea igual tanto para la clase 0 (victoria “local”) y la clase 1 (victoria “visitante”), ya que, a diferencia del fútbol, los eventos ocurren siempre en las mismas localizaciones, y por lo tanto estos dos valores deberían ser lo más cercanos posibles.

## MODELO NO SUPERVISADO

**KNeighbors (KNN)** es un algoritmo de aprendizaje no supervisado que se basa en la idea de que una muestra es similar a las muestras vecinas en su entorno. En un problema de clasificación, la clase de una muestra se determina en función de las clases de sus K vecinos más cercanos. KNN se utiliza a menudo en problemas de clasificación debido a su simplicidad y eficacia en manejar problemas no lineales. En nuestro caso logró una precisión de un 53,3% (Figura 13).

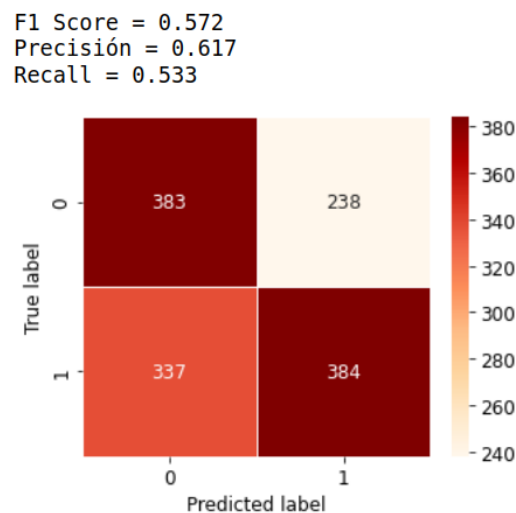
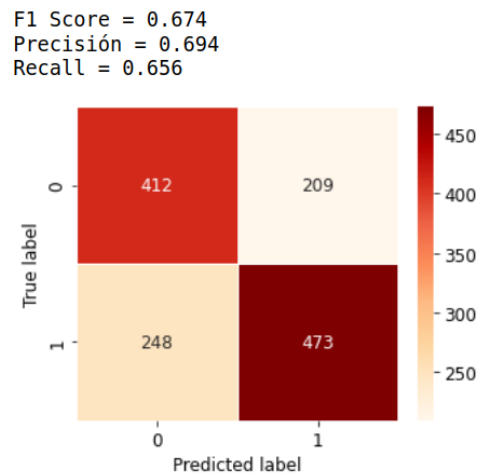


Figura 13: Matriz confusión regresión logística

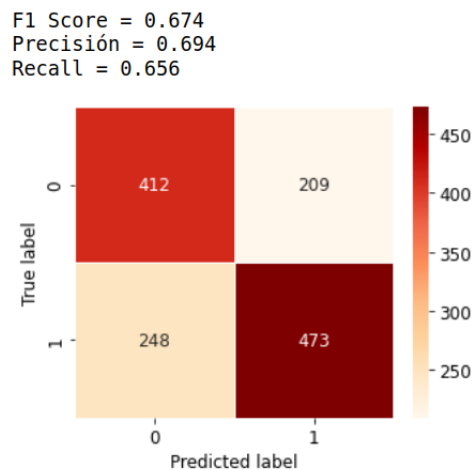
## MODELOS SUPERVISADOS DE CLASIFICACION

La **Regresión Logística** es un algoritmo sencillo y fácil de entender e implementar, y es una buena opción para una amplia gama de problemas de clasificación. Sin embargo, puede tener limitaciones en la capacidad de modelar relaciones no lineales entre las variables independientes y la clase objetivo. En nuestro caso logró una precisión de un 67,4% (Figura 14)



**Figura 14: Matriz confusión regresión logística**

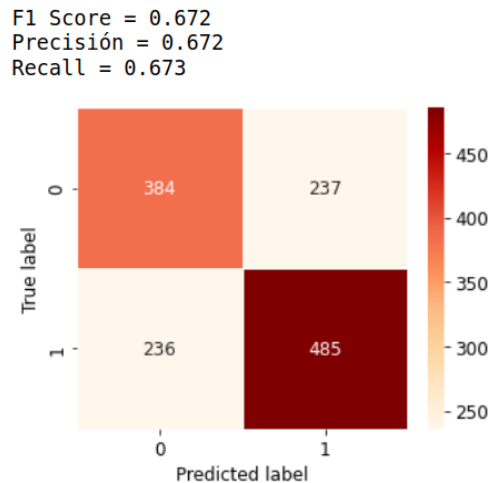
El **Support Vector Machine (SVM)** es un algoritmo de aprendizaje supervisado utilizado para resolver problemas de clasificación y regresión. El SVM es un algoritmo de alta dimensionalidad que busca un hiperplano que separe las clases en un espacio de características. En nuestro caso logró una precisión de un 67,4% (Figura 15)



