

# *LOL min 10 win predict*

David Lajas

LEAGUE<sup>OF</sup>  
LEGENDS

# *Introducción y Objetivo*

**Contexto:** El proyecto utiliza datos de partidas de alto nivel (Diamante I) en los primeros 10 minutos de juego.

**Objetivo:** Desarrollar un modelo de clasificación capaz de predecir qué equipo ganará la partida y las probabilidades que tiene de hacerlo, basándose únicamente en el estado inicial del juego.




# Dataset

## Fuente:

<https://www.kaggle.com/datasets/bobbyscience/league-of-legends-diamond-ranked-games-10-min>

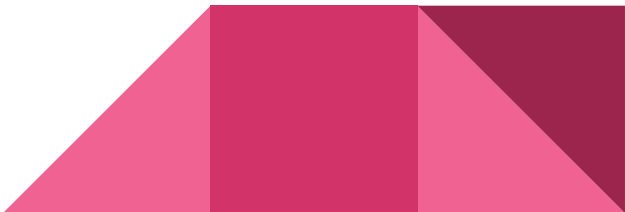
9879 filas × 40 columnas, sin nulos

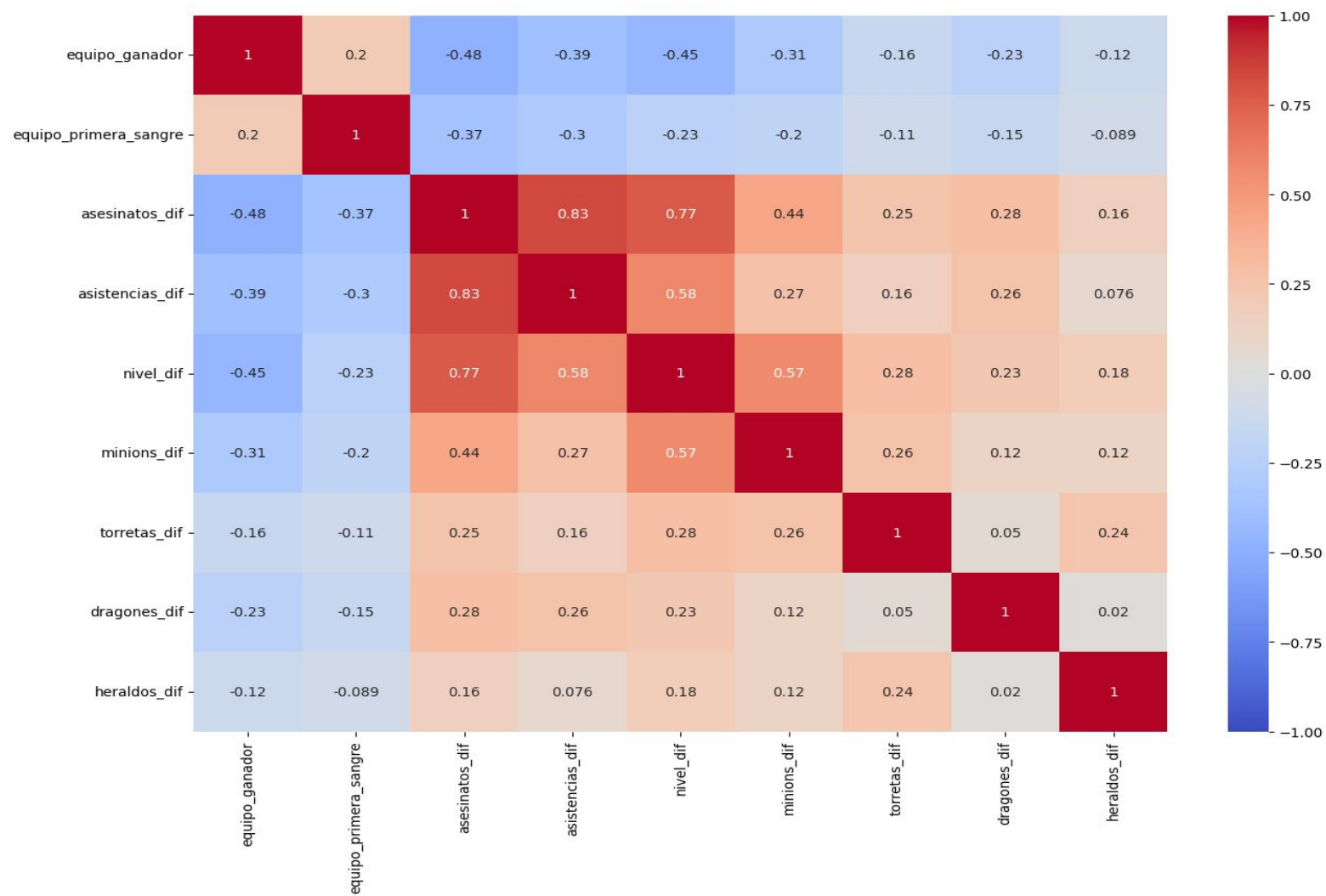
## Variables Clave:

