



# IMPLEMENTACIÓN DE ALGORITMOS DE APRENDIZAJE SUPERVISADO PARA EL JUEGO *Tic Tac Toe*

ANGELO BARBIERI, THOMAS BUSTAMANTE, EDWARD GUZMÁN  
{angelo.barbieri1901, thomas.bustamante1901, edward.guzman1901}@alumnos.ubiobio.cl

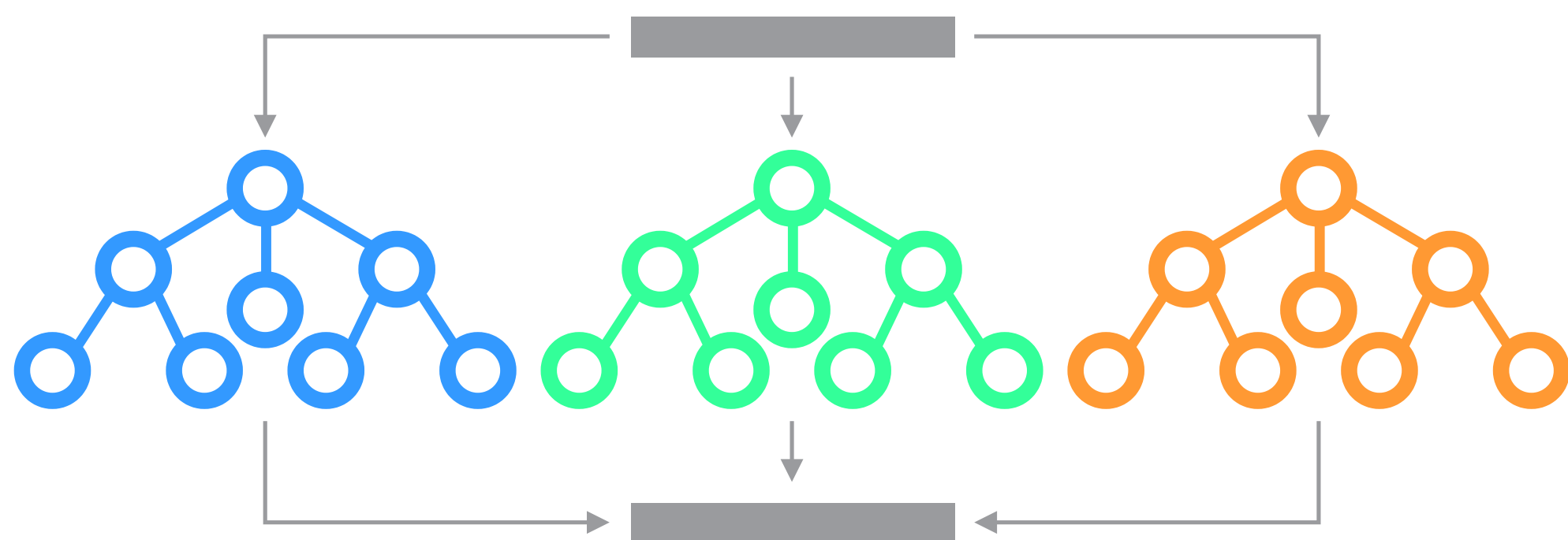
## RESUMEN

Este trabajo presenta un análisis comparativo de algoritmos de aprendizaje supervisado aplicados al juego *Tic Tac Toe*, evaluando la predicción de modelos como *Naive Bayes* (*Gaussian*), *Random Forest*, *Support Vector Machine* y *XGBoost* intentando predecir la mejor jugada a partir de juegos previamente jugados con el algoritmo *Minimax* proporcionando una perspectiva práctica sobre el uso del aprendizaje supervisado en sistemas interactivos.

## TEORÍA

El aprendizaje supervisado es una técnica de Machine Learning donde un modelo se entrena utilizando un conjunto de datos etiquetados para predecir resultados o clasificaciones futuras [1]. En este caso, se empleó para aprender la mejor jugada en el juego de *Tic Tac Toe*, integrando varios algoritmos para comparar su desempeño.

El algoritmo *Naive Bayes* (*Gaussian*) basado en el teorema de Bayes, asume independencia entre características. Es eficiente y adecuado para datos categóricos o continuos, debido a su simplicidad y rapidez en el entrenamiento y la predicción [2]. *Random Forest*, es un conjunto de árboles de decisión entrenados en distintas particiones del dataset (ver figura 1). Combina votación para mayor precisión y reduce el riesgo de sobreajuste [3].



**Figura 1:** Conjunto de árboles de decisión en *Random Forest*. Fuente: Elaboración Propia.

(*Support Vector Machines*), encuentra un hiperplano que separa las clases en el espacio de características. Es poderoso en datasets con una clara separación entre clases [4]. *XGBoost* (*Extreme Gradient Boosting*), es un algoritmo de *boosting* optimi-

## TEORÍA (CONTINUACIÓN)

zado que combina múltiples árboles débiles para lograr predicciones robustas y precisas, maximizando el rendimiento en problemas complejos [5].

