Universidad del Valle
Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación
Inteligencia artificial
Informe sobre *Machine learning*

League of legends es un juego multijugador online en el que 10 personas se enfrentan divididas en dos equipos de cinco jugadores cada uno. Cada jugador elige un personaje con habilidades únicas para luchar. El juego consiste en trabajar en conjunto para derrotar al equipo rival y destruir su base, enfrentando unas líneas de defensa enemigas conocidas como torretas y luchando contra monstruos neutrales. Dentro del juego, eliminar una estructura enemiga o monstruo le concede ventaja al equipo (estos se conocen como objetivos) y con la estrategia adecuada esta ventaja puede conducir a la victoria. En este taller se utilizará un conjunto de datos que contiene información de más de 50000 partidas de este juego y se quiere construir un modelo que ayude a predecir cuál de los dos equipos será el ganador. Cada partida se describe utilizando los 18 atributos que se presentan en la siguiente tabla. Estos atributos corresponden a avances del juego, logros, y decisiones que tomó cada equipo durante el desarrollo de la partida. La etiqueta de clase es el atributo *winner* cuyos valores pueden ser 1 ó 2 indicando el equipo que gana la partida. En la siguiente tabla se listan los atributos y se presenta su descripción y los posibles valores que pueden tomar.

#	Atributo	Descripción	Posibles valores
1	gameDuration	Duración de la partida en segundos	Valores mayores a cero
2	firstBlood	Equipo que eliminó al primer enemigo	1 - Si el equipo 1 eliminó al primer enemigo 2 - Si el equipo 2 eliminó al primer enemigo
3	firstTower	Equipo que eliminó la primera torreta	0 - Ninguno de los dos equipos consiguió eliminar una torreta antes de que acabara el juego 1 - Si el equipo 1 eliminó la primera torreta 2 - Si el equipo 2 eliminó la primera torreta
4	firstInhibitor	Equipo que eliminó al primer inhibidor	0 - Ninguno de los dos equipos consiguió eliminar un inhibidor antes de que acabara el juego 1 - Si el equipo 1 eliminó el primer inhibidor 2 - Si el equipo 2 eliminó el primer inhibidor
5	firstBaron	Equipo que eliminó al primer barón	0 - Ninguno de los dos equipos consiguió eliminar un barón antes de que acabara el juego 1 - Si el equipo 1 eliminó el primer barón 2 - Si el equipo 2 eliminó el primer barón
6	firstDragon	Equipo que eliminó al primer dragón	0 - Ninguno de los dos equipos consiguió eliminar un dragón antes de que acabara el juego 1 - Si el equipo 1 eliminó el primer dragón 2 - Si el equipo 2 eliminó el primer dragón
7	firstRiftHerald	Equipo que eliminó al primer heraldo	0 - Ninguno de los dos equipos consiguió eliminar un heraldo antes de que acabara el juego 1 - Si el equipo 1 eliminó el primer heraldo 2 - Si el equipo 2 eliminó el primer heraldo
8	t1_towerKills	Cantidad de torretas que ha eliminado el equipo 1	Valores enteros de 0 a 11

9	t1_inhibitorKills	Cantidad de inhibidores que ha eliminado el equipo 1	Valores mayores o iguales a 0
10	t1_baronKills	Cantidad de barones que ha eliminado el equipo 1	Valores mayores o iguales a 0
11	t1_dragonKills	Cantidad de dragones que ha eliminado el equipo 1	Valores mayores o iguales a 0
12	t1_riftHeraldKills	Cantidad de heraldos que ha eliminado el equipo 1	Valores mayores o iguales a 0
13	t2_towerKills	Cantidad de torretas que ha eliminado el equipo 2	Valores enteros de 0 a 11
14	t2_inhibitorKills	Cantidad de inhibidores que ha eliminado el equipo 2	Valores mayores o iguales a 0
15	t2_baronKills	Cantidad de barones que ha eliminado el equipo 2	Valores mayores o iguales a 0
16	t2_dragonKills	Cantidad de dragones que ha eliminado el equipo 2	Valores mayores o iguales a 0
17	t2_riftHeraldKills	Cantidad de heraldos que ha eliminado el equipo 2	Valores mayores o iguales a 0
18	winner	Equipo que gana la partida	1 - Si el equipo 1 es el ganador de la partida 2 - Si el equipo 2 es el ganador de la partida