- Oro total y experiencia total.
  - Asesinatos (Kills), muertes (Deaths) y asistencias.
  - Objetivos neutrales: Dragones y Heraldos de la Grieta.
  - Estructuras destruidas (Torretas) y súbditos eliminados (CS).
- 

[illegible]

# Metodología

- **Limpieza:** Eliminación de columnas redundantes y sin peso.
  - **Ingeniería de Características:**
    - blueWins - Equipo Ganador
    - redFirstBlood - Equipo primera sangre
    - Creación de métricas de "diferencia"
    - Traducción de columnas al español.
- 



# *Implementación del Modelo*


**Target:** equipo\_ganador

**Algoritmos:** Clasificación binaria (Equipo azul o equipo rojo)

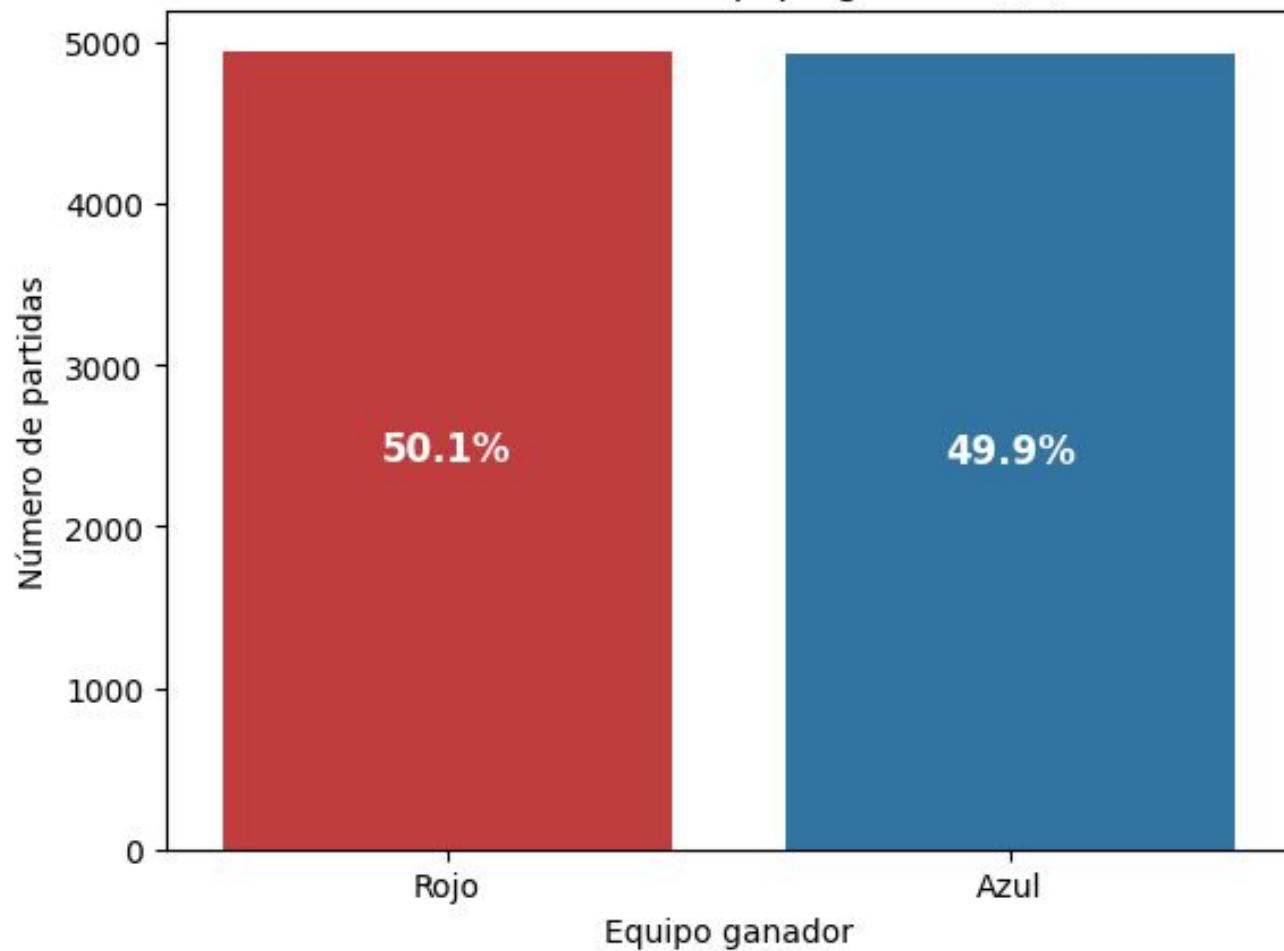
**División de Datos:** 80% para entrenamiento y 20% para prueba.  
Estratificación por equipo\_ganador.

