**Universidad Tecnológica de La Habana**

**José Antonio Echeverría**



**Práctica Profesional 2**

Creación de paneles Grafana para visualizar modelos de aprendizaje automático

(Árboles de decisión y Reglas de asociación)

**Autor:**

* Daniel Alejandro Deyne Rodríguez **G-42**

**Tutor:**

* Dr. C. Alejandro Rosete Suárez

**Año académico**: 4to

**Carrera**: Ing. Informática

La Habana

2025

RESUMEN

La ciencia de datos y el aprendizaje automático desempeñan un papel fundamental en el análisis y la visualización de datos, permitiendo extraer conocimiento útil a partir de grandes volúmenes de información. En este contexto, los modelos de aprendizaje automático, como los árboles de decisión y las reglas de asociación, son herramientas esenciales para identificar patrones y relaciones significativas.

Se exploró la funcionalidad de la herramienta Grafana y su complemento Business Chart, el cual integra las capacidades de la biblioteca Apache Echarts. Este complemento permite crear visualizaciones dinámicas y personalizables, elementos esenciales para representar los modelos de aprendizaje automático de manera clara y comprensible.

Para lograr el objetivo general del presente trabajo, se diseñó una propuesta para almacenar y persistir los modelos de árboles de decisión y reglas de asociación en PostgreSQL. Una vez almacenados, los modelos se recuperan en Grafana a través de consultas para su posterior visualización. Para adaptar la misma, fue necesario realizar ajustes en el código JavaScript de las visualizaciones que brinda Echarts, modificando primero la entrada de datos para que correspondiera con la consulta realizada, posteriormente se ajustó el código para representar correctamente los modelos.

Se llevaron a cabo varios casos de prueba utilizando grandes conjuntos de datos para validar la propuesta. Los resultados demostraron que la solución que se propone es funcional para el manejo de modelos complejos, proporcionando una visualización clara y flexible que facilita el análisis y la interpretación de los datos.

**Palabras claves**: Ciencia de Datos, Aprendizaje Automático, visualización de modelos, Grafana, casos de prueba.

ABSTRACT

Data science and machine learning play a fundamental role in data analysis and visualization, allowing to extract useful knowledge from large volumes of information. In this context, machine learning models, such as decision trees and association rules, are essential tools to identify significant patterns and relationships.

The functionality of the Grafana tool and its Business Chart plugin, which integrates the capabilities of the Apache Echarts library, was explored. This plugin allows to create dynamic and customizable visualizations, essential elements to represent machine learning models in a clear and understandable way.

To achieve the general objective of this work, a proposal was designed to store and persist decision tree models and association rules in PostgreSQL. Once stored, the models are retrieved in Grafana through queries for later visualization. To adapt it, it was necessary to make adjustments to the JavaScript code of the visualizations provided by Echarts, first modifying the data entry to correspond with the query performed, then adjusting the code to correctly represent the models.

Several test cases were carried out using large data sets to validate the proposal. The results demonstrated that the proposed solution is functional for handling complex models, providing a clear and flexible visualization that facilitates data analysis and interpretation.

**Keywords**: Data Science, Machine Learning, model visualization, Grafana, test cases.

ÍNDICE

[**Introducción** 1](#_Toc188004659)

[**Capítulo 1: Análisis de datos, visualización y modelos de aprendizaje automático** 5](#_Toc188004660)

[1.1 Ciencia de datos 5](#_Toc188004661)

[1.2 Visualización de datos 5](#_Toc188004662)

[1.2.1 Visualización de modelos de aprendizaje automático 10](#_Toc188004663)

[1.2.1.1 Árboles de decisión 12](#_Toc188004664)

[1.2.1.2 Reglas de asociación 16](#_Toc188004665)

[1.2.2 Visualización en Grafana 20](#_Toc188004666)

[1.3 Implementación de modelos y conexión a la base de datos 22](#_Toc188004667)

[1.3.1 Biblioteca *pg8000* 23](#_Toc188004668)

[1.3.2 Biblioteca *scikit-learn* 23](#_Toc188004669)

[1.3.3 Biblioteca *mlxtend* 25](#_Toc188004670)

[1.4 Conclusiones parciales 26](#_Toc188004671)

[**Capítulo 2: Almacenamiento y visualización de los modelos de aprendizaje automático, árboles de decisión y reglas de asociación** 27](#_Toc188004672)

[2.1 Propuesta para almacenar/ persistir el modelo obtenido 27](#_Toc188004673)

[2.1.1 Árboles de decisión 29](#_Toc188004674)

[2.1.2 Reglas de asociación 38](#_Toc188004675)

[2.2 Descripción de la solución 42](#_Toc188004676)

[2.2.1 Diagrama de componentes 42](#_Toc188004677)

[2.2.2 Diagrama de secuencia 44](#_Toc188004678)

[2.3 Visualización de los modelos 47](#_Toc188004679)

[2.3.1 Árboles de decisión 48](#_Toc188004680)

[2.3.2 Reglas de asociación 53](#_Toc188004681)

[2.4 Conclusiones parciales 59](#_Toc188004682)

[**Capítulo 3: Validación** 60](#_Toc188004683)

[3.1 Opciones de personalización para la visualización de modelos de árboles de decisión 60](#_Toc188004684)

[3.2 Opciones de personalización para la visualización de modelos de reglas de asociación 65](#_Toc188004685)

[3.3 Caso de prueba 1 71](#_Toc188004686)

[3.3.1 Preprocesamiento del dataset de encuestas 71](#_Toc188004687)

[3.3.2 Árboles de decisión 74](#_Toc188004688)

[3.3.3 Reglas de asociación 75](#_Toc188004689)

[3.4 Caso de prueba 2 78](#_Toc188004690)

[3.4.1 Preprocesamiento del dataset de comentarios emitidos 78](#_Toc188004691)

[3.4.2 Árboles de decisión 80](#_Toc188004692)

[3.4.3 Reglas de asociación 85](#_Toc188004693)

[3.5 Conclusiones parciales 89](#_Toc188004694)

[**Conclusiones** 90](#_Toc188004695)

[**Recomendaciones** 91](#_Toc188004696)

[**Referencias Bibliográficas** 93](#_Toc188004697)

ÍNDICE DE FIGURAS

[**Figura 1:** Árbol de decisiones para ayudar a la representación visual de datos 10](#_Toc188004708)

[**Figura 2:** Ejemplo de árbol de decisión simple para predecir la mortalidad en el Titanic 14](#_Toc188004709)

[**Figura 3:** Ejemplo de reglas de asociación a través de *arulesVizs* con: (a) un diagrama de dispersión, (b) visualización matricial y (c) visualización basada en grafos 18](#_Toc188004710)

[**Figura 4:** Ejemplo de reglas de asociación basada en matriz con barras 3D 19](#_Toc188004711)

[**Figura 5:** Ejemplos de visualizaciones de árboles 21](#_Toc188004712)

[**Figura 6:** Ejemplo de visualizaciones de grafos 22](#_Toc188004713)

[**Figura 7:** Tabla grafana\_ml\_model\_index para almacenar los modelos 28](#_Toc188004714)

[**Figura 8:** Código de creación de la tabla grafana\_ml\_model\_index 29](#_Toc188004715)

[**Figura 9:** Ejemplo de Modelo de Árbol de decisiones del dataset iris (datos numéricos) 33](#_Toc188004716)

[**Figura 10:** Código de creación de las tablas: grafana\_ml\_model\_arbol\_decision, valores\_prediccion, caracteristicas 33](#_Toc188004717)

[**Figura 11:** Almacenamiento del modelo de Árbol de decisiones con dataset iris en la tabla: grafana\_ml\_model\_arbol\_decision 34](#_Toc188004718)

[**Figura 12:** Almacenamiento del modelo de Árbol de decisiones con dataset iris en la tabla: caracteristicas 34](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004719)

[**Figura 13:** Almacenamiento del modelo de Árbol de decisiones con dataset iris en la tabla: valores\_prediccion 34](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004720)

[**Figura 14:** Ejemplo de Modelo de Árbol de decisiones del dataset de coches (datos no numéricos) 36](#_Toc188004721)

[**Figura 15:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: grafana\_ml\_model\_arbol\_decision 36](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004722)

[**Figura 16:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: caracteristicas 37](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004723)

[**Figura 17:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: valores\_prediccion 37](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004724)

[**Figura 18:** Ejemplo de Modelo de Reglas de Asociación 40](#_Toc188004725)

[**Figura 19:** Código de creación de la tabla grafana\_ml\_model\_reglas\_asociacion 40](#_Toc188004726)

[**Figura 20:** Almacenamiento del modelo de Reglas de asociación con dataset de transacciones de productos en la tabla: grafana\_ml\_model\_reglas\_asociacion 41](#_Toc188004727)

[**Figura 21:** Diagrama de componente 42](#_Toc188004728)

[**Figura 22:** Diagrama de secuencia del proceso: Carga inicial de los datos, creación, entrenamiento y almacenamiento del modelo 45](#_Toc188004729)

[**Figura 23:** Diagrama de secuencia del proceso: Recuperación y visualización del modelo en Grafana 46](#_Toc188004730)

[**Figura 24:** Ejemplo de consulta SQL para obtener modelo de árbol de decisión 48](#_Toc188004731)

[**Figura 25:** Variable en Grafana para mostrar modelos de árbol de decisión disponibles en la base de datos 49](#_Toc188004732)

[**Figura 26:** Código de visualización de un árbol en forma horizontal en Echarts 49](#_Toc188004733)

[**Figura 27:** Código JavaScript modificado para mostrar árbol de decisión 50](#_Toc188004734)

[**Figura 28:** Código JavaScript para obtener lo valores de la consulta SQL a partir de los mismos campos de la base de datos en árbol de decisión 50](#_Toc188004735)

[**Figura 29:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset iris 51](#_Toc188004736)

[**Figura 30:** Ejemplo de cursor sobre un nodo no hoja en árbol de decisón 52](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004737)

[**Figura 31:** Ejemplo de cursor sobre un nodo hoja en árbol de decisón 52](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004738)

[**Figura 32:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset de coches 52](#_Toc188004739)

[**Figura 33:** Ejemplo de consulta SQL para obtener modelo de reglas de asociación 53](#_Toc188004740)

[**Figura 34:** Variable en Grafana para mostrar modelos de reglas de asociación disponibles en la base de datos 54](#_Toc188004741)

[**Figura 35:** Código de visualización de un grafo en Echarts 54](#_Toc188004742)

[**Figura 36:** Código JavaScript modificado para mostrar reglas de asociación 55](#_Toc188004743)

[**Figura 37:** Código JavaScript para obtener lo valores de la consulta SQL a partir de los mismos campos de la base de datos en reglas de asociación 55](#_Toc188004744)

[**Figura 38:** Visualización de modelo de reglas de asociación en forma de grafo modo claro con dataset de transacciones de productos 56](#_Toc188004745)

[**Figura 39:** Ejemplo de cursor sobre una arista con forma de flecha en reglas de asociación 57](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004746)

[**Figura 40:** Ejemplo de cursor sobre una arista con forma de línea discontinua en reglas de asociación 57](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004747)

[**Figura 41:** Ejemplo de cursor sobre un nodo en reglas de asociación 57](#_Toc188004748)

[**Figura 42:** Visualización de modelo de reglas de asociación en forma de grafo modo claro con dataset de transacciones de prendas 58](#_Toc188004749)

[**Figura 43:** Variable en Grafana para mostrar las visualizaciones de modelos de árbol de decisión disponibles 60](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004750)

[**Figura 44:** Variable en Grafana para mostrar los modos de visualización de modelos de árbol de decisión disponibles 60](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004751)

[**Figura 45:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset iris 61](#_Toc188004752)

[**Figura 46:** Visualización de modelo de árbol de decisión en forma circular modo claro con dataset iris 61](#_Toc188004753)

[**Figura 47:** Visualización de modelo de árbol de decisión en forma doble modo claro: Modelo 1 - dataset iris, Modelo 11 - dataset iris (sin sepal width (cm) y petal width (cm)) 62](#_Toc188004754)

[**Figura 48:** Visualización de modelo de árbol de decisión en forma vertical modo oscuro con dataset iris 62](#_Toc188004755)

[**Figura 49:** Variable en Grafana para mostrar los tamaños de las etiquetas de modelos de árbol de decisión disponibles 63](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004756)

[**Figura 50:** Variable en Grafana para mostrar los grosores de aristas de modelos de árbol de decisión disponibles 63](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004757)

[**Figura 51:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas pequeñas con dataset iris 63](#_Toc188004758)

[**Figura 52:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas medianas con dataset iris 64](#_Toc188004759)

[**Figura 53:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas grandes con dataset iris 64](#_Toc188004760)

[**Figura 54:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y aristas gruesas con dataset iris 65](#_Toc188004761)

[**Figura 55:** Variable en Grafana para mostrar las visualizaciones de modelos de reglas de asociación disponibles 65](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004762)

[**Figura 56:** Variable en Grafana para mostrar los modos de visualización de modelos de árbol de decisión disponibles 65](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004763)

[**Figura 57:** Visualización de modelo de reglas de asociación en forma de matriz modo claro con dataset de transacciones de productos 66](#_Toc188004764)

[**Figura 58:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de transacciones de productos 67](#_Toc188004765)

[**Figura 59:** Visualización de modelo de reglas de asociación en forma 3D modo claro con dataset de transacciones de productos 67](#_Toc188004766)

[**Figura 60:** Visualización de modelo de reglas de asociación en forma 3D modo oscuro con dataset de transacciones de productos 68](#_Toc188004767)

[**Figura 61:** Variable en Grafana para mostrar las visualizaciones de grafos de modelos de reglas de asociación disponibles 68](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004768)

[**Figura 62:** Variable en Grafana para mostrar los colores de modelos de reglas de asociación disponibles 68](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004769)

[**Figura 63:** Visualización de modelo de reglas de asociación en forma de grafo (solo relaciones de conjunción) modo claro con dataset de transacciones de productos 69](#_Toc188004770)

[**Figura 64:** Visualización de modelo de reglas de asociación en forma de grafo (solo reglas de asociación) modo claro con dataset de transacciones de productos 69](#_Toc188004771)

[**Figura 65:** Visualización de modelo de reglas de asociación en forma de grafo modo claro (paleta 2) con dataset de transacciones de productos 70](#_Toc188004772)

[**Figura 66:** Visualización de modelo de reglas de asociación en forma de grafo modo claro (paleta 3) con dataset de transacciones de productos 70](#_Toc188004773)

[**Figura 67:** Flujo en Knime para el preprocesamiento del dataset de encuestas 72](#_Toc188004774)

[**Figura 68:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset de encuestas 74](#_Toc188004775)

[**Figura 69:** Visualización de modelo de reglas de asociación en forma de grafo (solo reglas de asociación) modo claro con dataset de encuestas 76](#_Toc188004776)

[**Figura 70:** Regla de asociación asociada a la valoración emitida por los clientes acerca de los servicios de un hotel durante su estadía 77](#_Toc188004777)

[**Figura 71:** Flujo en Knime para el preprocesamiento del dataset de comentarios emitidos 79](#_Toc188004778)

[**Figura 72:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset de comentarios emitidos: val\_h como variable objetivo 81](#_Toc188004779)

[**Figura 73:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset de comentarios emitidos: val\_llm como variable objetivo 83](#_Toc188004780)

[**Figura 74:** Visualización de modelo de árbol de decisión en forma doble modo claro: Modelo 7 - dataset de comentarios emitidos: val\_h como variable objetivo, Modelo 8 - dataset de comentarios emitidos: val\_llm como variable objetivo 84](#_Toc188004781)

[**Figura 75:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de comentarios emitidos (con val\_h y val\_llm) 86](#_Toc188004782)

[**Figura 76:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de comentarios emitidos (sin val\_h y val\_llm) 87](file:///C:\Users\Daniel\Desktop\Práctica%20Profesional%202\Práctica%20Profesional%202.docx#_Toc188004783)

# **Introducción**

La ciencia de datos es un campo que se encuentra en constante crecimiento, el cual ha transformado la manera en la que las empresas y organizaciones toman sus decisiones. El aumento del volumen de datos, ha impulsado esta ciencia en todas las esferas. Los científicos de este sector emplean técnicas avanzadas para el trabajo con los datos y la automatización de los sistemas [1, 2].

Dentro de este campo se encuentra el análisis de datos, el cual se ocupa del trabajo con la estadística, herramienta fundamental para las matemáticas, y el análisis estadístico. Este se ha convertido en uno de los conceptos más actuales y relevantes en el ámbito empresarial, permitiendo a las compañías procesar grandes cantidades de datos para mejorar procesos, conocer mejor a los clientes y tomar decisiones más precisas [3, 4].

El análisis de datos, al revelar patrones y relaciones claves, sirve como base sobre la cual los modelos de aprendizaje automático pueden realizar predicciones basadas en datos históricos, lo que facilita la toma de decisiones proactivas. Estos modelos, como los árboles de decisión o las reglas de asociación, analizan datos de manera eficiente, encontrando relaciones significativas y generando predicciones que pueden optimizar áreas como la gestión de clientes y la logística. Así, las empresas, pueden comprender mejores tendencias, patrones de comportamiento del consumidor y ofrecer experiencias personalizadas, lo cual es crucial en el panorama actual con vistas al futuro [5, 6].

Un componente esencial para complementar el uso de estos modelos es la visualización de datos, ya que facilita la representación gráfica de la información, permitiendo a las empresas y sus equipos no solo interpretar los resultados generados por los modelos de aprendizaje automático, sino también identificar de manera más eficiente tendencias, valores atípicos y patrones claves. Muchos de estos modelos, como los de tipo predictivo, se enfocan en hacer estimaciones sobre futuros eventos basados ​​en datos históricos, y al combinarlos con técnicas de visualización, es posible comunicar hallazgos complejos de manera clara y accesible. Esto facilita que los interesados ​​puedan comprender los patrones descubiertos y actuar con más efectividad [7, 8].

Existen numerosas herramientas tanto gratuitas como de pago para la visualización de datos. Este trabajo se enfoca en la herramienta gratuita Grafana, así como en el empleo de su extensión “Business Chart”, que integra las capacidades de la biblioteca Apache Echarts, la que ofrece una amplia variedad de tipos de gráficos y opciones de visualización. En este sentido, Grafana se presenta como una de las herramientas más completas para proyectos que requieren visualizaciones dinámicas y personalizables, con una gran capacidad de integración y soporte, superando en varios aspectos a otras opciones como Knime.

A pesar de que ECharts permite la creación de visualizaciones basadas en estructuras como árboles, grafos y otras, estas no están diseñadas originalmente para representar de manera directa modelos de aprendizaje automático, lo cual limita su utilidad para este propósito. Teniendo en cuanta la situación problemática mencionada, se define como **problema**: ¿Cómo visualizar modelos de aprendizaje automático en la herramienta Grafana a partir de la extensión “Business Chart”?

Se define como **Objetivo General**: Crear paneles en la herramienta Grafana para la visualización de modelos de aprendizaje automático, específicamente Árboles de decisión y Reglas de asociación.

Determinando como **Objetivos específicos:**

1. Sistematizar los referentes teóricos de la ciencia de datos.
2. Estudiar los modelos de aprendizaje automático.
3. Elaborar una propuesta para el almacenamiento y la visualización de estos modelos, Árboles de decisión y Reglas de asociación.
4. Establecer casos de prueba para cada uno de los modelos.

**Tareas:**

* 1. Sistematizar sobre los referentes teóricos de la ciencia de datos.

2.1 Estudiar los modelos de aprendizaje automático.

2.2 Profundizar en la visualización de los modelos de árboles de decisión y reglas de asociación.

2.3 Caracterizar la herramienta Grafana teniendo en cuenta su extensión “Business Chart” para la visualización de modelos de aprendizaje automático.

2.4 Seleccionar las bibliotecas a emplear para la creación, el entrenamiento y el almacenamiento de los modelos de árboles de decisión y reglas de asociación.

3.1 Elaborar propuesta para el almacenamiento de los modelos de árboles de decisión y reglas de asociación.

3.2 Desarrollar una propuesta para la visualización de los modelos de árboles de decisión y reglas de asociación.

4.1 Validar los casos de prueba para cada uno de los modelos de aprendizaje automático.

El **artefacto** que se obtiene de la investigación, según la definición de Offermann, es [9]:

* Diseño del sistema: descripción del proceso de obtención del modelo de aprendizaje automático y su visualización en Grafana con el uso de diagramas UML.

**Valor Práctico:** Ofrecer una adaptación de las visualizaciones que ofrece Apache Echarts para representar de manera más precisa, modelos de aprendizaje automático en la herramienta Grafana.

El documento de práctica que se presenta está estructurado de la siguiente manera:

El **Capítulo 1** se centra en los fundamentos teóricos relacionados con los modelos de aprendizaje automático y la visualización de datos. Se abordan conceptos claves sobre árboles de decisión y reglas de asociación junto con un análisis sobre la herramienta Grafana y su complemento Business Chart. Posteriormente se exploran aspectos relacionados con la implementación de los modelos y su conexión a la base de datos, identificando las limitaciones y las características más relevantes para su desarrollo. Finalmente, se arriba a conclusiones parciales.

En el **Capítulo 2**, se describe la propuesta para almacenar y visualizar los modelos de aprendizaje automático. Se detalla el diseño de la base de datos en PostgreSQL para almacenar tanto los árboles de decisión como las reglas de asociación. Posteriormente se hace una descripción de la solución a través de diagramas UML. También se describen los pasos a seguir para visualizar los modelos en Grafana. Finalmente, se arriba a conclusiones parciales.

El **Capítulo 3** se centra en la validación de la propuesta planteada en el Capítulo 2. Se describen las opciones de personalización para la visualización de los modelos. Posteriormente, se realizan dos casos de prueba con grandes conjuntos de datos, analizando los resultados obtenidos y evidenciando la funcionalidad de las configuraciones implementadas. Finalmente, se arriba a conclusiones parciales.

# **Capítulo 1: Análisis de datos, visualización y modelos de aprendizaje automático**

## 1.1 Ciencia de datos

La ciencia de datos es una disciplina que se enfoca en el estudio de datos con el fin de extraer información significativa para las empresas. Es un enfoque multidisciplinario que combina principios y prácticas de campos como las matemáticas, la estadística, la inteligencia artificial y la ingeniería de computación para analizar grandes cantidades de datos. Este análisis permite que los científicos de esta rama planteen y respondan preguntas como “qué pasó”, “por qué pasó”, “qué pasará” y “qué se puede hacer con los resultados” [10].

## 1.2 Visualización de datos

La visualización de datos es una técnica que se utiliza para transformar información y datos en representaciones gráficas empleando elementos como tablas, gráficos y mapas. Al emplear estas herramientas visuales, la visualización de datos proporciona un medio accesible para discernir tendencias, identificar valores atípicos y descubrir patrones dentro de conjuntos de datos. Además, desempeña un papel fundamental durante todo el ciclo de vida del análisis de datos dentro de la ciencia de datos, al sintetizar información, interpretar resultados y transmitir conocimientos sobre conjuntos de datos complejos. Este proceso facilita la comprensión al presentar los datos de manera clara y concisa [11-13].

La visualización se divide en 3 categorías [14-16]:

* Análisis exploratorio de datos: Es la búsqueda de patrones y tendencias en un conjunto de datos determinado. La clave aquí es ocultar la complejidad, para facilitar la comprensión de los datos y permitir que el principal enfoque sea en las principales tendencias y patrones.
* Cuadros de calidad de publicación/presentación: Su principal objetivo es facilitar la interpretación de la información por parte del usuario. Deben seguir una serie de características claves como: la claridad, precisión, estética y el contexto, con el fin de mejorar la comprensión de los datos [17].
* Visualización interactiva para aplicaciones externas: Crear paneles (en inglés se usa el término “dashboards”) que faciliten la interacción del usuario con conjuntos de datos propietarios es una tarea típica de los ingenieros de software orientados a la ciencia de datos. La misión típica aquí es crear herramientas que respalden el análisis exploratorio de datos para personal menos capacitado técnicamente y más orientado a las aplicaciones [18].

Dentro de sus principios se encuentra [14]:

* Maximizar la relación datos-tinta: En cualquier gráfico, parte de la tinta se utiliza para representar los datos subyacentes reales, mientras que el resto se emplea en efectos gráficos. En términos generales, las visualizaciones deben centrarse en mostrar los datos en sí. Esta relación datos-tinta, la define, el estadístico Edward Tufte, como:

Tinta de datos

Relación datos-tinta =

Tinta total utilizada en gráficos

* Minimizar el factor mentira: Una visualización busca contar una historia real sobre lo que dicen los datos. La forma más insolente de mentir es manipular los datos, pero sigue siendo muy posible informarlos con precisión y, al mismo tiempo, engañar deliberadamente a la audiencia sobre lo que dicen. Tufte, define el factor de mentira de un gráfico como:

(tamaño de un efecto en el gráfico)

Factor de mentira =

(tamaño del efecto en los datos)

Cabe destacar que la fórmula definida por Tufte [14], no es posible calcularla mecánicamente, pues requiere comprender quien está detrás de la creación de ese gráfico y con qué fin lo hizo.

* Minimizar basura gráfica: La eliminación de elementos en un gráfico que no sean esenciales para la representación de datos, evita criterios engañosos o modelos innecesarios, priorizando la información relevante sobre la decoración o elementos superfluos en los gráficos.
* Escalado y etiquetado adecuados: Las deficiencias en la escala y el etiquetado son la principal fuente de información errónea intencional o accidental en los gráficos. Las etiquetas deben informar la magnitud adecuada de los números, y la escala debe mostrar estos números con la resolución correcta, de manera que facilite la comparación.
* Uso eficaz del color y el sombreado: Los colores se asumen cada vez más como parte de cualquier comunicación gráfica. Estos desempeñan dos funciones principales en los gráficos: marcar distinciones de clases y codificar valores numéricos.
* El poder de la repetición: Los conjuntos de gráficos similares con elementos de datos diferentes pero relacionados, proporcionan una manera concisa y poderosa de permitir comparaciones visuales, reforzando patrones, tendencias o mensajes claves a través de la repetición de elementos visuales como colores, formas, tamaños, o estilos.

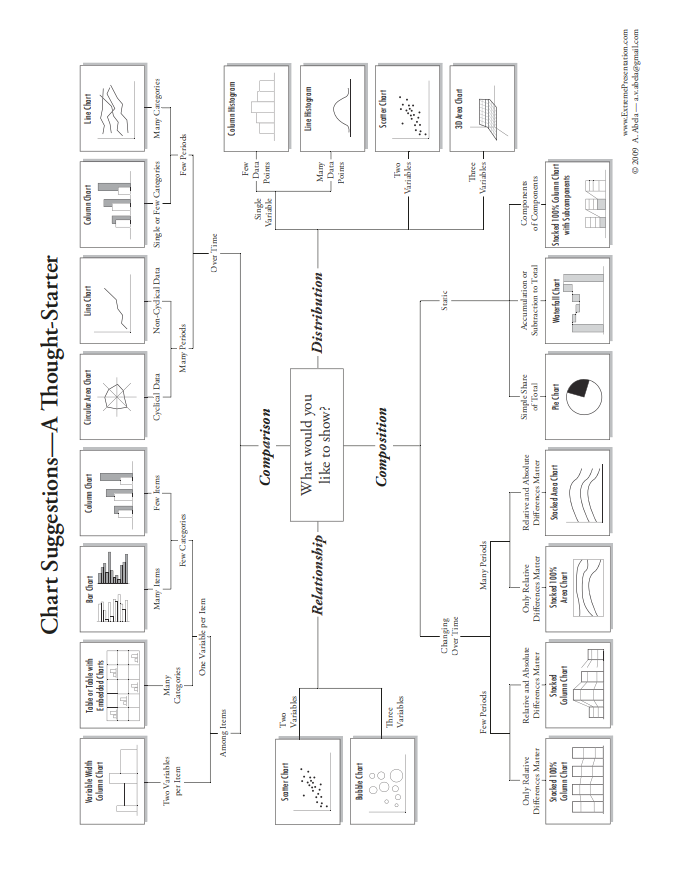
Existen diversos tipos de visualizaciones, cada uno con sus propias fortalezas y debilidades. Elegir la visualización adecuada no solo mejora la comprensión de los datos, sino que también facilita la toma de decisiones al resaltar patrones y tendencias importantes (ver **Figura.1**).

* Tipos de visualización de datos [14]:
* Cuadros
* Tablas
* Gráficos
* Mapas
* Infografía
* Paneles
* Ejemplos específicos de métodos para visualizar datos [14]:
* Gráfico de área
* Gráfico de barras
* Diagramas de caja y bigotes
* Nube de burbujas
* Gráfico de bala
* Cartograma
* Vista circular
* Mapa de distribución de puntos
* Gráfico de Gantt
* Mapa de calor
* Tabla de resaltado
* Histograma
* Matriz
* Red
* Área polar
* Árbol radial
* Diagrama de dispersión (2D o 3D)
* Gráfico de flujo
* Tablas de texto
* Escala de tiempo
* Diagrama de árbol
* Gráfico circular apilado
* Nube de palabras
* Combinaciones presentes en paneles

Según Skiena, para identificar el gráfico adecuado se debe (ver **Figura.1**) [14]:

1. Identificar el propósito: Antes de elegir un gráfico, se debe comprender el propósito de la visualización a mostrar. ¿Qué se quiere comunicar? ¿Se está mostrando una distribución, comparando valores o explorando relaciones?
2. Tipo de datos: ¿Qué tipo de datos se está manejando?

* Datos temporales: Si se está analizando cambios a lo largo del tiempo, se podrían usar gráfico de línea y gráfico de barras.
* Datos categóricos: Si se está comparando categorías, se recomienda gráfico de barras agrupadas y gráfico de pastel.
* Datos numéricos continuos: Si se quiere visualizar distribuciones, se puede emplear histogramas y diagramas de cajas.
* Relaciones y correlaciones: Si se está interesados en relaciones entre variables, se pudieran utilizar gráficos de dispersión (en inglés se usa el término “scatter plot”) y gráfico de burbujas.
* Datos geográficos: Si se quiere trabajar con datos geográficos, se emplean mapas de flujo y diagramas de red.



**Figura 1:** Árbol de decisiones inteligente para ayudar a identificar la mejor representación visual para representar datos. Foto tomada del libro “THE Data Science Design MANUAL” de Steven S. Skiena [14]

### 1.2.1 Visualización de modelos de aprendizaje automático

En la actualidad, los modelos de aprendizaje automático juegan un papel importante dentro de la ciencia de datos, debido tanto al incremento de los datos como del crecimiento computacional. Estos modelos funcionan mediante algoritmos que analizan datos históricos para identificar patrones y hacer predicciones. Dependiendo de su objetivo, pueden clasificarse en varias categorías, como los modelos de clasificación, los cuales se emplean para categorizar datos en grupos específicos, o los modelos de regresión, que estiman valores continuos [14].

La visualización en el aprendizaje automático (en inglés se usa el término “machine learning”) generalmente se refiere al proceso de representar modelos de aprendizaje automático, datos y sus relaciones a través de medios gráficos o interactivos. El objetivo es facilitar la comprensión de algoritmos complejos y los patrones de datos de un modelo, haciéndolo más accesible para las partes interesadas técnicas y no técnicas [19].

La visualización de modelos ML puede ayudar con una amplia gama de objetivos [19]:

* Visualización de la estructura del modelo: Los tipos de modelos comunes, como los árboles de decisión, las máquinas de vectores de soporte o las redes neuronales profundas, suelen constar de muchas capas de cálculos e interacciones que resultan difíciles de comprender para los humanos. La visualización nos permite ver con mayor facilidad cómo fluyen los datos a través de un modelo y dónde se producen las transformaciones.
* Visualización de métricas de rendimiento: Una vez entrenado un modelo, se debe evaluar su rendimiento. Visualizar métricas como la exactitud, la precisión, la recuperación y la puntuación F1 ayuda a ver qué tan bien está funcionando el modelo y dónde se necesitan mejoras.
* Análisis de modelos comparativos: Cuando se trabaja con múltiples modelos o algoritmos, la visualización de las diferencias en la estructura o el rendimiento permite elegir el mejor para una tarea particular.
* Importancia de las características: Es fundamental comprender qué características influyen más en las predicciones de un modelo. Las técnicas de visualización, como los gráficos de importancia de las características, facilitan la identificación de los factores críticos que impulsan los resultados del modelo.
* Interpretabilidad: Debido a su complejidad, los modelos de ML suelen ser “cajas negras” para sus creadores humanos, lo que dificulta la explicación de sus decisiones. Las visualizaciones pueden arrojar luz sobre cómo las características específicas afectan el resultado o qué tan sólidas son las predicciones de un modelo.
* Comunicación: Las visualizaciones son un lenguaje universal para transmitir ideas complejas de forma sencilla e intuitiva. Son esenciales para compartir información de manera eficaz con la dirección y otras partes interesadas no técnicas.

#### 1.2.1.1 Árboles de decisión

Un árbol de decisión es un modelo predictivo utilizado en áreas como la inteligencia artificial, aprendizaje automático, y toma de decisiones. Este se basa en un conjunto de decisiones estructuradas de manera jerárquica, donde cada nodo interno representa una "prueba" o condición sobre un atributo, cada rama representa un resultado de la prueba, y cada hoja final representa una clase o valor de predicción [14, 20, 21].

