Universidad del Desarrollo
Facultad de Ingeniería

PREDICCION DE PARTIDAS EN VALORANT

Utilizando algoritmos de Machine Learning, Random Forest y XGBoost.

POR: PEDRO ESTEBAN MILLA SANHUEZA

Capstone project presentado a la Facultad de Ingeniería de la Universidad del Desarrollo para optar al grado académico de Magíster en Data Science

PROFESOR(ES) GUÍA:

Sr. Alonso Astroza
Sr. Cristian Candia

Enero 2023
SANTIAGO

AGRADECIMIENTO

Agradezco a mi familia por el constante apoyo durante este difícil pero hermoso proceso de aprendizaje.

Muchas gracias a los profesores que, durante el programa, compartieron sus conocimientos y experiencias.

Muchas gracias a los compañeros con que me tocó trabajar en grupo, aprendí bastante de ellos gracias a la diversidad de profesionales en el programa.

TABLA DE CONTENIDO

RESUMEN **1**

1. INTRODUCCIÓN **2**

2. TRABAJO RELACIONADO **3**

3. HIPÓTESIS Y OBJETIVOS **4**

4. DATOS Y METODOLOGÍA **5**

4.1. DATOS 5

4.2. METODOLOGÍA 12

5. RESULTADOS **13**

6. CONCLUSIÓN Y TRABAJO FUTURO **19**

BIBLIOGRAFÍA **20**

Resumen

El mundo de los videojuegos experimentó un notable avance cuando fue posible jugar a distancia con cualquier persona en cualquier parte del mundo. El acceso a internet y la notable evolución de la transferencia de datos, permiten que actualmente existan grandes ligas de videojuegos y campeonatos a nivel mundial con equipos profesionales compitiendo.

Este trabajo presenta la implementación de modelos de Machine Learning para predecir el resultado de partidas del videojuego Valorant, utilizando los datos generados en Random Forest y XGBoost como algoritmos de clasificación.

Para realizar las predicciones de partidas futuras, los datos se fueron agrupando de manera que por cada partida existan dos observaciones, una por cada equipo. Además, se tomaron como variables los datos generados en partidas pasadas, generando así una predicción al inicio de la partida futura.

Los resultados indican que XGBoost tiene un mejor desempeño sobre Random Forest, debido a que el ajuste de hiperparámetros y la selección de variables logró un aumento en la capacidad de predecir tanto partidas ganadas como perdidas.

1. Introducción

Valorant es un videojuego online, de tipo shooter táctico en primera persona, desarrollado por la compañía Riot Games y lanzado oficialmente en junio 2020 para Microsoft y MacOs [1].

El juego se basa en enfrentamientos de 2 equipos de 5 jugadores cada uno, donde uno es atacante y el otro es defensor, donde, adicionalmente, cada jugador es un agente que, según sus distintas características (ie. Raze; Sage, entre otros), asume un determinado rol. Las partidas del videojuego se componen de rondas, adjudicándose la victoria el equipo que logre alcanzar la ronda número 13 ganada, una ronda se puede ganar si:

1) el equipo atacante planta una bomba en un espacio designado del mapa y el equipo defensor no es capaz de desarmarla.
2) El equipo elimina a todos los jugadores contrarios.

La existencia de un empate (ambos equipos con 12 rondas ganadas) se resuelve con la victoria de 2 rondas adicionales seguidas de un equipo.

La existencia de múltiples variables del juego, que adicionalmente se afectan entre ellas, generan una complejidad en la determinación a priori de un desenlace de los resultados de acuerdo a datos de inicio.

La analítica avanzada permite generar métodos predictivos y descriptivos, que generan información para la toma de decisiones [2]. En base a lo planteado, la analítica avanzada de los modelos algorítmicos de Machine Learning de aprendizaje supervisado, como Random Forest y XGBoost, se pueden utilizar para predecir [3], al inicio de partida, triunfos o derrotas, en el videojuego Valorant, utilizando datos generados de partidas pasadas.

2. Trabajo Relacionado

Existen varios deportes en donde se utilizan técnicas de Machine Learning para realizar predicciones con algoritmos de clasificación.

En League of Legends (Hitar-Garcia, et al, 2022), los autores desarrollan un modelo de predicción de partidas profesionales utilizando sólo datos previos a la partida, implementando varios algoritmos de clasificación obtuvieron un accuracy sobre 0,7 [4].

En DOTA 2 (Stanlly, et al, 2022), los autores utilizan Random Forest y XGBoost para predecir resultados de partidas, tomando en cuenta solo 2 variables, el personaje utilizado y el item escogido, logrando un accuracy de 0.93 con XGBoost [5].

En la liga profesional de Inglaterra (Pugsee and Pattawong, 2019), los autores desarrollan un modelo con el algoritmo Random Forest como clasificador para predecir el resultado de un determinado partido utilizando los datos históricos de esta liga, los resultados obtenidos indican accuracy sobre 0.7 [6].

3. Hipótesis y Objetivos

La hipótesis planteada para este trabajo es que se pueden realizar pronósticos en videojuegos a través del uso de técnicas de Machine Learning, utilizando los datos generados durante las partidas pasadas, para predecir triunfos o derrotas de partidas futuras, mediante algoritmos de aprendizaje supervisado.