En la siguiente tabla se muestra como ejemplo una parte del conjunto de datos donde se tiene la información de una partida. En esta partida que duró 2309 segundos (atributo 1) el equipo 1 eliminó al primer enemigo (atributo 2), el equipo 2 eliminó la primera torreta (atributo 3) y el primer inhibidor (atributo 4). Ningún equipo consiguió eliminar un barón (atributo 5). El atributo 13 indica que el equipo 2 eliminó 10 torretas mientras que el atributo 16 permite conocer que el equipo 2 eliminó a 4 dragones. En este caso el ganador de la partida fue el equipo 2 tal como se puede ver en el atributo 18.

Atributo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Valor	2309	1	2	2	0	2	0	2	0	0	0	0	10	2	0	4	0	2

El objetivo de este informe es crear dos notebooks. Uno donde se utilice la técnica de redes neuronales y otro para las técnicas de árboles de decisión y redes Bayesianas. Inicialmente se deben probar diferentes topologías de redes neuronales y modificar los hiperparámetros de tal manera que se puedan obtener modelos que permitan predecir el equipo ganador. Para esto, debe entregar un notebook donde se realicen las siguientes tareas:

1. Leer el archivo games.csv
2. Seleccionar aleatoriamente el 80% del conjunto de datos para entrenar y el 20% restante para las pruebas
3. Utilizar una estrategia para normalizar los datos
4. Construir 5 redes neuronales variando en la topología de la red la cantidad de capas ocultas y de neuronas por cada capa oculta. Puede también variar los hiperparámetros solver y la función de

activación. En todas las pruebas debe usar un random_state=123. Incluya en el notebook una tabla a manera de resumen con el *accuracy* obtenido en cada caso y también las matrices de confusión

5. Indique en el notebook usando una celda de tipo *Markdown* los hiperparámetros que por el momento le permiten obtener la red con mayor *accuracy*
6. Seleccione uno de los hiperparámetros disponibles en la documentación (https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html) que sea diferente al solver, a la función de activación, y al random_state. Realice dos variaciones en el hiperparámetro seleccionado manteniendo los otros hiperparámetros del punto anterior. Indique el *accuracy* obtenido al modificar el hiperparámetro seleccionado y analice si la red mejora, empeora, o mantiene su exactitud. Incluya en el notebook dicho análisis

En el segundo notebook se deben realizar las siguientes tareas:

1. Leer el archivo games.csv
2. Seleccionar aleatoriamente el 80% del conjunto de datos para entrenar y el 20% restante para las pruebas
3. Utilizar una estrategia para normalizar los datos
4. Configurar los hiperparámetros del árbol de decisión de la siguiente manera: criterion=gini, splitter=best, y random_state=123. Obtener 10 árboles de decisión que resultan de modificar el hiperparámetro max_depth desde 100 hasta 1000 con incrementos de 100
5. Incluya en el notebook una tabla con el *accuracy* para los 10 árboles del punto anterior
6. Repita el mismo procedimiento del punto 4 usando como hiperparámetros criterion=entropy, splitter=best, y random_state=123. Compare los 10 valores obtenidos con los del punto 5
7. Repita el mismo procedimiento del punto 4 usando como hiperparámetros criterion=entropy, splitter=random, y random_state=123.
8. Indique en el notebook los hiperparámetros que por el momento le permiten obtener el árbol con mayor *accuracy*
9. Seleccione uno de los hiperparámetros disponibles en la documentación (https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html) que sea diferente al criterion, splitter, max_depth, y random_state. Realice dos variaciones en el hiperparámetro seleccionado manteniendo los otros hiperparámetros del punto anterior. Indique el *accuracy* obtenido al modificar el hiperparámetro seleccionado y analice si el árbol de decisión mejora, empeora, o mantiene su exactitud.
10. Cree un modelo usando algún clasificador Bayesiano (GaussianNB(), BernoulliNB(), CategoricalNB(), o MultinomialNB())
11. Indique en el notebook el *accuracy* obtenido al usar clasificador Bayesiano
12. Indique en el notebook con cuál técnica, entre redes neuronales, árboles de decisión, y el clasificador Bayesiano seleccionado, se obtiene un mayor *accuracy* para este problema