Funcionamiento:

* Selección de un atributo: En el nodo raíz, se elige un atributo que divida mejor los datos según algún criterio (como la ganancia de información o el índice Gini).
* El criterio de ganancia de información se utiliza en la construcción de árboles de decisión para seleccionar el mejor atributo en cada nodo. Este consiste en medir cuánto reducir la incertidumbre (entropía) al dividir un conjunto de datos en subconjuntos basados ​​en dicho atributo [20].

Funcionamiento:

1. **Entropía:** Mide la "impureza" o el desorden de un conjunto de datos.
2. **Ganancia de información:** Se calcula como la diferencia entre la entropía original del conjunto y la entropía promedio de los subconjuntos tras una división.

El atributo que maximiza la ganancia de información es el que mejor separa los datos.

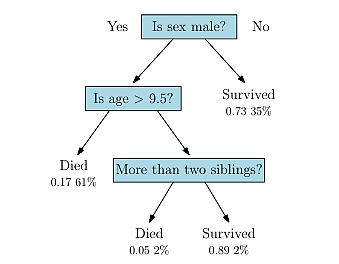
* El índice de Gini es un criterio utilizado en los árboles de decisión, específicamente en algoritmos como CART, para medir la pureza de un conjunto de datos. Este representa la probabilidad de que una instancia sea clasificada incorrectamente si se elige aleatoriamente según la distribución de las clases en dicho conjunto [14].

Funcionamiento:

1. **Valor de Gini:** Va de 0 a 1, donde 0 significa pureza total (todas las instancias pertenecen a una sola clase), y 1 indica máxima impureza.
2. **División:** Al seleccionar un atributo, se calcula el índice de Gini de los subconjuntos generados, buscando minimizar la impureza.

* Generación de nodos: Para cada posible valor del atributo seleccionado, se crean ramas que representan los posibles resultados de la "prueba". Cada rama lleva a un nuevo nodo, donde se repite el proceso con un nuevo atributo.
* Criterios de parada: El proceso continúa hasta que no se pueden realizar más divisiones útiles (por ejemplo, todos los datos en un nodo pertenecen a la misma clase o se alcanza un límite predefinido de profundidad). Cabe destacar que las hojas del árbol representan las predicciones finales.
* Predicción: Para predecir, se comienza en el nodo raíz y se sigue el camino en el árbol, dependiendo de los valores de los atributos de la nueva muestra, hasta llegar a una hoja que representa la clase o valor predicho.

La visualización de árboles de decisión implica una representación gráfica de su estructura general, mostrando las divisiones y las decisiones en cada nodo de forma clara e intuitiva. La profundidad y el ancho del árbol, así como los nodos de las hojas, se hacen evidentes a primera vista. Además, la visualización de árboles de decisión ayuda a identificar características cruciales, los atributos más discriminantes que conducen a predicciones precisas [19].



**Figura 2:** Ejemplo de árbol de decisión simple para predecir la mortalidad en el Titanic. Foto tomada del libro “THE Data Science Design MANUAL” de Steven S. Skiena [14]

En la **Figura.2** se muestra un árbol de decisión simple, el cual tiene como objetivo predecir la probabilidad de supervivencia de los pasajeros en el Titanic [14]. Este árbol refleja cómo se toman decisiones jerárquicas basadas en atributos específicos, que son evaluados secuencialmente en los nodos del árbol.

Explicación:

1. Raíz del árbol

El árbol comienza con el nodo raíz, donde se evalúa la pregunta "¿El sexo es masculino? “.

* Si la respuesta es **No**, el modelo predice directamente que el pasajero sobrevive con una probabilidad de 0,73 (73%) y un 35% del total de casos que se encuentran en esta categoría.
* Si la respuesta es **Sí**, pasa al siguiente nodo para evaluar otra característica.

1. Segunda división (profundidad 1):

Para los pasajeros hombres (**Sí** en el nodo raíz), el siguiente criterio es la edad del pasajero, expresado como "¿Es la edad mayor a 9.5?".

* Si la respuesta es **Sí**, el modelo predice que estos pasajeros fallecen con una probabilidad de 0.17 (17%), representando el 61% del total de casos.
* Si la respuesta es **No**, pase al siguiente nodo para evaluar una nueva característica.

1. Tercera división (profundidad 2):

En este nivel, para hombres menores de 9.5 años, se evalúa la pregunta "¿Más de dos hermanos?”.

* Si la respuesta es **Sí**, el modelo predice que los pasajeros fallecen con una probabilidad de 0.05 (5%), representando solo el 2% del total de casos.
* Si la respuesta es **No**, se predice que los pasajeros sobreviven con una probabilidad de 0.89 (89%), representando también el 2% del total de casos.

El camino hacia una predicción precisa se puede resumir en cuatro pasos [19]:

* Claridad de las características: La visualización de árboles de decisión es similar a mirar un diagrama de flujo de toma de decisiones, donde cada rama representa una característica y cada nodo de decisión contiene un aspecto crucial de los datos.
* Atributos discriminatorios: La belleza de una visualización de árbol de decisiones reside en su capacidad de destacar las características más discriminatorias. Estos factores influyen en gran medida en el resultado y guían al modelo en la realización de predicciones. Al visualizar el árbol, se pueden identificar estas características y, de esta manera, comprender los factores centrales que impulsan las decisiones del modelo.
* Camino hacia la precisión: Cada camino que recorre el árbol de decisiones es un viaje hacia la precisión. La visualización muestra la secuencia de decisiones que conducen a una predicción particular. Esto es de gran ayuda para comprender la lógica y los criterios que utiliza el modelo para llegar a conclusiones específicas.
* Simplicidad en medio de la complejidad: A pesar de la complejidad de los algoritmos de aprendizaje automático, la visualización de árboles de decisión tiene un elemento de simplicidad. Transforma cálculos matemáticos complejos en una representación intuitiva, lo que la hace accesible para las partes interesadas técnicas y no técnicas.

#### 1.2.1.2 Reglas de asociación

Las reglas de asociación son técnicas de minería de datos que permiten identificar relaciones o patrones frecuentes entre los elementos de un conjunto de datos, especialmente en contextos de transacciones. Su objetivo es descubrir cómo ciertos artículos o eventos están relacionados entre sí, es decir qué elementos tienden a aparecer juntos en un conjunto de transacciones, lo cual es útil en áreas como marketing, ventas minoristas y recomendaciones personalizadas [22, 23].

Funcionamiento:

1. **Identificar patrones frecuentes:** El primer paso es identificar qué combinaciones de artículos o eventos aparecen más comunes en las transacciones. Para ello:

* Se establece un umbral mínimo de *soporte*.

El *soporte* se calcula como la proporción de transacciones que contiene el conjunto de artículos respecto al total de transacciones (cabe destacar que este proceso se realiza por cada conjunto de artículos).

* Luego, solo los conjuntos que superen este valor se tienen en cuenta para la generación de reglas.

1. **Generación de reglas:** A partir de los conjuntos frecuentes de artículos o eventos (los que superaron el umbral mínimo de soporte), se generan reglas que describen las relaciones entre ellos. Cada regla se expresa en la forma: {A} ⇒ {B}, que significa que dada la condición A es probable que también ocurra B.
2. **Evaluación de reglas:** Las reglas generadas se evalúan utilizando medidas como:

* Soporte: Mide la frecuencia con la que aparece una regla en el conjunto de datos.

Se calcula como la proporción de transacciones que contienen ambos ítems (antecedente y consecuente) en relación con el total de transacciones [24].

***Fórmula:*** *supp(X⇒Y) = Número de transacciones (X U Y) /*

*Total de transacciones*

* Confianza: Mide la probabilidad en cuanto a la cantidad de veces de que una regla dada resulta ser verdadera.

Se calcula como la proporción de transacciones que contienen ambos ítems (antecedente y consecuente) en relación con el número de transacciones del ítem antecedente [24].

***Fórmula:*** *conf(X ⇒Y) = supp(X U Y)/ supp(X)*

* Ascenso (en inglés se usa el término “Lift”): No es tan básico como el soporte y la confianza, pero se utiliza para evaluar la independencia entre los artículos. Un valor de lift mayor que 1 indica que los ítems están positivamente asociados, es decir, ocurren juntos más de lo esperado, mientras que un lift igual a 1 indica que no hay asociación., es decir son independientes y un lift menor que 1 indica que los ítems están negativamente relacionados; la presencia de uno disminuye la probabilidad de que ocurra el otro.

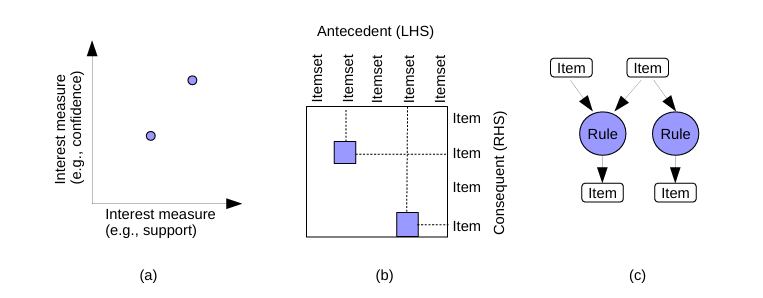
Se calcula como la proporción del soporte de ambos ítems (antecedente y consecuente) en relación a la confianza del ítem consecuente [24].

***Fórmula:*** *lift(X ⇒ Y) =supp(X U Y)/ (supp(X) supp(Y))*

*supp(X)*

1. **Interpretación y acciones:** Una vez culminado el proceso de evaluación, se procede a interpretar cada una de las reglas, lo cual permite tomar decisiones o acciones según el problema que se esté tratando.

La visualización de reglas de asociación implica una representación gráfica de las relaciones entre conjuntos de datos, mostrando de manera clara los patrones significativos en las transacciones. A través de gráficos, como matrices de dispersión y diagramas de barras en 3D (ver **Figura.3, Figura.4**), se pueden identificar rápidamente asociaciones relevantes, soporte y confianza en las reglas. Estos gráficos no solo presentan de forma accesible la frecuencia y relación entre los ítems, sino que también destacan las reglas más fuertes en función de la medida de interés, como por ejemplo el "lift" (ver **Figura.4**) [24, 25].



**Figura 3:** Ejemplo de reglas de asociación a través de *arulesVizs* con: (a) un diagrama de dispersión, (b) visualización matricial y (c) visualización basada en grafos. Foto tomada del artículo “arulesViz: Interactive Visualization of Association Rules” de Michael Hahsler [24]

En la **Figura.3** se presentan tres formas diferentes de visualizar reglas de asociación a través de la herramienta arulesViz, cada una destacando distintos aspectos de las relaciones entre los ítems [24]:

* Diagrama de dispersión (a):

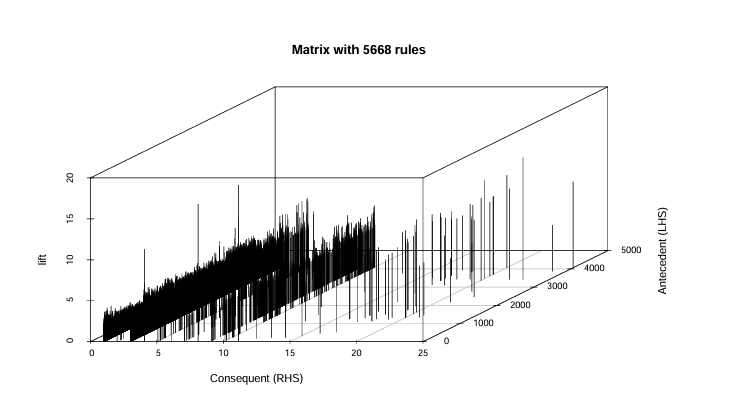
Este gráfico utiliza un plano cartesiano para representar las reglas de asociación, donde los ejes corresponden a métricas clave como el soporte (en inglés se usa el término “support”) y la confianza (en inglés se usa el término “confidence”). Cada punto en el diagrama representa una regla específica, lo cual permite identificar visualmente reglas más fuertes o significativas en función de estas métricas.

* Visualización matricial (b):

En esta representación, se utiliza una matriz para mostrar las relaciones entre ítems. El eje horizontal (LHS) corresponde a los antecedentes de la regla, mientras que el eje vertical (RHS) corresponde a los consecuentes. Las celdas en la matriz indican la relación entre los ítems, y su tamaño o color puede reflejar medidas de interés, como el soporte o la confianza.

* Visualización basada en grafos (c):

Este grafo representa las reglas como nodos y aristas en un diagrama de red. Los ítems se visualizan como nodos individuales, mientras que las reglas de asociación se representan como conexiones (aristas) entre los nodos. Los nodos centrales (Rule) corresponden a las reglas, y los nodos externos a los ítems que forman parte de los antecedentes o consecuentes.



**Figura 4:** Ejemplo de reglas de asociación basada en matriz con barras 3D. Foto tomada del artículo “Visualizing Association Rules in Hierarchical Groups” de Michael Hahsler, Sudheer Chelluboina [25]

En la **Figura.4** se muestra una representación tridimensional de reglas de asociación utilizando una matriz de barras en 3D [25]. Esta visualización permite analizar un gran número de reglas (5,668 en este caso), donde los ejes representan los diferentes elementos involucrados en las reglas:

* El eje X corresponde a los consecuentes (RHS) de las reglas.
* El eje Y representa los antecedentes (LHS).
* El eje Z muestra el lift como métrica.

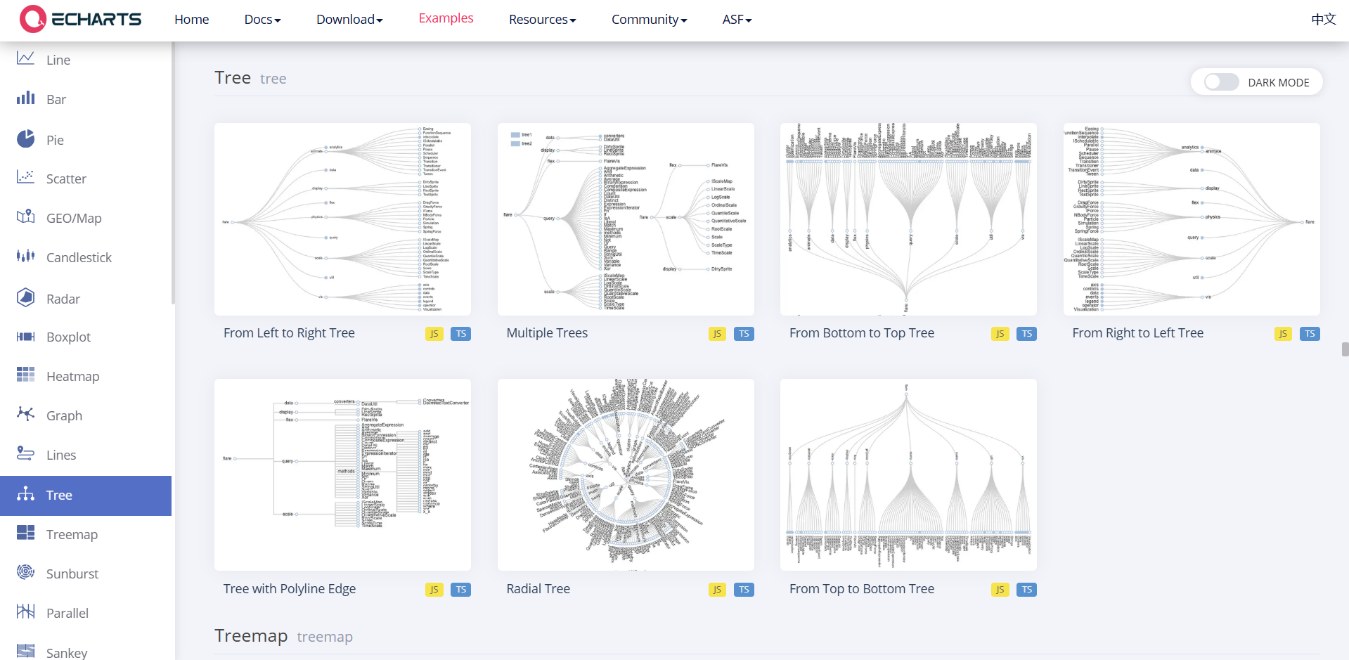
Componentes claves en la visualización de reglas de asociación [25]:

* Claridad de asociaciones: Similar a un diagrama de flujo, las visualizaciones en arulesViz (paquete en R diseñado para la visualización de reglas de asociación) permiten observar cómo se relacionan los ítems. Cada punto en un gráfico de dispersión o cada celda en una matriz representa una regla, ayudando a identificar patrones de compra o correlaciones fuertes entre productos.
* Medidas de interés: Atributos como el soporte, la confianza y el "lift" se representan visualmente para resaltar las asociaciones más significativas. La coloración y el tamaño en estos gráficos facilitan la diferenciación de reglas fuertes y débiles, lo cual es crucial para los análisis de marketing y comportamiento del consumidor.
* Camino hacia patrones clave: La visualización muestra de manera intuitiva cómo ciertas combinaciones de ítems se destacan en términos de relevancia, lo cual es esencial para el análisis de tendencias y predicciones de comportamiento. Por ejemplo, un gráfico de dispersión revela asociaciones populares que pueden sugerir recomendaciones personalizadas.
* Simplicidad en la exploración de grandes volúmenes de datos: A pesar de la complejidad inherente a los grandes conjuntos de datos transaccionales, la visualización de reglas de asociación simplifica este análisis mediante gráficos intuitivos que se adaptan a necesidades de exploración visual.

### 1.2.2 Visualización en Grafana

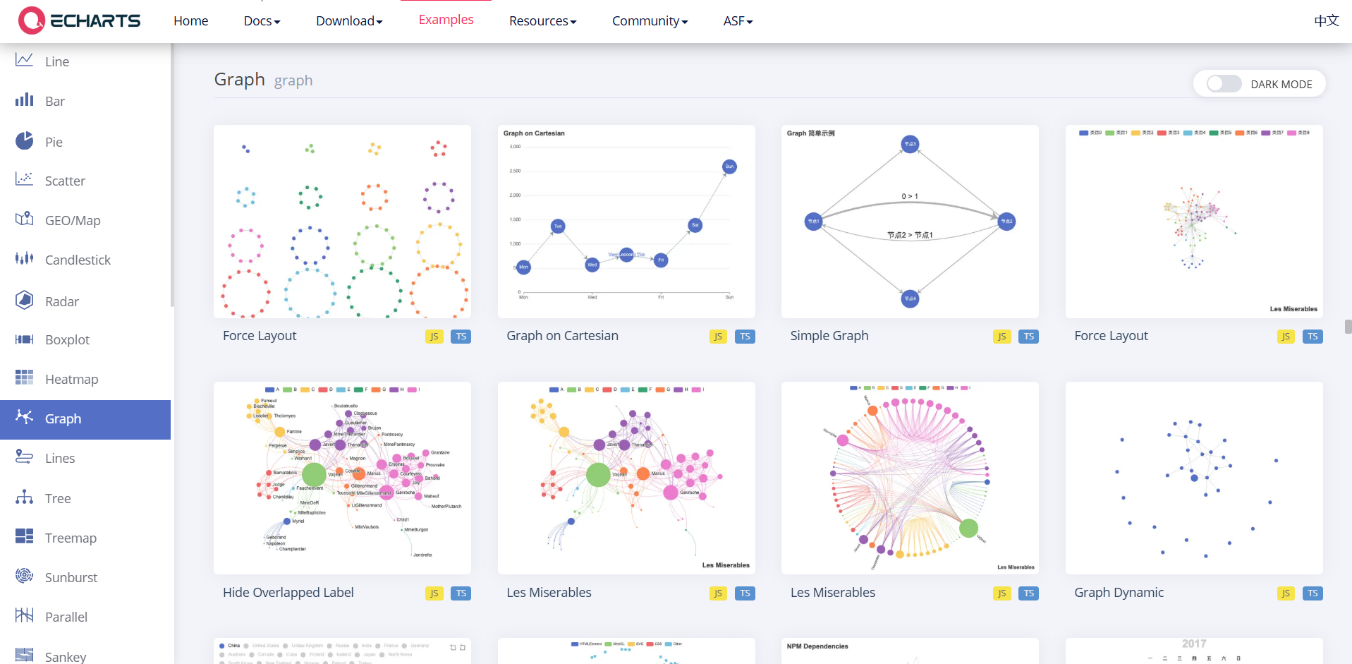
Grafana es una herramienta de visualización ampliamente utilizada para monitorizar datos en tiempo real, ofreciendo paneles interactivos y gráficos que permiten comprender mejor el comportamiento de sistemas y métricas. Sin embargo, en su forma base, Grafana está más orientada a la supervisión de infraestructura y métricas de rendimiento que a la visualización de modelos de aprendizaje automático o análisis de datos complejos [26].

Con el fin de mejorar sus capacidades de visualización, Grafana emplea extensiones, entre las que se encuentra "Business Chart", la cual aprovecha la flexibilidad de la biblioteca Apache Echarts. Apache Echarts es una biblioteca gráfica en JavaScript conocida por su versatilidad y por ofrecer una amplia variedad de tipos de gráficos y opciones interactivas, que incluyen gráficos de dispersión, árboles (ver **Figura.5**), grafos (ver **Figura.6**), diagramas de flujo, gráficos de líneas, mapas geográficos, entre otros. Esta extensión permite a los usuarios personalizar sus paneles en Grafana, ofreciendo opciones avanzadas para representar datos de manera más detallada y visualmente atractiva [27].



**Figura 5:** Ejemplos de visualizaciones de árboles. Foto tomada por el autor del sitio de ejemplos de Apache Echarts [28].

A través de "Business Chart" en Grafana, es posible adaptar visualizaciones complejas que pueden ayudar en el análisis de datos y, en ciertos casos, en la representación gráfica de datos relacionados con modelos de aprendizaje automático, como la visualización de predicciones o tendencias de datos. No obstante, a pesar de estas mejoras, aún existen limitaciones en cuanto a visualizaciones especializadas para modelos predictivos, como la importancia de características (en inglés se usa el término “feature import”), métricas de precisión o visualizaciones interactivas de la estructura de un modelo, aspectos que son fundamentales en la ciencia de datos avanzada [27].



**Figura 6:** Ejemplo de visualizaciones de grafos. Foto tomada por el autor del sitio de ejemplos de Apache Echarts [28].

## 1.3 Implementación de modelos y conexión a la base de datos

La implementación de modelos de aprendizaje automático y su integración con bases de datos relacionales en entornos de producción requieren tanto herramientas de entrenamiento eficientes como bibliotecas que faciliten la conexión estable y rápida con bases de datos. En el contexto de aplicaciones de ciencia de datos en Python, varias bibliotecas se han destacado en la comunidad por su rendimiento y simplicidad, tanto para la gestión de bases de datos como para el desarrollo de modelos [29, 30].

### 1.3.1 Biblioteca *pg8000*

Para interactuar con bases de datos PostgreSQL en Python, existen varias opciones ampliamente utilizadas, tales como *psycopg2* [31], *asyncpg* [32], *pg8000* [33], *sqlAlchemy* [34]. Cada una de estas bibliotecas ofrece características específicas para la gestión de conexiones y operaciones de lectura/escritura en PostgreSQL, un sistema de base de datos relacional reconocido por su robustez y capacidad de manejo de grandes volúmenes de datos [35].

Entre estas opciones, pg8000 sobresale como una alternativa eficiente y versátil, especialmente en escenarios donde se busca simplicidad de instalación y compatibilidad con entornos de Python puro. Dentro de sus ventajas destacan [33, 35]:

* A diferencia de psycopg2, que requiere compilaciones en C y puede implicar complicaciones en cuanto a dependencias, pg8000 está escrito completamente en Python, lo que evita problemas de compatibilidad y facilita su implementación en sistemas diversos.
* Cumple con la especificación DB-API 2.0, un estándar que garantiza la interoperabilidad con otros módulos y herramientas en Python.
* Es compatible con las versiones modernas de Python, lo cual garantiza su funcionamiento a largo plazo.

### 1.3.2 Biblioteca *scikit-learn*

Scikit-learn se ha consolidado como una de las bibliotecas de aprendizaje automático más populares en Python, en gran medida por su enfoque intuitivo y su flexibilidad. En el caso de los árboles de decisión, scikit-learn implementa el algoritmo CART (Classification and Regression Trees), una técnica ampliamente utilizada que optimiza la pureza de los nodos hijos en cada división, permitiendo clasificar o predecir con alta precisión a partir de criterios de división como la entropía o el índice de Gini. Además, permite ajustar parámetros clave, como la profundidad máxima del árbol, el número mínimo de muestras por nodo y otros, los cuales permiten adaptar el modelo a las necesidades específicas de los datos y mejorar su capacidad de generalización [36].

Ventajas de *scikit-learn* para Árboles de Decisión [36]:

* **Facilidad de uso y flexibilidad:** Es conocida por su diseño intuitivo, que sigue una estructura uniforme para todos los modelos. Su clase DecisionTreeClassifier es fácil de ajustar, permitiendo personalizar parámetros como la profundidad máxima del árbol y el criterio de división (Gini o entropía) sin requerir configuraciones complejas.
* **Optimización y eficiencia en cálculo:** La implementación de árboles de decisión en scikit-learn es rápida y bien optimizada, gracias al uso de Cython y otras técnicas de optimización de bajo nivel, lo cual es ideal para datos medianos y grandes. Aunque no cuenta con algoritmos de boosting nativos como XGBoost o LightGBM, el algoritmo CART de scikit-learn ofrece un equilibrio entre eficiencia y rendimiento, especialmente para modelos que no requieren boosting avanzado.
* **Compatibilidad y soporte amplio en la comunidad:** Es una de las bibliotecas más respaldadas por la comunidad de aprendizaje automático y ciencia de datos en Python. Cuenta con abundante documentación, tutoriales, y una comunidad activa que facilita el soporte.
* **Interoperabilidad con herramientas de evaluación y visualización:** Proporciona herramientas de evaluación y visualización, como plot\_tree, que facilitan el análisis y la interpretación de modelos de árboles de decisión. Esta capacidad es fundamental para aplicaciones que requieren explicabilidad y transparencia, y está menos desarrollada en otras bibliotecas enfocadas en boosting, como XGBoost o LightGBM.

### 1.3.3 Biblioteca *mlxtend*

Para la generación de reglas de asociación, mlxtend es una biblioteca que se destaca por su enfoque en la simplicidad de uso y la capacidad de personalización. Esta biblioteca incluye algoritmos como Apriori y FP-Growth, que permiten identificar patrones en grandes volúmenes de datos y encontrar asociaciones significativas entre ellos. La posibilidad de establecer umbrales de soporte y confianza ajustables facilita el ajuste de las reglas generadas según los requisitos específicos de cada análisis. Además, mlxtend proporciona métricas clave, como lift y confianza, que ayudan a evaluar la solidez de las reglas descubiertas y son esenciales para determinar la importancia de cada asociación encontrada [37].

Ventajas de *mlxtend* para Reglas de Asociación [38, 39]:

* **Interfaz intuitiva y configuración flexible:** Ofrece una implementación del algoritmo Apriori que permite ajustar configuraciones clave como el soporte y la confianza de las reglas de asociación de manera sencilla y directa. Esto la convierte en una opción más intuitiva que alternativas como apyori o efficient-apriori, que son más limitadas en cuanto a la personalización y configuración de los modelos.
* **Métricas integradas para evaluación de reglas:** Incluye métricas como lift, confianza, y soporte, que son esenciales para evaluar la calidad de las reglas generadas. Estas métricas no están presentes en bibliotecas más ligeras como apyori y son importantes debido a que permiten realizar un análisis detallado de la utilidad y relevancia de las reglas de asociación obtenidas.
* **Compatibilidad con el ecosistema de scikit-learn:** Se integra bien con el ecosistema de scikit-learn y otras herramientas de Python, lo cual permite una fluidez en el flujo de trabajo. A diferencia de SPMF, que es una herramienta robusta, pero requiere la programación en Java, mlxtend se adapta sin problemas a entornos de Python y permite una interacción sencilla con otros modelos o transformaciones en el mismo flujo de trabajo.
* **Documentación y soporte de la comunidad:** La documentación de mlxtend está bien desarrollada, lo cual facilita la implementación de reglas de asociación y el acceso a ejemplos prácticos.

## 1.4 Conclusiones parciales

* De este estudio, se detectó una limitación de la biblioteca scikit-learn, que radica en su implementación de árboles de decisión exclusivamente como árboles binarios. Esta restricción limita el trabajo con estructuras de árboles generales, ya que en un árbol binario cada nodo se divide en solo dos ramas (izquierda y derecha), mientras que en un árbol general los nodos pueden tener múltiples hijos.
* Se identificó que la extensión "Business Chart" de Grafana, basada en la biblioteca Apache Echarts, permite ampliar las opciones de visualización, integrando gráficos avanzados que son útiles en muchos contextos. Sin embargo, aunque Echarts ofrece versatilidad en tipos de gráficos, su integración actual en Grafana aún no dispone de visualizaciones para representar de manera directa modelos de aprendizaje automático, lo cual limita su aplicabilidad.
* Se identificó que los modelos de reglas de asociación pueden representarse mediante diversos enfoques visuales (bidimensional, tridimensional, entre otros), cada uno ajustado a los objetivos específicos del análisis. Esta diversidad de enfoques permite destacar diferentes aspectos clave, como la estructura, la complejidad o las métricas asociadas a las reglas.

# **Capítulo 2: Almacenamiento y visualización de los modelos de aprendizaje automático, árboles de decisión y reglas de asociación**

Al diseñar una solución para el almacenamiento y visualización de modelos de aprendizaje automático (árboles de decisión y reglas de asociación), fue necesario evaluar diferentes alternativas que permitieran garantizar una correcta integración entre los modelos generados y la herramienta Grafana. Una de las opciones que se tuvo en cuenta fue la exportación de los modelos en formatos como CSV, JSON o XML procedentes de otras herramientas como por ejemplo Knime [40], que podrían integrarse en Grafana mediante complementos como Infinity [41]. Sin embargo, este enfoque resulta limitado para escenarios que manejan grandes volúmenes de datos o modelos dinámicos. Otra alternativa analizada fue el uso de APIs REST [42], lo que permitiría consumir los modelos en tiempo real, aunque esto implicaría un mayor esfuerzo en el desarrollo y mantenimiento. Finalmente, se optó por una solución más robusta y escalable, basada en la integración de Python con un gestor de bases de datos relacionales como PostgreSQL, lo cual permitió estructurar y persistir los modelos generados de forma adecuada y optimizada. Esta propuesta, que se detalla en el epígrafe 2.1, está diseñada para garantizar la correcta organización de los modelos y facilitar su consulta y visualización en la herramienta Grafana.

## 2.1 Propuesta para almacenar/ persistir el modelo obtenido

Para almacenar el modelo obtenido (Árbol de decisión o reglas de asociación) se llevará a cabo lo siguiente:

1. Conectar el entorno Python de desarrollo con la base de datos, en este caso PostgreSQL, a través de la biblioteca *pg8000* (v1.31.2).
2. Crear una tabla en la base de datos para almacenar todos los modelos (ver **Figura.7**).

A continuación, se describe la misma:

* Las filas serán cada uno de los modelos almacenados con su información correspondiente.
* Las columnas son:
* **id**: Es el identificador único de cada modelo.

*Función:* Es clave primaria y permite referenciar cada modelo de forma

individual.

* **nombre**: Es el nombre de cada una de los modelos.

*Función:* Permite guardar una sola vez el nombre del modelo y de esta

manera se asegura la duplicidad de los modelos.

* **descripcion**: Es la descripción de cada modelo.

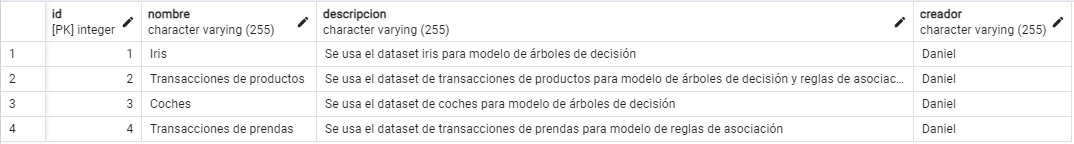
*Función:* Permite describir el conjunto de datos usado para la creación del

modelo, así como el/los modelos almacenados.

* **creador**: Es el nombre del creador del modelo.

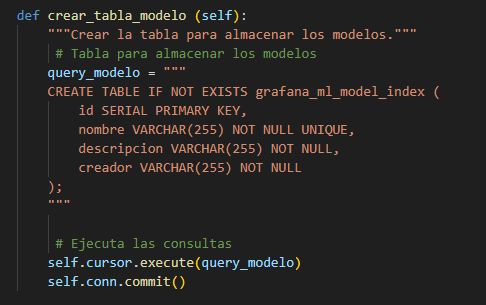
*Función:* Permite identificar al creador del modelo, lo cual facilita el control

con respecto a cada responsable de un modelo en específico.



**Figura 7:** Tabla *grafana\_ml\_model\_index* para almacenar los modelos. Foto tomada por el autor de PostgreSQL.

Para almacenar cada uno de los modelos fue necesario implementar un código de creación de una tabla en PostgreSQL (ver **Figura.8**) que permitiera almacenar dichos modelos, con los atributos definidos anteriormente, tanto para las filas como para las columnas.