El objetivo general es implementar los modelos Random Forest y XGBoost como algoritmos de clasificación para predecir resultados de partidas futuras en Valorant, tomando como variables predictoras, los datos generados en partidas pasadas. Evaluar cual modelo logra un mejor desempeño a través de la métrica F1 score.

Para alcanzar el objetivo general anteriormente descrito, se requiere cumplir con los siguientes objetivos específicos:

- Consolidar en un set de datos general, la información de los 6 eventos disponibles y configurar para que por cada partida existan 2 observaciones, una por cada equipo.
- Extraer y analizar los datos generados en las rondas, agruparlos por partida y por equipo para añadirlos al set de datos general.
- Generar nuevas variables, tomando en cuenta la información de daño y economía.
- Generar variables promedio, para que la predicción por equipo se realice tomando en cuenta los datos de las últimas 3 partidas jugadas.
- Implementar modelos Random Forest y XGBoost.
- Realizar optimización de hiperparámetros, selección de variables y evaluar su desempeño mediante métrica F1 score.

4. Datos y Metodología

4.1. Datos

Los datos fueron obtenidos desde la API de Riot Games (para este trabajo no fue necesario utilizar la API, ya que los datos fueron suministrados), los cuales consisten de 7 archivos del torneo VCT North America 2021 [7], que corresponden a 6 eventos realizados entre julio y octubre del periodo indicado. Los datos están en formato .xlsx que, a su vez, se dividen en pestañas que contienen información de las partidas, equipos, rondas, indicadores KDA (asesinatos, muertes y asistencias), kills (información detallada de los enfrentamientos), datos de economía, estadísticas de jugadores, ubicaciones y eventos.

Con la finalidad de trabajar los datos para utilizarlos en el modelo de Machine Learning se realizaron los siguientes pasos:

1. Modificar los datos que venían agrupados por partida, expandir para que cada enfrentamiento figure con 2 observaciones, una por cada equipo.
2. De acuerdo al resultado, cada observación fue etiquetada como ganador y perdedor, proceso realizado manualmente.
3. Finalmente, agrupar por partida y por equipo, dejando este set de datos como principal.
4. Se tomaron los datos de estadísticas de jugadores, se agruparon por partida y por equipo para poder añadir al set principal de datos. Además, a partir de estos datos fueron creadas las variables “KDR”, “KD”, “FKD” y “KDA”.
5. Posteriormente, de los datos de economía se realizó la misma operación, se agruparon los datos por partida y por equipo para ser añadida al set de datos principal. Adicionalmente, a partir de estos datos, se creó la variable “avgEconRating”.
6. Luego, se obtuvieron las fechas de los eventos para posteriormente poder dividir los sets de entrenamiento y prueba.

Posterior a estos procesos, el set de datos principal tiene 1.302 observaciones y se compone de las siguientes variables:

- **dates**: fecha correspondiente a la partida (datetime).
- **matchId**: código único por cada partida (int).
- **teamId**: código único por cada equipo (int).
- **teamName**: nombre de equipo (string).
- **map**: escenario en donde se lleva a cabo la partida (string).
- **oppTeamId**: identificador del equipo oponente (int).
- **oppTeamName**: nombre del equipo oponente (string).
- **teamScore**: total de rondas ganadas (int).
- **atkFirst**: 1 si el equipo ataca primero, 0 si defiende primero (int).
- **won**: 1 si el equipo gana la partida, 0 si pierde (int).
- **acs**: average combat score, puntaje promedio de combate, indicador generado por el juego (float).
- **kills**: total de asesinatos (int).
- **firstKills**: total de primeros asesinatos (int).
- **deaths**: total de muertes (int).
- **firstDeaths**: total de primeras muertes (int).
- **assists**: total de asistencias (int).
- **damage**: daño promedio por round en la partida (float).
- **headshots**: total de tiros en la cabeza (int).
- **bodyshots**: total de tiros al cuerpo (int).
- **legshots**: total de tiros a las piernas (int).
- **plants**: total de ocasiones en que se plantó la bomba (int).
- **defusals**: total de ocasiones en que se desactivó la bomba (int).
- **clutches**: total de ocasiones en que el equipo gana estando en desventaja por cantidad de agentes (int).

- **clutchOpponents**: total de ocasiones en que el equipo oponente gana estando en desventaja por cantidad de agentes (int).
- **clutchOpportunities**: total de oportunidades de hacer un clutch (int).
- **KDR**: relación entre kills y deaths (float).
- **KD**: diferencia entre kills y deaths (int).
- **FKD**: diferencia entre firstKills y deaths (int).
- **KDA**: suma de kills y assists sobre deaths (float).
- **spentCreds**: total de créditos utilizados (float).
- **loadoutValue**: valor total de armas, habilidades y escudo de un equipo, representado en créditos, indicador generado por el juego (float).
- **avgEconRating**: puntuación de economía, el cual corresponde al daño realizado por cada 1000 créditos gastados (float).