**Figura 15: Matriz confusión SVC**

**Random Forest** es un algoritmo de aprendizaje supervisado que se utiliza para resolver problemas de clasificación y regresión. Se basa en el concepto de crear muchos árboles de decisión individuales y combinar sus predicciones para obtener una solución más robusta y precisa. En nuestro caso logró una precisión de un 67,2% (Figura 16)

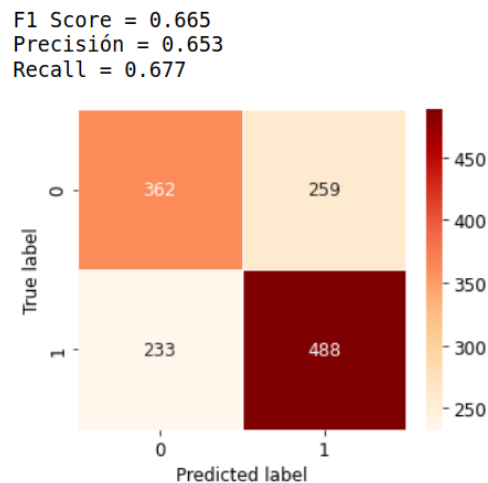




**Figura 16: Matriz confusión Random Forest**

**XGBoost** es un algoritmo de aprendizaje supervisado basado en árboles de decisión que se utiliza para resolver problemas de clasificación y regresión. Se diferencia de otros algoritmos de árboles de decisión como Random Forest en que XGBoost implementa una técnica llamada gradient boosting.

Gradient Boosting significa que, en lugar de crear árboles de decisión individuales y combinar sus predicciones, XGBoost crea un único árbol de decisión y luego lo mejora iterativamente mediante la adición de nuevos árboles que corrigen las predicciones incorrectas del árbol anterior. En nuestro caso logró una precisión de un 66,5% (Figura 17).



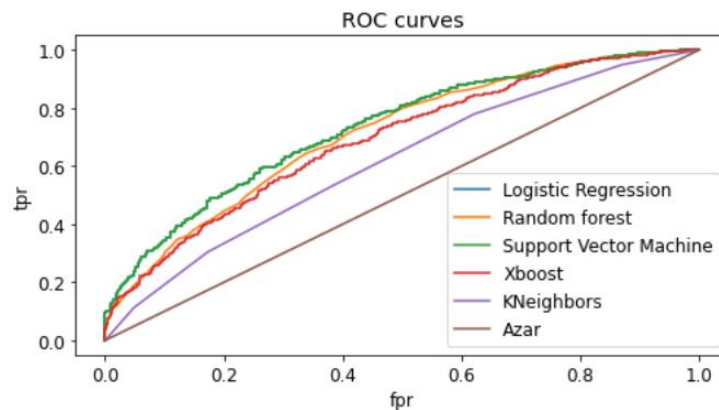
**Figura 17: Matriz confusión Xboost**

## COMPARACION RESULTADOS

En resumen, todos los modelos tuvieron resultados similares en cuanto a las curvas ROC y con precisiones cercanas al 66% (Figura 18). Esto puede deberse a varios factores, como la calidad de los datos disponibles y falta de variables o datos predictivos que

expliquen mejor el modelo. En general, estos resultados indican que todos los modelos tienen una buena capacidad de predecir el resultado, pero es posible que existan formas de mejorar su rendimiento. Por ejemplo, puede ser necesario realizar una selección de características o probar con diferentes hiperparámetros para mejorar los resultados obtenidos.

