Predicción del resultado de una partida de League of Legends usando métricas del juego temprano

Luis Adrián Uribe Cruz, estudiante
Instituto Tecnológico y de Estudios Superiores de Monterrey, Querétaro 76130

A01783129@tec.mx

Inteligencia Artificial Avanzada para la Ciencia de Datos

ABSTRACT Este reporte tiene como finalidad presentar el desarrollo y evaluación de un modelo de Regresión Logística para la predicción del resultado de partidas del videojuego *League of Legends*. Se utilizan únicamente métricas numéricas disponibles de los primeros 10 minutos de cada enfrentamiento para determinar los factores más decisivos en la victoria o derrota del equipo azul. La regresión se implementa de forma casera, minimizando el uso de librerías externas. El modelo se entrena con datos recolectados de aproximadamente 10,000 partidas obtenidas hace 5 años de divisiones altas, garantizando la consistencia de los datos. La clasificación se limita a un problema binario de victoria o derrota, proporcionando a su vez detalles sobre la importancia relativa de cada variable. La intención del modelo es priorizar la explicabilidad de los coeficientes resultantes por sobre las métricas evaluativas del mismo. Dados aquellos datos y relaciones no lineales que no pueden ser tomadas en cuenta, no se espera una precisión perfecta.

KEYWORDS Aprendizaje Automático, Regresión Logística, Función Sigmoide, Gradiente descendente, League of Legends, MOBA

I. INTRODUCCIÓN

A. CONTEXTO GENERAL

League of Legends (o LoL) es un juego de arena de batalla multijugador en línea, o MOBA por sus siglas en inglés, lanzado en 2009 donde dos equipos de 5 integrantes luchan por destruir la base del oponente mientras tienen que proteger la propia, siendo así que la partida culmina cuando el Nexo de cualquier equipo es destruido.

LoL es uno de los juegos más importantes de su género, con millones de jugadores diarios y torneos oficiales de asistencia internacional, lo que convierte a su escena competitiva en una muy codiciada en la que tener la capacidad de predecir un vencedor se Vuelve muy útil.

A pesar de que cada partida es su propio mundo, existen factores en común registrados en las métricas que se repiten entre los distintos equipos vencedores a lo largo del tiempo. Factores cuya relación lineal con el resultado serán explotados para la realización de un modelo.

B. OBJETIVO

La realización de un modelo de clasificación que permita la predicción de una partida, y que a su vez habilite una clara interpretación del valor de cada variable predictora involucrada en el proceso. Esto usando únicamente los primeros 10 minutos de una partida, tiempo conocido como *juego temprano* y donde se asientan las bases de cómo se desarrollarán las fases posteriores.

Se ha elegido un modelo lineal para poder extraer esta clase de relaciones de cada variable con respecto a la probabilidad de victoria, que permite un análisis donde cada pequeña ventaja acumula o resta al desenlace final. Esto como base para otros modelos que permitan analizar relaciones más complejas.

C. DATOS

El set utilizado recopila todas aquellas métricas numéricas de 9,879 partidas de la división de Diamante dentro de las clasificatorias del juego, lo que significa un rango muy alto comprendido por el 2.34% de los jugadores totales, pero no el rango más alto posible, Retador. Esto permite balancear

un buen nivel mecánico de los jugadores y la cantidad disponible de los mismos, garantizando la consistencia.

El set de nombre "League of Legends Diamond Ranked Games (10 min)" [1] fue extraído haciendo uso de la API oficial de *Riot Games*, la empresa detrás del juego. Contiene 40 columnas de las cuales 1 es categórica, 2 son binarias y el resto son numéricas.

II. METODOLOGÍA

A. ALGORITMO

La regresión logística es un método de clasificación, usualmente binaria, que permite obtener la probabilidad de pertenencia de una muestra a una determinada clase según un conjunto preestablecido de variables predictoras. Funciona en base a 3 principales componentes: la función sigmoidea, la función de entropía cruzada y el algoritmo del gradiente descendente.

1) FUNCIÓN SIGMOIDEA

La función sigmoidea, o función logística, es una función que desde infinito negative hasta infinito positivo sólo puede albergar una curva de valores entre 0 y 1. Actúa como función de activación.

$$\sigma(z) = \frac{1}{1+e^z} \quad (1)$$

donde:

- $z = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n (2)$ es la combinación lineal de las variables independientes más un sesgo.

Esta forma le permite ser derivable en cualquier punto de la línea, pero también la vuelve susceptible al problema del desvanecimiento de gradiente, que limita o impide el aprendizaje del modelo.

2) FUNCIÓN DE ENTROPÍA CRUZADA

La función de entropía cruzada, o pérdida logística, es una adaptación de la teoría de la entropía de la información que mide la diferencia entre dos distribuciones de probabilidades, aplicado a este caso, la distribución de los valores reales en contra de los valores predichos.

$$J(\beta) = -\frac{1}{n}\sum_1^n [y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)] \quad (3)$$

Donde:

- y_i es el valor real de la variable dependiente
- $\hat{y}_i$ es el valor predicho por la función sigmoide

Esta forma, en comparación con otras funciones de pérdida, penaliza en mayor medida una predicción errónea, pues el valor del gradiente escala mucho más.

3) GRADIENTE DESCENDENTE

El gradiente descendente es una función de optimización que permite ajustar dinámicamente los coeficientes del modelo de acuerdo al valor del gradiente de los mismos. El gradiente indica qué tanto cambia la función de costo con el cambio, pero, sobre todo, en qué dirección. Un gradiente positivo resultará en disminuir el coeficiente, uno negativo lo aumentará.

$$\beta_j = \beta_j - \alpha \frac{\partial J(\beta)}{\partial \beta_j} \quad (4)$$

Donde:

- β_j es el coeficiente a actualizar
- α es la tasa de aprendizaje
- $\frac{dJ(\beta)}{d\beta_j}(5)$ es el gradiente de la función de costo

La tasa de aprendizaje indica el tamaño de paso entre cada iteración del algoritmo. Dado que el objetivo es minimizar una función, un valor muy alto hará que el algoritmo pueda saltarse el mínimo local y no converja. Un valor pequeño elimina ese riesgo a cambio de incrementar las iteraciones necesarias.

B. Implementación

1) Estructura de datos

La estructura de datos principal en la que se están guardando todos los posibles datos y cálculos es el arreglo de Numpy [2], un arreglo homogenizado previamente optimizado para cálculos vectoriales y matriciales gracias al uso de funciones hechas directamente sobre C, saltando la interpretación de Python. Esta diferencia en eficiencia es vital para aprendizaje automático que trabaja con constantes cálculos de arreglos muy grandes que superan los varios miles de elementos. Los datos para su visualización se almacenan en un dataframe de Pandas[3], estructuras de datos organizadas de forma tabular que están escritas sobre el arreglo de Numpy, dándole acceso a sus beneficios.