A continuación, podemos observar la distribución según la variable a predecir.

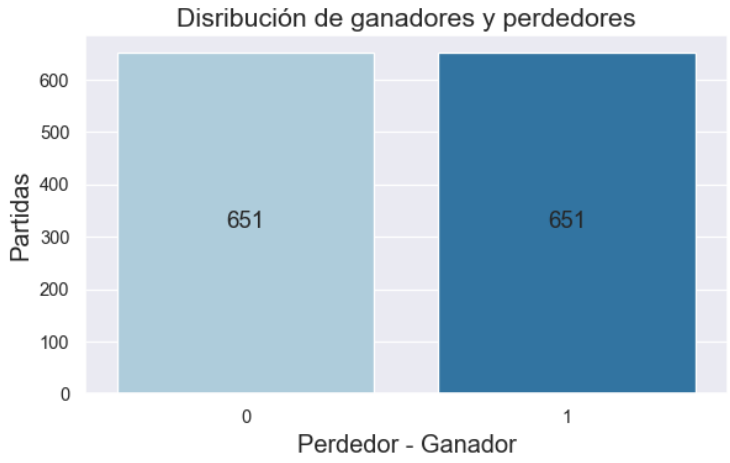

Figura 1: distribución de variable binaria "won".

Debido a la operación realizada descrita anteriormente, al generar que por cada partida existan 2 observaciones, una por cada equipo, en la figura 1 se observa que la variable won queda con la misma cantidad de victorias y de derrotas.

Otro aspecto a analizar son los mapas, ¿habrá significativa diferencia entre mapas?, ¿habrá mapas en donde se realice más daño, más headshots, etc.? En primera instancia vemos con frecuencia se utiliza cada mapa.

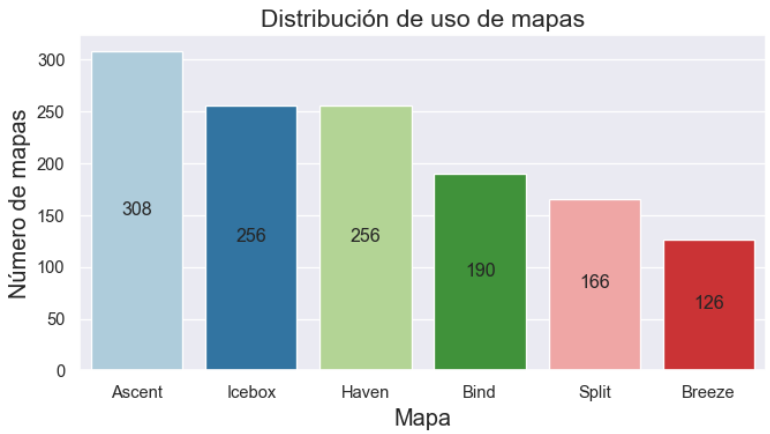

Figura 2: Conteo de mapas.

En la figura 2, se observa que Ascent es el mapa más utilizado, luego están Icebox y Haven. Ahora, visualizamos como se distribuyen algunas variables de daño y economía en cada mapa.

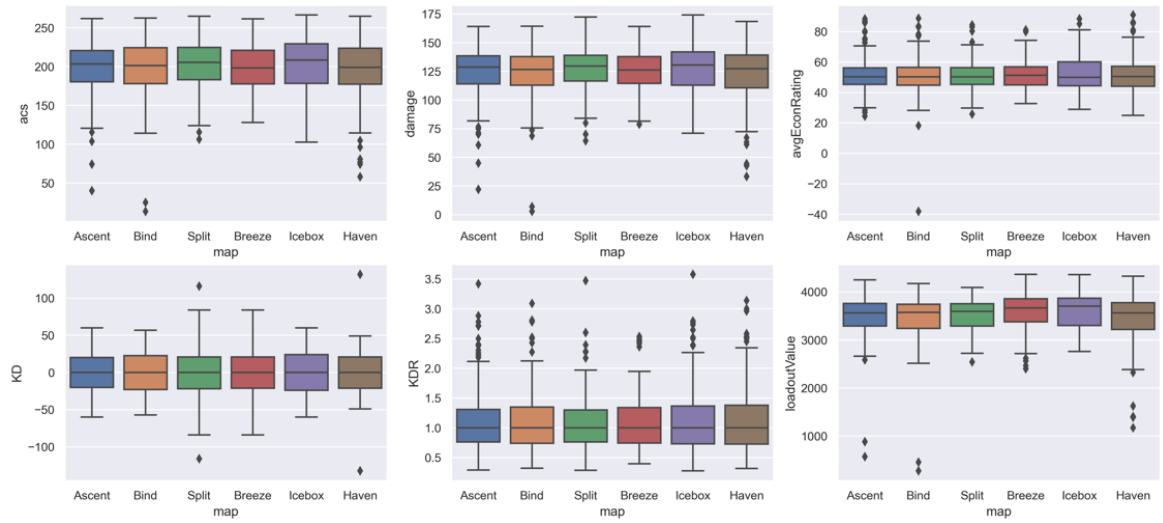

Figura 3: Distribución de variables acs, damage, avgEconRating, KD, KDR y loadoutValue por cada mapa.