**Figura 8:** Código de creación de la tabla *grafana\_ml\_model\_index*. Foto tomada por el autor del entorno de desarrollo en Visual Studio Code.

1. Crear una o varias tablas en la base de datos para almacenar un modelo en específico (árboles de decisión o reglas de asociación).

### 2.1.1 Árboles de decisión

En el caso del Árbol de Decisión, se optó por crear 3 tablas, reduciéndose la carga en la base de datos, evitando de esta manera la redundancia de los mismos. En el caso de este modelo, en vez de repetir los nombres de las características y de los valores de predicción, se referencia a la única vez que estos se guardan en la base de datos. A continuación, se describen cada una de estas tablas:

*1ra tabla* (grafana\_ml\_model\_arbol\_decision)

* Las filas serán cada uno de los nodos del árbol con sus atributos correspondientes.
* Las columnas son:
* **modelo\_id**: Es el identificador único de cada modelo de árbol de decisión.

*Función:* Permite que cada nodo tenga la referencia del modelo al que

pertenece.

* **nodo\_id**: Es el identificador único de cada nodo en el árbol.

*Función:* Es clave primaria y permite referenciar cada nodo de forma

individual.

* **nodo\_padre**: Es el identificador único del padre de cada nodo.

*Función:* Permite referenciar el padre en cada uno de los nodos. En el caso

de que el nodo sea raíz este campo será NULL.

* **característica**: Es el índice de la característica del conjunto de datos que el árbol de decisión utiliza para dividir los datos en este nodo. Cada característica es una de las columnas en el conjunto de datos originales.

*Función:* Indica qué característica se utiliza para realizar la división en este

nodo. Si el nodo es una hoja (no tiene hijos y predice un valor), este campo

puede ser NULL porque no hay más divisiones en las hojas.

* **umbral**: Este es el valor que se utiliza como punto de corte o umbral para dividir los datos en este nodo.

Por ejemplo, si el valor de la característica 2 es mayor que el umbral 3.5, entonces el árbol sigue por el nodo derecho; de lo contrario, sigue por el nodo izquierdo.

*Función:* Definir el valor que se compara con la característica para determinar

la dirección de la rama que debe seguirse en el árbol (nodo izquierdo o

derecho).

* **nodo\_izquierdo** y **nodo\_derecho**: Son referencias a otros nodos, los hijos izquierdo y derecho de cada nodo.

*Función:* Estas columnas permiten seguir el camino en el árbol hacia el

siguiente nodo. Si el nodo es una hoja (nodo terminal), ambos valores

pueden ser NULL porque no hay más nodos hijos que seguir.

* **es\_hoja**: Es un valor booleano que indica si el nodo actual es una hoja, es decir, un nodo terminal.

*Función:* Si es TRUE, significa que el nodo es una hoja (nodo terminal); si es

FALSE, el nodo sigue dividiendo los datos en más ramas.

* **valor predicción**: Es el valor de la predicción que hace el árbol en el nodo si es una hoja. Este valor se utiliza para predecir la clase o el valor final cuando se alcanza el nodo hoja.

*Función:* Se usa cuando se llega a una hoja durante el recorrido del árbol.

Este valor se devuelve como la predicción final.

*2da tabla* (caracteristicas)

* Las filas serán cada uno de las características del conjunto de datos.
* Las columnas son:
* **modelo\_id**: Es el identificador único de cada modelo de árbol de decisión.

*Función:* Permite que cada característica tenga la referencia del modelo al

que pertenece.

* **caracteristica\_id**: Es el identificador único de cada característica.

*Función:* Es clave primaria y permite referenciar cada característica de forma

individual.

* **nombre**: Es el nombre de cada una de las características del conjunto de datos.

*Función:* Permite guardar una sola vez el nombre de la característica y de

esta manera se evita la redundancia de los datos.

*3ra tabla* (valores\_prediccion)

* Las filas serán cada una de las clases o valores de predicción del conjunto de datos.
* Las columnas son:
* **modelo\_id**: Es el identificador único de cada modelo de árbol de decisión.

*Función:* Permite que cada clase tenga la referencia del modelo al que

pertenece.

* **prediccion\_id**: Es el identificador único de cada clase.

*Función:* Es clave primaria y permite referenciar cada una de las clases de

forma individual.

* **nombre\_clase**: Es el nombre de cada una de las clases del conjunto de datos.

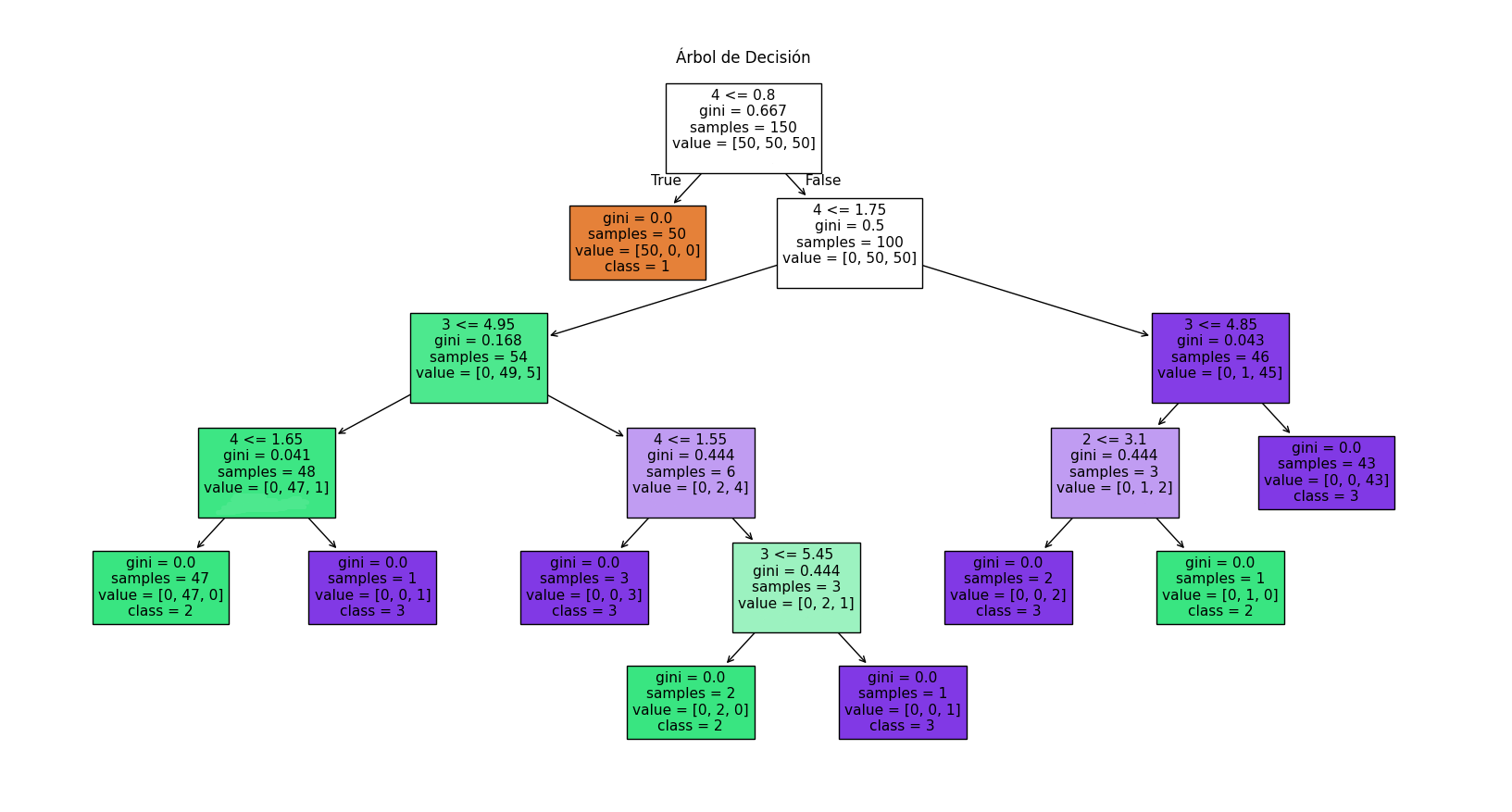
*Función:* Permite guardar una sola vez el nombre de la clase y de esta

manera se evita la redundancia de los datos.

A continuación, se muestra un ejemplo de un modelo de árbol de decisión (ver **Figura.9**) basado en el dataset “iris” y en el cual todos los datos son numéricos. Es importante destacar que en este documento se emplea la palabra inglesa “dataset” (que en español podría traducirse como juego de datos, conjunto de datos o tabla de datos) debido a su uso extensivo en el lenguaje técnico. Con este término se hace referencia a un conjunto de datos, los cuales se emplean comúnmente para probar el funcionamiento de algoritmos.

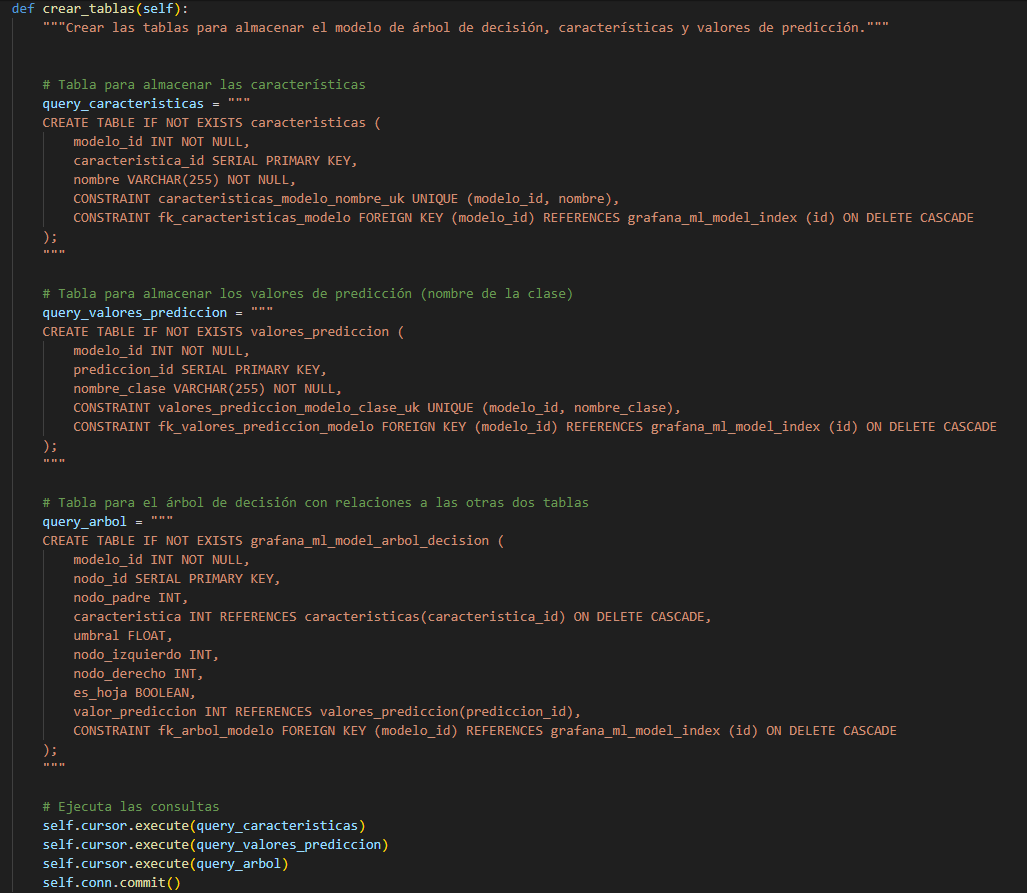
Este dataset cuenta con:

* 150 muestras (flores de iris).
* cada especie tiene 50 muestras.
* incluye 3 especies: Iris Setosa, Iris Versicolor, Iris Virginica.
* cada muestra en el conjunto de datos tiene 4 características que describen las flores: longitud del sépalo, anchura del sépalo, longitud del pétalo, anchura del pétalo.



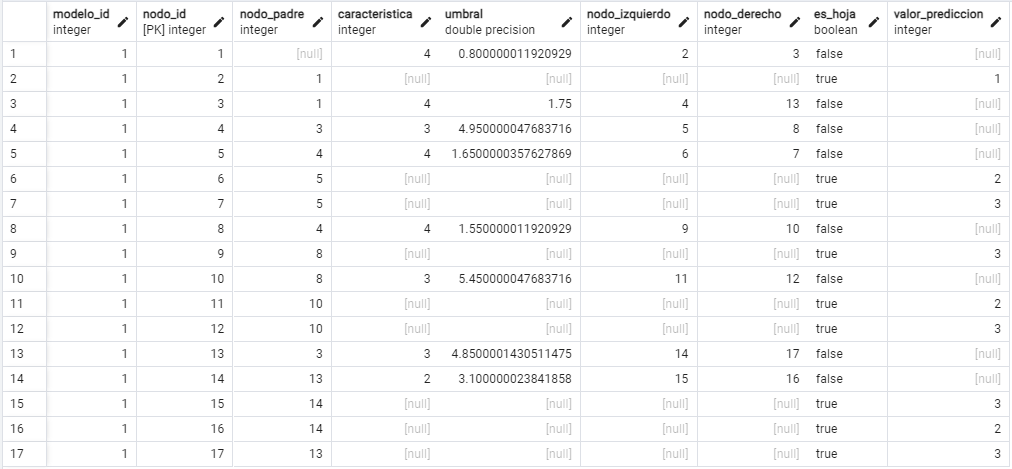
**Figura 9:** Ejemplo de Modelo de Árbol de decisiones del dataset “iris” (datos numéricos). Foto tomada por el autor de la ventana que muestra la biblioteca *matplotlib*.

Para almacenar este modelo fue necesario, primeramente, implementar un código de creación de las 3 tablas en PostgreSQL (ver **Figura.10**) que permitiera almacenar dicho modelo, con los atributos definidos anteriormente, tanto para las filas como para las columnas, en cada una de ellas.

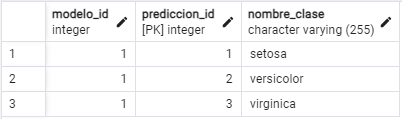
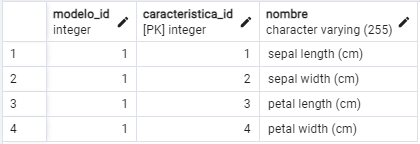


**Figura 10:** Código de creación de las tablas: *grafana\_ml\_model\_arbol\_decision*, *valores\_prediccion*, *caracteristicas*. Foto tomada por el autor del entorno de desarrollo en Visual Studio Code.

Luego se procede a ejecutar el algoritmo que permite tanto crear como entrenar dicho modelo, y una vez finalizada su ejecución, se puede observar cómo queda almacenado el modelo (ver **Figura.11**, **Figura.12**, **Figura.13**).



**Figura 11:** Almacenamiento del modelo de Árbol de decisiones con dataset “iris” en la tabla: *grafana\_ml\_model\_arbol\_decision*. Foto tomada por el autor de PostgreSQL.



**Figura 12:** Almacenamiento del modelo de Árbol de decisiones con dataset “iris” en la tabla: *caracteristicas*. Foto tomada por el autor de PostgreSQL.

**Figura 13:** Almacenamiento del modelo de Árbol de decisiones con dataset “iris” en la tabla: valores\_prediccion. Foto tomada por el autor de PostgreSQL.

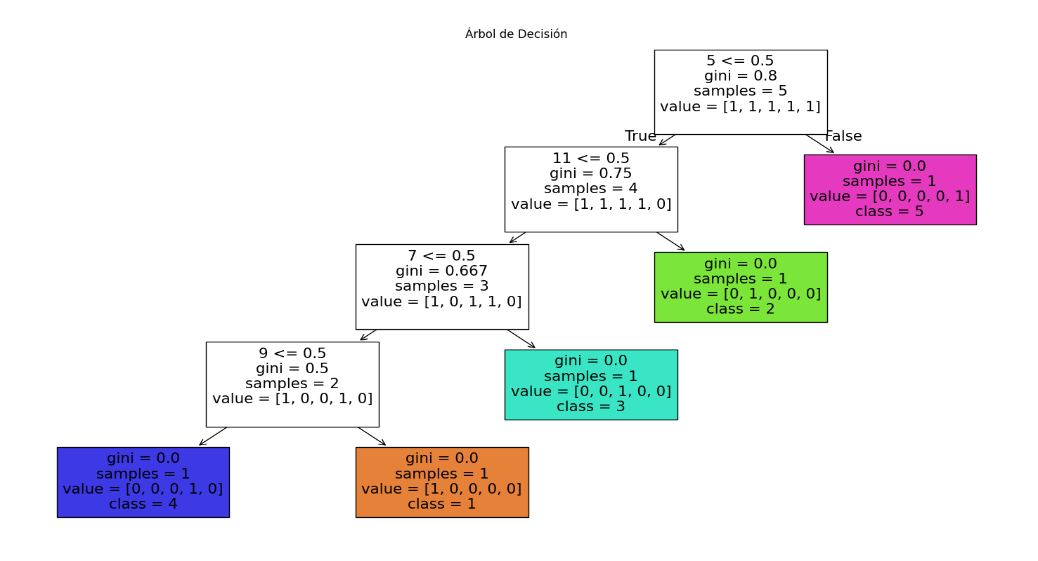
Como se mencionaba anteriormente, tanto para la columna *caracteristica* como para la de *valor\_prediccion* (ver **Figura.11)**, no se tienen en cuenta los nombres, sino que se asignan los identificadores que hagan referencia a dicho atributo en las otras dos tablas (ver **Figura.12**, **Figura.13**). En cuanto a la columna *valor\_prediccion*, lo primero que se analiza es si el nodo es hoja, de no serlo el campo se llena con NULL, de lo contrario se asigna el identificador correspondiente de la clase, lo mismo sucede con la columna *característica*, se analiza si el nodo es hoja, de serlo significa que no hay más divisiones en las hojas y por tanto el campo se establece en NULL, de lo contrario se asigna el identificador correspondiente de la característica.

Es importante destacar que el diseño físico adoptado para el almacenamiento del modelo de árboles se basa en mantener todos los nodos en una sola tabla (*grafana\_ml\_model\_arbol\_decision*), donde ciertos campos pueden contener valores NULL según el tipo de nodo. A pesar de optar por este enfoque, se evaluó también el uso de herencia de tablas, que aporta numerosas ventajas en ciertos contextos. Sin embargo, para este caso, se priorizó una solución que ofreciera simplicidad y flexibilidad. Este enfoque presenta varias ventajas:

* Al consolidar todos los nodos en una tabla única, las consultas, actualizaciones y recorridos jerárquicos se simplifican. Por ejemplo, no es necesario realizar *JOIN* para obtener la información completa de un árbol.
* Es más sencillo añadir o modificar atributos sin afectar la estructura general. Los cambios se realizan directamente en la tabla principal.
* Las operaciones sobre la jerarquía, como buscar nodos específicos o recorrer el árbol, se realizan de manera más directa y rápida.

Otro caso a tener en cuenta es cuando los datos no son numéricos, como se muestra en el siguiente ejemplo, donde se tiene un dataset de coches (ver **Figura.14**), en el cual todos los datos son no numéricos. Este conjunto de datos cuenta con:

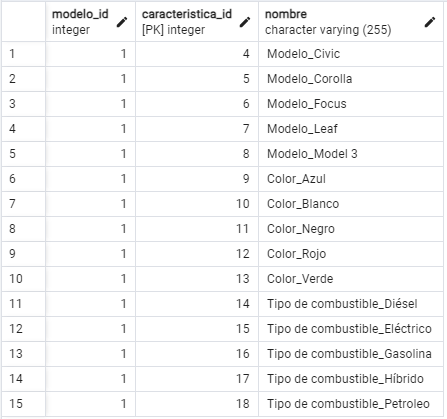
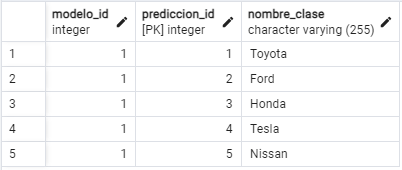
* Cada muestra en el conjunto de datos tiene 4 características que describen a los coches:
* Marca: Nombre de la empresa fabricante del coche (Toyota, Ford Honda, Tesla y Nissan).
* Modelo: Nombre específico del modelo del coche (Corolla, Focus, Civic, Model 3, Leaf).
* Color: Color del coche (Rojo, Azul, Negro, Blanco, Verde).
* Tipo de combustible: El tipo de combustible utilizado por el coche (Gasolina, Diésel, Híbrido, Eléctrico, Petróleo).



**Figura 14:** Ejemplo de Modelo de Árbol de decisiones del dataset de coches (datos no numéricos). Foto tomada por el autor de la ventana que muestra la biblioteca *matplotlib*.

Luego se procede a ejecutar el algoritmo que permite tanto crear como entrenar dicho modelo y en este caso, como son datos no numéricos, el algoritmo realiza un preprocesamiento donde convierte estos datos a numéricos para así poder llevar a cabo su funcionamiento. Una vez finalizada su ejecución, se puede observar cómo queda almacenado el modelo (ver **Figura.15**, **Figura.16**, **Figura.17**).

**Figura 15:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: *grafana\_ml\_model\_arbol\_decision*. Foto tomada por el autor de PostgreSQL.



**Figura 16:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: *valores\_prediccion*. Foto tomada por el autor de PostgreSQL.

**Figura 17:** Almacenamiento del modelo de Árbol de decisiones con dataset de coches en la tabla: *caracteristicas*. Foto tomada por el autor de PostgreSQL.

En el almacenamiento de este modelo con datos no numéricos, hay q destacar dos puntos clave:

1. Al aplicar el preprocesamiento, el algoritmo transforma las variables categóricas en un formato numérico, por lo que cada valor único dentro de una característica se convierte en una columna binaria (0 o 1) y cada nueva columna toma el nombre de la característica original seguido del valor específico, como, por ejemplo, Modelo\_Civic, Color\_Blanco, Tipo de combustible\_Gasolina.
2. Dado que las columnas binarias resultantes tienen solo dos valores posibles, es decir, 0 o 1, el algoritmo elige 0.5 como el umbral óptimo para realizar la división entre las dos clases.

En el ejemplo anterior no hay presencia de datos numéricos, solo nominales, pero puede darse el caso de que existan tantos campos numéricos como no. Para esa situación, se hace algo muy similar a cuando solo hay datos no numéricos, es decir el algoritmo realiza un preprocesamiento, pero solo convierte, a datos numéricos, las columnas que sean categóricas, mientras que las otras se mantienen igual y de esta forma se lleva a cabo su funcionamiento.

También se debe tener en cuenta cuando el modelo de árbol de decisión no se representa como un árbol binario, sino que se estructura como un árbol general, es decir que cada nodo puede tener dos o más hijos. Para este caso en específico, debido a las limitaciones de las diferentes implementaciones utilizadas, para crear y entrenar el modelo, en cuanto al manejo de árboles generales, se decide continuar investigando con el fin de lograr el manejo de este tipo de modelo especifico en dichas implementaciones.

### 2.1.2 Reglas de asociación

En el caso de las Reglas de Asociación, se optó por una sola tabla. A continuación, se describe la misma:

* Las filas serán cada una de las reglas generadas con sus respectivos atributos.
* Las columnas son:
* **modelo\_id**: Es el identificador único de cada modelo de regla de asociación.

*Función:* Permite que cada regla tenga la referencia del modelo al que

pertenece.

* **id**: Es el identificador único para cada regla de asociación en la tabla.

*Función:* Sirve como clave primaria para identificar cada regla de forma

individual.

* **antecedente**: Contiene los elementos o ítems que conforman la condición inicial o "antecedente" (por ejemplo, {A}) de la regla.

*Función:* Definir la condición inicial de la regla, es decir, los ítems cuya

presencia se analiza para determinar su relación con el consecuente.

* **consecuente**: Almacena los elementos o ítems que forman la "consecuencia" o resultado (por ejemplo, {B}) de la regla.

*Función:* Representa el resultado probable de la regla, indicando los

elementos que tienden a ocurrir junto con el antecedente.

* **soporte**: Mide la frecuencia con la que tanto el antecedente como el consecuente aparecen juntos en las transacciones.

*Función:* Calcula la proporción de transacciones en las que el antecedente y

el consecuente ocurren juntos.

* **confianza**: Representa la probabilidad de que el consecuente ocurra en una transacción dado que el antecedente está presente.

*Función:* Evalúa la certeza de la regla calculando la proporción de

transacciones con el antecedente y el consecuente, en relación con las

transacciones que contienen solo el antecedente.

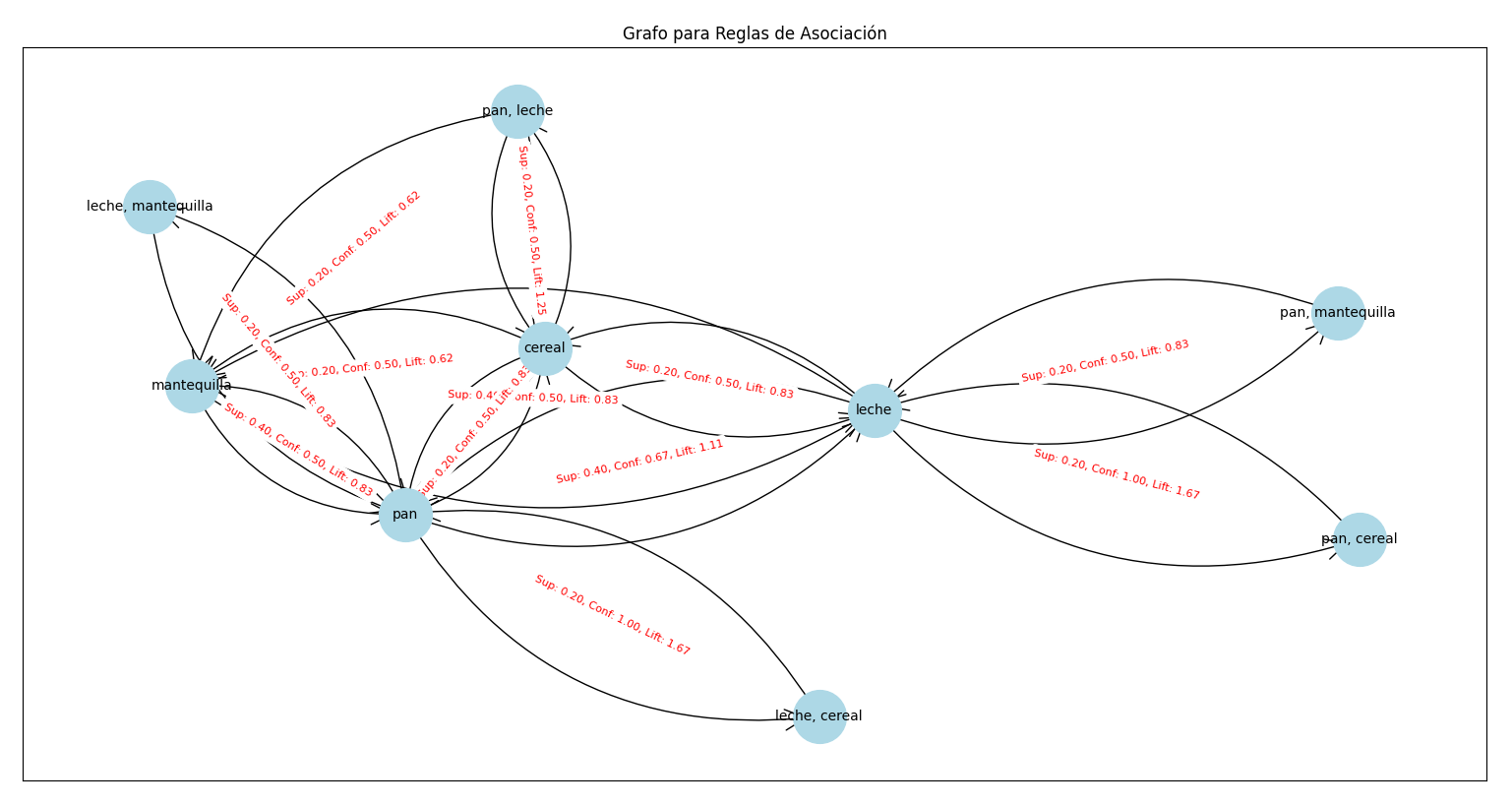
* **lift**: Es una medida que evalúa la independencia entre el antecedente y el consecuente.

*Función:* Calcula la fuerza de la asociación comparando el soporte conjunto

de A y B con la confianza de B.

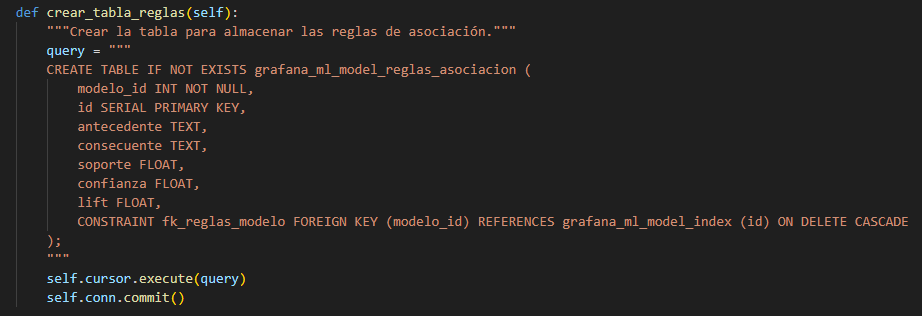
A continuación, se muestra un ejemplo de un modelo de Reglas de Asociación (ver **Figura.18**) basado en un dataset de transacciones de productos en el que cada fila representa una transacción, y cada columna corresponde a un producto específico que puede estar presente (1) o ausente (0) en dicha transacción. Este dataset cuenta con:

* cuatro productos: pan, leche, mantequilla y cereal.



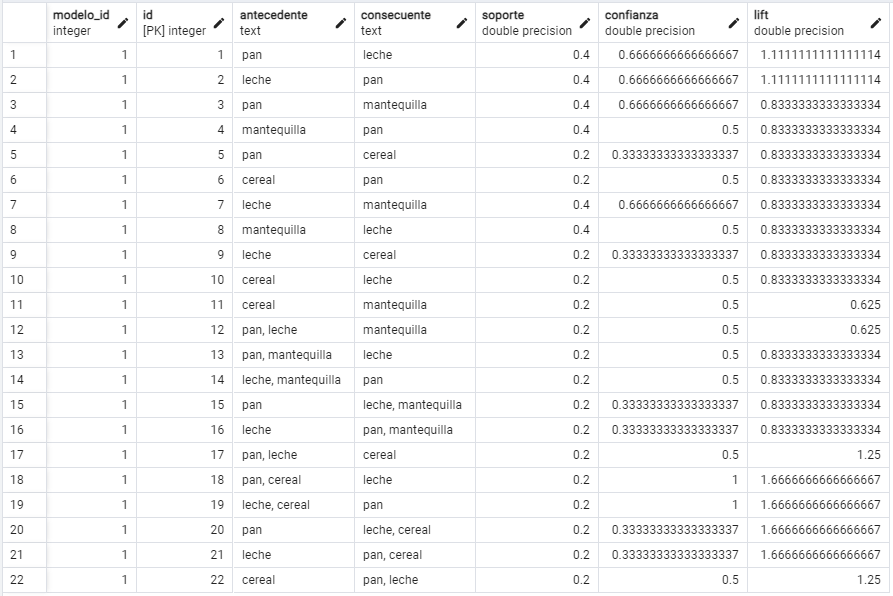
**Figura 18:** Ejemplo de Modelo de Reglas de Asociación. Foto tomada por el autor de la ventana que muestra la biblioteca *matplotlib*.

Para almacenar este modelo fue necesario, primeramente, implementar un código de creación de una tabla en PostgreSQL (ver **Figura.19**) que permitiera almacenar dicho modelo, con los atributos definidos anteriormente, tanto para las filas como para las columnas.



**Figura 19:** Código de creación de la tabla *grafana\_ml\_model\_reglas\_asociacion*. Foto tomada por el autor del entorno de desarrollo en Visual Studio Code.

Luego se procede a ejecutar el algoritmo que permite tanto crear como entrenar dicho modelo, y una vez finalizada su ejecución, se puede observar cómo queda almacenado el modelo (ver **Figura.20**)



**Figura 20:** Almacenamiento del modelo de Reglas de asociación con dataset de transacciones de productos en la tabla: *grafana\_ml\_model\_reglas\_asociacion*. Foto tomada por el autor de PostgreSQL.

En este ejemplo, los valores de soporte, confianza y lift están claramente definidos y se reflejan de forma precisa en la visualización (indicados en color rojo sobre cada flecha), lo cual permite verificar fácilmente su correspondencia con los datos almacenados en la tabla del modelo (ver **Figura.20**). Esto no solo asegura la coherencia entre la visualización y los datos subyacentes, sino que también refuerza la interpretabilidad de las reglas. Además, la dirección de las flechas en el grafo indica con claridad las relaciones entre los elementos, destacando quién actúa como antecedente y quién como consecuente en cada regla de asociación.