De esta forma, los cálculos se realizan con el bloque completo de información en lugar de tener que iterar elemento a elemento. El principal cambio se encuentra en el cálculo del gradiente, pues la ecuación (5) pasa a tener la siguiente forma:

$$gradW = X^T \cdot (\hat{y} - y) \quad (6)$$

Donde:

- X es una matriz de forma *(muestras, características)*
- X^{T} es la transpuesta de forma (características, muestras)
- y es el vector columna de valores reales de forma *(muestras, 1)*
- $\hat{y}$ es el vector columna de predicciones de forma *(muestras, 1)*
- $gradW$ es el vector columna de gradientes de cada coeficiente de forma (características, 1)

Otros ejemplos del uso de Numpy es a la hora manejar potenciales divisiones entre 0 [3] [4], el uso de la pseudoinversa [5] de matrices para prevenir singularidades, y cálculo directo de diagonales sin tener que construir la matriz completa [6].

2) Validación del modelo

El modelo será analizado bajo 6 métricas principales para validar su capacidad de predicción.

- Curvas ROC/AUC: Ver cómo se comporta el modelo a lo largo de diferentes puntos de activación para la predicción, y qué tanto se aleja del azar.
- Exactitud: Visión preliminar de los aciertos en general.
- Precisión: Optimizar hacia disminuir los falsos positivos.
- Sensibilidad: Optimizar hacia disminuir los falsos negativos.
- Puntuación F1: Optimizar un equilibrio entre precisión y sensibilidad.
- Matriz de Confusión: Visualización directa de los aciertos y fallos en general.
-

De forma inicial no se estará buscando activamente optimizar para alguna métrica en particular, sólo se analizan los resultados generales.

3) Validación de coeficientes

Dada la extensión del set, una prueba de student se quedaría corta, por lo que para este trabajo se estará usando una prueba de Wald, para demostrar que los coeficientes son significativamente distintos de 0. Donde se asume que bajo la hipótesis nula se sigue una distribución normal:

- Se calculan los errores estándar de cada coeficiente
- Con ellos se obtienen los z-score de cada uno
- Finalmente, con el uso de la Función de Distribución Acumulada [7] de la curva normal, se obtienen los p-valores correspondientes.
- También se generan los intervalos de confianza al 95%

4) Fases

Habrá 3 modelos distintos que van de menos a más limpieza y preprocesamiento de los datos. En común, todos partirán del primer modelo y todos pasarán por estandarización z-scores.

a) Modelo redundante

Modelo inicial y básico. Su set de datos únicamente será eliminado de variables directamente dependientes de otras de las que ya se conoce de antemano, pero cualquier otra posible correlación no será limpiada. No se aplicarán más transformaciones ni se crearán columnas nuevas derivadas. Los hiperparámetros serán los preestablecidos de la clase.

b) Modelo descorrelacionado

Modelo derivado, utiliza como set de datos el resultante del modelo redundante. Se eliminarán aquellas variables con correlaciones muy fuertes, dejando sólo la que se correlacione mejor con la variable objetivo y/o tenga menor correlación con otras. Analiza posible desbalance de clases. No se realizarán más transformaciones ni se agregarán columnas derivadas. Los hiperparámetro serán levemente ajustados.

c) Modelo pulido

Modelo derivado, su set de datos es el resultante del modelo redundante. Se aplicarán transformaciones y se crearán columnas nuevas antes de decidir si se eliminan las correlacionadas. Analiza posible desbalance de clases. Se realizarán diferentes ajustes de hiperparámetros.

B. EXPLICACIÓN DE VARIABLES

Tabla 1

DESCRIPCIÓN DEL SET

Variable	Descripción
gameId	Identificador único de la partida.
blueWins	Columna objetivo. Victoria del equipo azul.
wardsPlaced	Centinelas de visión colocados. Un centinela de visión permite ver zonas del mapa donde no haya aliados.
wardsDestroyed	Centinelas de visión del equipo contrario destruidos.
firstBlood	Primer asesinato de la partida. Otorga recursos adicionales al jugador que la consiguió.
kills	Asesinatos sobre jugadores enemigos. La muerte le deja fuera de juego por un tiempo y le regresa hasta su base.
deaths	Muertes de cada equipo, sea a manos de otros jugadores o de elementos del entorno.
assists	Asistencias de cada equipo. Una asistencia es haber participado de un asesinato pero no haber sido quien de el golpe final.
eliteMonsters	Monstruos épicos. Objetivos neutrales que dan beneficios especiales al equipo que los derroten.
dragons	Dragones por equipo. Cada dragón da una mejora permanente diferente. Reunir 4 da una aún mayor y desbloquea un dragón más poderoso.
heralds	Heraldos de la Grieta por equipo. Además de recursos inmediatos tras u derrota, permite ser invocado para asediar la base enemiga.
towersDestroyed	Torretas destruidas. Las torres protegen la base y el camino a ella con gran daño además de da visión.
totalGold	Oro total por equipo. El oro se convierte en objetos que potencian a cada personaje.

avgLevel	Nivel promedio por equipo. Subir de nivel da estadísticas adicionales, desbloquea nuevas habilidades y mejora las existentes.
totalExperience	Cantidad de experiencia por equipo. La experiencia es lo que permite subir de nivel.
totalMinionsKilled	Súbditos por equipo. Los súbditos marchan con el mismo objetivo que los jugadores, su ausencia fortalece torres enemigas. Dan pocos recursos, pero son numerosos y constantes.
totalJungleMinionsKilled	Los monstruos de la jungla son un objetivo fijo y neutral. Un rol se encarga de ellos como fuente de ingresos y demuestra su presencia en el mapa.
goldDiff	Diferencia entre el oro total de los equipos.
experienceDiff	Diferencia entre la experiencia total de los equipos.
csPerMin	Promedio de súbditos asesinados por minuto.
goldPerMin	Promedio de oro obtenido por minuto.
avgLevel	Ambos equipos. Información prácticamente idéntica a la experiencia total.
csPerMin	Ambos equipos. Directamente obtenido de lo súbditos totales divididos entre 10, no importa información nueva por ignorar a los monstruos de la jungla que también cuentan como “cs”.
goldPerMin	Ambos equipos. Directamente obtenido del oro total dividido entre 10. Es más útil saber el total que un promedio.
redFirstBlood	Sólo puede haber una, por lo que si el equipo azul no la tiene, será del equipo rojo. Basta dejar la del equipo azul.
redKills	Salvo muy contadas excepciones, los asesinatos de un equipo equivalen a las muertes del otro.
redDeaths	Salvo muy contadas excepciones, las muertes de un equipo son los asesinatos del otro.
redGoldDiff	Exacto opuesto que blueGoldDiff, ese basta.
redExperienceDiff	Exacto opuesto que blueExperienceDiff, ese basta.