En la figura 3, se observa que los rangos inter cuartiles son prácticamente similares en todas las variables analizadas, por lo que podemos decir que no hay diferencia significativa entre los mapas, en general son bastante similares en cuanto a los variables analizadas. Además, observamos datos que se escapan de los rangos, pero sabemos que estos datos, en el caso de la variable KDR, corresponden a equipos que realizan muchos más asesinatos que las muertes que reciben y en el caso de loadoutValue, vemos que hay equipos de menor rendimiento que no logran materializar su juego en armas, habilidades y escudo. Los valores negativos del indicador KD, representan los equipos que tienen más deaths que kills.

A continuación, se analizan otras variables que son de interés saber su distribución dentro de cada mapa.

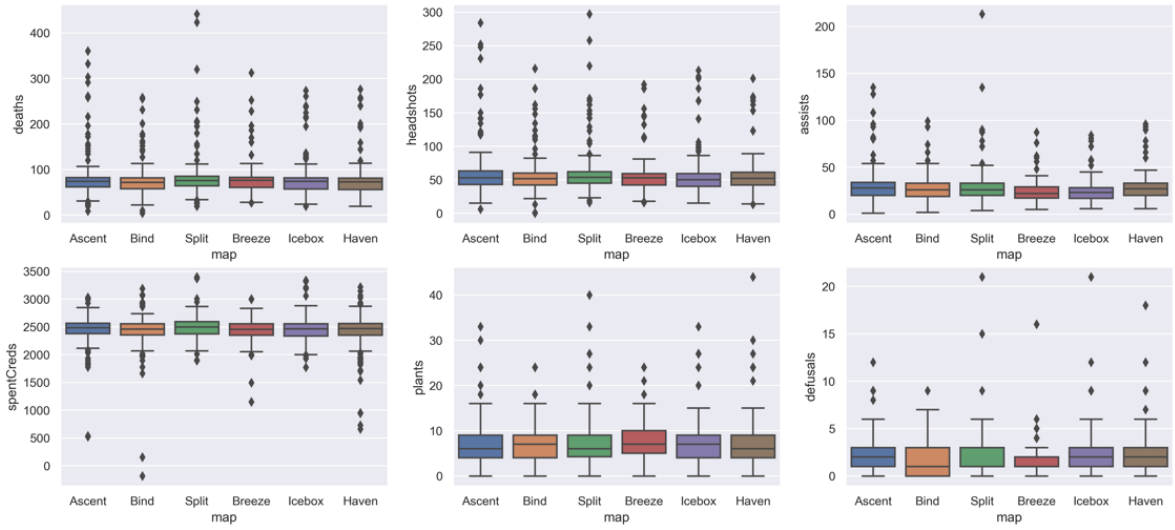

Figura 4: Distribución de variables deaths, headshots, assists, spentCreds, plants y defusals por cada mapa.

En la figura 4, se observa que el rango inter cuartil de las variables deaths y headshots, son bastante similares, pero que el área del rango sea menor indica que hay bastantes datos por sobre la media, hay una cantidad considerable de equipos que sufren muchas deaths y también una cantidad considerable de equipos que realizan gran cantidad de headshots.

Para la variable spentCreds, también se observa un rango inter cuartil similar entre los mapas, sólo hay menor cantidad de equipos que gastan pocos créditos durante la partida.

En base a este análisis, podemos decir que no existe una diferencia considerable entre mapas, en general son bastante equilibrados y no se observa que en alguno éstos se tenga una mayor ventaja en algún aspecto.

Como se mencionó anteriormente, el data set consta de 6 eventos, por lo que a continuación se verá cuáles son los equipos más frecuentes y más ganadores.

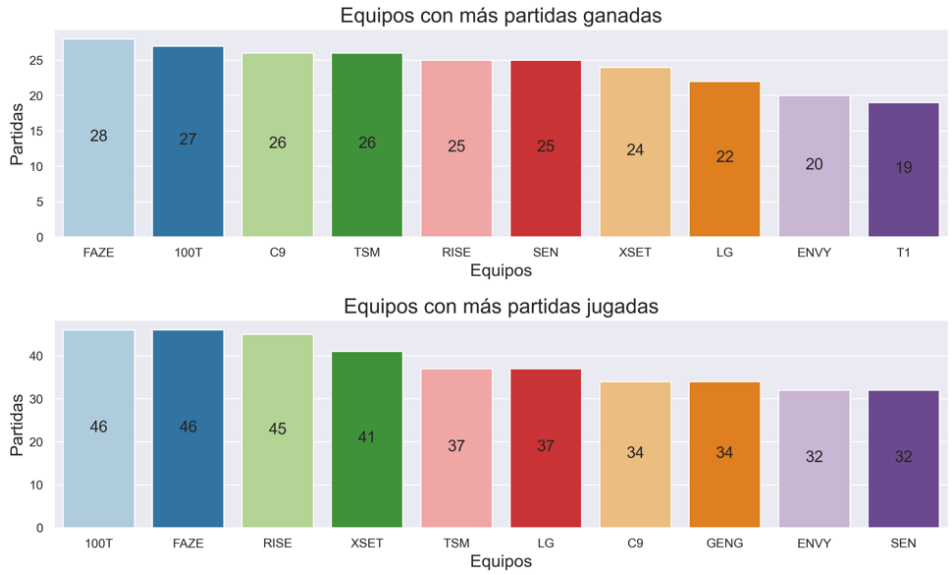

Figura 5: Equipos con más partidas ganadas y con más partidas jugadas.

