Documentación: KNIME ETL-Workflow (Proceso detallado)

Resumen del Proyecto

Este flujo de trabajo en KNIME fue desarrollado para limpiar y transformar un conjunto de datos de propiedades inmobiliarias. Durante el proceso, eliminamos duplicados, corregimos valores faltantes en varias columnas, unimos datos relevantes y generamos un conjunto de datos estructurado para su análisis.

El dataset original con el que trabajamos proviene del archivo **DATA_Barcelona_Fotocasa_HousingPrices_Augmented.csv**, que contenía registros con problemas como valores faltantes en múltiples columnas, datos inconsistentes y duplicados. Este documento describe detalladamente cada nodo del flujo de trabajo, explicando su propósito y cómo contribuye a la limpieza y transformación de los datos.

---

1. Importación de Datos y Preprocesamiento

1.1 Carga de Datos (CSV Reader)

- Se utilizó el nodo **CSV Reader** para importar el archivo **DATA_Barcelona_Fotocasa_HousingPrices_Augmented.csv**.
- Se verificó que los tipos de datos fueran asignados correctamente:
    - Columnas numéricas como *price*, *square_meters*, *bathroom*, *rooms* fueron detectadas como valores enteros o decimales.
    - Columnas categóricas como *neighborhood* y *real_state* fueron interpretadas como texto.
- Se identificaron valores faltantes en varias columnas, incluyendo:
    - *real_state*
    - *square_meters*
    - *rooms*
    - *bathroom*

1.2 Análisis Estadístico de los Datos (Statistics & Extract Table)

- Se utilizó el nodo **Column Filter** para eliminar columnas innecesarias en el dataset, como "Unnamed: 0".

  Se utilizó el nodo **Statistics** para obtener información descriptiva de las columnas:

  - Se calcularon estadísticas como la media, la mediana, la desviación estándar, valores mínimos y máximos.
  - Se identificaron valores atípicos y distribuciones de datos mediante histogramas.
  - Se verificó la cantidad de valores ausentes en cada columna

- Se aplicó el nodo **Extract Table Spec** para obtener detalles técnicos sobre las columnas:

  - Se listaron los nombres y tipos de datos de cada columna.
  - Se establecieron los límites inferior y superior de los valores numéricos.
  - Se confirmó la presencia de datos nulos (*NaN*) y su impacto en el dataset.

- En Python, este tipo de análisis se puede realizar con los siguientes comandos:

  - Para obtener estadísticas descriptivas: df.describe()
  - Para conocer los tipos de datos: df.dtypes

1.3 Corrección de Tipos de Datos y Redondeo (Math Node)

- Se utilizó el nodo **Math Expression** para corregir problemas en los valores numéricos.
- Se aplicó la función round($$CURRENT_COLUMN$$) en columnas que deberían ser enteros:
    - *rooms*
    - *bathroom*
    - *square_meters*
    - *price*
- Se convirtió el tipo de dato de estas columnas a **entero (INT)** para evitar inconsistencias en cálculos posteriores.

1.4 Renombrado de Columnas (Column Renamer)

- Se utilizó el nodo **Column Renamer** para cambiar el nombre de la columna **real_state** a **housing type**.
- Esto se realizó para estandarizar los nombres de las columnas y facilitar su comprensión en el análisis posterior.

1.5 Filtrado de Columnas para Relleno de Valores Faltantes (Column Filter)

- Se implementaron dos filtros de columnas para preparar los datos antes de rellenar valores ausentes:

    1. **Columnas numéricas (rooms, bathroom, square_meters):**

        - Estas columnas fueron seleccionadas para su posterior imputación utilizando la mediana de los valores existentes.

    2. **Columna categórica (housing type):**

        - Se aisló esta columna para rellenar los valores ausentes con la categoría "Unknown".

1.6 Filtrado de Columnas para Evitar Duplicaciones en la Unión de Datos (Column Filter)

- Se eliminó del conjunto principal las columnas *rooms*, *bathroom* y *housing type*.
- Esto se hizo para evitar que las columnas se dupliquen al volver a unir los datos más adelante en el flujo de trabajo.

2. Limpieza de Datos

2.1 Imputación de Valores Faltantes (Missing Value Nodes)

- Se implementaron dos nodos **Missing Value** para tratar los valores ausentes en diferentes columnas:
    1. **Columnas numéricas (rooms, bathroom, square_meters):**
        - Los valores faltantes en estas columnas fueron reemplazados utilizando la mediana de los valores existentes.
    2. **Columna categórica (housing type):**
        - Los valores faltantes en esta columna fueron reemplazados con la categoría "Unknown" para mantener la consistencia de los datos.

2.2 Verificación de la Imputación (Statistics Nodes & Value Counter)

- Se utilizaron nodos **Statistics** después de la imputación para confirmar que no quedaran valores ausentes en las columnas corregidas.

2.3 Conversión de Datos Numéricos a Enteros (Double to Integer)

- Se utilizó un nodo **Double to Integer** después de la imputación de la mediana.
- Esto fue necesario porque la imputación con la mediana convirtió automáticamente las columnas a tipo *double*.

2.4 Unificación de las columnas con valores faltantes

- Se utilizó un nodo **Joiner** para unir las columnas previamente separadas con valores faltantes. Esto asegura que las columnas procesadas vuelvan a estar en una única tabla.

2.5 Unión con el conjunto de datos principal

- Se realiza otra unión mediante un distinto **Joiner**. Aquí, las columnas previamente procesadas y reunidas en el punto 2.4 se reincorporan al conjunto de datos principal

2.6 Nodo de estadísticas para la validación de valores faltantes

- El nodo de se utiliza para verificar los valores faltantes restantes.
  El resultado muestra que en las columnas previamente procesadas ya no hay valores faltantes. Los únicos valores faltantes que aún existen se encuentran en la columna *square_meters_price*. Esto confirma que los pasos anteriores se realizaron correctamente.