III. RESULTADOS

De cada modelo se obtienen las gráficas de sus distintas métricas, una tabla impresa en consola con la prueba de significancia de cada coeficiente por orden descendente y se exporta el archivo del modelo, la matriz de correlaciones y el archivo separado por comas resultante de la limpieza y transformación.

Sin aplicar normalización, ningún modelo logra converger adecuadamente debido a la enorme diferencia que existe entre los dominios de cada variable: mientras algunas sólo son 0 o 1, otras tienen rangos continuos hasta los miles, lo cual complica el aprendizaje del modelo.

A. Modelo redundante

Modelo limpiado de variables directamente dependientes. Aquellas columnas eliminadas se encuentran en la siguiente tabla.

Tabla 2

MODELO REDUNDANTE: COLUMNAS ELIMINADAS

Variable	Motivo
gameId	No aporta ningún tipo de información.
eliteMonsters	Ambos equipos. Dado que cada monstruo épico aporta algo distinto, este total de ambos no da información más relevante.

Sus hiperparámetros son los predeterminados:

- Tasa de aprendizaje: 0.01
- Iteraciones máximas: 1000
- Tolerancia: 0.000001
- Límite de clasificación: 0.5
- Tamaño set de entrenamiento: 60%
- Tamaño set validación: 20%
- Tamaño set de prueba: 20%

El modelo no logra converger en tan limitado número de iteraciones, por lo que su resultado final depende completamente del azar con el que sus coeficientes se hayan inicializado.

1) Métricas

La línea de convergencia varía mucho de entrenamiento a entrenamiento dado ese componente aleatorio, con valores de pérdida de hasta 0.85, más de 1 en situaciones donde el paro temprano se dispara.

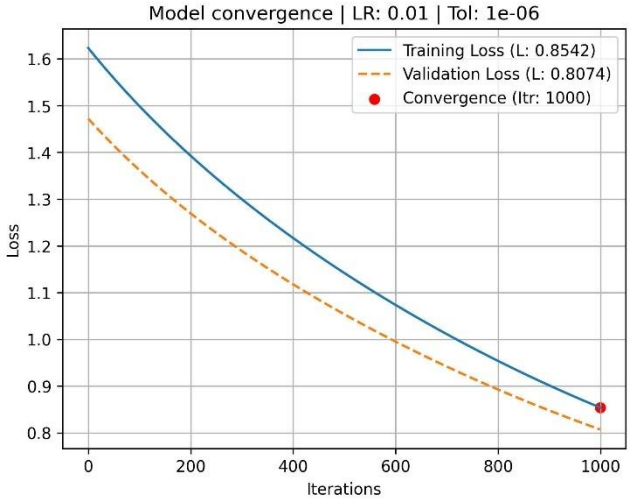

Fig. 1. Línea de convergencia entre los sets de entrenamiento y validación en el peor caso, primer modelo.

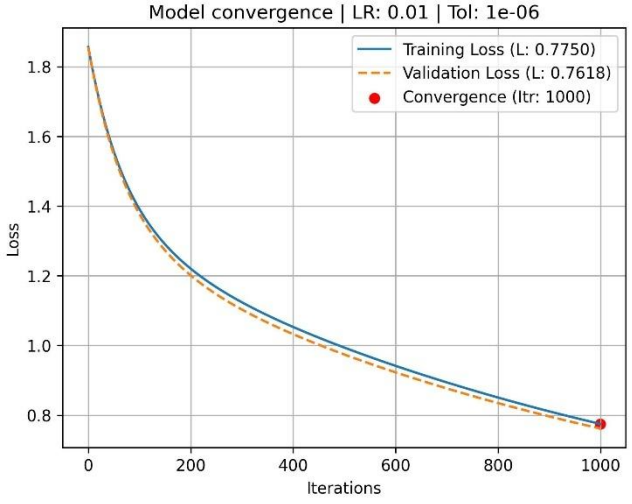

Fig. 2. Línea de convergencia entre los sets de entrenamiento y validación en el mejor de los casos, primer modelo.

La curva ROC y su área bajo la curva se estabilizan de mejor manera, estableciéndose en 0.72x en la gran mayoría de intentos, siendo que su mejor valor es de 0.76x.

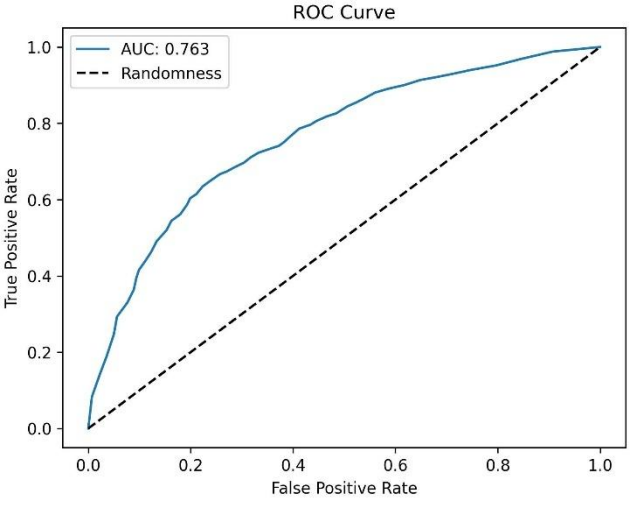

Fig. 2. Curva ROC/AUC en el mejor escenario del primer modelo.

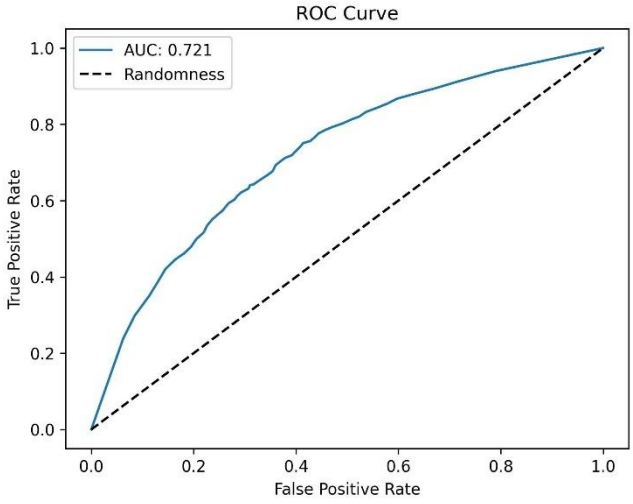

Fig. 4. Curva ROC/AUC en el caso habitual del primer modelo.