En la figura 5, se observa que dentro de este grupo de equipos en la figura 5, hay 11 equipos distintos, por lo que estos son los equipos que llegan a instancias más avanzadas del torneo.

También resulta interesante visualizar algunos indicadores por equipo, ¿los equipos que más ganan y juegan, tienen mejores indicadores de daño y de economía?

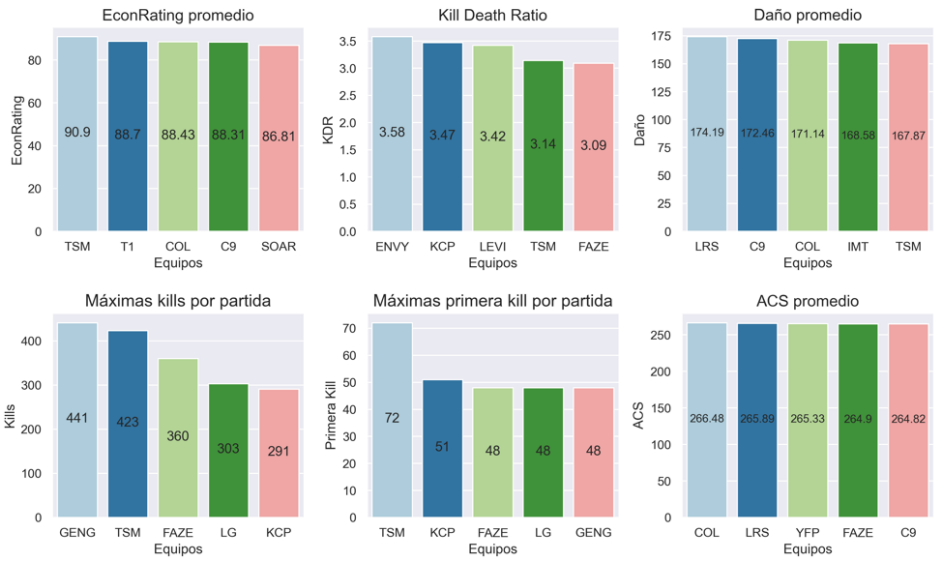

Figura 6: Equipos con mayores indicadores de avgEconRating, KDR, damage, kills, firstKills, acs.

En la figura 6, se puede observar que hay equipos de los más ganadores que están dentro de los equipos con mejores indicadores, hay equipos como TSM que es líder tanto en avgEconRating como firstKills, ENVY es líder en KDR y GENG es líder kills.

A continuación, se implementa un mapa de calor para observar la correlación de las variables predictoras con la variable a predecir.

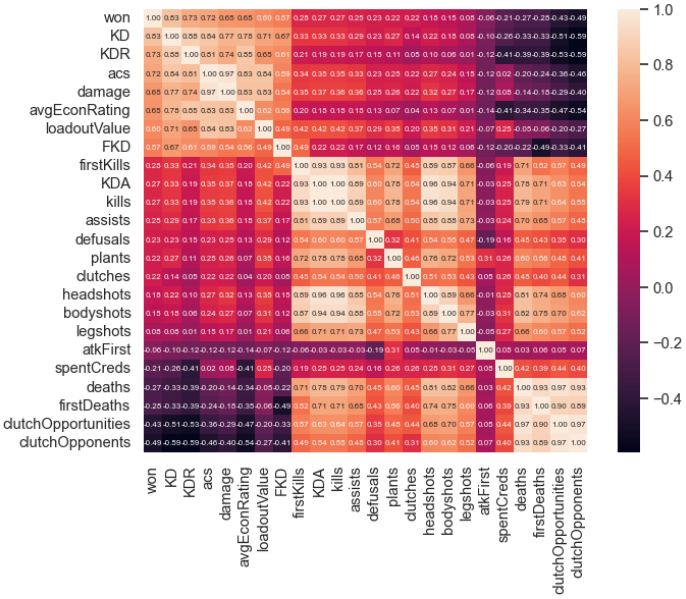

Figura 7: Mapa de calor, indica correlación entre variables predictoras y variable a predecir.

En la figura 7, se observa en el mapa de calor que las variables de indicadores de ataque y economía KD, KDR, acs, damage, avgEconRating, loadoutValue y FKD poseen una alta correlación positiva respecto a la variable won. Además, las variables deaths, firstDeaths, clutchOpportunities y clutchOponnents presentan la mayor correlación negativa. Con esta implementación, se entrega una primera idea de que variables nos ayudarían a predecir mejor la variable objetivo.