**A**



**B**

Model name	F1	Precision	Recall	AUC
Logistic Regression	0.674	0.694	0.656	0.731
Support Vector Machine	0.674	0.694	0.656	0.731
Random forest	0.672	0.672	0.673	0.710
Xboost	0.665	0.653	0.677	0.690
KNeighbors	0.572	0.617	0.533	0.614

**Figura 18: Métricas modelos supervisados y no supervisados. A** Curvas ROC de los modelos utilizados. El eje y representa el ratio de verdaderos positivos (tpr) y el eje x el ratio de falsos positivos (fpr). **B** Tabla comparativa de las métricas F1, Precisión y recall AUC de los modelos.

## AJUSTE DE HIPERPARAMETROS

De los algoritmos empleados se seleccionó Random Forest para un ajuste de los hiperparámetros en busca de una mejora del modelo. A pesar ello, no se logró una mejora significativa en la precisión del modelo mejorando tan solo un 0,6%. Se decidió continuar con la experimentación y probar una Red Neuronal para mejorar la precisión del modelo.

## RED NEURONAL

Las **redes neuronales** Sequential en Tensorflow son un tipo de modelo de aprendizaje profundo que están compuestas por varias capas consecutivas de nodos (neuronas) conectados en capas. Se ajustó la capa final utilizando una función de activación

adecuada para el tipo de problema que se está abordando, en este caso una función de activación sigmoide para problemas de clasificación binaria.

Se utilizaron los siguientes parámetros para la evaluación inicial de la red neuronal:

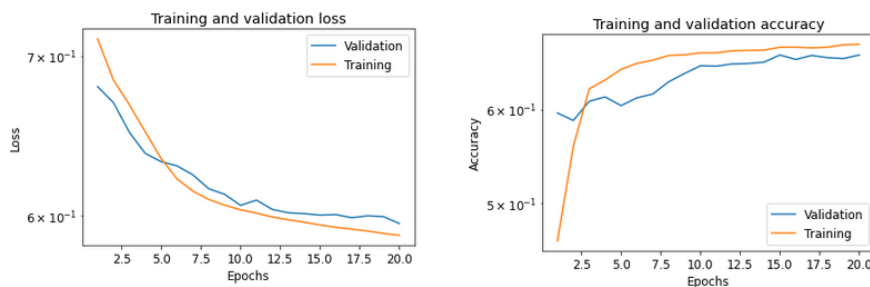
- neurons = 30
- layers = 10
- activation = 'relu'
- optimizer = 'adam'
- batch\_size = 300
- epochs = 20
- función de coste = 'binary\_crossentropy'

Se obtuvo una precisión de un 66.9% en el conjunto de validación en la epoch 5. A partir de este ciclo, la red neuronal comienza a empeorar su rendimiento en el conjunto de validación ya que aumenta el overfitting al conjunto de entrenamiento.

A continuación, se realizó un ajuste de los hiperparámetros consiguiendo los más óptimos para nuestro caso:

- optimizador: Adam
- activador: leaky\_relu
- layers: 20
- neuronas: 20
- batch size: 350
- learning rate: 0.001

Pese a ello, no se consiguió una mejora significativa del modelo (Figura 19) consiguiendo una precisión del 66% en el conjunto de test tras el ajuste de los hiperparámetros. Esto apoya la hipótesis de que los datos disponibles no son suficiente para explicar la totalidad de los resultados.



**Figura 19: Grafica de la evolución de la función de coste y la precisión con cada ciclo de entrenamiento de la red neuronal tras la optimización de los hiperparámetros.**

## FRONTEND

Por último, se utilizó Streamlit para construir una aplicación web que da acceso a probar el modelo desarrollado para la predicción del ganador de una pelea de MMA en UFC. Se crearon varias páginas en la aplicación, incluyendo una página de inicio, una página de

predicciones, una página de estadísticas y una de ayuda con la memoria que incluirá el manual de usuario. Para más información, pasad al punto final de la memoria donde se muestran capturas de pantalla del frontend y se explica el funcionamiento.

## RESULTADOS

---

Además de ser comentados en los notebooks, me gustaría comentar en este documento los resultados obtenidos, posibles maneras de mejorar y logros.

En primer lugar, la variable que se ha tratado de predecir es categórica y tiene tres posibles resultados: victoria “local”, empate y victoria “visitante”. En principio, la probabilidad de adivinar el resultado de una pelea sin tener en cuenta ningún factor adicional es prácticamente un 50%, ya que es muy poco común que ocurran empates o combates nulos en MMA. Se realizaron modelos preliminares tras la limpieza de los datos y la creación de las nuevas variables obteniendo en todos los casos resultados similares con una precisión aproximada del 66%.