Y por último, las puntuaciones del modelo se mantienen estables en valores de 0.68 a lo largo de los entrenamientos y de 0.7 en el set de prueba.

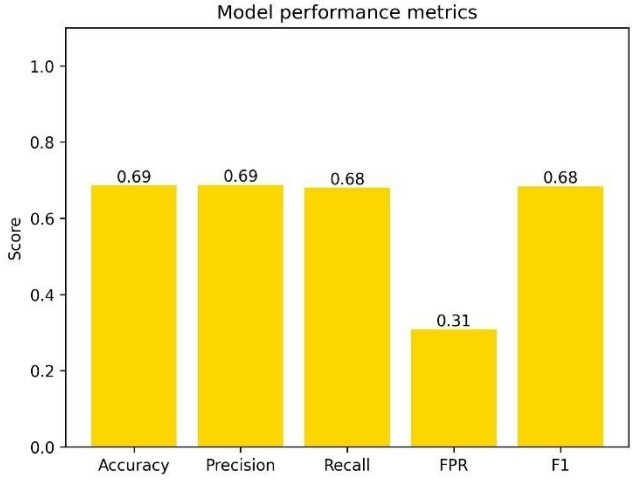

Fig. 5. Puntuaciones sobre el set de entrenamiento. Estables en esos valores. Primer modelo

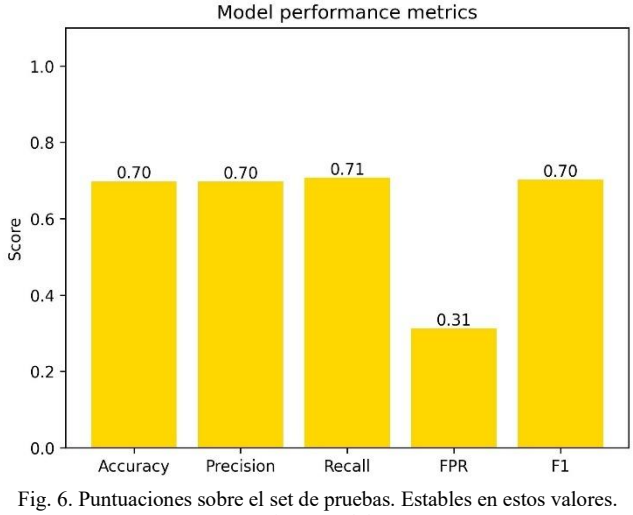

Fig. 6. Puntuaciones sobre el set de pruebas. Estables en estos valores. Primer modelo

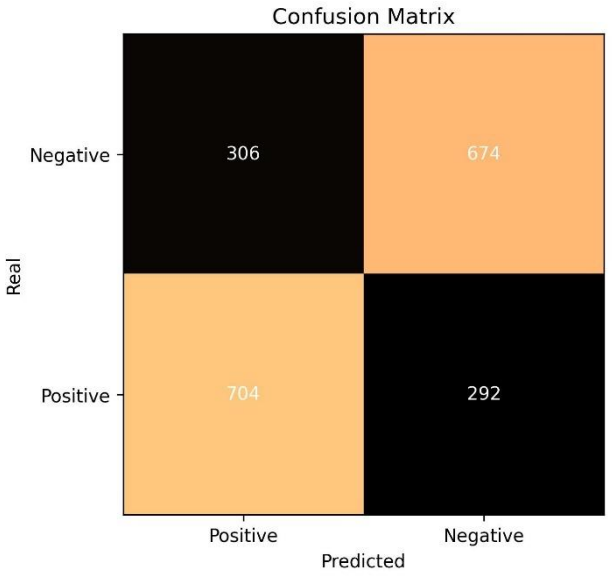

Fig. 7. Matriz de confusión sobre el set de pruebas del primer modelo.

B. Modelo decorrelacionado

Modelo que, tomando de inicio la limpieza del modelo pasado, elimina las variables que posean correlaciones cercanas y mayores a 0.8. La variable que permanece es aquella que posee la menor correlación con el resto de variables y la mayor con la variable objetivo, siempre y cuando esta no supere el límite establecido.

Tabla 3

MODELO DECORRELACIONADO: COLUMNAS ELIMINADAS

Variable	Motivo
totalGold	Ambos equipos. Tiene correlaciones con 4 variables. Permanecerá "blueGoldDiff".
totalExperience	Ambos equipos. Tiene correlaciones con 2 variables.
assists	Ambos equipos. Correlacionada con 2 variables. Permanecerán los asesinatos.
blueExperience Diff	Correlación alta con el oro. Permanece el oro, dada su mejor correlación con la variable objetivo.

El único hiperparámetro ajustado fueron las iteraciones máximas, las cuales fueron incrementadas a 10,000 para permitirle converger al modelo. Gracias a este cambio, todas las métricas se han establecido en los valores próximos de ver.

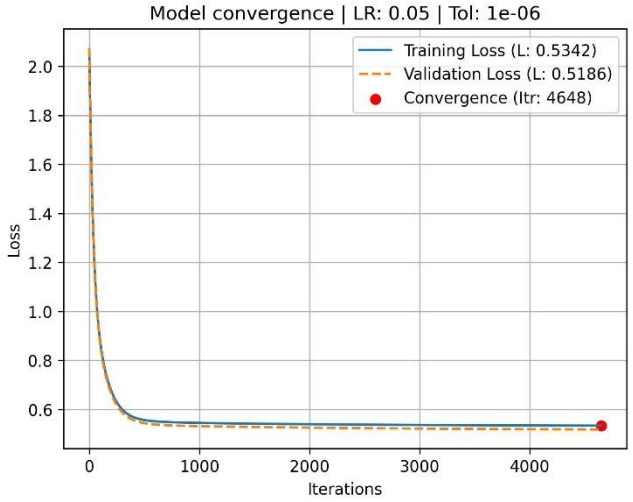

Fig. 8. Línea de convergencia entre los sets de entrenamiento y validación del segundo modelo.

La curva tiene un codo bien definido y un límite aparente en la convergencia de ambas pérdidas. Diferentes intentos mantienen el valor de 0.53xx en el entrenamiento y de 051xx en la validación.

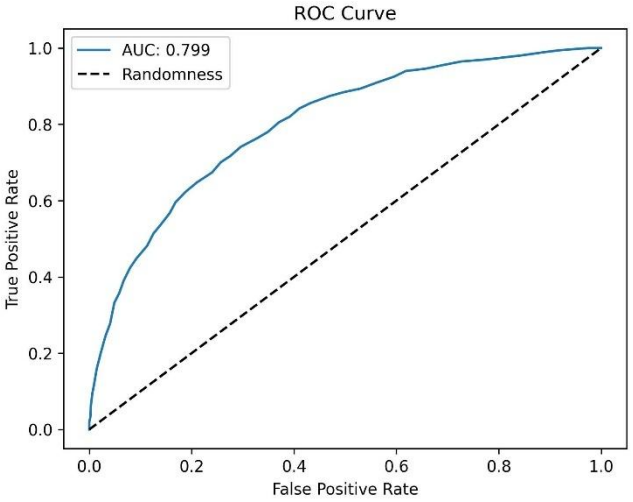

Fig. 9. Curva ROC/AUC del segundo modelo.