## 2.2 Descripción de la solución

El uso de diagramas UML (Unified Modeling Language) ofrece múltiples beneficios, ya que permite visualizar, documentar, analizar y comunicar los distintos aspectos de la solución propuesta. A continuación, en los siguientes subepígrafes, se emplearán alguno de ellos, con el fin de describir de manera más estructurada el proceso de obtención y visualización de los modelos.

### 2.2.1 Diagrama de componentes

Los diagramas de componentes presentan una vista de alto nivel de los elementos principales del sistema y sus conexiones. Su enfoque está en la arquitectura estática y en cómo los componentes colaboran y dependen uno de los otros.



**Figura 21:** Diagrama de componente. Foto tomada por el autor de Enterprise Architect.

Componentes:

* Database (PostgreSQL): Componente central que almacena los datos originales y los modelos entrenados (árboles de decisión o reglas de asociación).
* EntornoPython: Representa el entorno donde ocurre la creación y el entrenamiento de los modelos. Incluye los siguientes subcomponentes:
* Controler: Se encarga de gestionar el flujo de datos y coordinar la interacción entre los modelos y la base de datos.
* DecisionTreeModel / AssociationRulesModel: Subcomponentes responsables de implementar los algoritmos de aprendizaje automático y generar las estructuras del modelo.
* Grafana: Responsable de la visualización de los modelos. Contiene los siguientes subcomponentes:
* Grafana Model: Recupera los datos de el/los modelo/s almacenado/s en PostgreSQL mediante consultas SQL.
* Grafana Controler: Representado por el código JavaScript que el usuario introduce. Este código se encarga de procesar los datos obtenidos del modelo y convertirlos en un formato visualizable.
* Grafana Vista: Panel de visualización donde los datos procesados son representados gráficamente, ya sea como árbol de decisión o reglas de asociación.

En este diagrama (ver **Figura.21**) se muestran las relaciones de dependencias de los componentes y como están estructurados e interconectados. A continuación, se muestra una tabla que detalla y explica cada una de estas relaciones:

|  |  |
| --- | --- |
| **Dependencia** | **Descripción** |
| Grafana Controler -> Grafana Model | Grafana Controler depende de los datos proporcionados por Grafana Model para transformarlos según sea necesario para la visualización. |
| Grafana Vista -> Grafana Controler | Grafana Vista depende de los resultados procesados por Grafana Controler para generar las visualizaciones finales. |

**Tabla 1:** Descripción detallada de cada una de las dependencias relacionadas con la visualización.

|  |  |
| --- | --- |
| **Dependencia** | **Descripción** |
| Entorno Python -> Database (PostgreSQL) | El Entorno Python depende de Database para guardar los modelos entrenados y recuperar datos de entrada. |
| Controler -> DecisionTreeModel / AssociationRulesModel | Controler depende tanto de los algoritmos implementados como de las estructuras generadas por DecisionTreeModel / AssociationRulesModel para coordinar las interacciones entre los modelos y la base de datos. |
| Grafana -> Database (PostgreSQL) | Grafana depende de Database para ejecutar consultas SQL y obtener datos de los modelos. |

**Tabla 2:** Descripción detallada de cada una de las dependencias relacionadas con la gestión de datos.

### 2.2.2 Diagrama de secuencia

Los diagramas de secuencia muestran el orden temporal de las interacciones entre componentes y actores en un flujo específico. Permite visualizar cómo se desarrolla un proceso dentro del sistema y en qué secuencia los elementos se comunican para completar una tarea en específico. A continuación, se muestran dos diagramas de secuencia (ver **Figura.22**, **Figura.23**) asociado a los procesos:

1. Carga inicial de los datos, creación, entrenamiento y almacenamiento del modelo.
2. Recuperación y visualización del modelo en Grafana.



**Figura 22:** Diagrama de secuencia del proceso: Carga inicial de los datos, creación, entrenamiento y almacenamiento del modelo. Foto tomada por el autor de Enterprise Architect.

Explicación del Flujo:

1. Inicio del proceso

**Cliente de Análisis Visual -> Controler**: El usuario inicia el proceso de creación y entrenamiento del modelo solicitándolo al *Controler*.

1. Creación y entrenamiento del modelo

**Controler -> DecisionTreeModel****/ AssociationRulesModel**: *Controler* solicita a *DecisionTreeModel/ AssociationRulesModel* que cree y entrene un modelo de árbol de decisión o de reglas de asociación.

**DecisionTreeModel/ AssociationRulesModel -> Controler**: *DecisionTreeModel/ AssociationRulesModel* entrena el modelo y devuelve la estructura del árbol o la regla de asociación al *Controler*.

1. Almacenamiento del modelo en la base de datos

**Controler -> Database (PostgreSQL)**: *Controler* guarda la estructura del modelo entrenado en la base de datos.

**Database -> Controler**: La base de datos confirma que la estructura del modelo ha sido almacenada.

1. Notificación del almacenamiento del modelo

**Controler -> Cliente de Análisis Visual**: *Controler* envía notificación al usuario de que el modelo ha sido almacenado.



**Figura 23:** Diagrama de secuencia del proceso: Recuperación y visualización del modelo en Grafana. Foto tomada por el autor de Enterprise Architect.

Explicación del Flujo:

1. Inicio del proceso

**Cliente de Análisis Visual -> Grafana Vista**: El usuario accede a Grafana y solicita visualizar un modelo desde un panel de visualización.

1. Solicitud de datos para visualización

**Grafana Vista -> Grafana Controler**: *Grafana Vista* solicita a *Grafana Controler* los datos necesarios para generar la visualización del modelo.

1. Ejecución de consulta SQL

**Grafana Controler -> Grafana Model**: *Grafana Controler*delega a *Grafana Model* la tarea de ejecutar una consulta SQL para recuperar el modelo desde la base de datos.

**Grafana Model ->** **Database (PostgreSQL)**: *Grafana Model*envía la consulta SQL a la base de datos.

1. Recuperación de datos desde la base de datos

**Database (PostgreSQL) -> Grafana Model**: La base de datos procesa la consulta SQL y devuelve los datos del modelo solicitado a *Grafana Model*.

1. Procesamiento de los datos recuperados

**Grafana Model -> Grafana Controler**: *Grafana Model*pasa los datos recuperados a *Grafana Controler*.

**Grafana Controler -> Grafana Vista**: *Grafana Controler*procesa los datos para adaptarlos y enviarlos a *Grafana Vista*.

1. Visualización del modelo

**Grafana Vista -> Cliente de Análisis Visual**: *Grafana Vista* utiliza los datos procesados ​​para generar y presentar la visualización del modelo al usuario en el panel de Grafana.

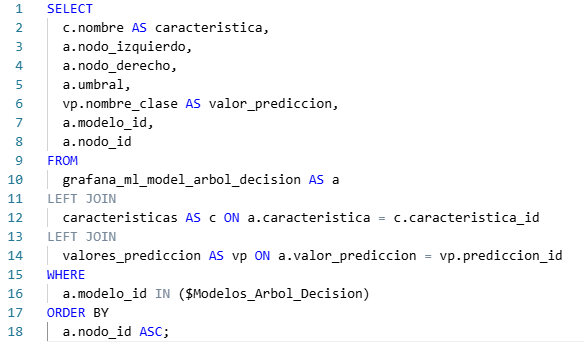
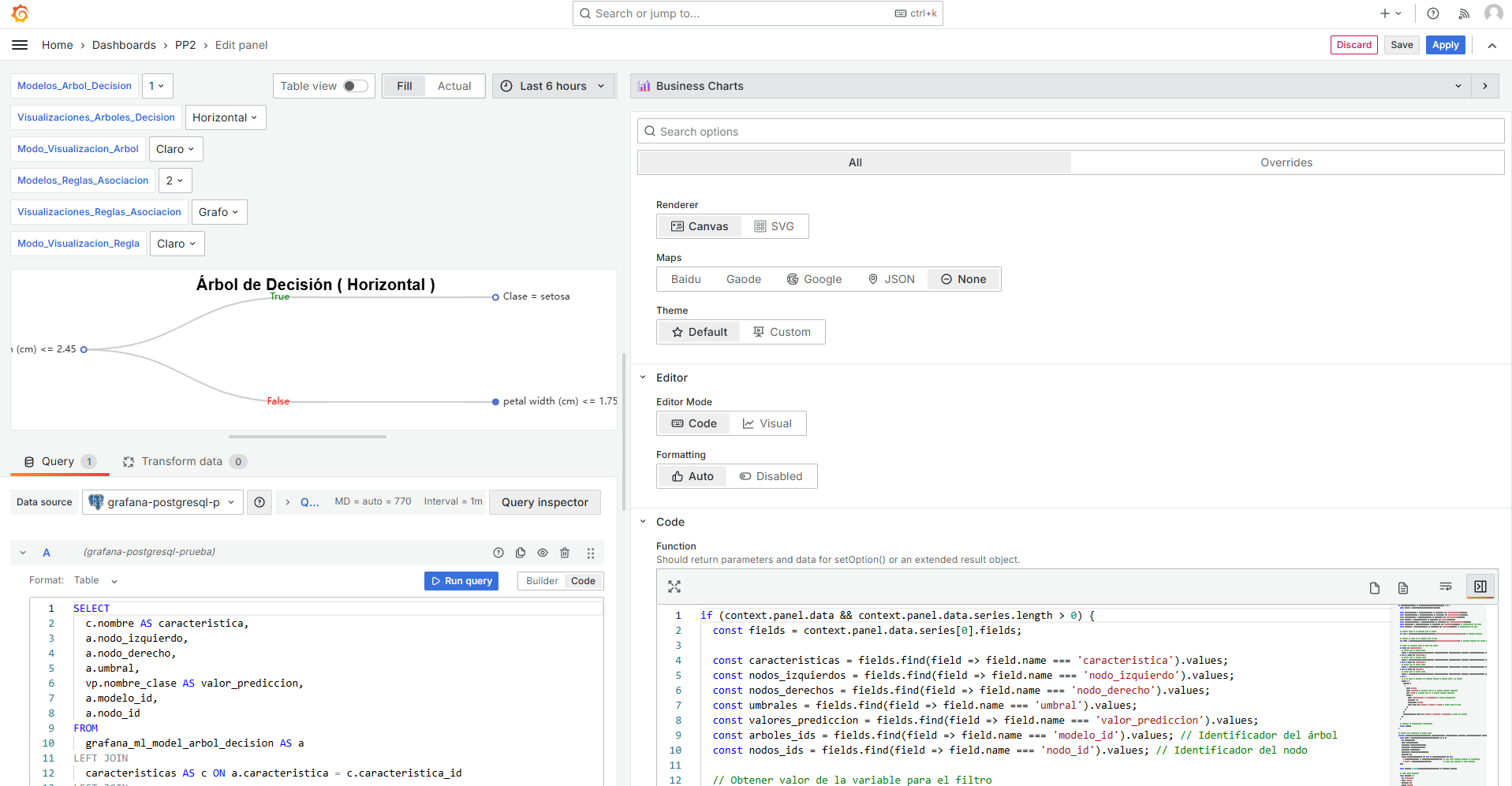
## 2.3 Visualización de los modelos

Una vez realizado los procesos de crear, entrenar y almacenar los modelos, se pasa a la siguiente etapa, que no es más que la visualización de los mismos. Para ello se tuvieron en cuenta los siguientes pasos:

1. Ejecutar una consulta que permita obtener los datos del modelo que se desea visualizar.
2. Modificar el código de la visualización seleccionada del sitio de Echarts, con el fin de ajustar la entrada de datos y la estructura del modelo que se desea mostrar.
3. Finalmente, se muestra la visualización del modelo.

### 2.3.1 Árboles de decisión

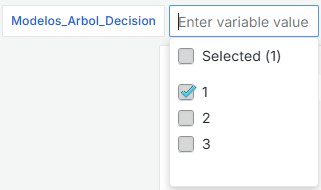
En el caso de los árboles de decisión, se muestra, a continuación, un ejemplo del proceso para visualizar el modelo.

**Consulta SQL**

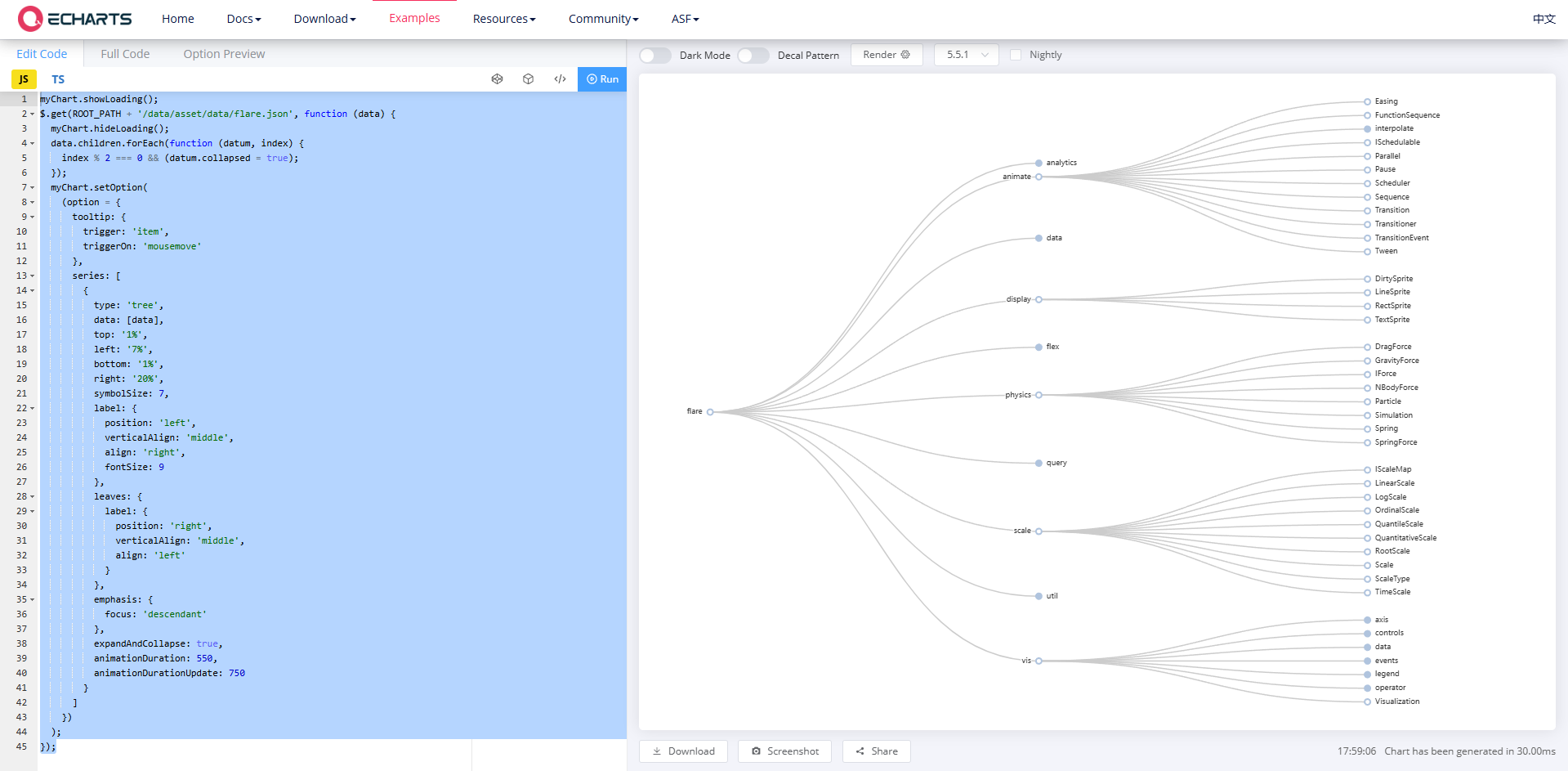
**Figura 24:** Ejemplo deconsulta SQL para obtener modelo de árbol de decisión. Foto tomada por el autor de Grafana.

Como se muestra en la **Figura.24**, se realizauna consultapara obtener los datos del modelo de árbol de decisión. En este caso es necesario hacer un LEFT JOIN con el fin de no solo obtener los nodos cuyos identificadores de características y de valores de predicción coincidan con los de las tablas donde se guardan los mismos (*caracteristicas* o *valores\_prediccion*), sino también aquellos que no coincidan, pues el fin de la sentencia LEFT JOIN [43] no es más que devolver todas las filas de la tabla izquierda (*grafana\_ml\_model\_arbol\_decision*), y las coincidencias de la tabla derecha (*características*, *valores\_prediccion*). También se hizo uso de la sentencia WHERE con el fin de mostrar en el panel los modelos de árboles de decisión (ver **Figura.25**) que estén disponibles en la base de datos, a partir de una variable en Grafana, en este caso, “Modelos\_Arbol\_Decision”. Dicha variable se configuró como multi-value para lograr visualizar más de un árbol en el mismo panel, más adelante se verá el mismo. Además, se realizó un ORDER BY de forma ascendente para garantizar que los nodos siempre se obtengan ordenadamente por su id.



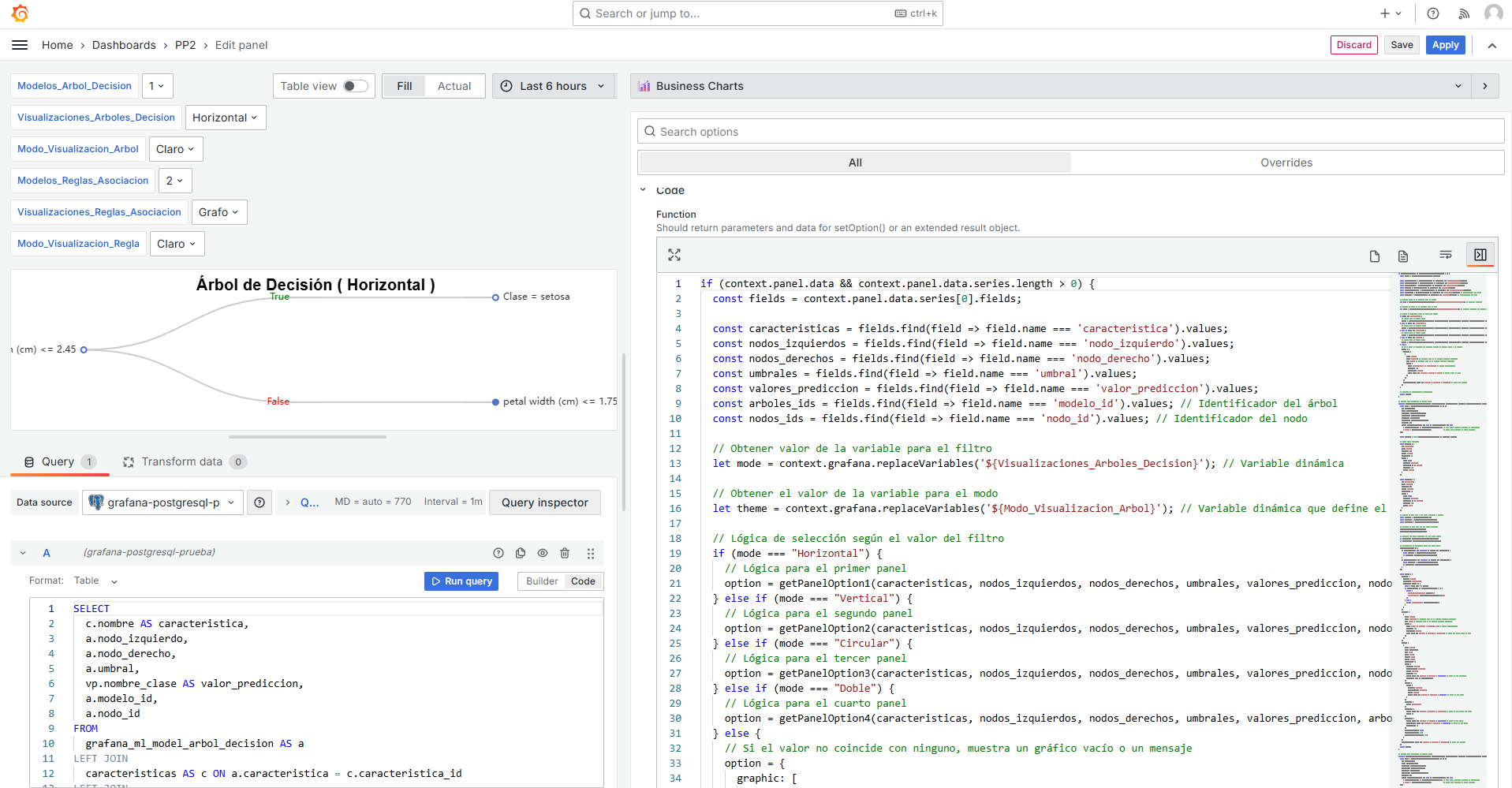
**Figura 25:** Variable en Grafana para mostrar modelos de árbol de decisión disponibles en la base de datos. Foto tomada por el autor de Grafana.

Luego se selecciona el código de la visualización, que se desea modificar en Grafana para mostrar el modelo, en el sitio de Echart (ver **Figura.26**).



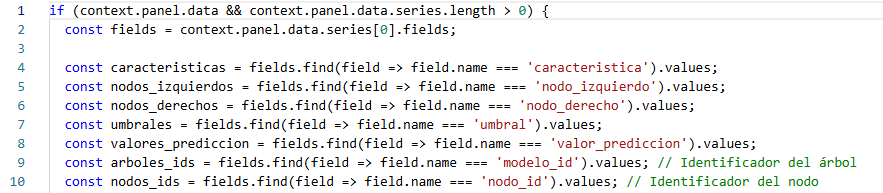
**Figura 26:** Código de visualización de un árbol en forma horizontal en Echarts. Foto tomada por el autor del sitio de Echarts.

Posteriormente, se modifica el código seleccionado (JavaScript) en Grafana (ver **Figura.27**). En este código se hicieron varios ajustes, uno de ellos es que la entrada de datos sea directamente proporcionada de la consulta ejecutada con anterioridad especificando cada campo de las columnas de la base de datos (ver **Figura.28**), esto facilita que, al almacenar otro modelo, solo cambien los datos proporcionados y de esta forma, el código, adopta una estructura genérica.



**Código JavaScript**

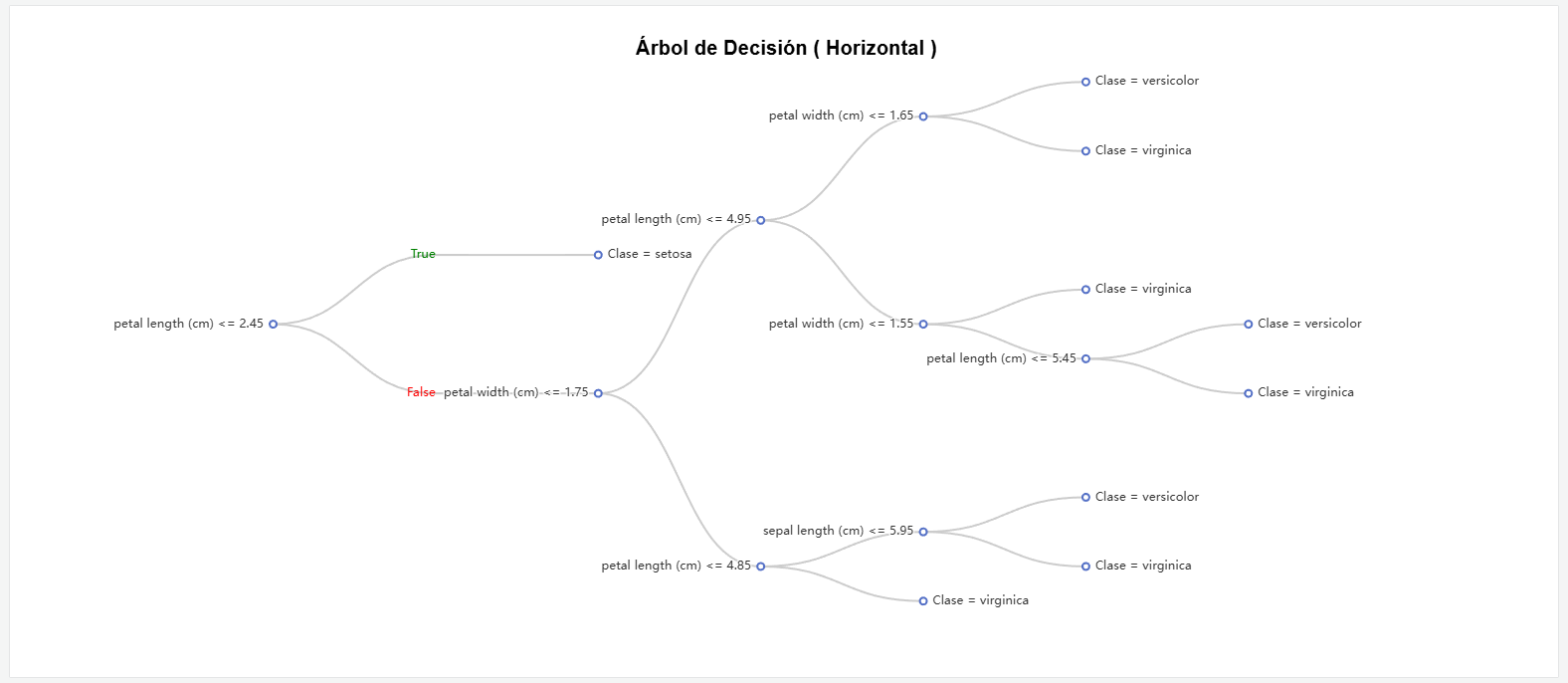
**Figura 27:** Código JavaScript modificado para mostrar árbol de decisión. Foto tomada por el autor de Grafana.



**Figura 28:** Código JavaScript para obtener lo valores de la consulta SQL a partir de los mismos campos de la base de datos en árbol de decisión. Foto tomada por el autor de Grafana.

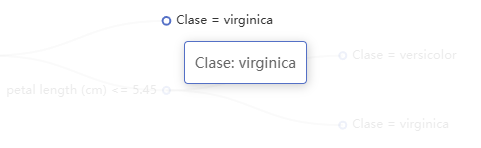
Otro ajuste es con respecto a la forma de visualizar el modelo, donde se tiene en cuenta, por ejemplo, la comparación de la característica con el umbral, lo cual representa una decisión clave en cada nodo del árbol y define cómo se separan los datos. Esta comparación permite identificar si una instancia cumple con la condición establecida, lo cual se refleja en las etiquetas True o False asignadas a las ramas del árbol. Dichas etiquetas son fundamentales para entender de forma clara y sencilla el flujo lógico del modelo, pues indican el camino que sigue cada decisión, simplificando la interpretación de cómo las características y los valores conducen a una predicción final.

Finalmente, se muestra la visualización del modelo (ver **Figura.29**) con los datos del dataset iris, el cual fue utilizado para este ejemplo.

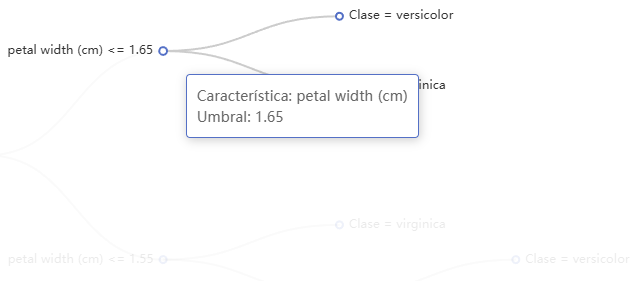


**Figura 29:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset iris. Foto tomada por el autor de Grafana.

Una de las ventajas de usar las visualizaciones que ofrece Echarts, y por la cual fue seleccionada para mostrar los modelos, es por su capacidad de interacción (ver **Figura.30**, **Figura.31**). En este caso cuando el cursor se sitúa encima de un nodo se muestra la información del mismo, cuando es hoja se muestra la clase a la que pertenece y cuando no, la característica y el umbral. Además, ofrece la posibilidad de extender o contraer el árbol haciendo clic en los nodos.

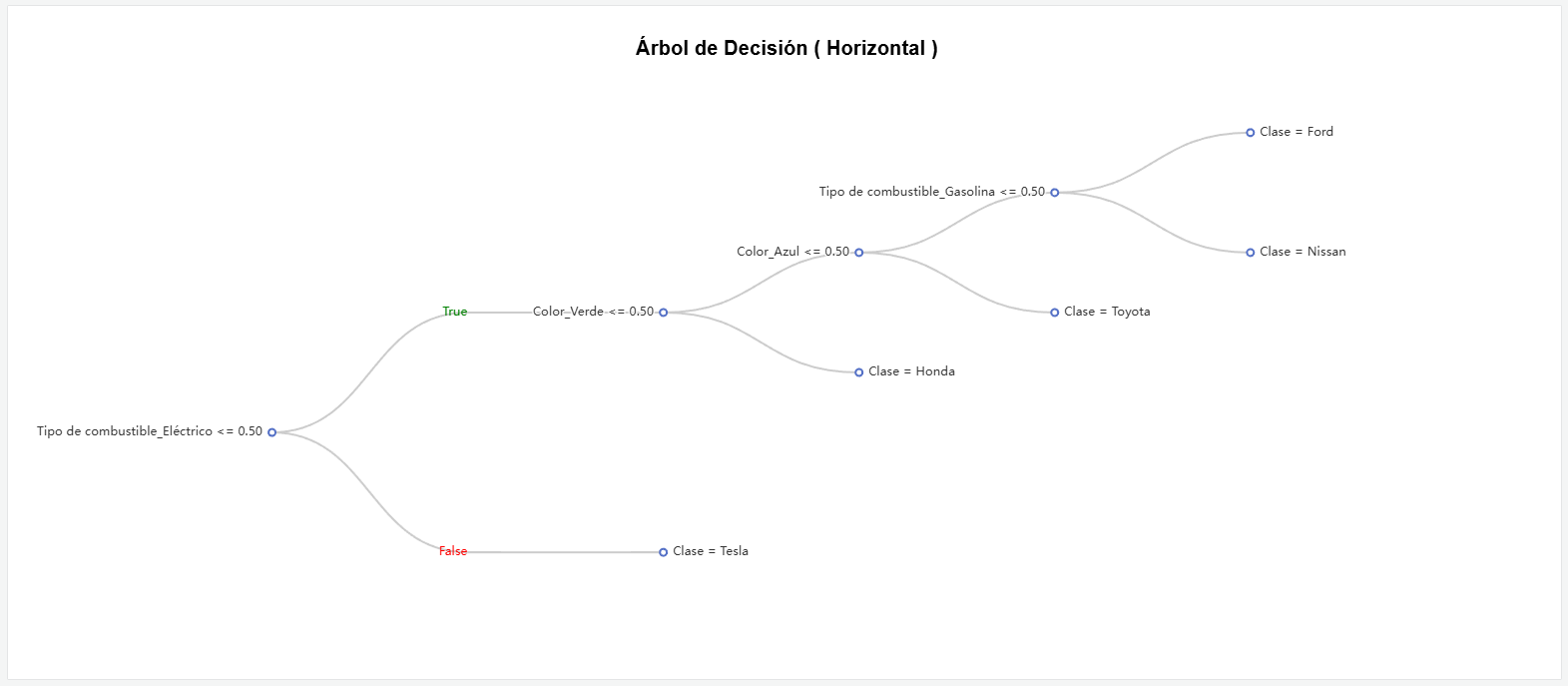


**Figura 30:** Ejemplo de cursor sobre un nodo no hoja en árbol de decisón. Foto tomada por el autor de Grafana.



**Figura 31:** Ejemplo de cursor sobre un nodo hoja en árbol de decisón. Foto tomada por el autor de Grafana.

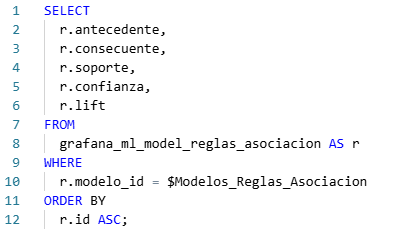
Como se mencionaba anteriormente, al contar con un código genérico diseñado para la entrada de datos de cualquier modelo de árbol de decisión almacenado, se logra una mayor flexibilidad y adaptabilidad en el manejo de diferentes modelos. Esto significa que el código puede procesar estructuras de árboles con distintas configuraciones y características, permitiendo que el enfoque principal esté en los datos específicos de cada modelo. Gracias a esta generalización, los cambios necesarios se limitan únicamente a los datos que se deben visualizar, sin necesidad de modificar la lógica base. A continuación, se muestra la visualización del árbol de decisión (ver **Figura.32**) con el dataset de coches. De esta forma se logra ver como la estructura se mantiene y solo cambian los datos que se obtuvieron de la consulta.



**Figura 32:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset de coches. Foto tomada por el autor de Grafana.

### 2.3.2 Reglas de asociación

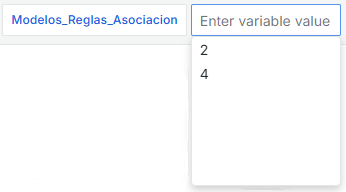
En el caso de las reglas de asociación, se muestra, a continuación, un ejemplo del proceso para visualizar el modelo.