A continuación, se eligió el modelo de Random Forest para el ajuste de hiperparámetros. Para ello, se utilizó los diferentes modelos explicados en el apartado anterior consiguiendo tan solo una mejora de entorno al 1%

Finalmente, se probó una red neuronal para intentar mejorar el modelo. Tras el ajuste de los hiperparámetros tampoco se consigue una mejora del modelo por lo que se descarta esta opción y se selecciona el modelo de Random Forest para usarlo en el front end ya que es el que presenta una precision y recall más similares entre sí (lo más lógico en este caso es el que el orden de los peleadores es al azar).

Por otro lado, se probó otro enfoque: actualizar los datos usando los datos de golpes y derribos de cada una de las peleas, en vez de usar las estadísticas intrínsecas estables utilizadas. Sin embargo, esta aproximación supuso un empeoramiento del modelo probablemente porque tan solo se tiene en cuenta las peleas de UFC y una gran parte de las peleas ocurren entre peleadores que tienen pocas peleas en la compañía. Esto se debe a que por intereses del propio peleador o de la UFC, puede irse a otra compañía de eventos. En estos casos usar las variables intrínsecas estables que tienen en cuenta todas las estadísticas de cualquier pelea profesional puede ser más predictivo que el dato obtenido de su última o últimas peleas. Además, la UFC es la mejor compañía de eventos de combate a nivel mundial por lo la mayoría de peleadores que llegan tienen un largo recorrido profesional en otras compañías que habría que tener en cuenta.

Por lo tanto, como próximos pasos en el proyecto, se probará a entrenar el modelo tan solo usando los peleadores que tengan un mayor número de peleas dentro de la UFC y calcular las mismas estadísticas que disponemos, pero teniendo en cuenta solo las anteriores 5 peleas. Con esto aumentamos el número de variables y podemos diferenciar peleadores en un estado de forma en ascenso de peleadores en descenso.

## CONCLUSIONES

---

Como resultado de este proyecto extraemos las siguientes conclusiones:

- Se ha cumplido el objetivo de conseguir un modelo predictivo que mejora la probabilidad del azar.
- Se procedió a un cálculo exhaustivo de las estadísticas actualizadas a fecha de cada pelea, pero no resultó en una mejora del modelo
- Variables que no se han contemplado en este estudio o el azar tiene una componente importante en las MMA ya que no hemos podido explicar un 30% de los resultados
- Se ha diseñado un frontend basado en Streamlit donde fácilmente se pueden seleccionar los peleadores que se enfrentan y obtener un ganador y su probabilidad de victoria.
- Además, se ha creado una funcionalidad de "Top 10 peleadores" donde usamos el histórico de datos para mostrar el Top 10 de peleadores según una estadística seleccionada

## REPOSITORIO Y MANUAL USUARIO

---

### ESTRUCTURA DEL PROYECTO

3 Jupyter Notebooks:

- **Data\_wrangling.ipynb**: Contiene el código necesario para la obtención de los datos mediante Web Scraping, limpieza de los datos y *Feature Engineering*.
- **Modelo\_predictivo.ipynb**: Contiene el código de normalización de los datos y creación y evaluación de varios modelos predictivos.
- **Frontend.ipnb**: Contiene el código para la elaboración de la interfaz de usuario usando la librería Streamlit.

**modelo\_random\_forest.joblib**: Tras la evaluación se guardó el modelo en este archivo para cargarlo en el frontend

**scaler.joblib**: Tras la normalización del dataset de predicción se guardó los parámetros del scaler para las predicciones que se harán en el frontend

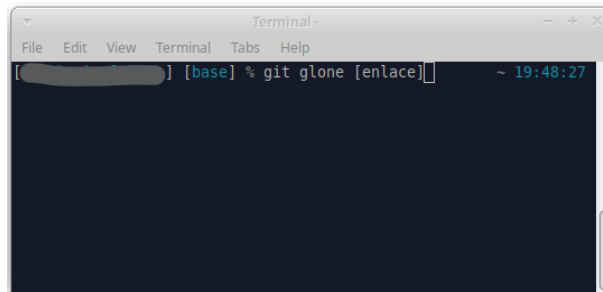
**Requirement.txt**: Incluye las librerías necesarias para replicar el proyecto