El área bajo la curva de este modelo alcanza valores de 0.8, distanciándose con mayor fuerza del azar, debido a la eliminación del ruido de la multicolinealidad y permitir la convergencia con más iteraciones.

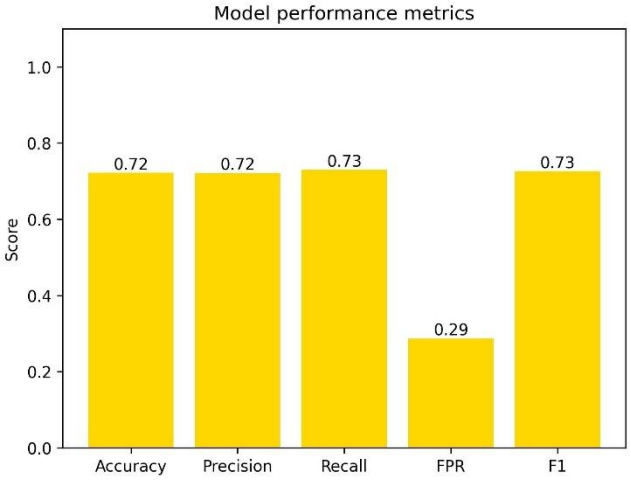

Fig. 10. Puntuaciones sobre el set de entrenamiento del segundo modelo.

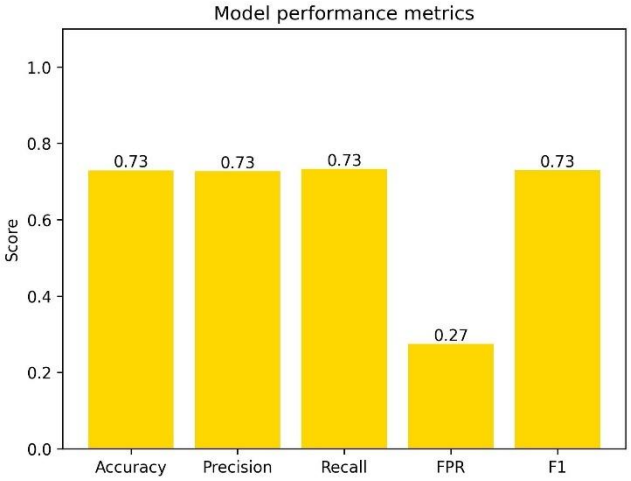

Fig. 11. Puntuaciones sobre el set de entrenamiento del segundo modelo.

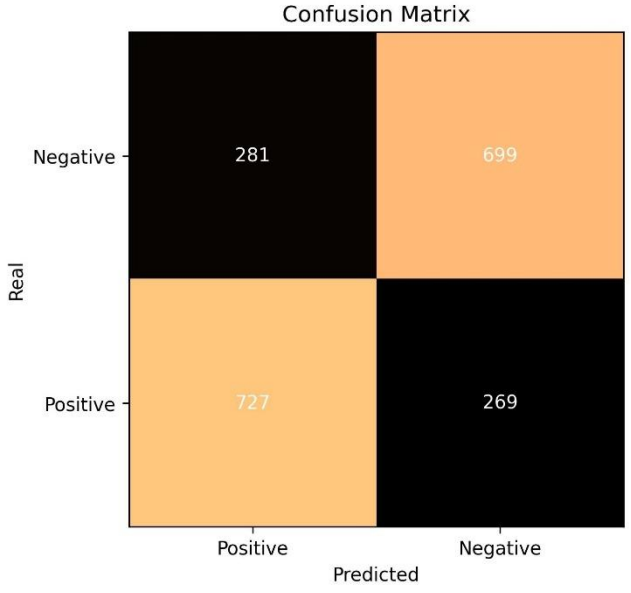

Fig. 12. Matriz de confusión sobre set de pruebas del segundo modelo.

Dado un límite de clasificación neutro de 0.5, el modelo mantiene buena capacidad de generalización al notar que no favorece ninguna métrica en particular.

Además de las métricas, esta versión del modelo provee de coeficientes significativos muy estables a lo largo de varios entrenamientos, como se puede ver a continuación.

Model results:

	Coefficient	StdError	ZScore	PValue	CI_Lower	CI_Upper
blueKills	0.435647	0.084116	5.179128	2.229256e-07	0.270780	0.600514
blueDeaths	-0.372396	0.081546	-4.566672	4.955282e-06	-0.532227	-0.212565
blueDragons	0.194403	0.039637	4.904615	9.361078e-07	0.116715	0.272090
blueTotalMinionsKilled	0.136159	0.043685	3.116823	1.828115e-03	0.050536	0.221782
blueTotalJungleMinionsKilled	0.142938	0.034942	4.090773	4.299383e-05	0.074453	0.211424
blueGoldDiff	0.661666	0.138754	4.768629	1.854837e-06	0.389709	0.933624
redDragons	-0.107849	0.040068	-2.691634	7.110300e-03	-0.186383	-0.029315
redTotalMinionsKilled	-0.100272	0.044186	-2.269324	2.324863e-02	-0.186875	-0.013668
redTotalJungleMinionsKilled	-0.096491	0.035400	-2.725752	6.415515e-03	-0.165875	-0.027107

Fig. 13. Resultados de significancia en una iteración del segundo modelo.

Model results:

	Coefficient	StdError	ZScore	PValue	CI_Lower	CI_Upper
blueKills	0.435683	0.084116	5.179553	2.224187e-07	0.270816	0.600550
blueDeaths	-0.372431	0.081546	-4.567095	4.945290e-06	-0.532262	-0.212599
blueDragons	0.194401	0.039637	4.904573	9.363091e-07	0.116713	0.272089
blueTotalMinionsKilled	0.136171	0.043685	3.117092	1.826445e-03	0.050548	0.221794
blueTotalJungleMinionsKilled	0.142944	0.034942	4.090939	4.296307e-05	0.074458	0.211430
blueGoldDiff	0.661602	0.138754	4.768167	1.859093e-06	0.389644	0.933560
redDragons	-0.107849	0.040068	-2.691627	7.110451e-03	-0.186383	-0.029315
redTotalMinionsKilled	-0.100283	0.044186	-2.269590	2.323250e-02	-0.186887	-0.013679
redTotalJungleMinionsKilled	-0.096497	0.035400	-2.725914	6.412367e-03	-0.165880	-0.027113

Fig. 14. Resultados de significancia en una segunda iteración del segundo modelo.