**Consulta SQL**

**Figura 33:** Ejemplo deconsulta SQL para obtener modelo de reglas de asociación. Foto tomada por el autor de Grafana.

Como se muestra en la **Figura.33**, se realizauna consultapara obtener los datos del modelo de reglas de asociación. En el caso de este modelo, no es necesario hacer un LEFT JOIN debido a que los datos de las reglas de asociación se encuentran en una misma tabla. También se hizo uso de la sentencia WHERE con el fin de mostrar en el panel los modelos de árboles de decisión (ver **Figura.34**) que estén disponibles en la base de datos, a partir de una variable en Grafana, en este caso, “Modelos\_Reglas\_Asociacion”. Además, se realizó un ORDER BY de forma ascendente para garantizar que las reglas siempre se obtengan ordenadamente por su id.



**Figura 34:** Variable en Grafana para mostrar modelos de reglas de asociación disponibles en la base de datos. Foto tomada por el autor de Grafana.

Luego se selecciona el código de la visualización, que se desea modificar en Grafana para mostrar el modelo, en el sitio de Echart (ver **Figura.35**).



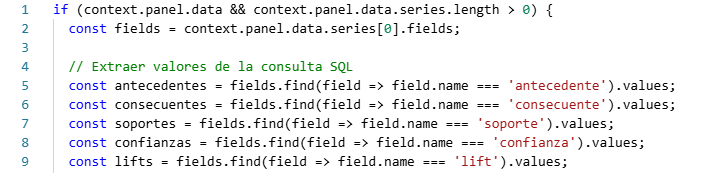
**Figura 35:** Código de visualización de un grafo en Echarts. Foto tomada por el autor del sitio de Echarts.

Posteriormente, se modifica el código seleccionado (JavaScript) en Grafana (ver **Figura.36**). En este código se hicieron varios ajustes, uno de ellos es que la entrada de datos sea directamente proporcionada de la consulta ejecutada con anterioridad especificando cada campo de las columnas de la base de datos (ver **Figura.37**), esto facilita que, al almacenar otro modelo, solo cambien los datos proporcionados y de esta forma, el código, adopta una estructura genérica.



**Código JavaScript**

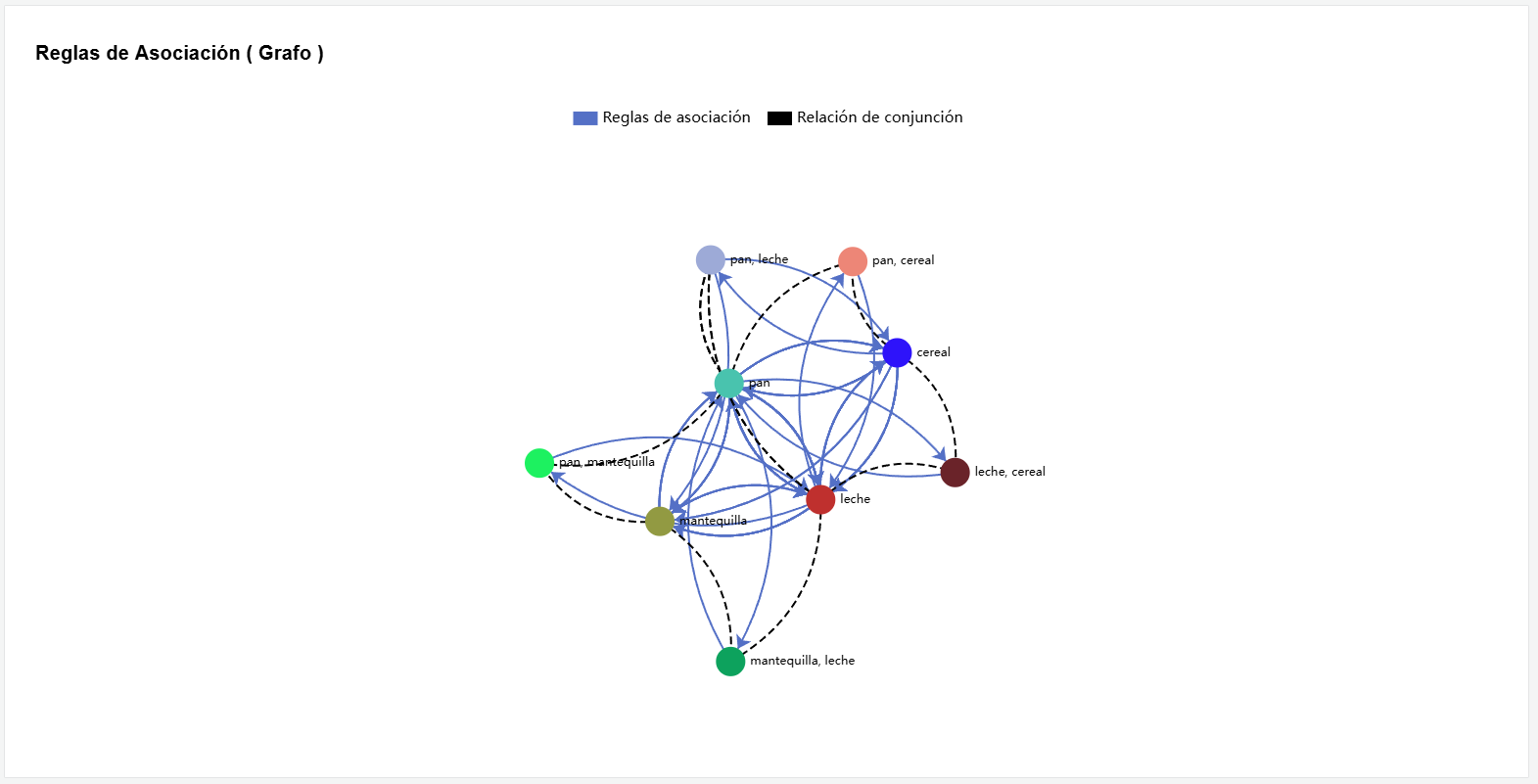
**Figura 36:** Código JavaScript modificado para mostrar reglas de asociación. Foto tomada por el autor de Grafana.



**Figura 37:** Código JavaScript para obtener lo valores de la consulta SQL a partir de los mismos campos de la base de datos en reglas de asociación. Foto tomada por el autor de Grafana.

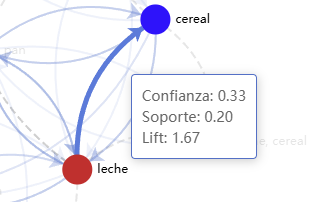
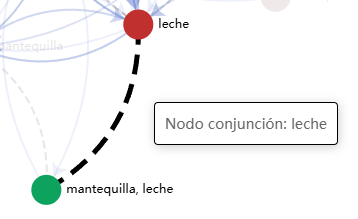
Otro ajuste es con respecto a la forma de visualizar el modelo, donde se tiene en cuenta, por ejemplo, etiquetas para mostrar la confianza, el soporte, el lift, las cuales permiten interpretar la calidad y relevancia de las reglas generadas. Además, la dirección de las relaciones mediante flechas es otro aspecto importante a tener en cuenta a la hora de visualizar este tipo de modelo, pues de esta forma el usuario logra entender cuál es el antecedente y el consecuente, así como también las relaciones de conjunción, que surgen cuando un elemento compuesto, como por ejemplo “**pan, leche**,” mantiene conexiones individuales con los elementos que lo conforman, como “**pan**” y “**leche**.” Estas relaciones son cruciales porque muestran cómo los elementos individuales contribuyen a formar combinaciones más complejas dentro del modelo. Por último y no menos importante, el color de los nodos, pues cada uno representa un elemento en particular (antecedente o consecuente).

Finalmente, se muestra la visualización del modelo (ver **Figura.38**) con los datos del dataset de transacciones de productos, el cual fue utilizado para este ejemplo.



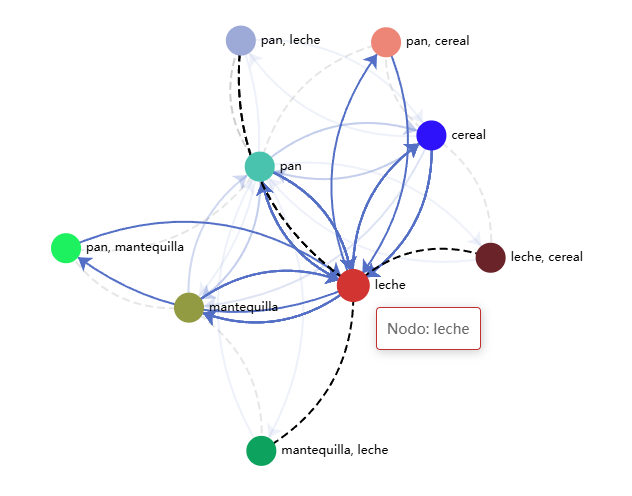
**Figura 38:** Visualización de modelo de reglas de asociación en forma de grafo modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

En el epígrafe anterior se comentó sobre la ventaja de usar las visualizaciones que ofrece Echarts tanto para los árboles de decisión como para las reglas de asociación y en esta sección se vuelve a demostrar las capacidades de interacción que ofrecen estas visualizaciones a la hora de interacción del usuario (ver **Figura.39**, **Figura.40**, **Figura.41**). En este caso cuando el cursor se sitúa encima de un nodo se muestra la información del nodo, especificando su nombre, para este ejemplo el nombre del o los productos, así como también las relaciones, es decir las reglas de asociación y las relaciones de conjunción, que tiene con los demás nodos. Algo similar sucede cuando el usuario se sitúa encima de una arista (forma de flecha o línea discontinua), se visualiza una etiqueta que contiene los valores de confianza, soporte y lift, así como la dirección de la misma, en el caso de la arista con forma de flecha, mientras que se muestra una etiqueta que contiene el elemento del nodo de conjunción cuando la arista tiene forma de línea discontinua.



**Figura 39:** Ejemplo de cursor sobre una arista con forma de flecha en reglas de asociación. Foto tomada por el autor de Grafana.

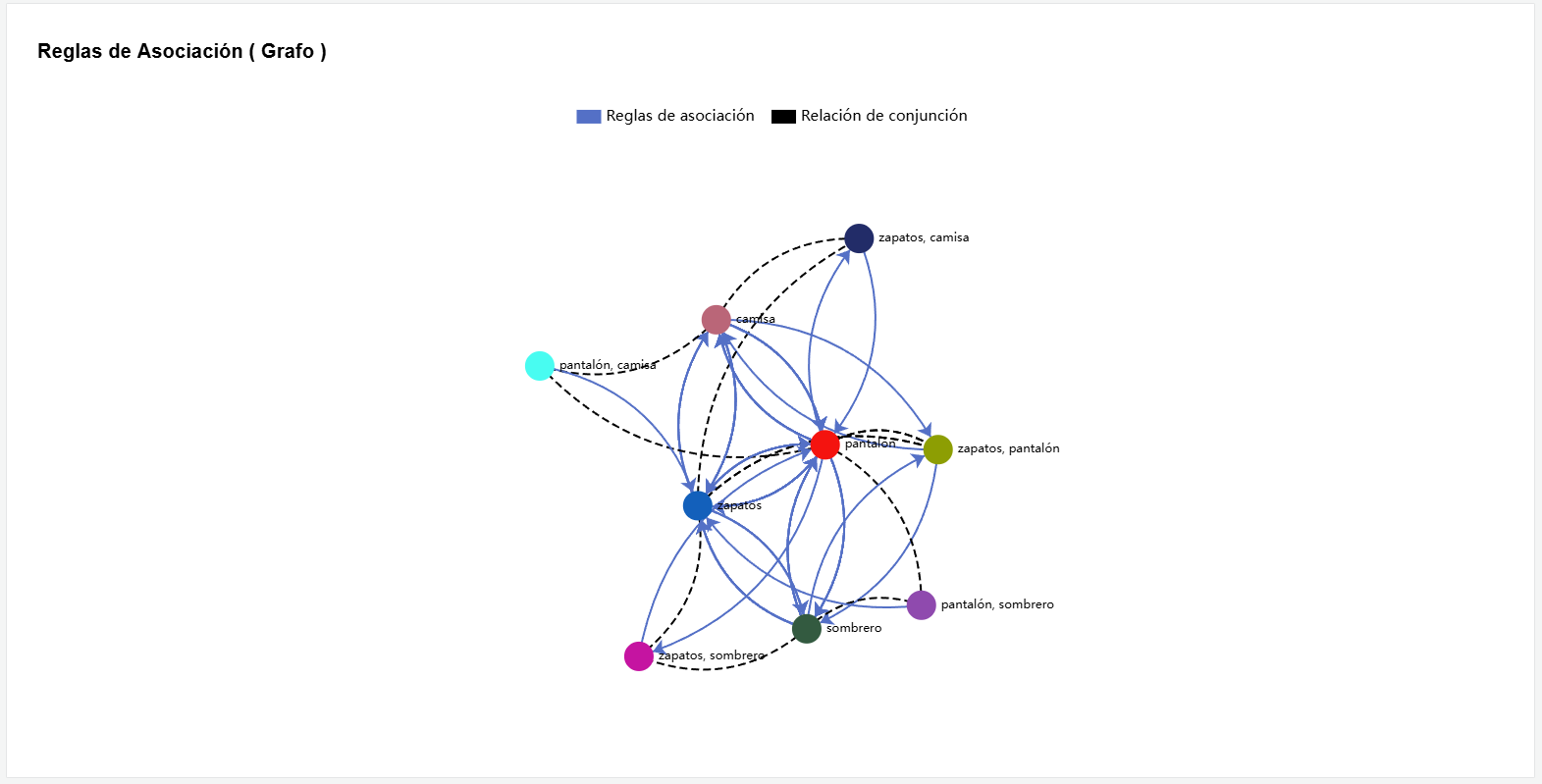
**Figura 40:** Ejemplo de cursor sobre una arista con forma de línea discontinua en reglas de asociación. Foto tomada por el autor de Grafana.



**Figura 41:** Ejemplo de cursor sobre un nodo en reglas de asociación. Foto tomada por el autor de Grafana.

Como se mencionó anteriormente, disponer de un código genérico para procesar la entrada de datos de cualquier modelo de reglas de asociación almacenado permite una mayor flexibilidad y adaptabilidad en el manejo de diferentes configuraciones. Esto implica que el código es capaz de interpretar reglas generadas a partir de distintos conjuntos de datos y métricas, manteniendo el enfoque en los valores específicos de cada modelo. Gracias a esta generalización, los ajustes necesarios se limitan únicamente a los datos a visualizar, sin requerir modificaciones en la lógica subyacente. A continuación, se presenta la visualización de las reglas de asociación (ver **Figura.42**) con dataset de transacciones de prendas. Este dataset cuenta con:

* cuatro prendas: camisa, pantalón, zapatos y sombrero.



**Figura 42:** Visualización de modelo de reglas de asociación en forma de grafo modo claro con dataset de transacciones de prendas. Foto tomada por el autor de Grafana.

Tanto para los árboles de decisión como para las reglas de asociación, existe un conjunto de parámetros configurables que permiten personalizar la forma en que se visualizan los modelos. Estos ajustes incluyen opciones para definir cómo se presentan las estructuras, los datos y las métricas asociadas, adaptándose a las necesidades específicas del usuario. En el próximo capítulo, se explorarán en detalle estas opciones de personalización.

## 2.4 Conclusiones parciales

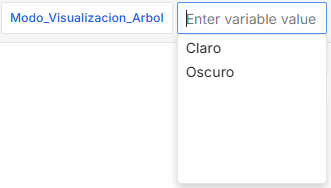
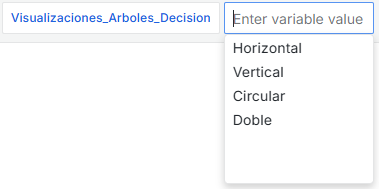
* Se logró establecer un diseño adecuado para el almacenamiento de modelos de aprendizaje automático (árboles de decisión y reglas de asociación) en PostgreSQL, destacando el uso de una tabla central para índices de modelos (*grafana\_ml\_model\_index*) y estructuras específicas para árboles de decisión y reglas de asociación. Este diseño facilita la consulta y recuperación de los datos sin redundancias innecesarias.
* En el caso de los árboles de decisión, se comprobó la utilidad de estructurar la información en tres tablas separadas (*grafana\_ml\_model\_arbol\_decision*,*valores\_prediccion* y *características*). Este enfoque optimiza la organización y reduce la redundancia de datos, facilitando su gestión y consulta.
* Para las reglas de asociación, se determinó que una única tabla (*grafana\_ml\_model\_reglas\_asociacion*) es adecuada para almacenar los antecedentes, los consecuentes y las métricas clave, como el soporte, la confianza y el lift, logrando una estructura consolidada y fácil de manejar.
* La integración de Grafana con Echarts mostró ser funcional para la visualización de modelos, brindando opciones como árboles y grafos. Sin embargo, se evidenció que el ajuste del código es fundamental para adaptar las visualizaciones a los datos obtenidos desde la base de datos.
* Las figuras incluidas en este capítulo demuestran cómo las representaciones visuales pueden adaptarse a diferentes tipos de modelos y conjuntos de datos, destacando la importancia de la flexibilidad en el diseño de las visualizaciones para garantizar su aplicabilidad en diversos contextos.

# **Capítulo 3: Validación**

En este capítulo se muestran distintas formas disponibles para visualizar los modelos de aprendizaje automático (árboles de decisión y reglas de asociación) a partir de los casos simples presentados en la sección anterior. Además, para evaluar la funcionalidad de estas visualizaciones, se presentan dos casos de prueba que permiten verificar su capacidad para mostrar modelos creados a partir de grandes conjuntos de datos. Estos casos permiten observar cómo las configuraciones visuales pueden ser utilizadas para representar modelos complejos.

## 3.1 Opciones de personalización para la visualización de modelos de árboles de decisión

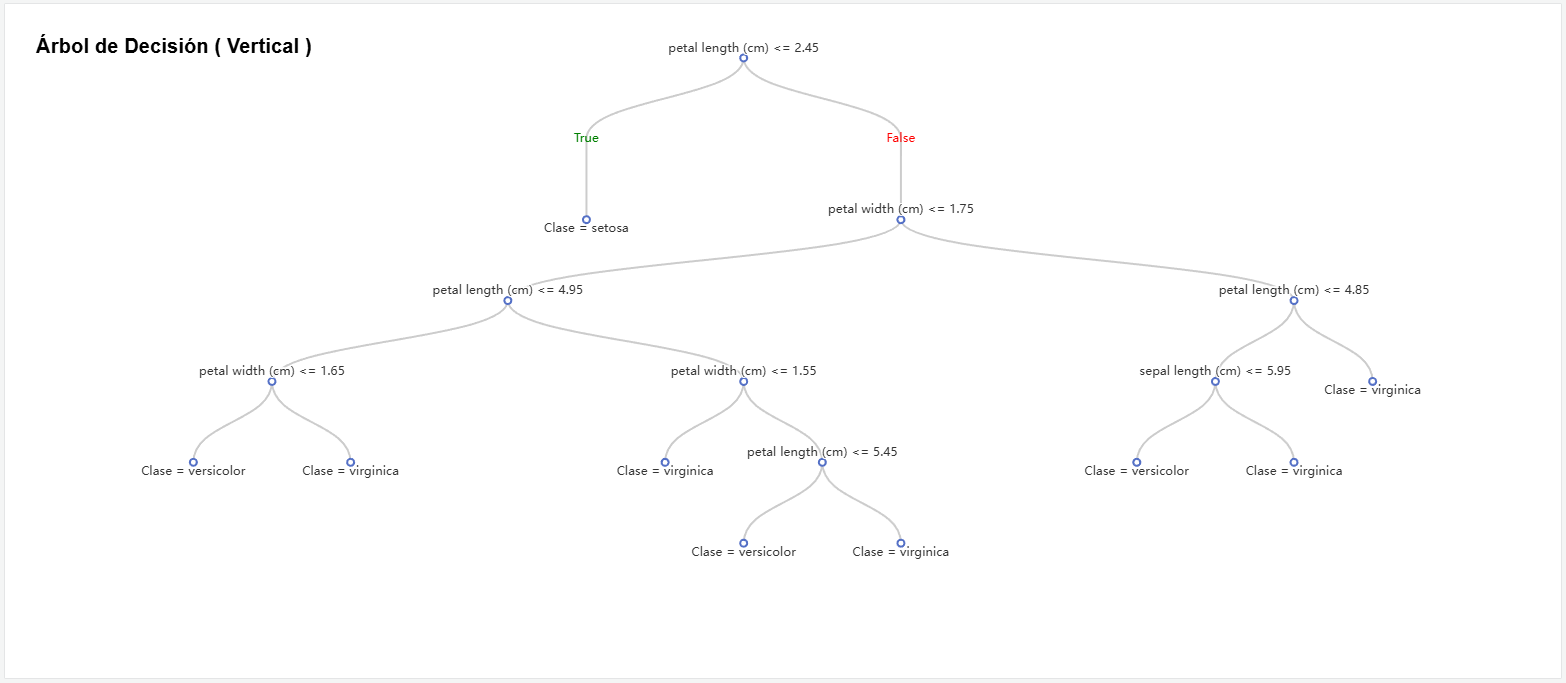
En el caso de los árboles de decisión, se muestran a continuación, las distintas variables que permiten adaptar las visualizaciones según desee el usuario.



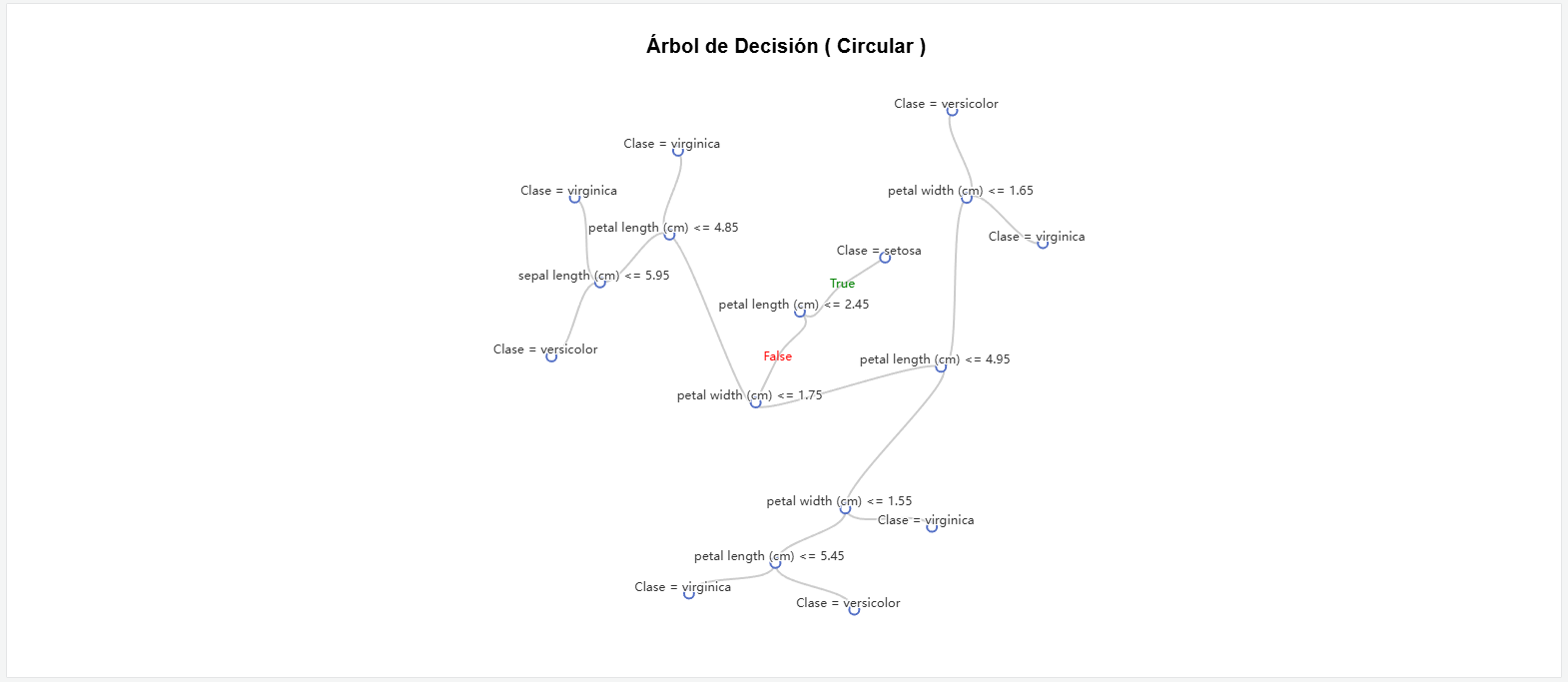
**Figura 43:** Variable en Grafana para mostrar las visualizaciones de modelos de árbol de decisión disponibles. Foto tomada por el autor de Grafana.

**Figura 44:** Variable en Grafana para mostrar los modos de visualización de modelos de árbol de decisión disponibles. Foto tomada por el autor de Grafana.

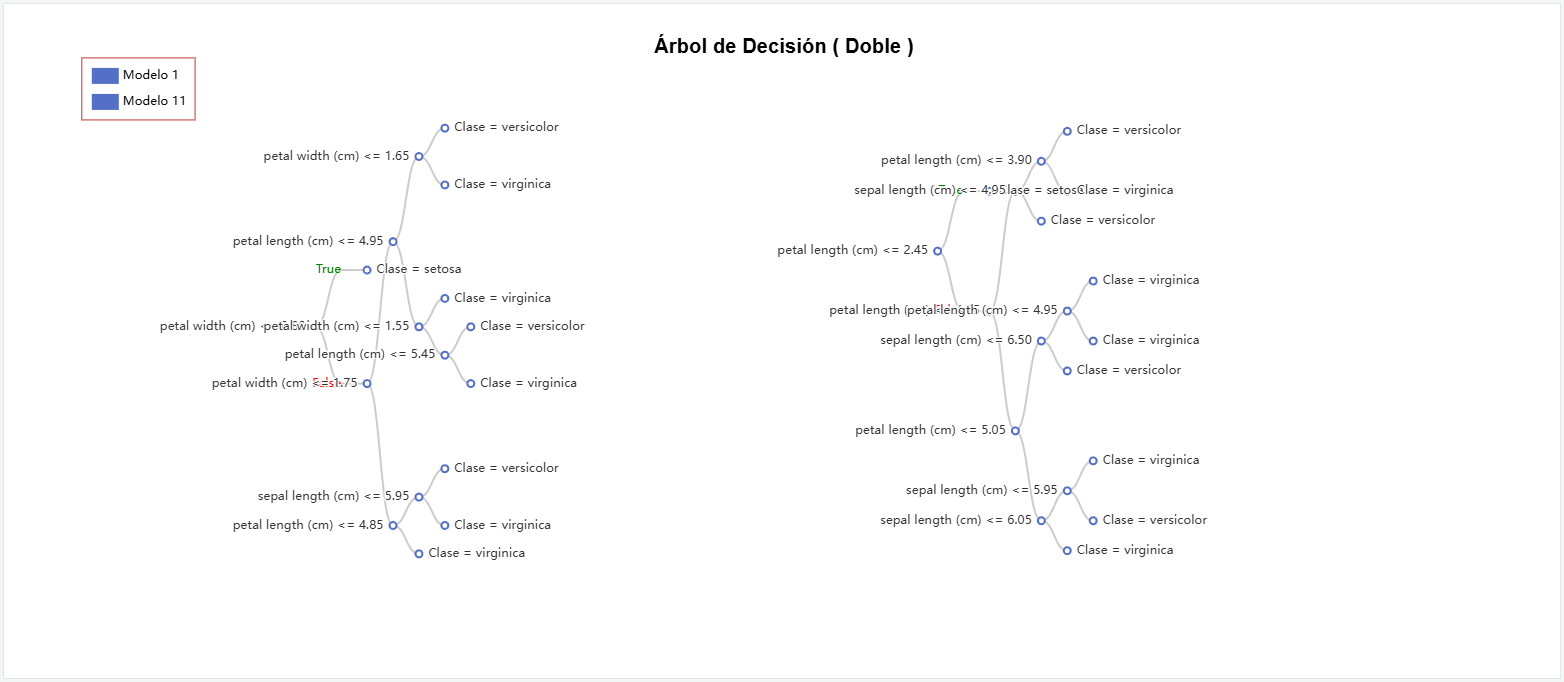
En la **Figura.43** se presentan las distintas formas de visualizar los modelos de árboles de decisión, destacando cuatro enfoques: Horizontal (ver **Figura.29**, **Figura.32**), donde las ramas del árbol se despliegan de izquierda a derecha, ofreciendo una perspectiva clara de las jerarquías y decisiones en un formato lineal; Vertical (ver **Figura.45**), que organiza los nodos de arriba hacia abajo, facilitando la interpretación desde la raíz hasta las hojas como un flujo descendente; Circular (ver **Figura.46**), que dispone los nodos en un diseño radial; y Doble(ver **Figura.47**), que permite comparar dos estructuras arbóreas en un solo esquema visual.



**Figura 45:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset iris. Foto tomada por el autor de Grafana.

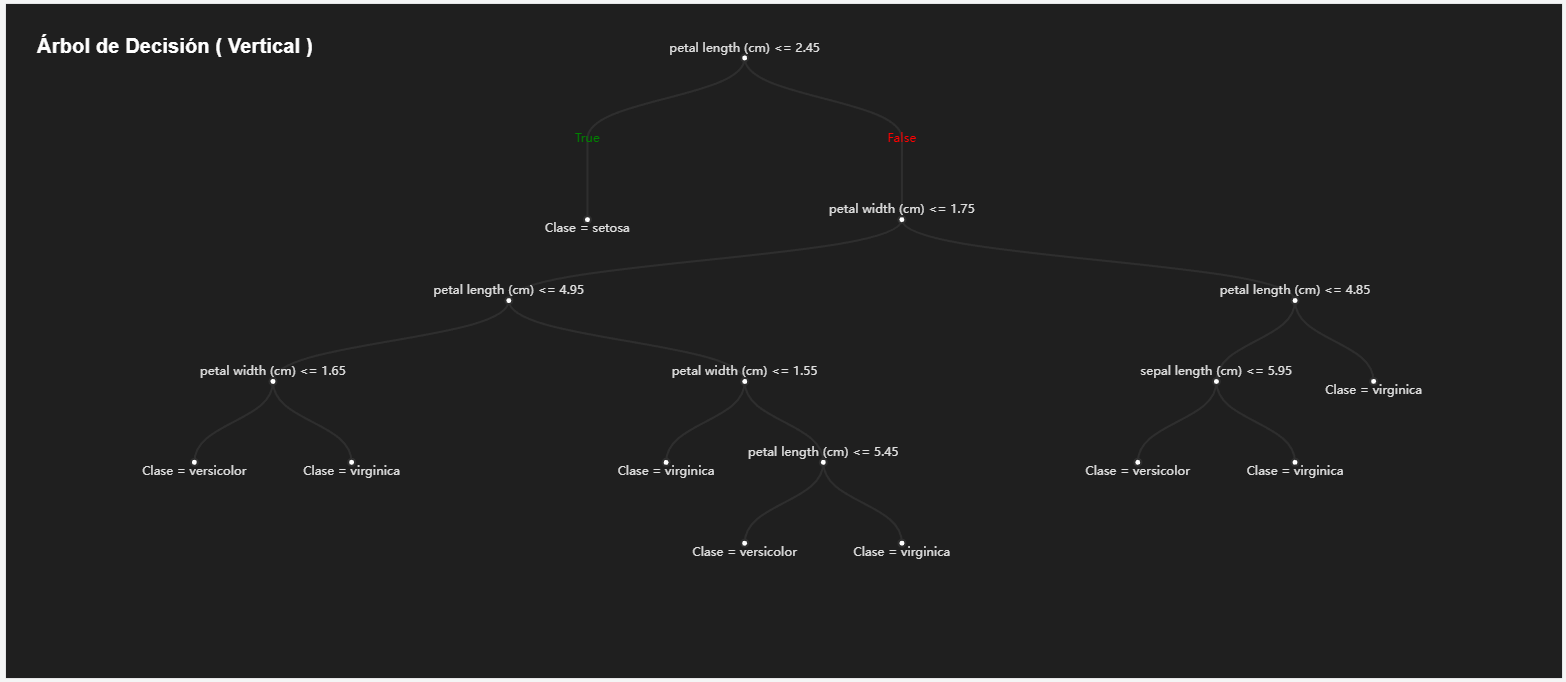


**Figura 46:** Visualización de modelo de árbol de decisión en forma circular modo claro con dataset iris. Foto tomada por el autor de Grafana.



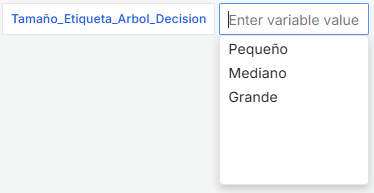
**Figura 47:** Visualización de modelo de árbol de decisión en forma doble modo claro: Modelo 1 - dataset iris, Modelo 11 - dataset iris (sin “sepal width (cm)” y “petal width (cm)”). Foto tomada por el autor de Grafana.

En la **Figura.44** se muestran los diferentes modos de visualización para representar los modelos de árboles de decisión: el modoClaro y el modo Oscuro. El modo Claro utiliza un esquema de colores con fondo claro y texto oscuro (ver **Figura.45**), mientras que el modo Oscuro invierte esta paleta, empleando fondo oscuro con texto claro (ver **Figura.48**).