**README.md**: Breve resumen del proyecto, de cómo obtener los datos y cómo replicar el proyecto

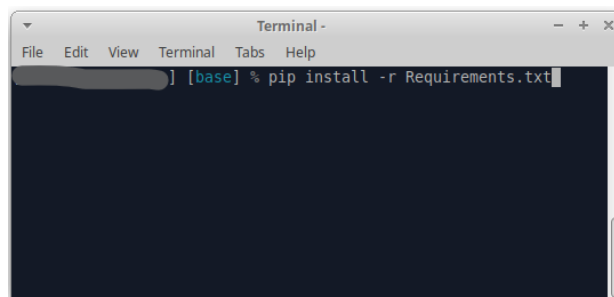
**Memoria.pdf**: Memoria del Proyecto de final de Master y el manual de usuario.

## INSTRUCCIONES PARA REPLICAR EL PROYECTO

1. Clonar el proyecto a tu repositorio local. Enlace:  
<https://github.com/PacoRodriguezLozano/MoneyFighter>



2. Instalar las librerías necesarias especificadas en el archivo "Requirement.txt".



3. Si quieres realizar de nuevo la adquisición de los datos debes ejecutar todos los notebooks en el siguiente orden:
  - Data\_wangling.ipynb
  - Modelo\_predictivo.ipynb
  - Frontend.ipnb

Ten en cuenta que cambios en la estructura de las páginas que se utilizaron para obtener los datos o cambios en su código podrían resultar en un error en el notebook. Además, si se introducen nuevas barreras para evitar el Web Scraping puede que se produzca un error.

Durante el desarrollo del proyecto, tan solo se encontró el problema de limitación de las solicitudes del servidor que se solucionó incorporando una gestión del error y un tiempo de pausa usando la librería "time".

4. Si prefieres usar los datos que se obtuvieron en el momento del desarrollo del proyecto (datos actualizados hasta 05/02/2023):
  - Descargar datasets obtenidos tras el Web Scraping del siguiente link:  
[https://drive.google.com/drive/folders/1kdoIGkDWWuYlr\\_DpAqAgxTs2cf0cZfss](https://drive.google.com/drive/folders/1kdoIGkDWWuYlr_DpAqAgxTs2cf0cZfss). Y colocarlos en la misma carpeta en la que clonaste el proyecto.

- Correr los notebooks en el mismo orden que antes, pero comenzando por los apartados marcados en el propio notebook para este caso saltando así la parte de adquisición de los datos.