C. Modelo pulido

Modelo que toma el set de datos resultante del modelo redundante y, antes de aplicar la limpieza de correlaciones del modelo pasado, genera nuevas columnas derivadas para posteriormente eliminar las columnas que utilice de base.

Tabla 4

MODELO PULIDO: COLUMNAS CREADAS

Variable	Explicación
KDA	Promedio de derribos (asesinatos y asistencias) por cada muerte. Elimina los asesinatos, asistencias y muertes de ambos equipos.
visionControlDiff	Diferencia de control de visión, definido en este proyecto como la relación de centinelas colocados sobre centinelas destruidos por el equipo enemigo. Elimina los centinelas colocados y destruidos de ambos equipos.
csDiff	Diferencia de súbditos más monstruos de la jungla asesinados entre ambos equipos. Elimina el total de súbditos y el total de monstruos de la jungla.

Los hiperparámetros se mantienen iguales a los del modelo pasado, con 10,000 iteraciones máximas para permitir una correcta convergencia.

Las métricas como la línea de convergencia y la curva ROC/AUC se mantienen estables y constantes, gracias al ruido eliminado de tantas variables que ha sido resumido en las columnas nuevas.

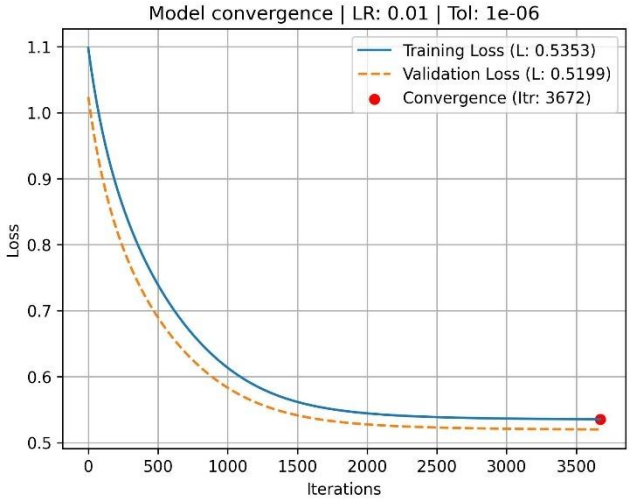

Fig. 15. Línea de convergencia entre los sets de entrenamiento y validación para el tercer modelo.

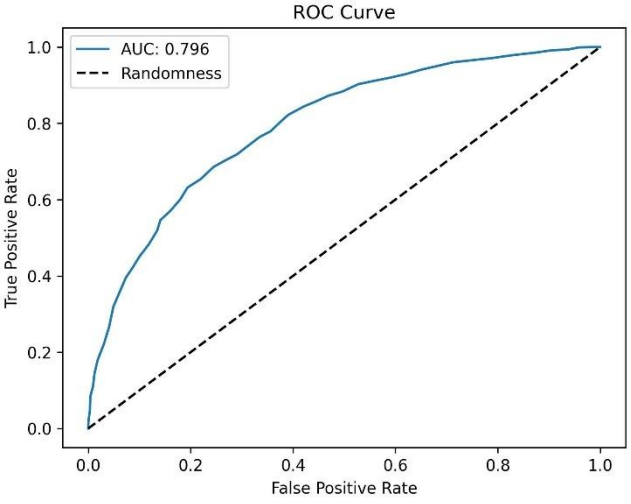

Fig. 16. Curva ROC/AUC del tercer modelo.

La variación para el área bajo la curva requiere 3 decimales de precisión para notar una diferencia, mientras que sólo 2 ya cambian las pérdidas en ambos sets graficados.

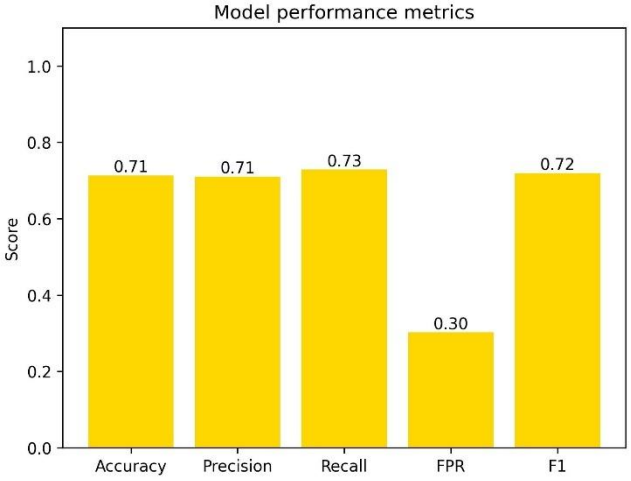

Fig. 17. Puntuaciones para el set de prueba del tercer modelo.

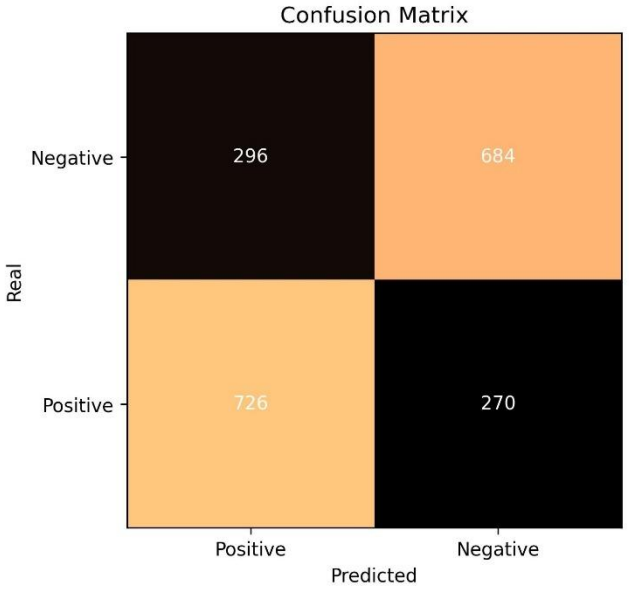

Fig. 18. Matriz de confusión para el set de prueba del tercer modelo.

Este modelo ha incrementado los falsos positivos que produce con respecto a los otros dos modelos, pero no es el único detalle.