**Figura 48:** Visualización de modelo de árbol de decisión en forma vertical modo oscuro con dataset iris. Foto tomada por el autor de Grafana.

También, fueron empleadas otras variables, “Tamanno\_Etiqueta\_Arbol\_Decision” y “Grosor\_Arista\_Arbol\_Decision” (ver **Figura.49**, **Figura.50**)

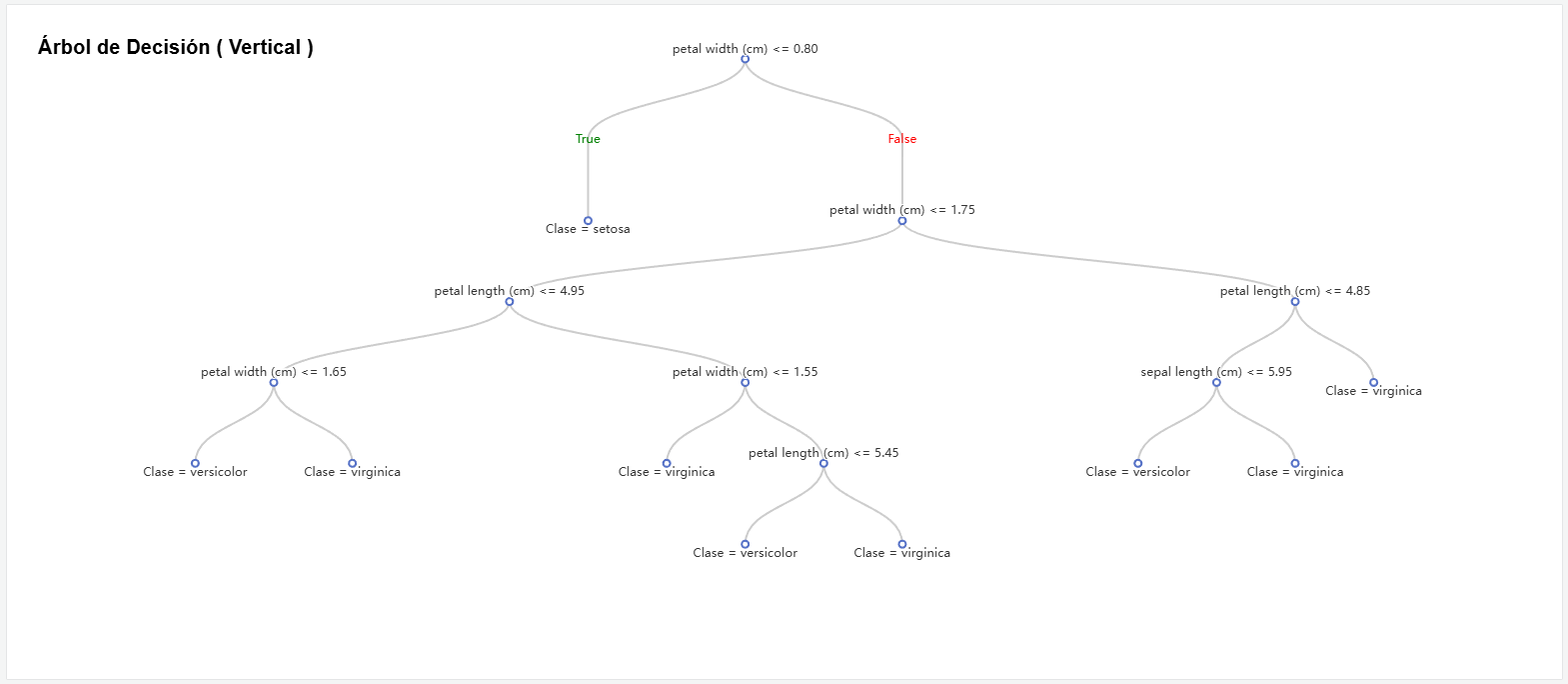


**Figura 49:** Variable en Grafana para mostrar los tamaños de las etiquetas de modelos de árbol de decisión disponibles. Foto tomada por el autor de Grafana.

**Figura 50:** Variable en Grafana para mostrar los grosores de aristas de modelos de árbol de decisión disponibles. Foto tomada por el autor de Grafana.

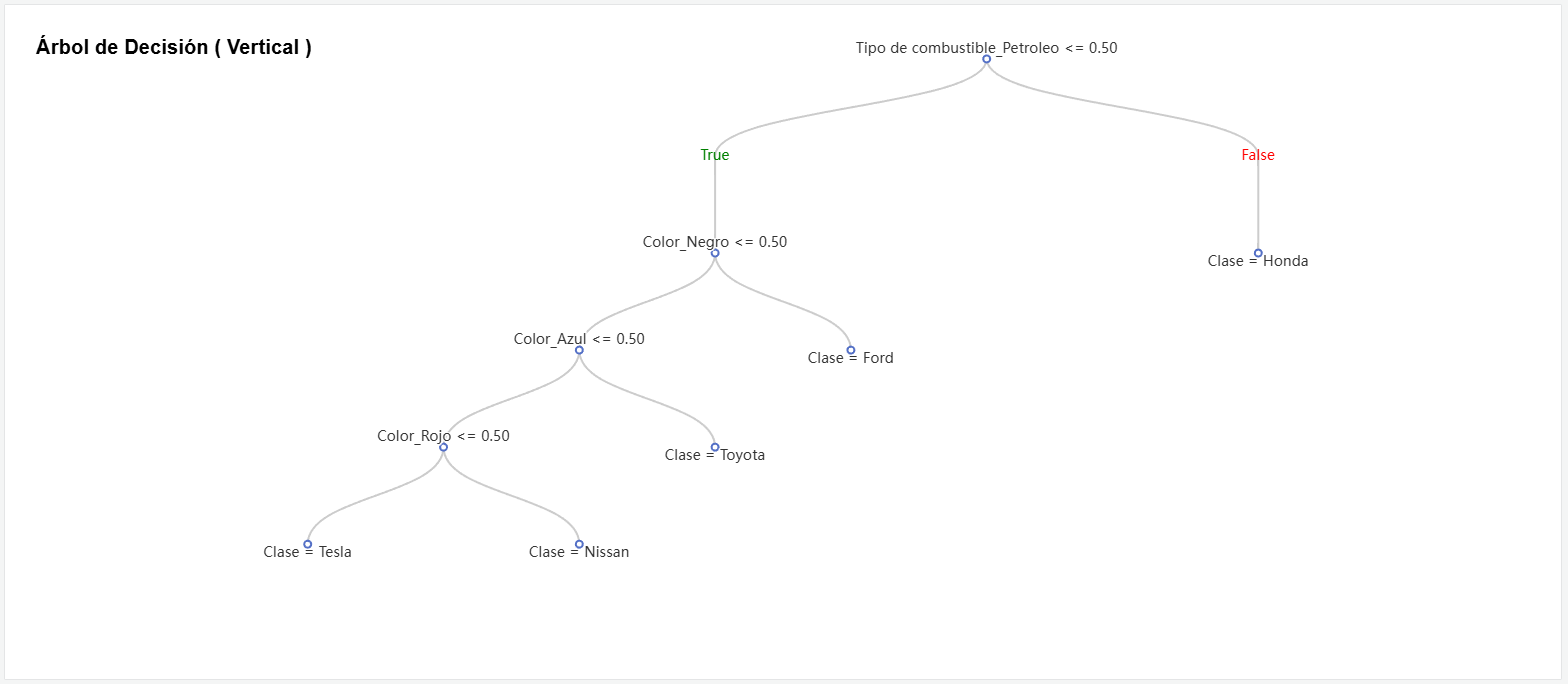
En la **Figura.49** se presentan las distintas opciones para ajustar el tamaño de las etiquetas:

* Pequeño (ver **Figura.51**): Utiliza etiquetas compactas para minimizar el espacio ocupado en la visualización, ideal para árboles con un gran número de nodos.



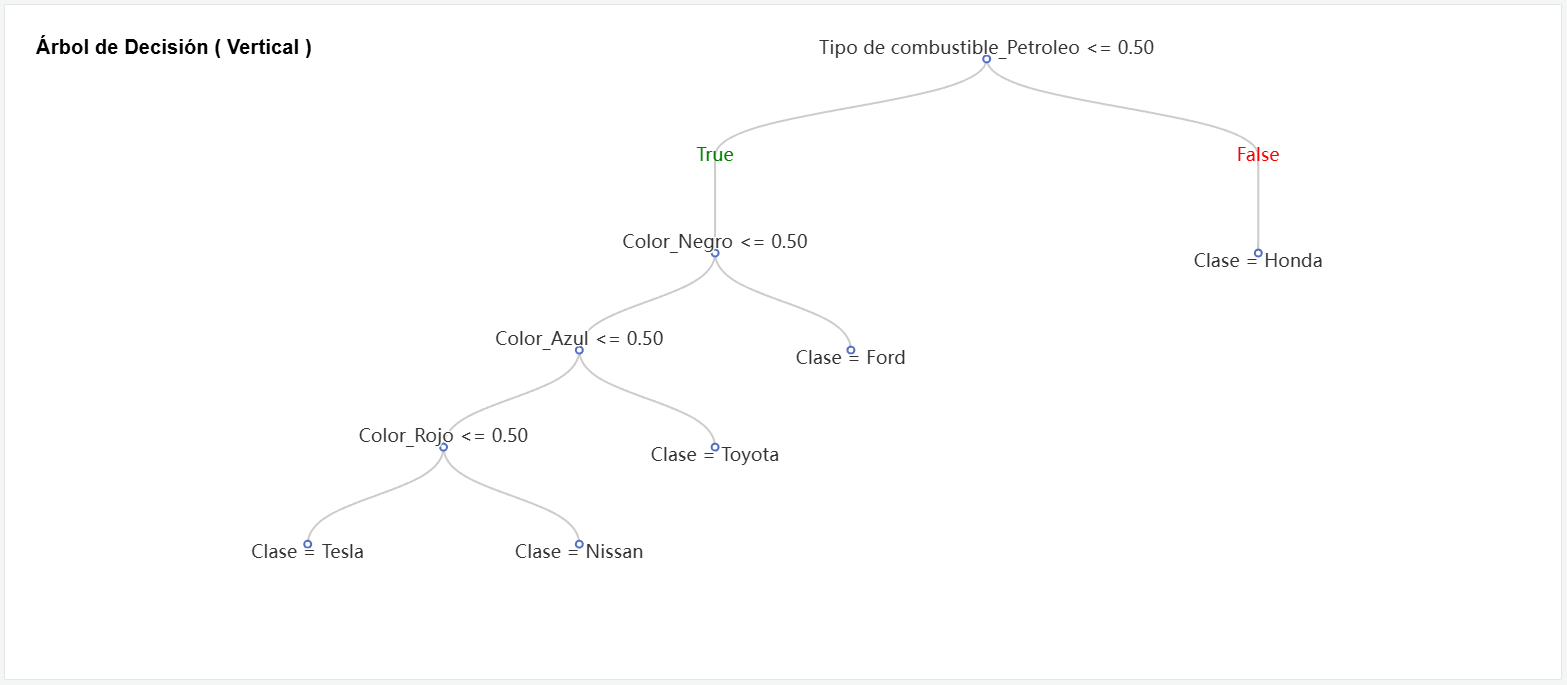
**Figura 51:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas pequeñas con dataset iris. Foto tomada por el autor de Grafana.

* Mediano (ver **Figura.52**): Ofrece un equilibrio entre legibilidad y espacio, siendo adecuado para la mayoría de los casos.

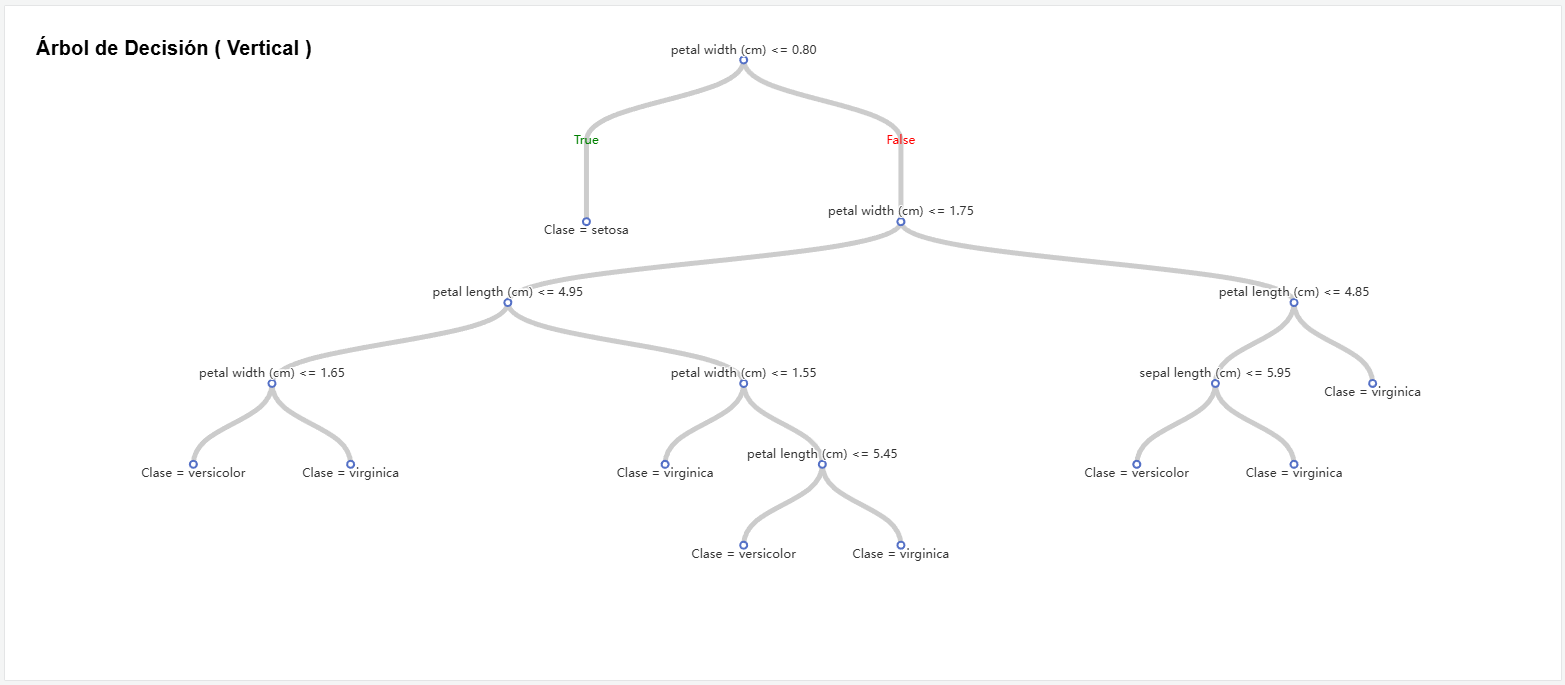


**Figura 52:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas medianas con dataset iris. Foto tomada por el autor de Grafana.

* Grande (ver **Figura.53**): Amplía significativamente las etiquetas para resaltar el contenido, especialmente útil en árboles con menos nodos donde la claridad textual es prioritaria.



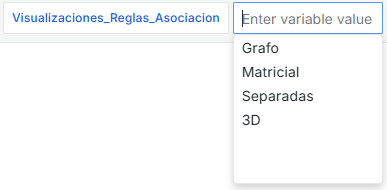
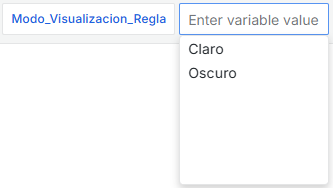
**Figura 53:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y etiquetas grandes con dataset iris. Foto tomada por el autor de Grafana.

En la **Figura.50** se muestran las diferentes opciones de visualización para ajustar el grosor de las aristas: el grosor Fino y el grosor Grueso. El grosor Fino utiliza líneas delgadas que destacan por su sutileza, permitiendo un enfoque en los nodos y etiquetas del árbol (ver **Figura.51**). Por otro lado, el grosor Grueso emplea líneas más anchas que resalta las conexiones entre los nodos, lo cual es ideal para enfatizar la estructura jerárquica del árbol (ver **Figura.54**).

**Figura 54:** Visualización de modelo de árbol de decisión en forma vertical, modo claro y aristas gruesas con dataset iris. Foto tomada por el autor de Grafana.

## 3.2 Opciones de personalización para la visualización de modelos de reglas de asociación

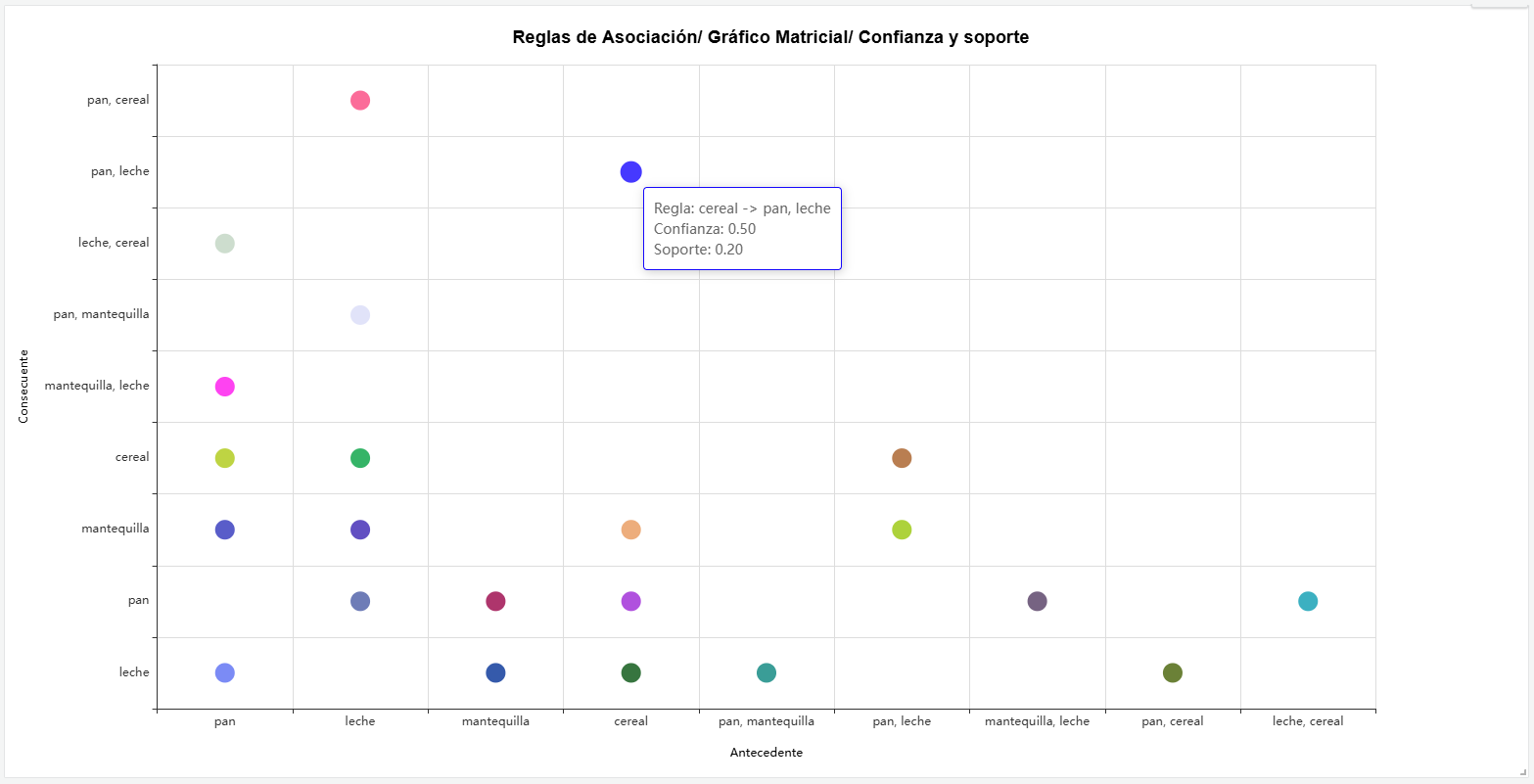
En cuanto a las reglas de asociación, se presentan a continuación las diferentes opciones disponibles para personalizar las visualizaciones.



**Figura 55:** Variable en Grafana para mostrar las visualizaciones de modelos de reglas de asociación disponibles. Foto tomada por el autor de Grafana.

**Figura 56:** Variable en Grafana para mostrar los modos de visualización de modelos de árbol de decisión disponibles. Foto tomada por el autor de Grafana.

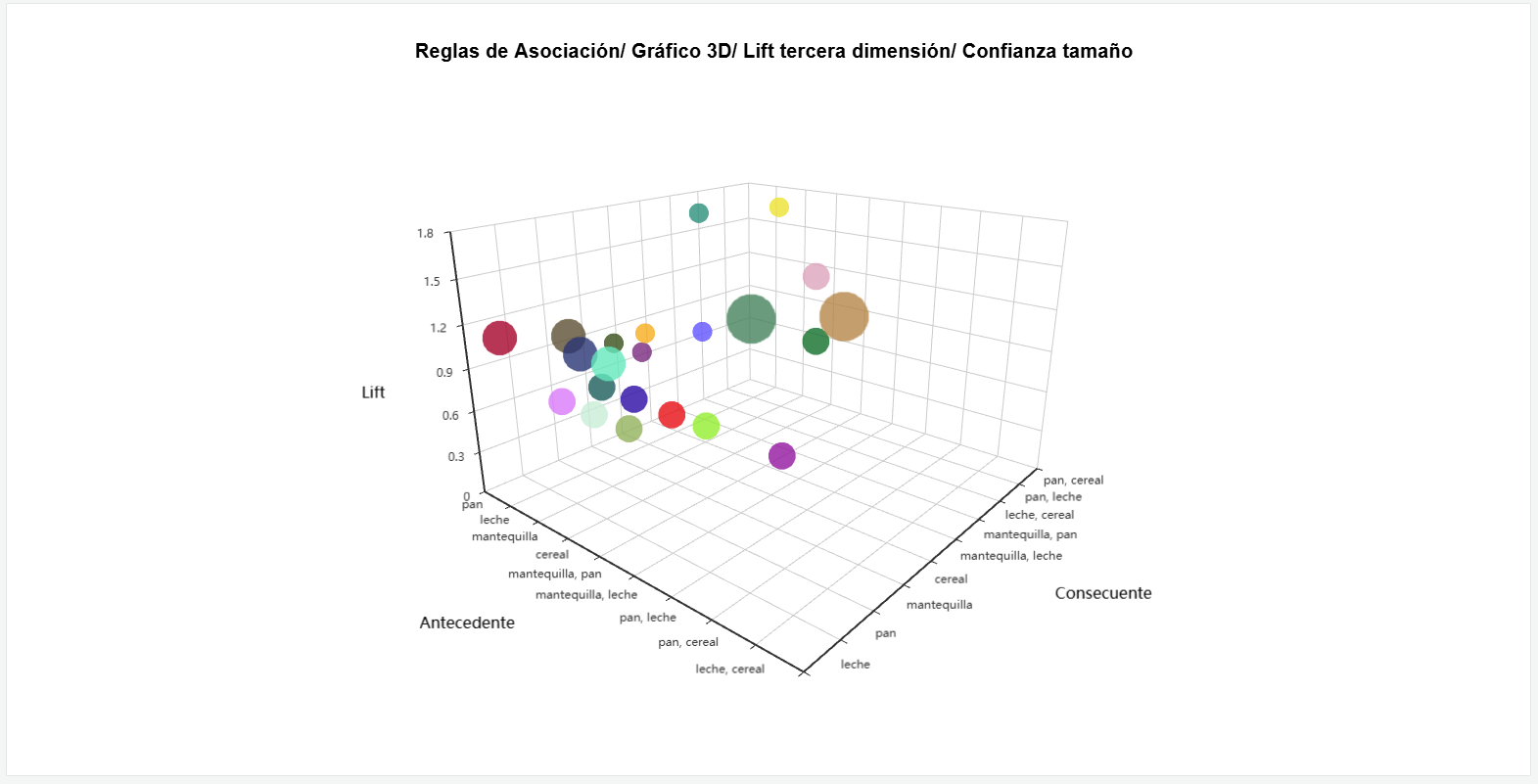
En la **Figura.55** se presentan las distintas formas de visualizar los modelos de reglas de asociación, destacando cuatro enfoques: Grafo (ver **Figura.38**, **Figura.42**), donde se utilizan nodos y aristas para mostrar las relaciones entre los elementos, representando los antecedentes y consecuentes como nodos conectados por flechas que indican la dirección de la regla, así como también nodos conectados por líneas discontinuas para representar relaciones de conjunción; Matricial (ver **Figura.57**), donde el eje “x” corresponde a los antecedentes y el eje “y” a los consecuentes. Los nodos dentro de la matriz simbolizan reglas específicas e incluyen métricas clave, como la confianza y el soporte; Separadas (ver **Figura.58**), que distribuye las reglas en grupos individuales, organizando los antecedentes y consecuentes de manera independiente; y 3D(ver **Figura.59**), que permite introducir una dimensión adicional como es el Lift, además de los antecedentes (eje x) y los consecuentes (eje y), y al igual que en la matricial, los nodos representan reglas específicas e incluyen métricas como la confianza y el soporte.



**Figura 57:** Visualización de modelo de reglas de asociación en forma de matriz modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

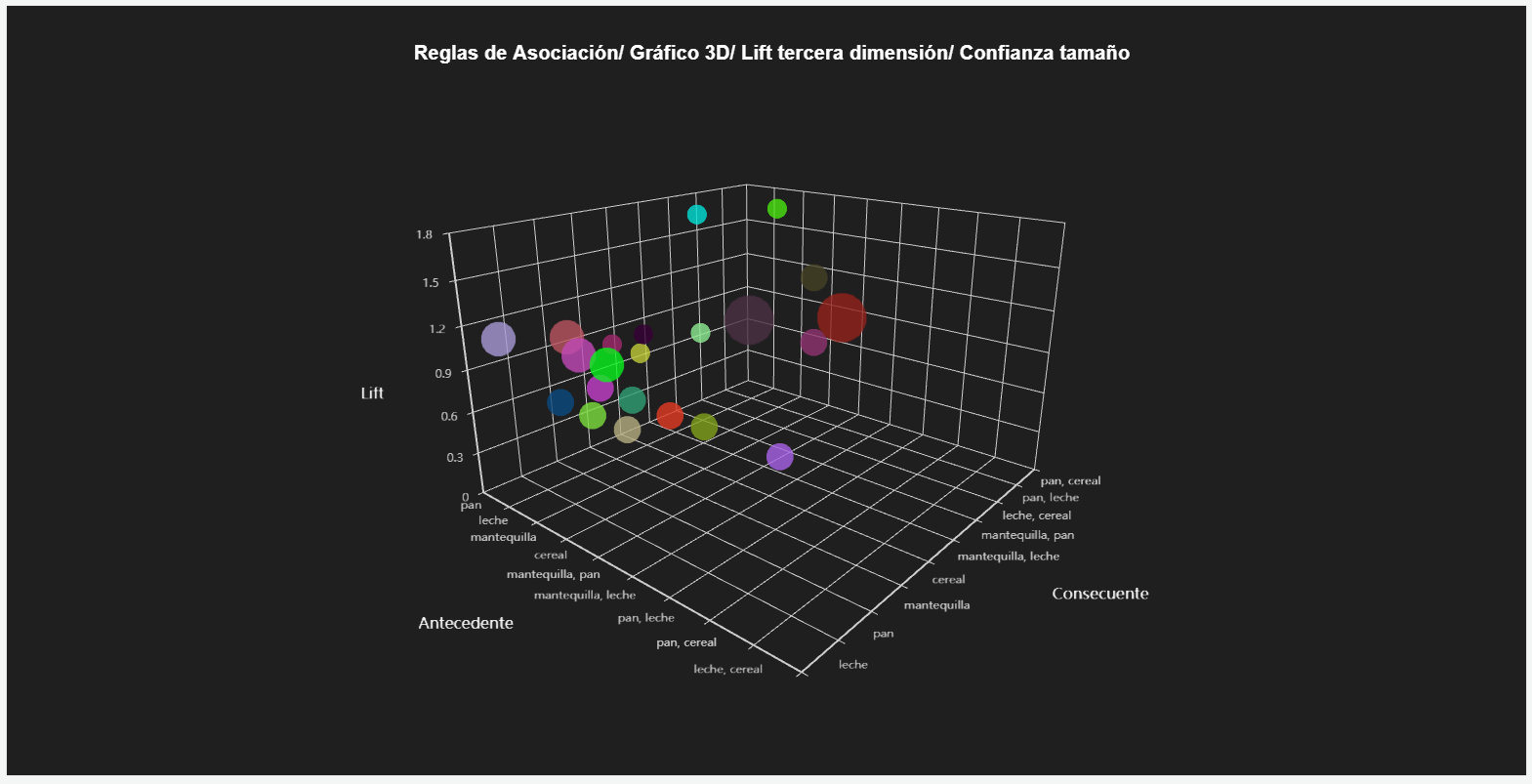


**Figura 58:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.



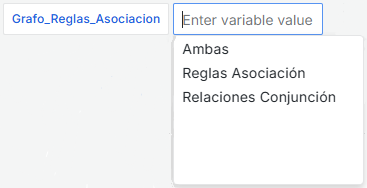
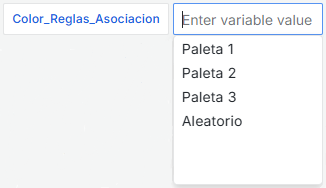
**Figura 59:** Visualización de modelo de reglas de asociación en forma 3D modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

En la **Figura.56** se muestran los diferentes modos de visualización para representar los modelos de reglas de asociación: el modoClaro y el modo Oscuro. El modo Claro utiliza un esquema de colores con fondo claro y texto oscuro (ver **Figura.59**), mientras que el modo Oscuro invierte esta paleta, empleando fondo oscuro con texto claro (ver **Figura.60**).



**Figura 60:** Visualización de modelo de reglas de asociación en forma 3D modo oscuro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

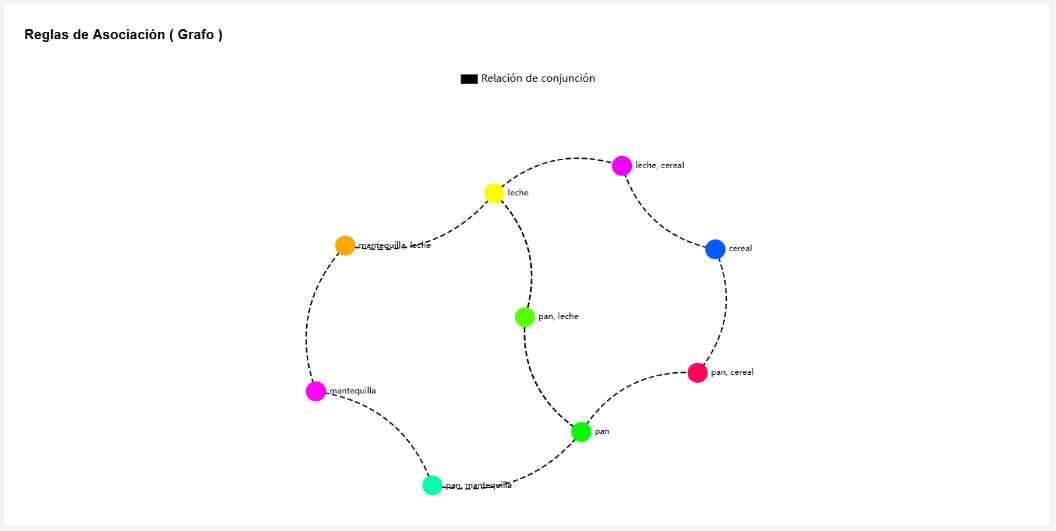
También, fueron empleadas otras variables, “Grafo\_Reglas\_Asociacion” y “Color\_Reglas\_Asociacion” (ver **Figura.61**, **Figura.62**).