**Escalado:** Standard Scaler

**Entrenamiento:** Los modelos aprenden a asignar pesos a cada variable para determinar su impacto en la probabilidad de victoria.



Distribución del equipo ganador (%)





# Modelos Utilizados

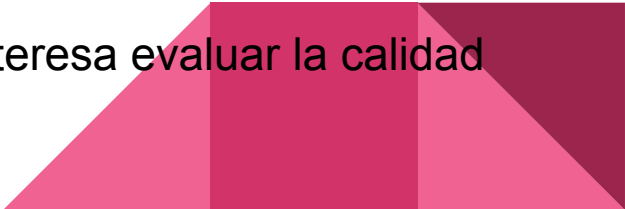
## Algoritmos evaluados:

- Supervisados: Regresión logística, Gradient Boosting, Random Forest, AdaBoost, XGBoost, KNN, SVC, Red neuronal secuencial (Keras) y Stacking Classifier.
- No supervisados: KMeans.

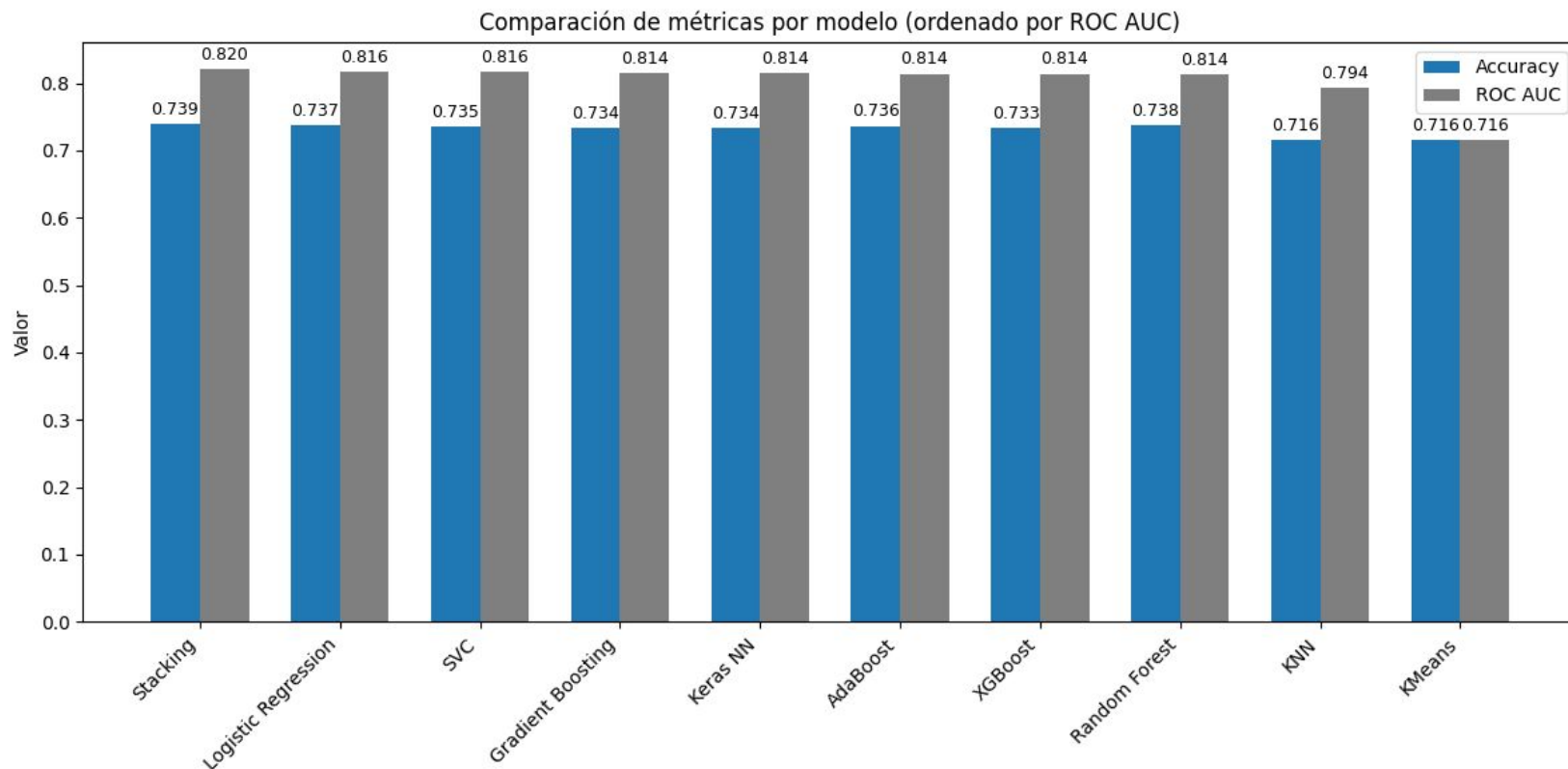
## Validación:

- GridSearchCV / RandomizedSearchCV para ajuste de hiperparámetros
- Validación cruzada de 3-5 folds según modelo.

**Medida de valoración escogida: ROC-AUC** porque nos interesa evaluar la calidad de las probabilidades del modelo, no solo si acierta o falla.



# Resultados de los modelos



# Stacking Classifier

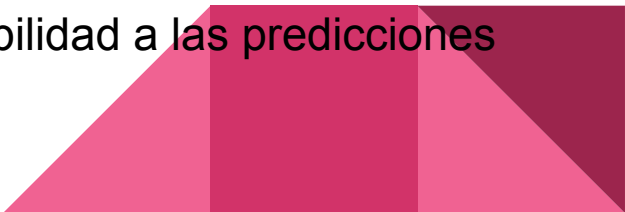
**Modelos base (nivel 0):** Regresión logística, SVC, XGBoost