Model results:

	Coefficient	StdError	ZScore	PValue	CI_Lower	CI_Upper
blueKills	0.247487	0.089278	2.772095	5.569683e-03	0.072502	0.422472
blueDeaths	-0.238924	0.086546	-2.760643	5.768776e-03	-0.408554	-0.069293
blueDragons	0.254989	0.039611	6.437269	1.216418e-10	0.177351	0.332628
blueTowersDestroyed	-0.088305	0.041918	-2.106602	3.515210e-02	-0.170466	-0.006145
blueGoldDiff	1.317150	0.144092	9.141034	0.000000e+00	1.034729	1.599570
redDragons	-0.081645	0.039617	-2.060847	3.931765e-02	-0.159294	-0.003995
redTowersDestroyed	0.134985	0.037884	3.563064	3.665519e-04	0.060731	0.209238
blueKDA	-0.202990	0.055914	-3.630388	2.829958e-04	-0.312582	-0.093399

Fig. 19. Coeficientes significativos en una iteración del tercer modelo.

Model results:

	Coefficient	StdError	ZScore	PValue	CI_Lower	CI_Upper
blueKills	0.398918	0.089741	4.445202	0.000009	0.223025	0.574811
blueDeaths	-0.390405	0.086941	-4.490473	0.000007	-0.560809	-0.220001
blueDragons	0.159232	0.039189	4.063168	0.000048	0.082421	0.236043
blueHeralds	0.091542	0.032343	2.830357	0.004650	0.028150	0.154934
blueGoldDiff	0.673884	0.141159	4.773926	0.000002	0.397212	0.950557
redDragons	-0.156889	0.039293	-3.992758	0.000065	-0.233903	-0.079874
CSDiff	0.232220	0.055187	4.207882	0.000026	0.124054	0.340386

Fig. 20. Coeficientes significativos en una segunda iteración del tercer modelo.

Este modelo tiene una mayor variación de los coeficientes entre entrenamientos a pesar de que ninguna variable, incluidas las creadas, tiene una gran correlación.

D. Comparación de puntos de clasificación.

Hasta ahora todos los resultados se reportaban utilizando un punto de clasificación neutral de 0.5, donde ambas clases tienen la misma probabilidad de ser elegidas. Para esta sección y usando como base el tercer modelo se van a mostrar las métrica obtenidas de un modelo conservador (punto en 0.6) y un modelo liberal (punto en 0.4).

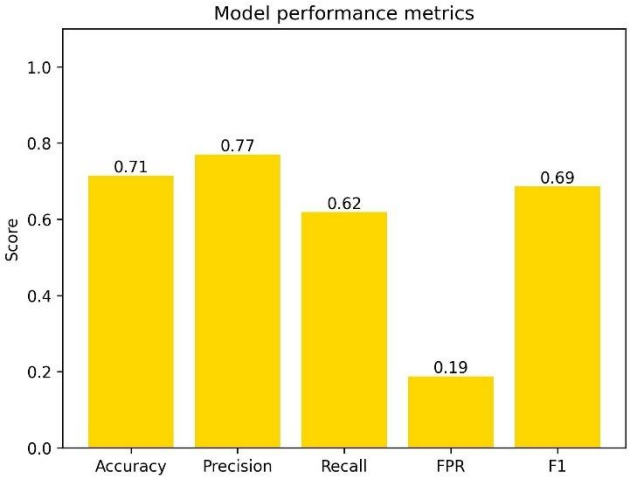

Fig. 21. Puntuaciones de un modelo conservador.

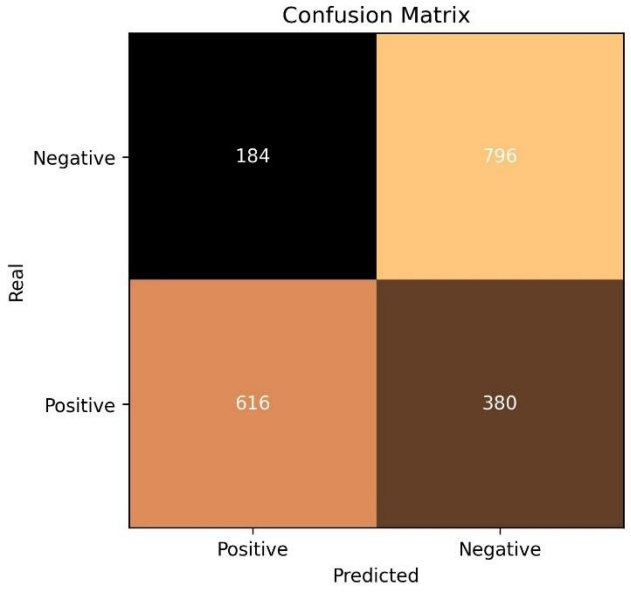

Fig. 22. Matriz de confusión de un modelo conservador.

En un modelo conservador, se intentan minimizar los falsos positivos potenciando la puntuación de Precisión, lo cual se muestra en ambas gráficas como aumentó la precisión pero disminuyó la sensibilidad, con un leve impacto negativo en la puntuación F1.

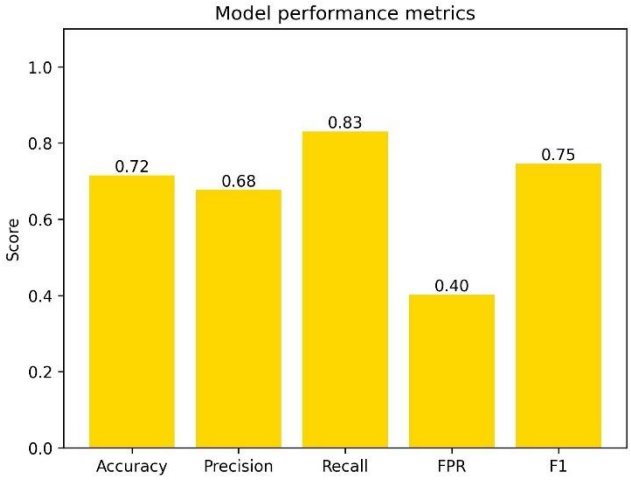

Fig. 23. Puntuaciones de un modelo liberal.

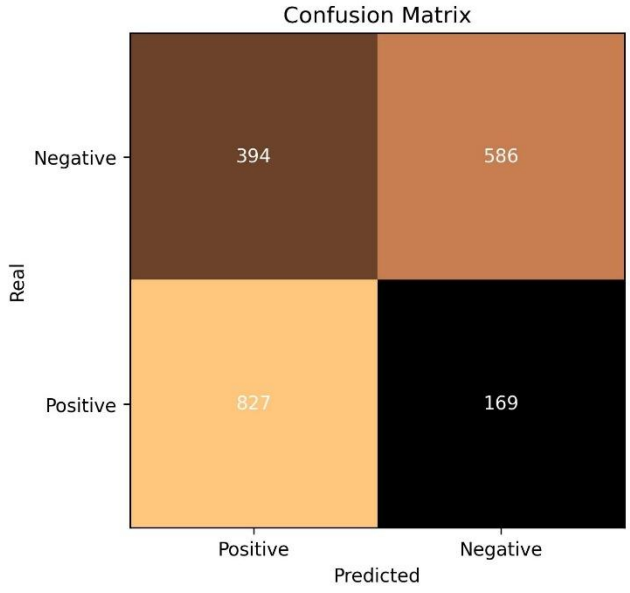

Fig. 24. Matriz de confusión de un modelo liberal.