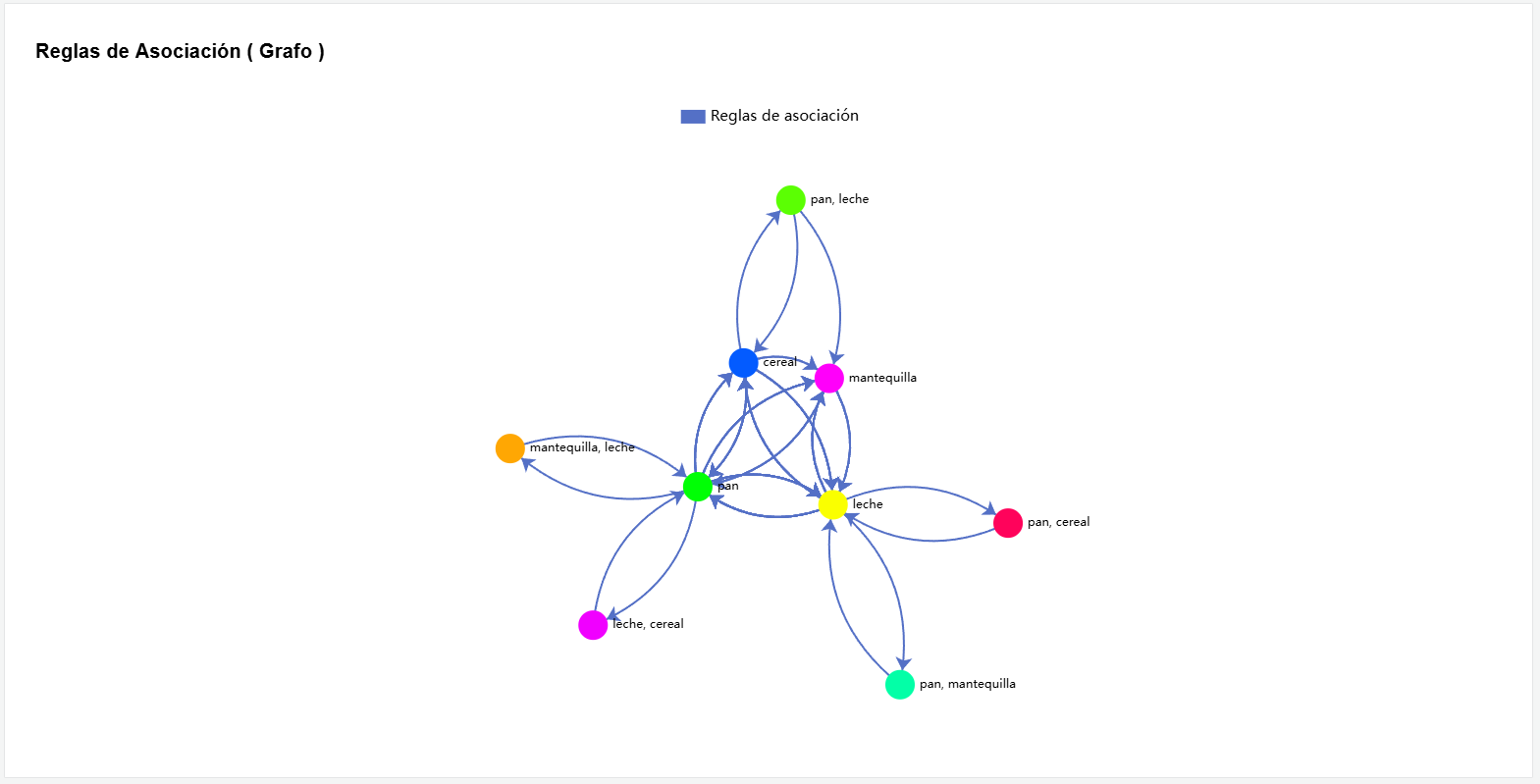
**Figura 61:** Variable en Grafana para mostrar las visualizaciones de grafos de modelos de reglas de asociación disponibles. Foto tomada por el autor de Grafana.

**Figura 62:** Variable en Grafana para mostrar los colores de modelos de reglas de asociación disponibles. Foto tomada por el autor de Grafana.

En la **Figura.61** se presentan las distintas opciones para ajustar las visualizaciones del grafo en las reglas de asociación: Ambas (ver **Figura.42**), que muestra tanto las relaciones de conjunción como las reglas de asociación, proporcionando una visión completa de las conexiones y las implicaciones entre los ítems; Relaciones Conjunción (ver **Figura.63**), visualiza únicamente las conexiones de conjunción entre los elementos, enfocándose en las relaciones entre los ítems (simples o compuestos); Reglas Asociación (ver **Figura.64**), resalta únicamente las reglas de asociación, mostrando las relaciones entre los antecedentes y consecuentes.

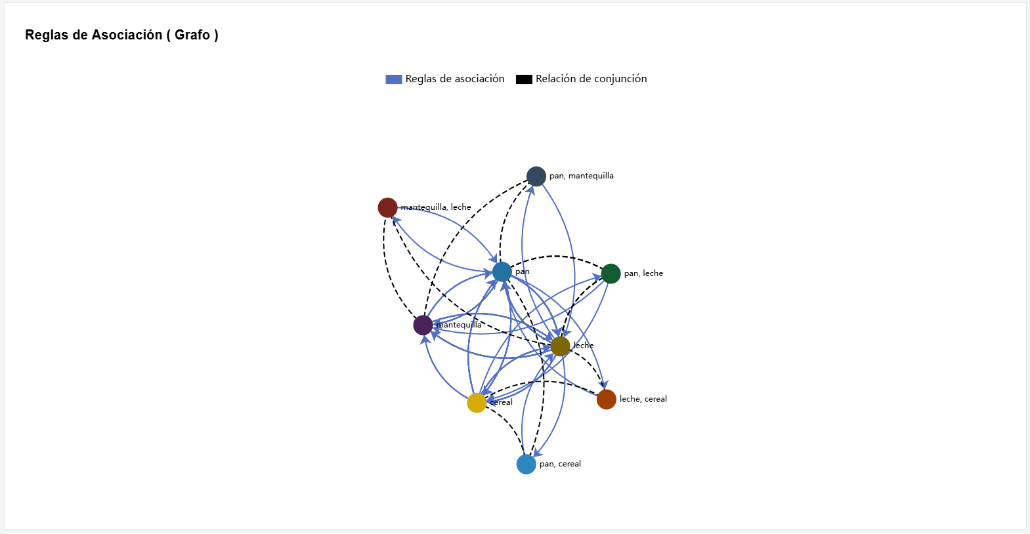


**Figura 63:** Visualización de modelo de reglas de asociación en forma de grafo (solo relaciones de conjunción) modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

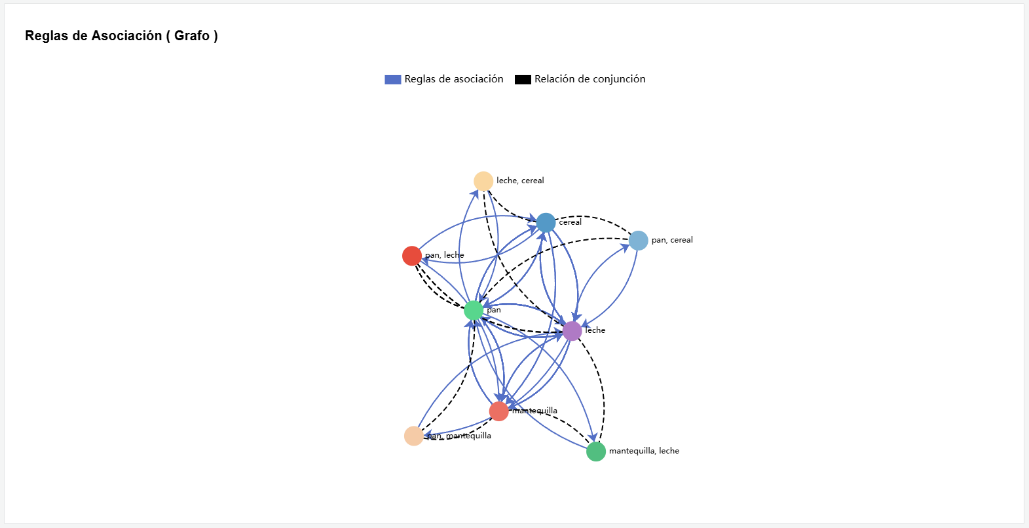


**Figura 64:** Visualización de modelo de reglas de asociación en forma de grafo (solo reglas de asociación) modo claro con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

En la **Figura.62** se muestran las distintas opciones para ajustar las paletas de colores en la visualización de las reglas de asociación con cuatro enfoques: Paleta 1 (ver **Figura.63**, **Figura.64**), utiliza colores saturados para resaltar las relaciones; Paleta 2 (ver **Figura.65**), emplea colores más neutros para lograr una visualización clara y equilibrada; Paleta 3 (ver **Figura.66**), utiliza tonos pasteles para una visualización más suave y discreta, lo cual evita la sobrecarga visual; y Aleatorio (ver **Figura.42**), que asigna colores de manera aleatoria a las reglas de asociación, proporcionando una variabilidad.



**Figura 65:** Visualización de modelo de reglas de asociación en forma de grafo modo claro (paleta 2) con dataset de transacciones de productos. Foto tomada por el autor de Grafana.



**Figura 66:** Visualización de modelo de reglas de asociación en forma de grafo modo claro (paleta 3) con dataset de transacciones de productos. Foto tomada por el autor de Grafana.

## 3.3 Caso de prueba 1

El siguiente caso de prueba se basa en un conjunto de datos, el cual corresponde a una encuesta realizada a personas según su estadía en un determinado hotel. Este cuenta con 37 columnas y un total de 8842 filas, las cuales hacen referencia a cada encuesta realizada.

Las columnas se dividen en varias categorías:

* Datos personales de los clientes: Estas contienen valores descriptivos de las personas que opinan, como lo son: sexo, país, edad o cualquier otro valor.

Columnas: **Q00001**, **Q00002**, **Q00003**, **Q00004**.

* Valoraciones específicas de servicios: Estas contienen la valoración de las personas en cuento a los servicios del hotel. Puede ser 1 (positiva), 0 (neutra) y -1 (negativa).

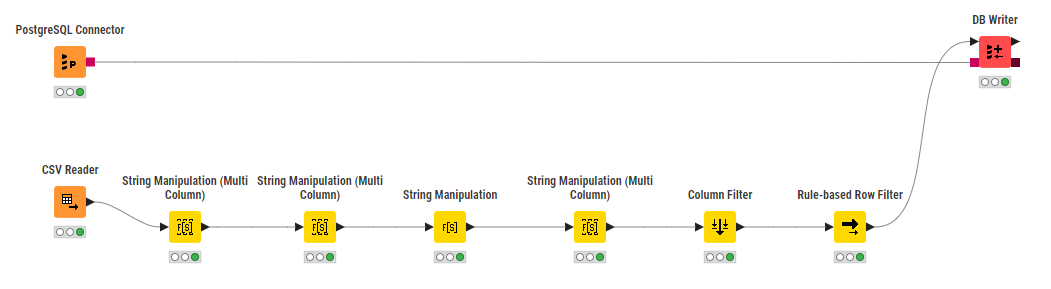
Columnas: **FrontDesk01[G02Q05]**, **FrontDesk01[G02Q06]**, **Room01[G03Q10]**, ……, **Buffet01[G04Q11]**, **Pool01[G10Q31]**, **Suggest01**.

* Campos adicionales: Estas contienen otros campos que contribuyen en la encuesta.

Columnas: **idpolo**, **dia\_de\_semana**, **mes**, **dia\_del\_mes**.

### 3.3.1 Preprocesamiento del dataset de encuestas

Debido a que existen columnas con todos sus valores faltantes, se tomó la decisión de hacer un preprocesamiento en la herramienta Knime (ver **Figura.67**), la cual facilita este proceso mediante el uso de nodos, con el fin de eliminar dichas columnas que no aportan ninguna información para el entrenamiento de los modelos. También, se decidió obviar las filas que tuvieran valores faltantes, lo cual no implicó una gran pérdida de datos pues inicialmente había 8842 filas y luego quedaron 7043, esto no significa una gran diferencia dado el gran volumen da datos que se está analizando.



**Figura 67:** Flujo en Knime para el preprocesamiento del dataset de encuestas. Foto tomada por el autor de Knime.

Como se muestra en la **Figura.67**, además de los ajustes mencionados anteriormente se realizaron otros, los cuales se detallan a continuación en la descripción del flujo realizado:

1. **CSV Reader**

* Este nodo se utilizó para leer los datos de las encuestas realizadas, provenientes de un archivo **CSV**.

1. **String Manipulation (Multi Column)** *(Reemplazo de "NULL")*

* Se utilizó con el fin de reemplazar las cadenas con valores "NULL" por celdas vacías, lo cual fue necesario, pues, al intentar convertir las columnas de tipo String (FrontDesk01[G02Q05], FrontDesk01[G02Q06], Room01[G03Q10], ……, Buffet01[G04Q11], Pool01[G10Q31], Suggest01) a tipo Integer, el valor "NULL" como texto no es permitido.

1. **String Manipulation (Multi Column)** *(Conversión de vacías a null)*

* Se cambiaron las celdas vacías por valores **null**, lo cual garantizó que los campos vacíos se interpretaran correctamente durante las operaciones posteriores.

1. **String Manipulation** *(Reemplazo de valores en la columna "Suggest")*

* Se reemplazaron los valores existentes en la columna **"Suggest"** por números, lo que facilitó la interpretación numérica de los datos, dándoles un sentido específico para su análisis posterior.

1. **String Manipulation (Multi Column)** *(Conversión a tipo entero)*

* Se transformó el tipo de datos de las columnas seleccionadas (FrontDesk01[G02Q05], FrontDesk01[G02Q06], Room01[G03Q10], ……, Buffet01[G04Q11], Pool01[G10Q31], Suggest01) de valoraciones de los servicios de String a Integer, pues, de esta manera se garantiza la correcta interpretación de los datos.

1. **Column Filter** *(Eliminación de columnas con valores faltantes completos)*

* Se eliminaron las columnas que contienen únicamente valores faltantes (Buffet01[G04Q11], Buffet01[G04Q12], Buffet01[G04Q13]).

1. **Rule-based Row Filter** *(Filtrado de filas con celdas vacías)*

* Se establecieron reglas para filtrar las filas donde exista al menos una celda vacía, por lo que todas las filas que contienen valores faltantes se eliminan del conjunto de datos.

1. **PostgreSQL Connector**

* Este nodo permitió establecer la conexión con la base de datos PostgreSQL.

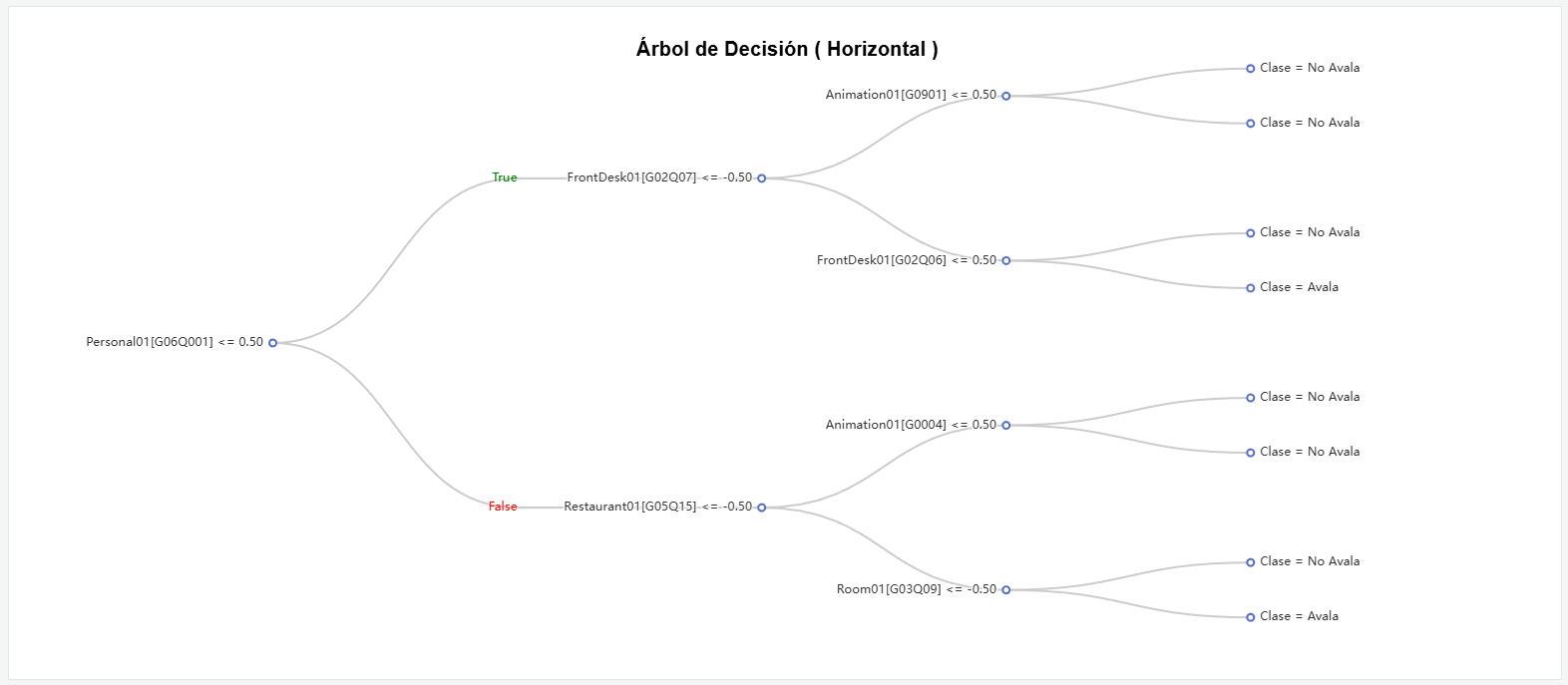
1. **DB Writer**

* Por último, se utilizó este nodo para almacenar los datos procesados y limpios en la base de datos PostgreSQL.

### 3.3.2 Árboles de decisión

A partir de los datos preprocesados ​​en Knime, donde se eliminaron columnas y filas con valores faltantes, se procedió a crear un modelo de Árbol de Decisión. Para la creación de este, solo se tuvo en cuenta las columnas relacionadas con las valoraciones de los servicios del hotel, es decir las características del modelo están definidas por cada columna que hace referencia a los servicios exceptuando “Suggest”, la cual se utilizó como variable objetivo y que tiene como fin avalar o no el hotel al cual hace referencia la encuesta.

Una vez que se ejecutó el algoritmo para crear y entrenar el modelo de árbol de decisión, a partir de los datos definidos, se pudo visualizar en la herramienta Grafana y este fue el resultado (ver **Figura.68**). Para lograr una representación más clara y comprensible del árbol, fue necesario ajustar la profundidad máxima a 3 niveles y aplicar un peso a las clases con el fin de garantizar un balance adecuado en el modelo debido a que los datos presentaban un desequilibrio en la distribución de las clases, es decir, algunas categorías estaban representadas con mayor frecuencia que otras. Este desequilibrio conlleva a que el modelo favorezca las clases mayoritarias al realizar predicciones, lo que reduce su capacidad para identificar correctamente las clases minoritarias.



**Figura 68:** Visualización de modelo de árbol de decisión en forma horizontal modo claro con dataset de encuestas. Foto tomada por el autor de Grafana.

Este modelo permite:

* Identificar servicios clave: Determinar cuáles de los servicios evaluados (recepción, habitaciones, personal, animación, entre otros) tienen mayor impacto en las valoraciones positivas, neutras o negativas de los clientes.
* Facilitar la toma de decisiones: Proporcionar información concreta y visualmente intuitiva para que los responsables del hotel puedan implementar estrategias enfocadas en mejorar los servicios que más afectan la percepción de los clientes.
* Priorizar áreas de mejora: Ayudar a identificar qué servicios tienen el mayor impacto en las valoraciones para enfocar los esfuerzos de mejora en las áreas críticas.

Observaciones:

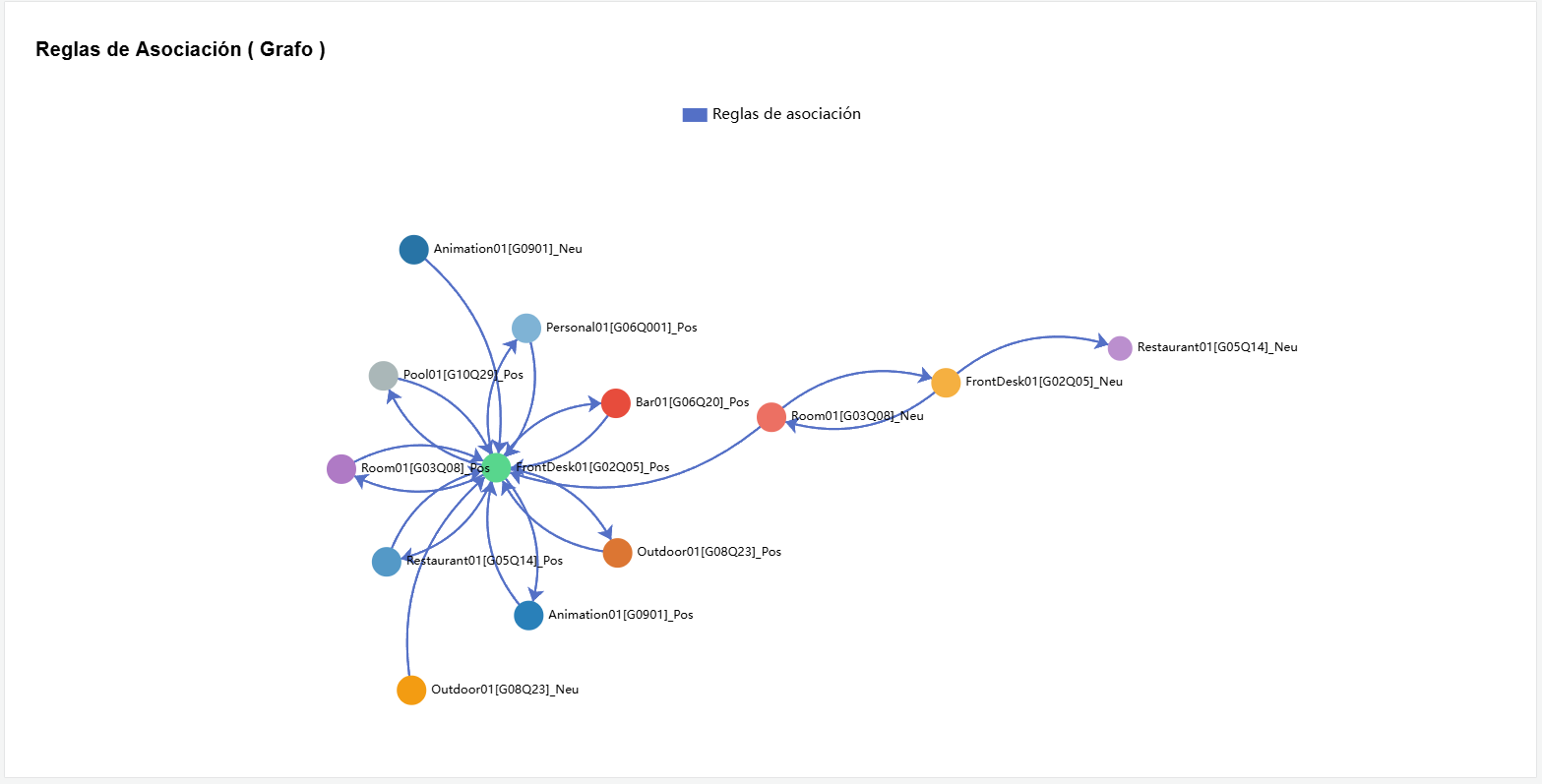
* Un cliente que otorga una valoración > 0.50 (positiva) al personal (Personal01[G06Q001]), una valoración > -0.50 (positiva o neutra) al restaurante (Restaurant01[G05Q15]) y una valoración de > -0.50 (positiva o neutra) a la habitación (Room01[G03Q09]) tiende a avalar el hotel, lo que sugiere que experiencias satisfactorias en el personal, el restaurante y la habitación son clave para una percepción global positiva del hotel.
* Un cliente que otorga valoraciones de ≤ -0.50 (negativas) a la recepción (FrontDesk01[G02Q07]) y valoraciones de ≤ 0.50 (neutra) a la animación (Animation01[G0901]) suele tener una percepción general negativa del hotel (No lo avala). Esto indica que ambos servicios son fundamentales para garantizar una experiencia positiva.

### 3.3.3 Reglas de asociación

En el caso de las reglas de asociación, para la creación del modelo, se seleccionaron 8 columnas correspondientes a las valoraciones de los servicios del hotel. La elección de esta cantidad de columnas se debe a consideraciones técnicas, pues al incluir un mayor número de características incrementaba significativamente el tiempo de ejecución del algoritmo y, en algunos casos, provocaba fallos en el sistema debido a limitaciones de memoria RAM.

A diferencia del modelo de Árbol de Decisión, en esta sección no se utilizó la columna "Suggest" como variable objetivo, pues el enfoque de las reglas de asociación, en este caso, es descubrir relaciones frecuentes entre las valoraciones de los servicios, sin necesidad de predefinir una variable dependiente. En lugar de visualizar directamente las valoraciones, este modelo revela las relaciones existentes entre los ítems (servicios) con base en su co-ocurrencia en las encuestas.

Tras ejecutar el algoritmo para crear y entrenar el modelo de reglas de asociación con los datos seleccionados, los resultados fueron visualizados en la herramienta Grafana, tal como se muestra en la **Figura.69**. Para garantizar una visualización clara y evitar la sobrecarga del panel, fue necesario limitar la consulta a las 20 primeras reglas mediante un LIMIT en la configuración.



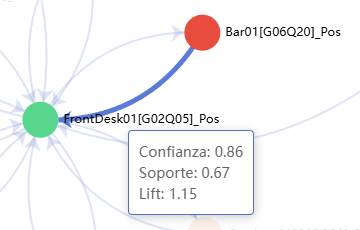
**Figura 69:** Visualización de modelo de reglas de asociación en forma de grafo (solo reglas de asociación) modo claro con dataset de encuestas. Foto tomada por el autor de Grafana.

Este modelo permite:

* Descubrir patrones frecuentes: Encontrar combinaciones de servicios que se mencionan de manera recurrente, lo cual permite detectar tendencias en las preferencias o menciones de los clientes.
* Ofrecer reglas interpretables: Generar reglas del tipo *“Si el servicio A es mencionado, entonces es probable que también se mencione el servicio B”*, facilitando la interpretación y el análisis de los resultados.
* Optimizar estrategias de mejora: Utilizar las relaciones identificadas para diseñar estrategias de mejora enfocadas en servicios que suelen estar conectados.

Observaciones:

* Por ejemplo, si un cliente da una valoración positiva con respecto al bar existe un 86% de probabilidad de que también valore de forma positiva la recepción (ver **Figura.70**).



**Figura 70:** Regla de asociación asociada a la valoración emitida por los clientes acerca de los servicios de un hotel durante su estadía. Foto tomada por el autor de Grafana.

* El servicio de recepción tiene una alta frecuencia de aparición en las valoraciones de la encuesta, lo que sugiere que es un aspecto central en la experiencia del cliente.

## 3.4 Caso de prueba 2

El segundo escenario de prueba, se apoya en un conjunto de datos que corresponde a comentarios emitidos por clientes en relación con su experiencia en hoteles. Este cuenta con 29 columnas y un total de 43 443 filas, las cuales representan cada comentario.

Las columnas se dividen en varias categorías:

* Palabras clave en comentarios (Binarias): Estas indican la presencia (**1**) o ausencia (**0**) de palabras específicas en los comentarios. Cada palabra representa un aspecto relevante de la experiencia del cliente.

Columnas: **hotel**, **resort**, **staff**, **beach**, **cuba**, …..., **buffet**, **excelente**.

* Valoraciones del comentario: Describen la evaluación del comentario en base a percepciones humanas o modelos automáticos.

Columnas:

* **val\_h**: Valoración realizada por humanos sobre el comentario

(1: Positiva, **0**: Neutra, -1: Negativa).

* **val\_llm**: Valoración realizada por un modelo de lenguaje (LLM)

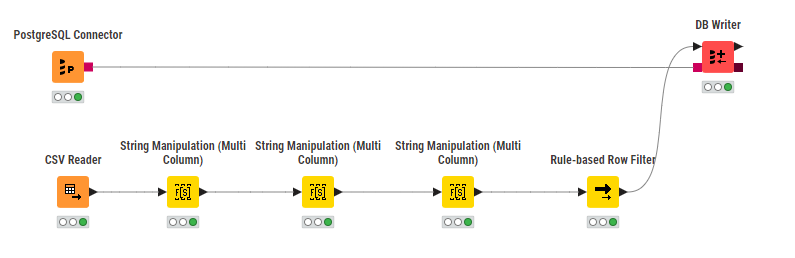
(1: Positiva, **0**: Neutra, -1: Negativa).

* Atributos descriptivos adicionales: Describen información adicional sobre el contexto del comentario y el perfil del hotel.

Columnas: **l\_comentario**, **polo**, **modality**, **segment**, **dia\_de\_semana**, **mes**, **dia\_del\_mes**.

### 3.4.1 Preprocesamiento del dataset de comentarios emitidos

Al igual que en el dataset anterior (sección 3.3.1), se decidió obviar las filas con valores faltantes, para ello se tomó la decisión de realizar un preprocesamiento en la herramienta Knime (ver **Figura.71**), la cual facilita este proceso mediante el uso de nodos, con el fin de eliminar dichas filas que no aportan información completa para el análisis. Como resultado, se descartaron las filas con valores incompletos, lo cual no implicó una gran pérdida de datos, ya que inicialmente había 43 443 filas y luego quedaron 41 492. Esta diferencia no representa un impacto significativo debido al gran volumen de datos analizados.



**Figura 71:** Flujo en Knime para el preprocesamiento del dataset de comentarios emitidos. Foto tomada por el autor de Knime.

Como se muestra en la **Figura.71**, además de la eliminación de las filas con valores faltantes se realizaron otros ajustes, los cuales se detallan a continuación en la descripción del flujo realizado:

1. **CSV Reader**

* Este nodo se utilizó para leer los datos de los comentarios emitidos, provenientes de un archivo **CSV**.

2. **String Manipulation (Multi Column)** *(Reemplazo de "NULL")*

* Se utilizó con el fin de reemplazar las cadenas con valores "NULL" por celdas vacías, lo cual fue necesario pues al intentar convertir las columnas de tipo String (l\_comentario, val\_llm) a tipo Integer, el valor "NULL" como texto no es permitido.

3. **String Manipulation (Multi Column)** *(Conversión de vacías a null)*

* Se cambiaron las celdas vacías por valores **null**, lo cual garantizó que los campos vacíos se interpretaran correctamente durante las operaciones posteriores.

4. **String Manipulation (Multi Column)** *(Conversión a tipo entero)*

* Se transformó el tipo de datos de las columnas seleccionadas (l\_comentario, val\_llm) de String a Integer, pues era necesario garantizar que los valores fueran interpretados como enteros.

5. **Rule-based Row Filter** *(Filtrado de filas con celdas vacías)*

* Se establecieron reglas para filtrar las filas donde exista al menos una celda vacía, por lo que todas las filas que contienen valores faltantes se eliminan del conjunto de datos.

6. **PostgreSQL Connector**

* Este nodo se utilizó para establecer la conexión con la base de datos PostgreSQL.

7. **DB Writer**

* Finalmente, se almacenaron los datos limpios y procesados en la base de datos PostgreSQL para su análisis y uso posterior.

### 3.4.2 Árboles de decisión

A partir de los datos preprocesados ​​en Knime, donde se eliminaron filas con valores faltantes, se procedió a crear dos modelos de árbol de decisión. Para la creación del 1er modelo, se consideraron las columnas relacionadas con las palabras claves en comentarios, así como el contenido textual del comentario (l\_comentario) y los meses en el que se realizó. La variable objetivo seleccionada fue “val\_h”, la cual representa la valoración realizada por humanos (1: Positiva, 0: Neutra, -1: Negativa).

Una vez ejecutado el algoritmo para crear y entrenar el modelo de Árbol de Decisión, utilizando las características definidas, se pudo visualizar en la herramienta Grafana y este fue el resultado (ver **Figura.72**). Para lograr una representación más clara y comprensible, fue necesario:

* Ajustar la profundidad máxima del árbol a 3 niveles, con el fin de facilitar la interpretación de los resultados.
* Aplicar peso a las clases para garantizar un balanceo adecuado entre las categorías de la variable objetivo (como mismo ocurre en la sección 3.3.2).



**Figura 72:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset de comentarios emitidos: “val\_h” como variable objetivo. Foto tomada por el autor de Grafana.