4.2. Metodología

El objetivo es realizar predicciones en el inicio de partidas futuras, tomando la información de partidas pasadas, debido a esto, no es posible utilizar algunas variables que se van generando durante el juego, ya que en el inicio de la partida los únicos datos que se tienen son el mapa, si el equipo ataca primero y la fecha. Para las demás variables, se utilizará la estrategia de que, para realizar las predicciones, se tome en cuenta la media de las últimas 3 partidas como variables predictoras. Por ejemplo, la variable promedio acs en la partida a predecir, corresponda al promedio acs de las últimas 3 partidas.

Para realizar las predicciones se utilizarán los modelos de Machine Learning, Random Forest y XGBoost como clasificadores. Random Forest es un meta estimador que ajusta una serie de clasificadores de árboles de decisión en varias submuestras del conjunto de datos y utiliza el promedio para mejorar la precisión predictiva y controlar el sobreajuste; para controlar el consumo de memoria, la complejidad y el tamaño de los árboles se configurarán los valores de hiperparámetros [8]. XGBoost es un optimizador del refuerzo de gradiente distribuido, diseñado para ser altamente eficiente, flexible y portátil [9]; en donde se debe establecer tres tipos de hiperparámetros: generales, de refuerzo o boosting y de tareas de aprendizaje [10].

Como métrica de evaluación y comparación para los modelos Random Forest y XGBoost, se utilizará F1 score. La métrica F1 score se puede interpretar como una media armónica de precision y recall, donde el F1 score alcanza su mejor puntuación en 1 y su peor en 0.

Es decir, la contribución relativa de precision y recall al F1 score es igual. La fórmula para F1 score es [11]:

$$F1 = \frac{2 \times (precision \times recall)}{(precision + recall)}$$

Se implementarán ambos modelos inicialmente con hiperparámetros por defecto, luego, se realizará ajuste de hiperparámetros y se evaluará nuevamente el modelo. Se visualizarán la importancia de las variables para analizar cuáles son mas o menos significativas y finalmente se realizará un resumen de los modelos evaluados con sus respectivos indicadores F1 score.

5. Resultados

La primera operación a realizar es crear las variables promedio, para esto, se agrupa el set de datos por equipo y se ordenan las partidas de acuerdo a la fecha en que se realizaron, se toma un tamaño de ventana móvil 3 para realizar el cálculo de la nueva variable, no tomando en cuenta la variable actual a predecir, es decir, la nueva variable del cuarto partido de un determinado equipo, corresponde a la media de los 3 anteriores. Al realizar esta operación se generarán datos nulos debido a que los equipos que no pasan las primeras etapas del torneo, no alcanzan a tener más de 3 partidas, por lo que el set de datos queda con 837 observaciones.

Las nuevas variables a las cuales se le calculó la ventana móvil, son las siguientes:

- teamScore_rolling.
- acs_rolling.
- kills_rolling.
- firstKills_rolling.
- deaths_rolling.
- firstDeaths_rolling.
- assists_rolling.

- damage_rolling.
- headshots_rolling.
- bodyshots_rolling.
- legshots_rolling.
- plants_rolling.
- defusals_rolling.
- clutches_rolling.
- clutchOpponents_rolling.
- clutchOpportunities_rolling.
- KDR_rolling.
- KD_rolling.
- FKD_rolling.
- KDA_rolling.
- spentCreds_rolling.
- loadoutValue_rolling.
- avgEconRating_rolling.

La variable map se trabajará como dummy, de tal forma que exista una columna por cada mapa donde 1 = la partida pertenece a ese mapa y 0 = la partida no pertenece a ese mapa. Las nuevas variables son las siguientes:

- map_Ascent.
- map_Bind.
- map_Breeze.
- map_Haven.
- map_Icebox.
- map_Split.

Para dividir el set de datos, se tomarán los primeros 4 eventos como entrenamiento y los 2 últimos eventos como prueba a predecir, quedando con las siguientes dimensiones:

- Set de entrenamiento, variables predictoras: 685 observaciones, 30 columnas.
- Set de entrenamiento, variable a predecir: 685 observaciones, 1 columna.
- Set de prueba, variables predictoras: 152 observaciones, 30 columnas.
- Set de prueba, variable a predecir: 152 observaciones, 1 columna.

Se procede a implementar el modelo Random Forest, sólo se ingresan los siguientes hiperparámetros, los demás se dejan por defecto:

- n_estimators = 500.
- random_state = 18.

Con esta configuración, se obtiene un F1 score de 0.62. Por lo que se procede a realizar una optimización de hiperparámetros para determinar con que valores podríamos tener un mejor resultado. Los hiperparámetros a utilizar son los siguientes:

- n_estimators = 1200.
- min_samples_split = 5.
- min_samples_leaf = 1.
- max_features = 'auto'.
- max_depth = 10.
- Bootstrap = True.
- n_jobs = -1.
- random_state = 32.

Con esta configuración de hiperparámetros se obtuvo un F1 score de 0.64. Para analizar la importancia de las variables se implementa la siguiente gráfica.