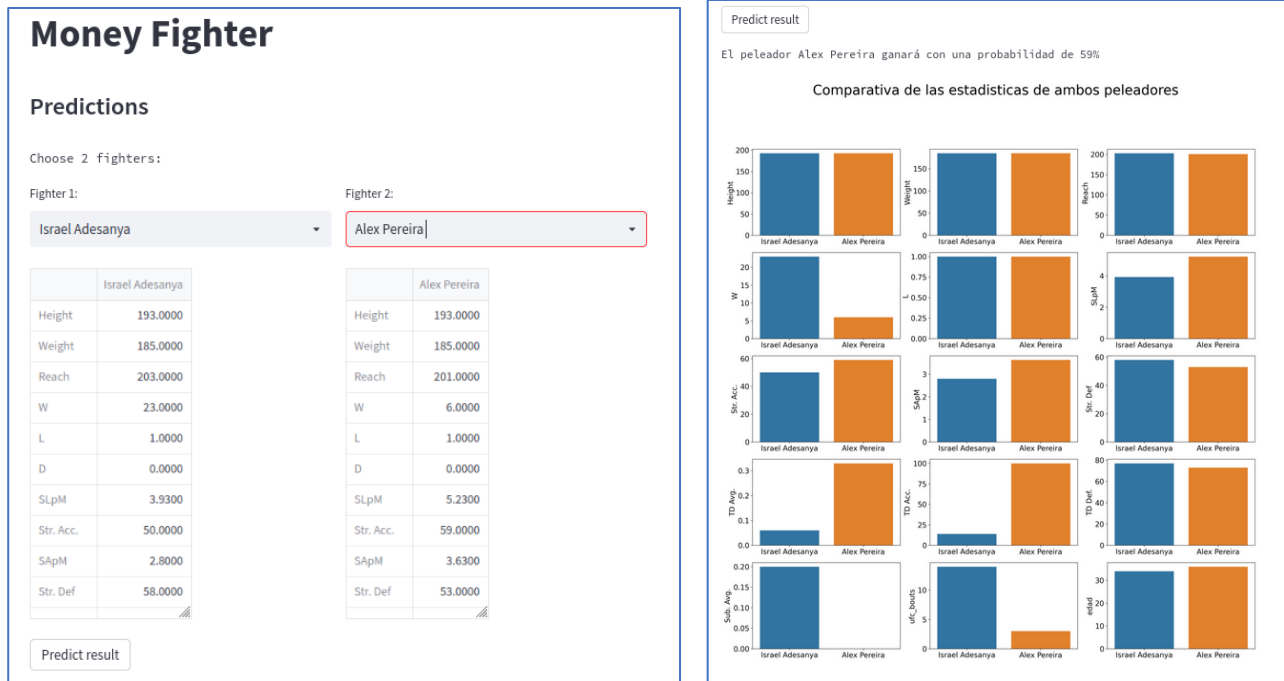
## INSTRUCCIONES DE USO DE LA APP

El frontend seleccionado para desarrollar este proyecto ha sido una aplicación web de Streamlit. Se muestra el menú a la izquierda como una barra lateral donde se podrá seleccionar entre varias pestañas:

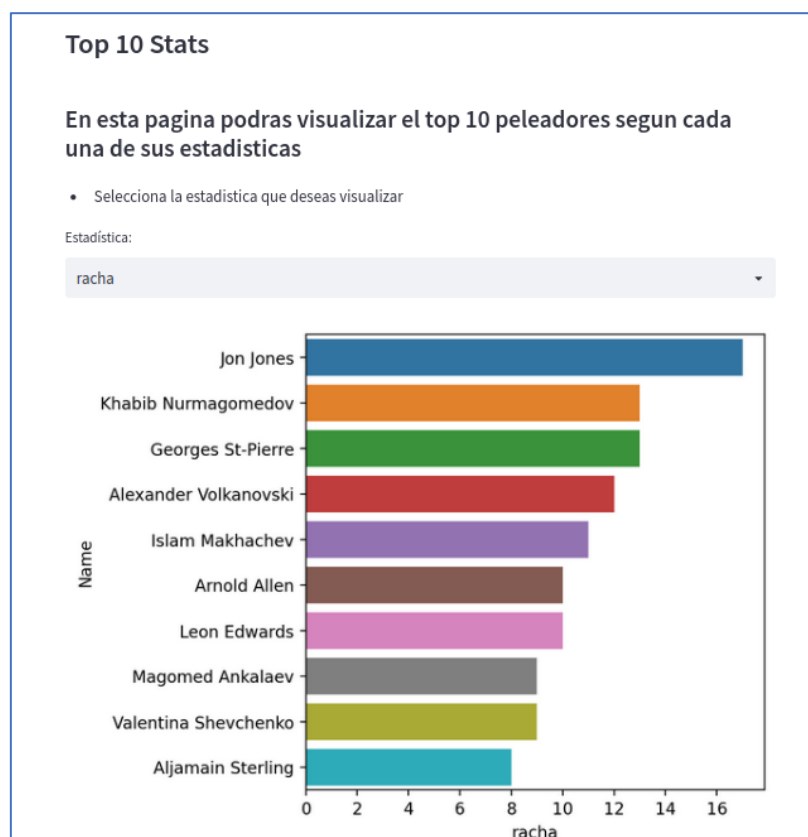
- **Home:** Pagina de presentación del proyecto, con una pequeña introducción del funcionamiento y los objetivos del mismo.



- **Predictions:** Pagina donde se podrá interactuar con el modelo. Esta página estará dividida en dos columnas con un desplegable en cada una de ellas para seleccionar el peleador. Se recomienda escribir el nombre del peleador para hacer la búsqueda más sencilla.



- **Top 10 stats:** Página donde podrás visualizar el top 10 peleadores para las estadísticas que selecciones en el menú desplegable





- **User Manual:** aquí encontraras el manual de usuario.