Este modelo permite:

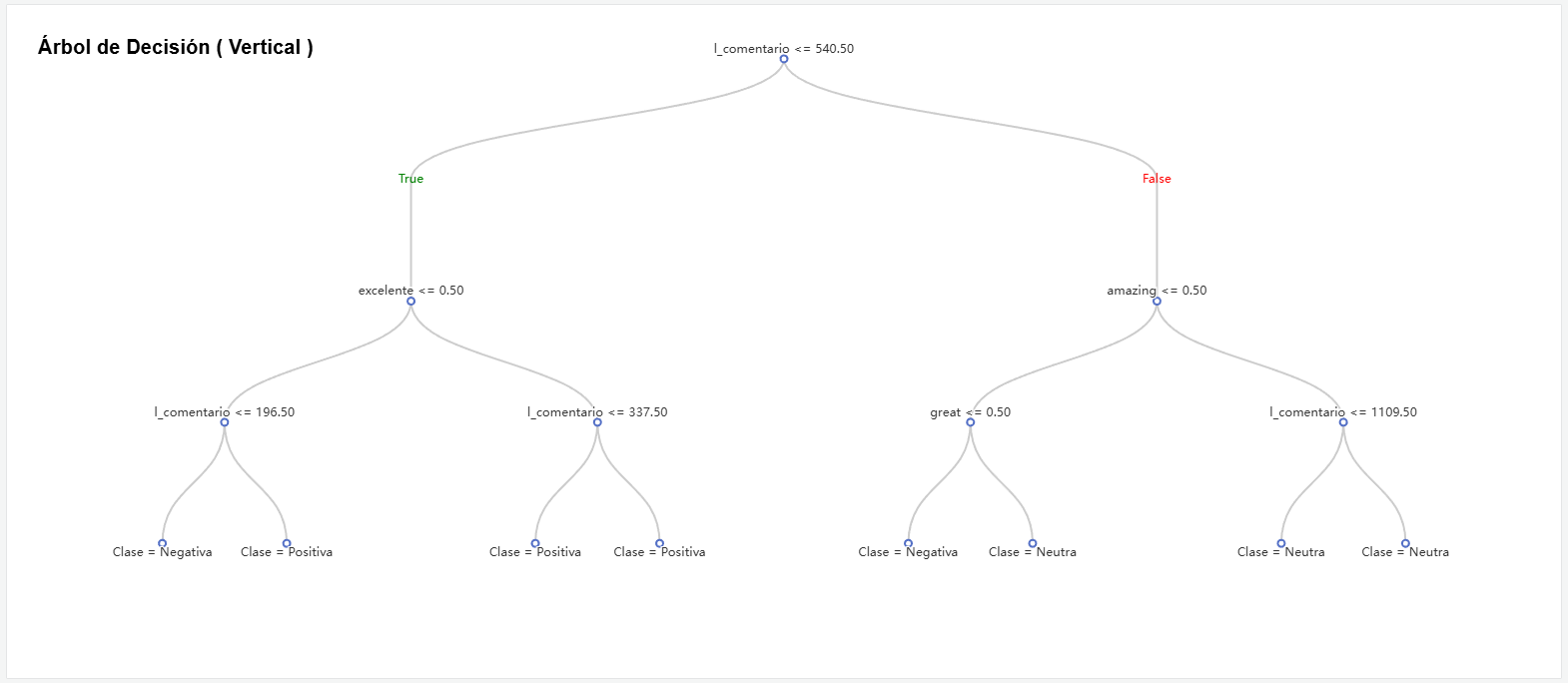
* Identificar factores claves: Determinar qué palabras claves y otros atributos, como el mes o el contenido del comentario (l\_comentario), tienen un mayor impacto en las valoraciones humanas.
* Comprensión de Patrones:Visualizar de forma estructurada cómo las características seleccionadas afectan la percepción de los comentarios. El modelo ofrece una representación clara que guía a una valoración específica, facilitando el entendimiento de los factores determinantes.
* Apoyo en la Gestión de Opiniones: Proporcionar información útil que permita identificar palabras claves o periodos críticos en los que se producen opiniones negativas o positivas, lo cual facilita la toma de decisiones para mejorar la experiencia del cliente.

Observaciones:

* Si el comentario no contiene la palabra "excelente", pero tiene más de “196 letras”, suele ser clasificado como “Positivo”, lo cual indica que la longitud del comentario puede reflejar una percepción favorable, incluso sin palabras muy marcadas.
* La ausencia de la palabra clave "great" en comentarios con más de “520letras” suele derivar en clasificaciones “Neutras”o“Negativas”, dependiendo de otros términos como "bien". Esto sugiere que términos positivos en inglés, como "great", pueden tener un peso significativo en las valoraciones humanas, especialmente en comentarios extensos.

Para la creación del 2do modelo, se consideraron las mismas columnas utilizadas en el modelo anterior, cambiando solamente la variable objetivo, que para este caso fue “val\_llm**”**, la cual representa la valoración realizada por un modelo de lenguaje (1: Positiva, 0: Neutra, -1: Negativa).

Al igual que en el 1er modelo se aplicaron los mismos ajustes, tanto el de la profundidad máxima del árbol como el del balanceo del peso de las clases. A continuación, se muestra cómo queda finalmente visualizado el modelo (ver **Figura.73**):



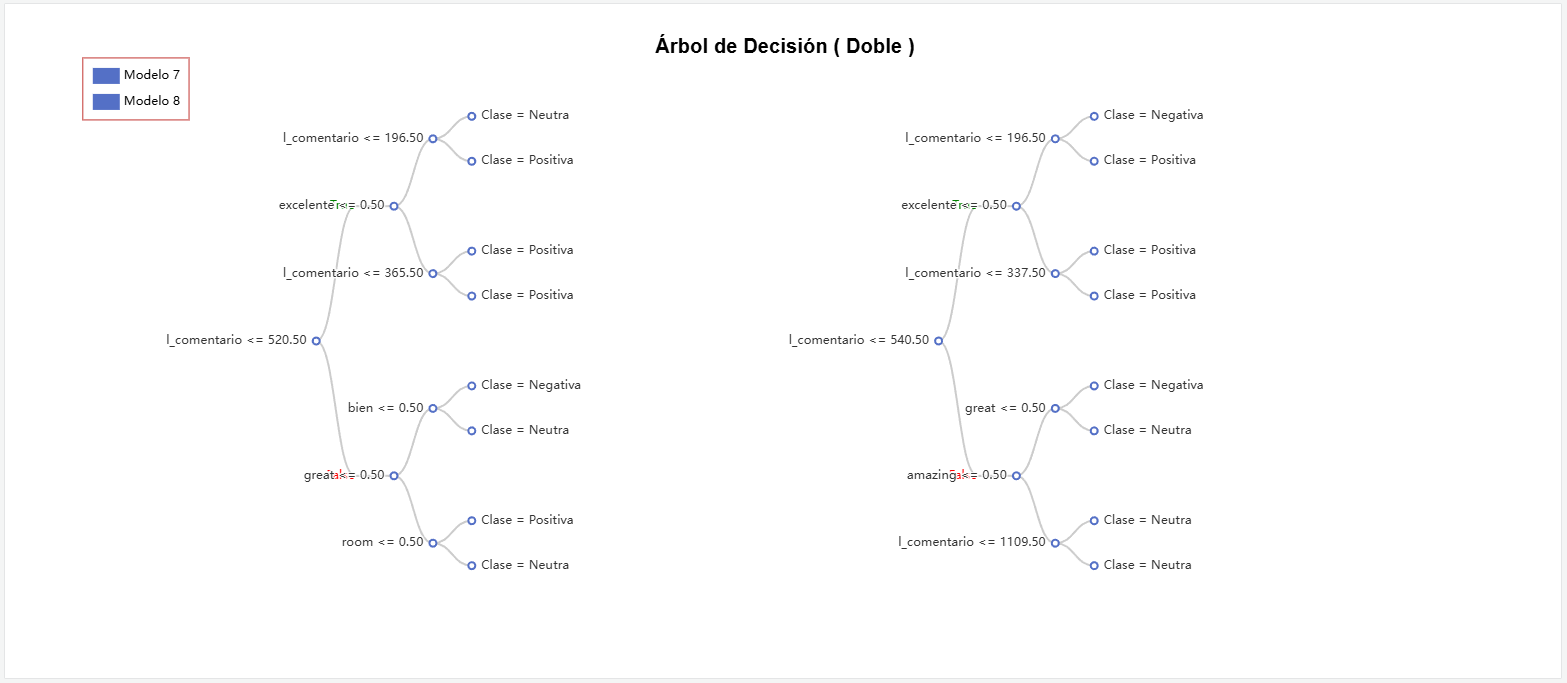
**Figura 73:** Visualización de modelo de árbol de decisión en forma vertical modo claro con dataset de comentarios emitidos: “val\_llm” como variable objetivo. Foto tomada por el autor de Grafana.

Este modelo permite:

* Identificar Factores Claves: Determinar cuáles son las palabras claves, los patrones temporales (mes), o las características del contenido textual (l\_comentario) que tienen mayor influencia en la clasificación automática de los comentarios como positivos, neutros o negativos.
* Analizar el Comportamiento del Modelo: Comprender cómo el modelo automático procesa la información, identificando qué características aportan mayor peso en la decisión final del modelo (val\_llm).
* Detección de Posibles Sesgos: Analizar si ciertas palabras clave o períodos específicos (meses) influyen de manera desproporcionada en las predicciones del modelo automático. Esto permite identificar posibles tendencias o sesgos en la valoración generada.

Observaciones:

* Si el comentario no contiene la palabra "excelente" y además tiene menos de “196letras”, es más probable que se clasifique como “Negativo”, mientras que comentarios más largos (entre 196y540letras) suelen ser clasificados como “Positivos”.
* Comentarios que no contienen palabras clave como "excelente", "amazing" o "great" y que son cortos o medianamente largos tienden a clasificarse como “Negativos”. Esto sugiere que el modelo prioriza la existencia de términos enfáticos en su evaluación.



**Figura 74:** Visualización de modelo de árbol de decisión en forma doble modo claro: Modelo 7 - dataset de comentarios emitidos: “val\_h” como variable objetivo, Modelo 8 - dataset de comentarios emitidos: “val\_llm” como variable objetivo. Foto tomada por el autor de Grafana.

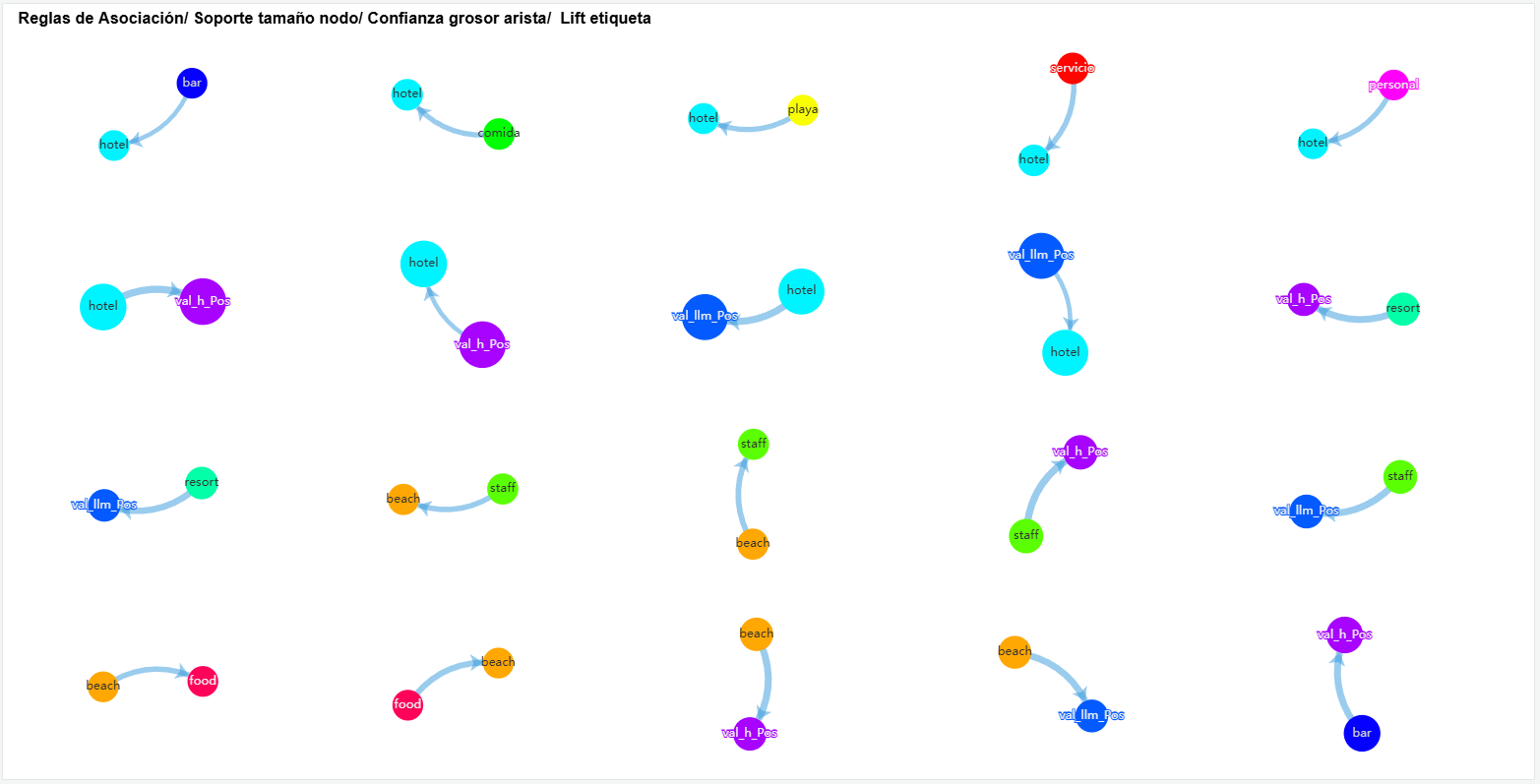
Al comparar ambas visualizaciones (ver **Figura.74**), se identificaron diferencias notables entre los modelos con "val\_h”y “val\_llm" como variable objetivo.

* El modelo con "val\_llm" incorpora atributos adicionales como "amazing", lo cual indica que es capaz de detectar matices más específicos en los comentarios, comparado con "val\_h". Este detalle sugiere que el modelo automático analiza una mayor variedad de características lingüísticas, mientras que el modelo humano tiende a centrarse en patrones más evidentes.
* El modelo con "val\_h" tiende a predecir más clases positivas y menos negativas, lo que refleja una posible percepción optimista por parte de las valoraciones humanas. Sin embargo, el modelo con "val\_llm" muestra más clases negativas y menos positivas, es decir, todo lo contrario, lo cual podría deberse a su capacidad para identificar patrones sutiles o negativos que el modelo humano no percibe tan claramente.
* En el modelo con “val\_llm” se observan umbrales más elevados de “l\_comentario”, sugiriendo que el modelo automático con “val\_llm” tiene en cuenta más detalles o variabilidad en los datos que el de “val\_h”.

### 3.4.3 Reglas de asociación

En el caso de las reglas de asociación, una vez preprocesados ​​los datos en Knime, se procedió a crear dos modelos. Para la creación del 1er modelo, se seleccionaron las columnas correspondientes a las palabras clave en comentarios, además de las variables de valoración val\_h (humana) y val\_llm (modelo de lenguaje). La inclusión de estas variables adicionales tuvo como propósito analizar no solo las co-ocurrencias entre las palabras claves, sino también descubrir patrones frecuentes asociados a las valoraciones, tanto humanas como del modelo.

Tras ejecutar el algoritmo para crear y entrenar el modelo, los resultados se visualizaron en la herramienta Grafana como se muestra en la **Figura.75**. Para garantizar una representación clara y comprensible, se limitó la consulta a las 20 primeras reglas, mediante un LIMIT en la configuración del panel, con el fin de evitar la sobrecarga y facilitar la interpretación de las mismas.



**Figura 75:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de comentarios emitidos (con “val\_h” y “val\_llm”). Foto tomada por el autor de Grafana.

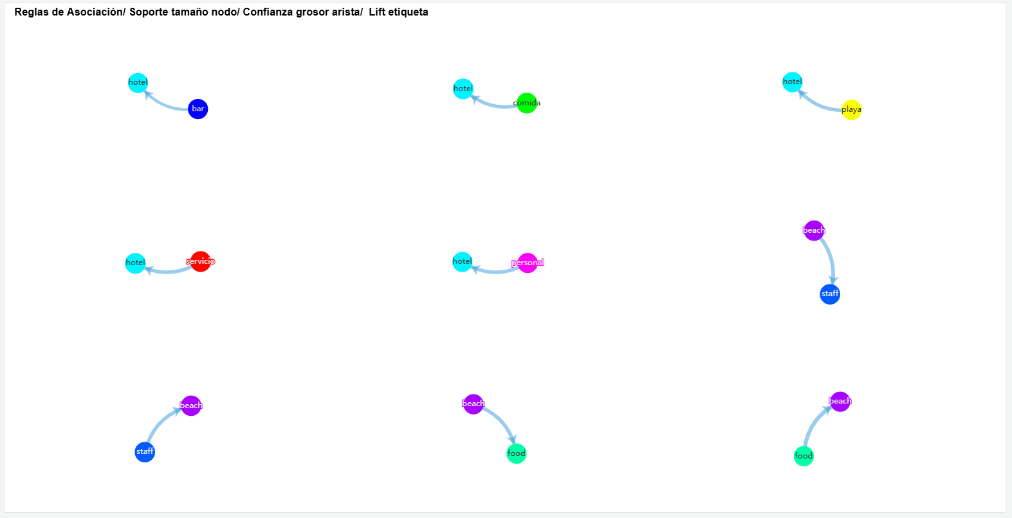
Este modelo permite:

* Descubrir patrones frecuentes: Identificar combinaciones de palabras claves que aparecen de manera recurrente en los comentarios y analizar cómo estas relaciones se reflejan en las valoraciones humanas (val\_h) y automáticas (val\_llm), y viceversa.
* Comparar perspectivas de valoración: Evaluar si las asociaciones entre palabras claves son consistentes al comparar las valoraciones humanas y automáticas, detectando posibles diferencias o similitudes en la interpretación de los comentarios.
* Revelar tendencias implícitas: Descubrir relaciones complejas entre las palabras claves y cómo estas co-ocurrencias pueden destacar preferencias o aspectos específicos percibidos en los comentarios, incluso cuando las valoraciones varían.

Observaciones:

* Si un comentario contiene la palabra "hotel", existe un alto nivel de confianza de que sea valorado de forma positiva tanto por "val\_h" (valoración humana) como por "val\_llm" (valoración generada por modelo).
* Si un comentario contiene la palabra “beach”, es poco probable (bajo nivel de confianza) que aparezcan las palabras “food” o “staff”, lo cual indica que las experiencias relacionadas con la playa son percibidas de manera independiente frente a otros aspectos del servicio del hotel.

Para la creación del 2do modelo (ver **Figura.76**), se consideraron las mismas columnas utilizadas en el modelo anterior, excluyendo las variables de valoración val\_h y val\_llm. El objetivo principal de este modelo es descubrir relaciones frecuentes entre las palabras clave sin la influencia directa de las valoraciones.

En este caso no fue necesario aplicar un LIMIT en la configuración del panel, como en el caso del 1er modelo debido a que no se generaron tantas reglas y por ende no afecta la visualización.

**Figura 76:** Visualización de modelo de reglas de asociación de forma separadas modo claro con dataset de comentarios emitidos (sin “val\_h” y “val\_llm”). Foto tomada por el autor de Grafana.

Este modelo permite:

* Descubrir patrones frecuentes: Identificar combinaciones de palabras claves que se mencionan de manera recurrente en los comentarios, lo cual permite detectar tendencias en los servicios o aspectos más mencionados por los clientes.
* Revelar relaciones entre servicios: Analizar cómo ciertos términos claves tienden a aparecer juntos en las encuestas, destacando posibles vínculos implícitos entre los ítems evaluados.
* Analizar contenido sin sesgos: Explorar las relaciones directas entre los términos claves del contenido textual, sin la influencia de las valoraciones humanas o automáticas, permitiendo una visión más neutral de las preferencias o menciones recurrentes.

Observaciones:

* Si un comentario contiene palabras como “comida”, “playa” o “servicio”, existe una probabilidad media (confianza media) de que también contenga la palabra “hotel”, lo cual indica que estos aspectos pueden ser valorados como elementos específicos de la experiencia del cliente y no siempre son generalizados para describir la percepción global del hotel.
* Las relaciones que se muestran presentan un bajo nivel de soporte, lo que indica que los comentarios que incluyen estas combinaciones de palabras no son mencionados con frecuencia por la mayoría de los clientes en sus valoraciones.

Al comparar ambas visualizaciones (ver **Figura.75**, **Figura.76**), se identificaron diferencias notables entre los modelos con y sin "val\_h”y “val\_llm".

* El modelo con "val\_h”y “val\_llm", muestra un conjunto más amplio de reglas de asociación, combinando percepciones humanas y automáticas, es decir se identifican patrones influenciados por ambas valoraciones, lo cual añade una perspectiva mixta a la clasificación. Sin embargo, el modelo sin "val\_h”y “val\_llm" presenta un conjunto más limitado y directo de reglas, enfocándose únicamente en atributos descriptivos y binarios del conjunto de datos, como palabras claves ("comida", "playa", "personal").
* El modelo con "val\_h" y "val\_llm" refleja una mayor diversidad en las asociaciones entre atributos, conectando términos menos intuitivos y generando reglas más complejas que capturan una mayor variabilidad en los datos. Sin "val\_h" y "val\_llm", el modelo produce reglas más simples y directas, basadas en relaciones más evidentes entre atributos frecuentes.

## 3.5 Conclusiones parciales

* Las distintas opciones de visualización de los modelos (árboles de decisión y reglas de asociación) demostraron ser esenciales para ofrecer flexibilidad y adaptabilidad en la representación de los mismos. Estas opciones permiten ajustes como el tamaño de las etiquetas, el grosor de las aristas y la selección de paletas de colores, asegurando que las visualizaciones puedan adaptarse a diferentes necesidades y contextos analíticos, facilitando su comprensión incluso en escenarios complejos.
* Los casos de prueba realizados validaron la funcionalidad de estas opciones de personalización al aplicarlas a grandes conjuntos de datos. Estos casos no solo permitieron identificar patrones relevantes y relaciones significativas en los modelos, sino que también evidenciaron cómo las configuraciones visuales pueden ajustarse para resaltar aspectos claves, garantizando un análisis más claro de los resultados.

# **Conclusiones**

En el desarrollo de este trabajo, se cumplió de manera satisfactoria con cada uno de los objetivos propuestos inicialmente. Se profundizó en los conceptos teóricos necesarios para comprender los modelos de aprendizaje automático, específicamente árboles de decisión y reglas de asociación. Se realizó un análisis de la herramienta Grafana y su complemento Business Chart, donde se logró identificar las limitaciones existentes en cuanto a visualizaciones específicas para estos modelos.

Se diseñó una solución para el almacenamiento y la persistencia de los modelos en PostgreSQL, desarrollando una estructura optimizada que permitió almacenar tanto árboles de decisión como reglas de asociación. Esta solución facilitó la recuperación de los modelos almacenados mediante consultas SQL, integrándose con Grafana para su posterior visualización.

Los paneles creados en Grafana requirieron ajustes en el código de Echarts, lo que permitió no solo procesar los datos provenientes de las consultas SQL, sino también la representación correcta de los modelos. Estos ajustes posibilitaron la personalización de elementos clave de las visualizaciones, como el tamaño de las etiquetas, el grosor de las aristas, las paletas de colores, entre otros, garantizando flexibilidad y adaptabilidad a las necesidades específicas del usuario.

Los casos de prueba realizados permitieron validar la funcionalidad de los paneles creados en Grafana. Estos casos demostraron que, dichos paneles no solo son capaces de representar correctamente los modelos de aprendizaje automático, sino que también logran visualizar grandes volúmenes de datos. Además, las pruebas evidenciaron cómo estos permiten identificar patrones claves, resaltar relaciones significativas entre los datos y ofrecer una mayor claridad en la visualización.

# **Recomendaciones**

A partir del desarrollo de este proyecto, centrado en la creación de paneles en Grafana para la visualización de modelos de aprendizaje automático, se formulan las siguientes recomendaciones con el objetivo de extender la funcionalidad de la solución planteada:

* Explorar alternativas para implementar árboles generales: Actualmente, la implementación se limita a árboles binarios. Ampliar esta capacidad permitiría trabajar con estructuras más complejas y adaptarse a un mayor rango de modelos.
* Continuar investigando cómo guardar el estado de los árboles en la visualización: Esto evitaría que los árboles se reinicien cada vez que se realice un cambio en la configuración visual, mejorando la experiencia del usuario y facilitando el análisis continuo.
* Profundizar en la representación matricial para las reglas de asociación: Específicamente, se recomienda investigar cómo relacionar los nodos que representan las conjunciones dentro de las visualizaciones, para lograr una representación más intuitiva y funcional de estas reglas.
* Optimizar la gestión de variables en Grafana: Se sugiere organizar las variables por panel, evitando que todas aparezcan juntas en un mismo lugar. Esto mejoraría la claridad y la navegabilidad en proyectos con múltiples visualizaciones.
* Seguir investigando la integración con otras herramientas complementarias: Considerar cómo Grafana puede combinarse con otras plataformas de visualización o bases de datos para extender su funcionalidad.
* Incorporar configuraciones que permitan ajustar la cantidad de reglas a mostrar y el criterio de ordenamiento: Implementar una opción que permita al usuario definir el número de reglas visibles y establecer el criterio de ordenamiento como soporte, confianza o lift, así como otros filtros que permitan la selección de reglas según parámetros personalizados, como valores mínimos de métricas, entre otros.
* Explorar alternativas para gestionar las columnas del dataset de encuestas: Dado que los nombres de las columnas: Q00001, Q00002, Q00003, Q00004, dificultan la interpretación de los datos, se sugiere buscar soluciones que permitan identificarlas y organizarlas de manera más clara para su posterior uso.
* Evaluar opciones para reducir la redundancia de clases en árboles de decisión al aplicar un max\_depth: Cuando se establece una profundidad máxima en los árboles de decisión, puede generarse redundancia en las clases de los nodos, lo que afecta la claridad de la visualización. Se recomienda investigar métodos que reduzcan estas duplicidades con el fin de optimizar la interpretación de los modelos y garantizar representaciones más claras.

# **Referencias Bibliográficas**

[1] E. A. R. Sánchez. (2023, 2024.02.09). *La importancia de conocer la ciencia de datos en el mundo moderno*. Available: <https://www.linkedin.com/pulse/la-importancia-de-conocer-ciencia-datos-en-el-mundo-rosario-s%C3%A1nchez?originalSubdomain=es%20%20>

[2] Datademia. (2024.02.09). *Sube la demanda de los Científicos de Datos*. Available: <https://datademia.es/blog/sube-la-demanda-de-los-cientificos-de-datos>

[3] U. T. d. Bolívar. (2022, 2024.02.09). *Ciencia de Datos, una excelente oportunidad para el fortalecimiento del perfil profesional*. Available: <https://www.utb.edu.co/blog/blog-ciencias-basicas/la-importancia-de-la-ciencia-de-datos-para-el-profesional/>

[4] A. García. (2023, 2024.02.09). *La importancia del análisis de datos en el presente y el futuro*. Available: <https://www.labrujulaverde.com/2023/11/la-importancia-del-analisis-de-datos-en-el-presente-y-el-futuro>

[5] M. S. Technologies. (2024.10.25). *Machine Learning & Predictive Analytics: A Comprehensive Guide*. Available: <https://www.msystechnologies.com/blog/machine-learning-predictive-analytics-a-comprehensive-guide/>

[6] R. Malhotra. (2024.10.25). *The Role of Machine Learning in Predictive Analytics and Decision - Making*. Available: <https://www.valuecoders.com/blog/analytics/role-of-machine-learning-in-predictive-analytics-and-decision-making/>

[7] Z. C. Qianwen Wang, Yong Wang, Huamin Qu. (2021, 2024.10.25). *A Survey on ML4VIS: Applying Machine Learning Advances to Data Visualization*. Available: <https://ar5iv.labs.arxiv.org/html/2012.00467v1>

[8] T. Analytics. (2023, 2024.10.25). *9 Top Machine Learning Algorithms for Predictive Modeling*. Available: <https://www.towardsanalytic.com/machine-learning-algorithms-for-predictive/>

[9] S. B. Philipp Offermann, Marten Schönherr, Udo Bub. (2010, 2024.02.25). *Artifact Types in Information Systems Design Science – A Literature Review*. Available: <https://www.researchgate.net/publication/221581354_Artifact_Types_in_Information_Systems_Design_Science_-_A_Literature_Review>

[10] AWS. (2024.02.09). *¿Qué es la visualización de datos?* Available: <https://aws.amazon.com/es/what-is/data-visualization/>

[11] W. Zhu. (2021, 2024.02.25). *A Study of Big-Data-Driven Data Visualization and Visual Communication Design Patterns*. Available: <https://doi.org/10.1155/2021/6704937>

[12] D. A. A. A. J. S. J. T. Rui Dan, Haitao Chu, Christopher H. Schmid, Yongchen, "Origami Plot a Novel Multivariate Data Visualization," *Journal of Clinical Epidemiology,* no. 156, pp. 85-94, 2023.

[13] V. V. e. G. Evgeniy Yur’evich Gorodov, "Analytical Review of Data Visualization Methods in Application to Big Data," *Journal of Electrical and Computer Engineering,* vol. 2013, pp. 1-7, 2013.

[14] S. S. Skiena, *The Data Science Design Manual*, 1ra ed. New York: Stony Brook University, 2017.

[15] Tableau. (2024.02.09). *What is data visualization? A definition, examples, and learning resources*. Available: <https://www.tableau.com/es-es/learn/articles/data-visualization>

[16] M. C. Gómez. (2021, 2024.02.25). *Visualización de datos: qué es, ejemplos, técnicas y soluciones*. Available: <https://blog.hubspot.es/marketing/visualizacion-de-datos>

[17] J. M. G.-B. David Moreno-Lumbreras, Gregorio Robles, Valerio Cosentino, "The influence of the city metaphor and its derivates in software visualization," *The Journal of Systems and Software,* vol. 210, 2024.

[18] A. P. Ioannis Tsouros, Athena Tsirimpa, Ioannis Karakikes, Shahram Tahmasseby, Anas Ahmad Nemer Mohammad, Wael K. M. Alhajyaseen, "From Raw Data to Informed Decisions: The Development of an Online Data Repository and Visualization Dashboard for Transportation Data," presented at the The 14th International Conference on Emerging Ubiquitous Systems and Pervasive Networks 2023.

[19] Neptune.ai. (2024, 2024.10.26). *How to Do Model Visualization in Machine Learning?* Available: <https://neptune.ai/blog/visualization-in-machine-learning>

[20] J. R. QUINLAN, *Induction of Decision Trees*. Sydney: New South Wales Institute of Technology, 2007.

[21] L. Gonzalez. (2019, 2024.10.26). *Árboles de Decisión Clasificación – Teoría*. Available: <https://aprendeia.com/arboles-de-decision-clasificacion-teoria-machine-learning/>

[22] IBM. (2024.10.26). *Reglas de asociación*. Available: <https://www.ibm.com/docs/es/spss-modeler/18.4.0?topic=nodes-association-rules>

[23] U. DSLab. (2024.10.26). *8 Reglas de asociación*. Available: <https://urjcdslab.github.io/AprendizajeAutomaticoI/reglas.html>

[24] M. Hahsler, "arulesViz: Interactive Visualization of Association Rules with R," *The R Journal,* vol. 9, pp. 163–174, Diciembre de 2017 2017.

[25] S. C. Michael Hahsler, "Visualizing Association Rules in Hierarchical Groups," presented at the 42nd Symposium on the Interface: Statistical, Machine Learning, and Visualization Algorithms, May 19, 2011, 2011.

[26] G. Labs. (2024.02.20). *Technical Documentation*. Available: <https://grafana.com/docs/>

[27] V. Labs. (2024.10.27). *Business Charts*. Available: <https://volkovlabs.io/plugins/business-charts/>

[28] ECharts. (2024.11.1). *ECharts Examples*. Available: <https://echarts.apache.org/examples/en/index.html>

[29] InfluxData. (2024.10.27). *MLOps: A Comprehensive Guide to Machine Learning Operations*. Available: <https://www.influxdata.com/glossary/mlops/>

[30] Bytescrum. (2024, 2024.10.27). *How to Deploy Machine Learning Models in Production: Key Challenges and Fixes*. Available: <https://blog.bytescrum.com/how-to-deploy-machine-learning-models-in-production>

[31] PyPl. (2024, 2024.10.26). *psycopg2 2.9.10*. Available: <https://pypi.org/project/psycopg2/>

[32] PyPI. (2024, 2024.10.26). *asyncpg 0.30.0*. Available: <https://pypi.org/project/asyncpg/>

[33] PyPI. (2024, 2024.10.26). *pg8000 1.31.2*. Available: <https://pypi.org/project/pg8000/1.31.2/>

[34] PyPI. (2024, 2024.10.26). *SQLAlchemy 2.0.36*. Available: <https://pypi.org/project/SQLAlchemy/>

[35] J. Ahmed. (2024.10.27). *Top PostgreSQL Drivers for Python*. Available: <https://www.timescale.com/learn/top-postgresql-drivers-for-python>

[36] G. V. Fabian Pedregosa, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay, " Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research 12* pp. 2825-2830, 2011.

[37] Mlxtend. (2024.10.26). *Welcome to mlxtend's documentation!* Available: <https://rasbt.github.io/mlxtend/>

[38] J. A. Rodrigo. (2023, 2024.10.26). *Reglas de asociación con python*. Available: <https://cienciadedatos.net/documentos/py50-reglas-de-asociacion-python>

[39] DataCamp. (2024, 2024.10.26). *Tutorial de minería de reglas de asociación en Python*. Available: <https://www.datacamp.com/es/tutorial/association-rule-mining-python>

[40] Knime. (2023, 2024.10.30). *Exporting learning model results from analysis nodes*. Available: <https://forum.knime.com/t/exporting-learning-model-results-from-analysis-nodes/75793>

[41] G. Labs. (2024.10.30). *Infinity data source plugin for Grafana*. Available: <https://grafana.com/docs/plugins/yesoreyeram-infinity-datasource/latest/>

[42] M. L. Challenge. (2024, 2024.10.30). *Implementación y puntuación de un modelo de aprendizaje automático mediante un punto de conexión en línea*. Available: <https://learn.microsoft.com/es-es/azure/machine-learning/how-to-deploy-online-endpoints?view=azureml-api-2&tabs=cli>

[43] C. J. Date, *Introducción a los sistemas de bases de datos*, 7ma ed. PEARSON EDUCACIÓN, 2001.