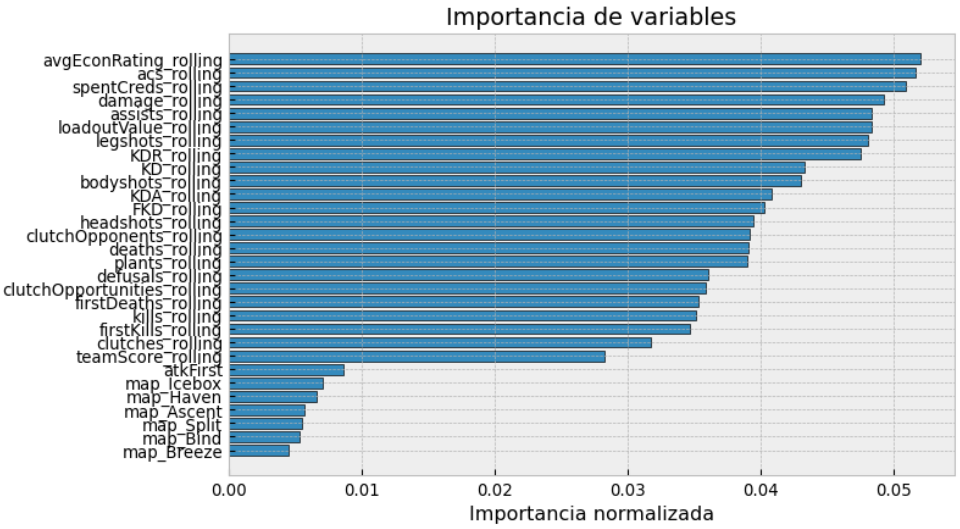

Figura 8: Importancia de variables según implementación del modelo Random Forest.

Dada la importancia de variables, podemos decir que indicadores de economía avgEconRating_rolling, spentCreds_rolling y loadoutValue_rolling tienen una alta importancia en el modelo, así mismo los indicadores de daño acs_rolling, damage_rolling y assists_rolling. Se observa que la información de los mapas es la menos relevante, probablemente debido a la uniformidad de sus distribuciones.

Se procede a implementar el modelo XGBoost, inicialmente, sólo se ingresan los siguientes hiperparámetros, los demás se dejan por defecto:

- random_state = 14.

Con esta configuración, se obtiene un F1 score de 0.56. Por lo que se procede a realizar una optimización de hiperparámetros para determinar con que valores podríamos tener un mejor resultado. Los hiperparámetros a utilizar son los siguientes:

- base_score = 0.2.
- booster = 'gblinear'.
- learning_rate = 0.1.
- max_features = 'auto'.
- n_estimators = 500.

- reg_alpha = 0.5.
- reg_lambda = 5.
- random_state = 15.

Con esta configuración de hiperparámetros se obtuvo un F1 score de 0.68. Para analizar la importancia de las variables se implementa la siguiente gráfica.

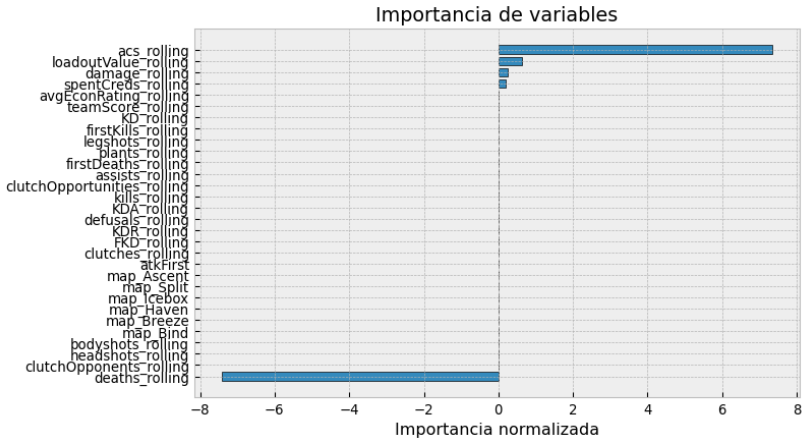

Figura 9: Importancia de variables según implementación del modelo XGBoost.

Analizando la figura 9, podemos decir que hay gran cantidad de variables que no generan importancia positiva ni negativa, vemos que nuevamente variables de daño acs_rolling, damage_rolling y variables de economía loadoutValue_rolling, spentCreds_rolling, tienen una alta importancia. La variable deaths_rolling destaca por su importancia negativa.

Se procede a seleccionar las siguientes variables para evaluar el modelo, acs_rolling, damage_rolling, deaths_rolling, headshots_rolling, loadoutValue_rolling, spentCreds_rolling y clutchOpponents_rolling, con la misma configuración de hiperparámetros obteniendo un F1 score de 0.68. Para analizar la importancia de las variables se implementa la siguiente gráfica.

