Universidad de San Carlos de Guatemala

Facultad de ingeniería

Escuela de ciencias y sistemas

Inteligencia Artificial 1

Ing. Luis Fernando Espino Barrios

Aux. Erick Eden Sandoval Ramírez

Manual técnico

José Luis Reynoso Tiu

201345126

“A”

2 de noviembre de 2024, Guatemala

Índice

Tecnologías Utilizadas

- Tytus.js para el entrenamiento y predicciones de los diferentes modelos de ML
- JavaScript para la lógica del proyecto (archivos app.js, kmeans.js, decision_tree.js, regresion_lineal.js).
- HTML y CSS para la interfaz de usuario (archivo index.html).
- Biblioteca vis-network para la visualización de árboles de decisión.
- Chart.js para la representación de gráficas de datos.

Estructura de Archivos

- index.html: Contiene la estructura de la página web, incluyendo elementos de entrada y botones de interacción.
- app.js: Controla la lógica de selección de modelos, carga de archivos CSV y entrenamiento de modelos.
- kmeans.js: Define funciones para el entrenamiento y visualización del modelo K-Means.
- decision_tree.js: Implementa la lógica para la configuración, entrenamiento y visualización del árbol de decisión.
- regresion_lineal.js: Contiene funciones para configurar, entrenar y mostrar gráficas de la regresión lineal.
- vis-network.min.js: Biblioteca externa para visualización de redes y árboles de decisión.

Descripción de los Modelos

Regresión Lineal

- Función de entrenamiento: startRegressionTraining() en regresion_lineal.js procesa datos de dos columnas seleccionadas y ajusta un modelo de regresión lineal.
- Visualización: showGraphLineal() muestra los datos de entrenamiento y la línea de predicción en un gráfico de dispersión.
- Predicción: predictModelLineal() toma un conjunto de valores X y devuelve los valores Y correspondientes.

K-Means

- Configuración: setupKmeansSelectors() permite al usuario seleccionar la columna a analizar y definir los parámetros de los clústeres.
- Entrenamiento: startKmeansTraining() en kmeans.js aplica el algoritmo K-Means y muestra los resultados.
- Visualización: showGraphKmeans() crea un gráfico de dispersión que muestra los clústeres formados.

Árbol de Decisión

- Configuración: setupTreeSelectors() permite la selección de columnas y prepara los datos para el entrenamiento.
- Entrenamiento: startTreeTraining() en decision_tree.js utiliza la clase DecisionTreeID3 para entrenar el modelo.
- Visualización: showGraphTree() muestra la estructura del árbol utilizando vis-network.