## MATERIALES & MÉTODOS

El dataset utilizado para entrenar los modelos fue construido empleando el algoritmo *Minimax*, un enfoque clásico de teoría de juegos que minimiza el error máximo posible [6]. Cada estado del tablero se etiqueta con una o más posibles clases que representan las mejores jugadas. Para abordar casos con múltiples clases, el dataset se extiende repitiendo cada estado con sus posibles clases. La métrica de desempeño utilizada es *accuracy* calculada de manera personalizada. Una predicción se considera correcta si la clase predicha es una de las posibles clases para el estado del tablero. La optimización de los hiperparámetros se realizó mediante *Exhaustive Grid Search*, probando combinaciones de parámetros relevantes para cada algoritmo. Este proceso utilizó *K-Fold* con 5 particiones para evaluar el desempeño de cada combinación.

Los clasificadores utilizados se implementaron con *Scikit-learn*. Además, se desarrolló una interfaz gráfica utilizando *Qt*, permitiendo a los usuarios interactuar y visualizar las jugadas generadas por los algoritmos.

## RESULTADOS

**Tabla 1:** *Accuracy* de los clasificadores en la predicción de jugadas óptimas en *Tic Tac Toe*. Fuente: Elaboración Propia.

| Clasificador | Accuracy (%) |
|--------------|--------------|
| GNB          | 65           |
| RFC          | 80           |
| SVM          | 68           |
| XGB          | 89           |

Como se observa en la tabla 1, *XGBoost* alcanza el mayor desempeño con un *accuracy* del 89%, seguido por *Random Forest* con un 80%. Por otro lado, *Gaussian Naive Bayes* y *SVM* presentan desempeños más bajos, lo que sugiere que algunos clasificadores podrían ser más adecuados para este tipo de problema.

En la tabla 2, *XGBoost* lidera con 73 victorias y solo 15 derrotas, mientras que *Random Forest* alcanza 61 victorias pero con un mayor tiempo total (24.45 s)

**Tabla 2:** Resultados del desempeño de distintos algoritmos en 100 partidas de *Tic Tac Toe* contra un jugador que realiza movimientos aleatorios. La tabla muestra el número de triunfos, derrotas, empates y el tiempo total. Fuente: Elaboración propia.

| Algoritmos | Triunfos | Derrotas | Empates | Tiempo [s] |
|------------|----------|----------|---------|------------|
| Random     | 26       | 67       | 7       | 17.36      |
| DFS        | 43       | 53       | 4       | 15.13      |
| BFS        | 39       | 58       | 3       | 15.26      |
| GBFS       | 58       | 1        | 51      | 16.1       |
| RFC        | 61       | 28       | 11      | 24.45      |
| XGB        | 73       | 15       | 12      | 15.95      |
| GNB        | 35       | 42       | 23      | 15.63      |
| SVM        | 43       | 51       | 6       | 16.35      |

Por otro lado, *GBFS* destaca por evitar derrotas (solo 1) gracias a su estrategia defensiva, aunque con menos victorias (58). Algoritmos como *Gaussian Naive Bayes* y *SVM* muestran un desempeño inferior, con menos triunfos y más derrotas, reflejando su menor efectividad.

## CONCLUSIONES

- La comparación entre los resultados de victorias y los valores de *accuracy* muestra que los algoritmos con mayor precisión predictiva, como *XGBoost* (89%) y *RFC* (80%), son también los que obtuvieron más triunfos en las partidas simuladas. Esto evidencia que la construcción del dataset con *Minimax* fomenta un estilo de juego más ofensivo en comparación a las estrategias basadas en algoritmos de búsqueda.
- El análisis demuestra que *XGBoost*, como algoritmo de aprendizaje supervisado, logra los mejores resultados en victorias (73) con un tiempo de ejecución (15.95 s) comparable e incluso inferior al mejor algoritmo de búsqueda, *GBFS* (16.1 s). Esto evidencia que los algoritmos de aprendizaje supervisado no solo ofrecen mejores resultados, sino que lo hacen en tiempos similares o inferiores a los de búsqueda, siendo la opción más eficiente y efectiva.

## REFERENCIAS

- [1] LJ Muhammad, Ebrahim A Algehyne, Sani Sharif Usman, Abdulkadir Ahmad, Chinmay Chakraborty, and Ibrahim Alh Mohammed. Supervised machine learning models for prediction of covid-19 infection using epidemiology dataset. *SN computer science*, 2(1):1–13, 2021.
- [2] Or Peretz, Michal Koren, and Oded Koren. Naive bayes classifier—an ensemble procedure for recall and precision enrichment. *Engineering Applications of Artificial Intelligence*, 136:108972, 2024.
- [3] Aakash Parmar, Rakesh Katariya, and Vatsal Patel. A review on random forest: An ensemble classifier. In *International conference on intelligent data communication technologies and internet of things (ICICI) 2018*, pages 758–763. Springer, 2019.
- [4] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.
- [5] Khansa Afifah, Intan Nurma Yulita, and Indra Sarathan. Sentiment analysis on telemedicine app reviews using xgboost classifier. In *2021 International Conference on Artificial Intelligence and Big Data Analytics*, pages 22–27. IEEE, 2021.
- [6] Issah Musah, Douglas Kwasi Boah, and Baba Seidu. A comprehensive review of solution methods and techniques for solving games in game theory. *Journal of Game Theory*, 9(2):25–31, 2020.