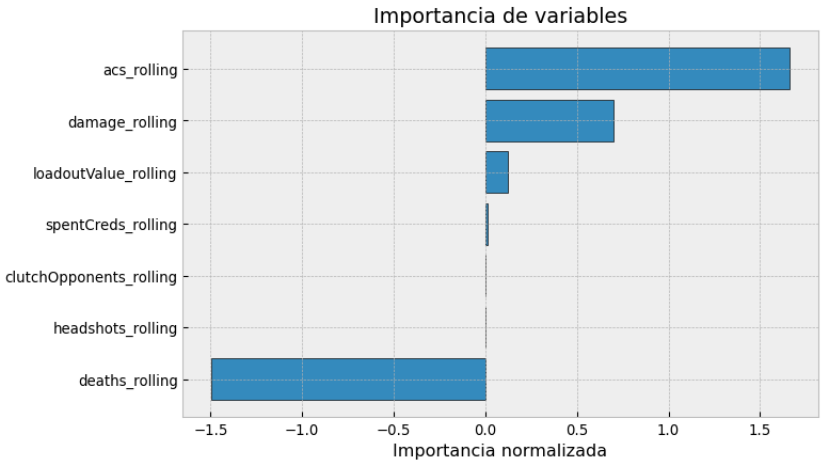

Figura 10: Importancia de variables seleccionadas según implementación del modelo XGBoost.

En la figura 10, se observa que las variables de daño seleccionadas tienen una mayor importancia al evaluar el modelo XGBoost.

Finalmente, estas son los indicadores F1 score por cada modelo implementado:

- Random Forest sin ajuste de hiperparámetros: 0.62.
- Random Forest con ajuste de hiperparámetros: 0.64.
- XGBoost sin ajuste de hiperparámetros: 0.56.
- XGBoost con ajuste de hiperparámetros: 0.68.

6. Conclusiones

El presente trabajo muestra que es posible realizar predicciones futuras basado en datos del pasado, utilizando modelos de Machine Learning de aprendizaje supervisado, utilizados como algoritmos de clasificación.

Es fundamental preparar los datos adecuadamente para lograr el objetivo planteado, en este caso, adaptar los datos para predecir partidas.

De acuerdo a la métrica F1 score implementada para evaluar los modelos, se obtuvo que XGBoost tuvo un mejor desempeño, destacando un aumento de 0.12 entre el modelo inicial por defecto y el modelo con parámetros optimizados. Además, es positivo que, al disminuir la cantidad de variables, el modelo mantiene el F1 score en 0.68.

A futuro es recomendable evaluar los modelos con mayor cantidad de datos, ya que con la operación de cálculo de variables promedio, quedan equipos fuera al generarse datos nulos.

Además, hay información a nivel de rondas, que no fue posible ingresar en el modelo, por ejemplo, los agentes, las armas, etc. Sería una buena práctica, buscar la forma en que estas variables se vean reflejadas a nivel de partida y evaluar su importancia para los modelos.

Un trabajo similar se puede realizar a nivel de rondas, donde se pueden considerar los aspectos anteriormente mencionados, llegando a nivel de jugador en cuanto a análisis y predicciones.

Finalmente, los análisis, visualizaciones y predicciones realizadas en este trabajo, serían de gran ayuda para equipos que juegan Valorant a nivel competitivo, ya que es posible analizar puntos altos y bajos, defectos y virtudes, en cuanto a variables del juego, lo que permite tomar decisiones para ir mejorando el desempeño del equipo.

Bibliografía

[1] “Valorant”, Valorant Wiki, 2021, [En línea]. Disponible en: https://valorant.fandom.com/es/wiki/Valorant.

[2] Harshdeep Singh, “Using Analytics for Better Decision-Making”, 2018, [En línea]. Disponible en: https://towardsdatascience.com/using-analytics-for-better-decision-making-ce4f92c4a025.

[3] Rohit Dwivedi, “Random Forest Vs XGBoost – Comparing Tree-Based Algorithms (With Codes)”, 2020, [En línea]. Disponible en: https://analyticsindiamag.com/random-forest-vs-xgboost-comparing-tree-based-algorithms-with-codes/.

[4] J. -A. Hitar-Garcia, L. Moran-Fernandez and V. Bolon-Canedo, "Machine Learning Methods for Predicting League of Legends Game Outcome," 2022 in IEEE Transactions on Games.

[5] Stanlly, F. A. Putra and N. N. Qomariyah, "DOTA 2 Win Loss Prediction from Item and Hero Data with Machine Learning," *2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, 2022, pp. 204-209.

[6] Pakawan Pugsee and Pattarachai Pattawong. 2019. Football Match Result Prediction Using the Random Forest Classifier. In Proceedings of the 2nd International Conference on Big Data Technologies (ICBDT '19). Association for Computing Machinery, New York, NY, USA, 154–158.

[7] “VALORANT Champions Tour 2021”, Esports Charts, 2021, [En línea]. Disponible en: https://escharts.com/events/valorant-champions-tour-2021.

[8] “sklearn.ensemble.RandomForestClassifier”, Scikit Learn, [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[9] “XGBoost Documentation”, DMLC XGBoost, [En línea]. Disponible en: https://xgboost.readthedocs.io/en/stable/.

[10] “XGBoost Parameters”, DMLC XGBoost, [En línea]. Disponible en: https://xgboost.readthedocs.io/en/stable/parameter.html.

[11] “sklearn.metrics.f1_score”, Scikit Learn, [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.