En un modelo liberal, por el otro lado, se intentan minimizar los falsos negativo potenciando la calificación de Sensibilidad, visto en ambas gráficas en el gran aumento de predicciones positivas. La sensibilidad aumentó en gran medida, compensando la bajada de precisión y por ende teniendo un impacto positivo en la puntuación F1.

IV. DISCUSIÓN

A. Hallazgos

No se pudo observar un caso de sobreajuste debido a que el modelo tenía implementado un paro temprano, por lo que en cuanto la pérdida de validación comenzara a incrementar o parase de disminuir, regresaría el modelo a su mejor estado, dejando así un modelo subajustado en su lugar al no lograr converger.

A pesar de que el primer modelo parece tener los mismos resultados en cuestión de métricas que los dos posteriores, sus coeficientes eran tan variables que en cada iteración, el orden, cantidad y magnitud de la significancia cambiaba

radicalmente, debido a la enorme cantidad de ruido en forma de multicolinealidad y del poco tiempo que disponía para poder adaptarse. Entonces, no había nada realmente útil que interpretar de este modelo.
Respecto al primer y segundo modelo, el notable incremento en la curva ROC indica que el potencial esta ahí a pesar de las bajas métricas, siendo cuestión de decidir con cuál comprometerse para mejorar esos resultados. Aún con un threshold neutral, alcanzaban una buena capacidad de generalizar los datos ingresados y mantener las métricas equilibradas y consistentes.
El segundo modelo es el que alcanza la mayor consistencia entre sus coeficientes, manteniendo apenas cambios en significancia entre entrenamientos.
El tercer modelo, por el otro lado, experimenta una caída en consistencia con los nuevos datos y la eliminación de otros más, demostrando que a veces menos es más en cuestión de preprocesamiento.

B. Interpretación

Gracias a la normalización por z-scores, los coeficientes pueden compararse entre sí directamente en una misma medida: desviaciones estándar.
A pesar de las variaciones, el coeficiente que la regresión siempre considera como el más impactante con los asesinatos y las muertes. Los asesinatos implican una gran presión en el otro equipo, una ventaja para la que el muerto no puede responder y que se sigue acumulando. Los dragones son otro objetivo repetido, pues el alma, el beneficio de obtener 4 dragones, es una ventaja decisiva en etapas tardías, así como el dragón mayor que se desbloquea tras ello. Un tercer factor siempre decisivo es el oro, casi todas las variables ven su ventaja reflejada en oro, pues es la principal recompensa de cualquier objetivo y la forma de fortalecimiento más confiable al ser fija. Con una confianza del 95%, se puede decir que el equipo que domine estos resultados será el que más probabilidades tenga de resultar ganador.

C. Limitaciones

A pesar de la limpieza, las transformaciones y los constantes intentos, un modelo lineal ya no es capaz de extraer más información de las métricas provistas en el set de datos. No era un problema de falta de datos, porque incluso cuando se expandía el set de entrenamiento, las puntuaciones llegaban incluso a disminuir.
Además, esta implementación no incluía validación cruzada, por lo que no existía una forma totalmente veraz de aceptar o desmentir los resultados de un determinado modelo más allá de comparar varias iteraciones.

D. Comparaciones

De forma totalmente objetiva, el primer modelo es el peor de todos, pero era de esperarse tras ser deliberadamente limitado para servir como una base. El tercer modelo es el que no compensa, pues se sacrifica la gran consistencia de coeficientes del segundo modelo por poder incrementar la precisión en una centésima, además de incrementar su costo computacional por tener que optimizar más columnas con relaciones lineales débiles.
El segundo modelo, el que únicamente limpiaba correlaciones, es el que mejores y más replicables resultados ofrece, pues a pesar de no tener números más altos, mantiene el principal objetivo del proyecto que era poder dar una interpretación.

V. CONCLUSIÓN Y TRABAJO FUTURO

Existen ciertas relaciones lineales entre las variables predictoras y el resultado final que pueden dar una buena idea del posible desenlace del juego, pero un modelo lineal se queda corto ante la complejidad que no se toma en cuenta, como combinaciones de variables en vez de ventajas aisladas que se sumen, además del factor humano. A pesar de ello, se obtienen resultados aceptables y que permiten entender la importancia de cada aspecto en una partida, y demostrar que los comportamientos el juego pueden resumirse en gran medida en los números registrados en el principio, capaces de determinar el desarrollo del resto del juego.
Los modelos son consistentemente sesgados, pues ninguna transformación puede acercar una métrica a un porcentaje casi perfecto, pero decidir optimizar haca no dejar fuera ninguna victoria o que cuando se prediga una haya certeza de que lo vaya a ser será clave para poder mejorar los modelos. Otro dato importante es escoger un modelo más complejo capaz de poder entender las relaciones no lineales de los datos y hacer las adecuadas asociaciones.
El sesgo siempre es elevado, pero la varianza se mantiene media en los modelos 1 y 3, mientras que para el segundo modelo, la varianza es baja

VI. REFERENCIAS

[1] Y. L. Ma, «League of Legends Diamond Ranked Games (10 min),» Kaggle, 2020. [En línea]. Available: https://www.kaggle.com/datasets/bobbyscience/league-of-legends-diamond-ranked-games-10-min.

[2] Documentation, Numpy, «numpy.array,» Numpy, [En línea]. Available: https://numpy.org/devdocs//reference/generated/numpy.array.html.

[3] Numpy Documentation, «numpy.divide,» Numpy, [En línea]. Available: https://numpy.org/devdocs//reference/generated/numpy.divide.html#numpy-divide.

[4] Numpy Documentation, «numpy.where,» Numpy, [En línea]. Available: https://numpy.org/devdocs//reference/generated/numpy.where.html#numpy.where.

[5] Numpy Documentation, «numpy.linalg.pinv,» Numpy, [En línea]. Available: https://numpy.org/devdocs//reference/generated/numpy.linalg.pinv.html#numpy.linalg.pinv.

[6] Numpy Documentation, «numpy.diag,» Numpy, [En línea]. Available: https://numpy.org/doc/2.2/reference/generated/numpy.diag.html.

[7] SciPy Documentation, «scipy.stats.rv_continuous.,» SciPy, [En línea]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.rv_continuous.cdf.html.