**Modelo meta (nivel 1):** Regresión logística

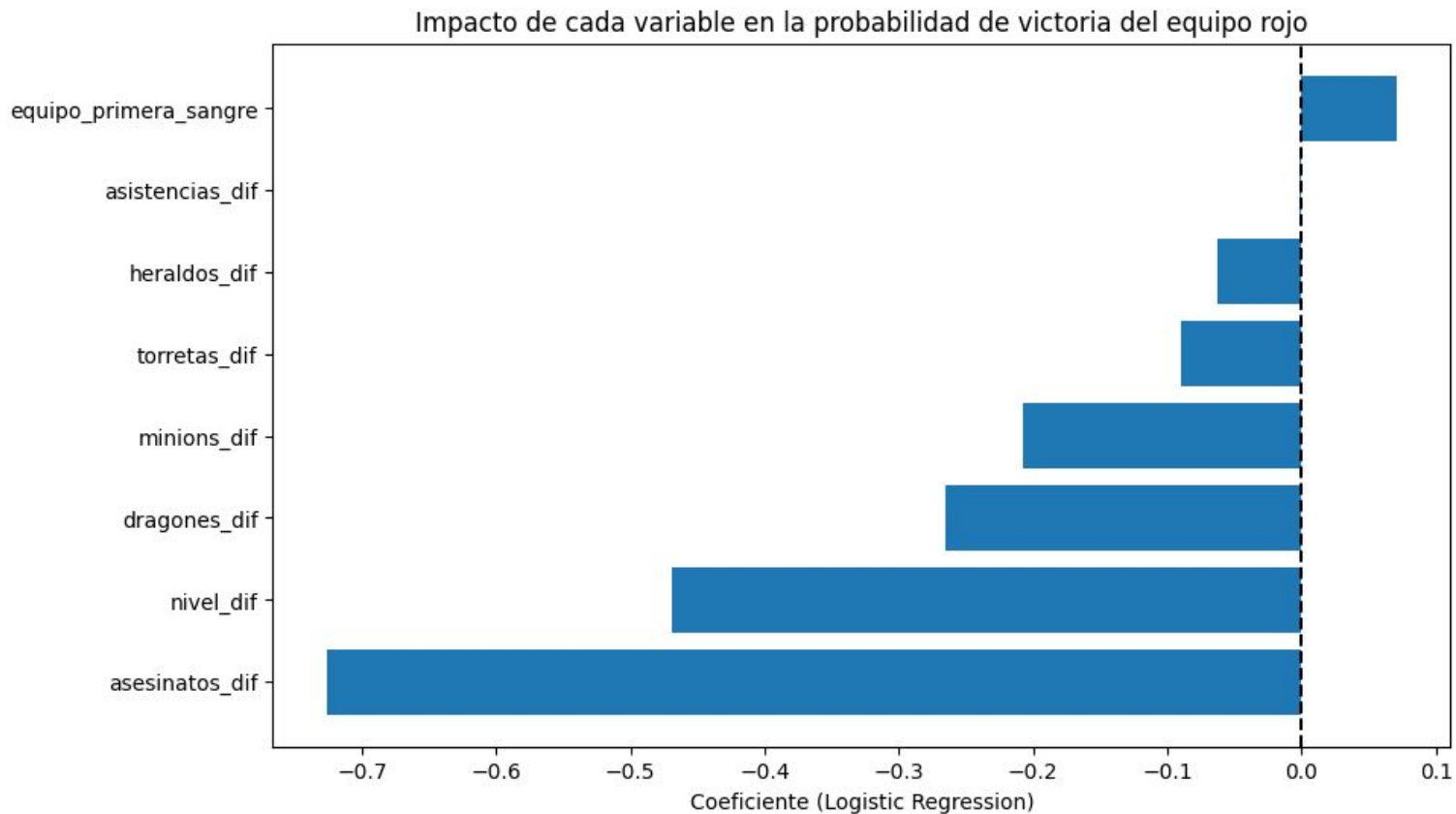
En el **stacking**, los modelos base generan predicciones que se utilizan como variables de entrada para entrenar un modelo meta, que es el encargado de realizar la predicción final.

Al aprovechar complementariedad entre modelos, generaliza mejor

El modelo tiene buena capacidad para asignar mayor probabilidad a las predicciones correctas que a las incorrectas.



# Coeficientes de Stacking




# ***Limitaciones y mejoras***


## **Limitaciones:**

- Mal funcionamiento de la API de Riot Games
- Solo se ve la ventaja global
- Ausencia de campeones elegidos

## **Mejoras:**

- Computer vision
  - Predicción por minuto
  - Introducir los campeones y roles seleccionados
- 

# Conclusiones

- Crear **nuevas variables** comparando los equipos ayudó al modelo a entender mejor los datos y mejorar las predicciones.
  - El **Stacking** combina modelos complementarios, aumentando robustez y estabilidad.
  - El modelo puede predecir resultados desde el inicio del juego con bastante precisión (**ROC AUC  $\approx 0.82$** ).
  - El modelo es **generalizable** y refleja ventajas tempranas de forma consistente.
- 

¡Muchas gracias por vuestra atención!