2.7 Identificación de Duplicados (Duplicate Row Filter)

- Se utilizó un nodo **Duplicate Row Filter** para identificar y analizar registros duplicados.
- Se generó una columna adicional llamada **Duplicate Status** que indica:
  - "chosen": Registros duplicados donde solo se conservará uno.
  - "duplicate": Registros que serán eliminados posteriormente por ser repetidos
  - "unique": Registros sin duplicados.
- Se utilizó un **Table View** para visualizar los duplicados detectados y asegurarse de que los datos sean correctos antes de eliminarlos.
- Se detectaron los siguientes valores:
  - *Valores únicos*: 14,976
  - *Valores "chosen"*: 592
  - *Valores duplicados*: 808

2.8 Eliminación de Duplicados y Normalización de Datos

- Se utilizó un segundo nodo **Duplicate Row Filter** para eliminar definitivamente los registros duplicados.
- Se redujo el conjunto de datos a **15,568 registros finales**.

2.9 Manipulación de cadenas de texto

- Se utilizó el nodo de **String Manipulation** para modificar los valores en la columna *housing type*. En este proceso, el valor "study" fue reemplazado por "studio", asegurando una consistencia en la nomenclatura de los tipos de vivienda. Esto permite mejorar la calidad y homogeneidad de los datos en la base de datos.

2.10 Identificación de Valores Faltantes en square_meters_price (Row Filter)

- Se utilizó un **Row Filter** para identificar los valores faltantes en la columna square_meters_price.
- Se detectaron **439 valores faltantes** en esta columna.

2.11 Imputación de Valores Faltantes en square_meters_price (Missing Value)

- Se utilizó un nodo **Missing Value** para reemplazar los valores faltantes en la columna *square_meters_price* con la mediana de los valores existentes.
- Un nodo **Statistics** fue utilizado para confirmar que ya no quedaban valores ausentes en ninguna columna.
- La revisión final en el nodo **Statistics** muestra que ahora **todas las columnas están completas**, sin valores faltantes.

2.12 Nodo Number Rounder

- Este nodo redondea el precio en la columna m2 a 2 decimales
- En la configuración del nodo, la opción "column m2 price rounded to just 2 decimals" está activada
- El nodo recibe los precios en la columna m2 y los devuelve redondeados a 2 decimales
- Este redondeo asegura que los precios se presenten en un formato uniforme y legible

3. Revisión de valores irracionales y outliers

Después del nodo *Number Rounder*, se han creado cinco grupos de filtros de filas para identificar y eliminar valores irracionales y posibles *outliers* en los datos.
Cada uno de los siguientes filtros cuenta con un Table View que ilustra las explicaciones dadas.

3.1 Rule-based Row Filter 1

- El primer filtro de filas se encuentra en la parte superior. Se encarga de filtrar valores ilógicos en la columna *rooms*, eliminando registros donde el número de habitaciones es 0. Cantidad de valores >>> 392.

Se aplicó un Row Splitter para dividir estos 392 registros en dos grupos:

- 88 registros con 0 rooms y housing type igual a studio, ya que en este tipo de viviendas es común no tener habitaciones definidas. Estos valores serán conservados y reincorporados al conjunto de datos principal.

- 304 registros con 0 rooms, pero con otro housing type. Como no es posible determinar dónde se encuentra exactamente el error en el conjunto de datos, estos registros serán descartados. Representan solo el 2% de la base de datos total y no afectan significativamente el análisis. Sin embargo, si los resultados de la predicción futura no son satisfactorios, se puede reconsiderar este filtro y analizarlo nuevamente.

3.2 Rule-based Row Filter 2

- Ubicado debajo del primer filtro, este filtro se enfoca en valores ilógicos en la columna *bathroom*. Se identificaron 4 valores sospechosos.
  No es posible determinar en qué columna exacta se encuentra el error, por lo que estos registros serán eliminados del conjunto de datos.

3.3 Rule-based Row Filter 3 - Conjunto de datos principal

- Este filtro representa el conjunto de datos principal sin los valores filtrados arriba y los dos que vienen ahora más abajo. Se puede considerar como la versión depurada de los datos.

3.4 Rule-based Row Filter 4

- Se encarga de detectar valores atípicos en la columna *square_meters*.
  Se han identificado 10 registros sospechosos.
  Este filtro contiene 10 valores, de los cuales 4 también están presentes en el filtro 5. Como los 4 valores en el filtro 5 serán conservados, solo los 6 valores restantes serán eliminados. La razón sigue siendo la misma: no se puede determinar con exactitud el origen del error, y estos 6 registros no afectan el procesamiento adecuado del conjunto de datos.

3.5 Rule-based Row Filter 5

- Este último filtro examina la columna *price* en busca de valores ilógicos.
  Se detectaron 4 valores sospechosos.
  Sin embargo, tras una revisión detallada, se ha determinado que estos precios son plausibles en el contexto de los demás detalles de cada entrada.
  Por lo tanto, estos valores serán retenidos en la base de datos.

3.6 Consolidación de los datos

Nodo Concatenate 1
Concatenate 1 une los datos del Filtro 1 (88 valores con *0 rooms* y *housing type* igual a *studio*) con el conjunto de datos principal.

Nodo Concatenate 2
Concatenate 2 fusiona el conjunto de datos principal con los 4 valores del Filtro 5.

4. Tratamiento de valores desconocidos

4.1 Filtro de valores 'Unknown' en la columna *housing type*

- Se utiliza un **Row Splitter** para separar los valores "Unknown" en la columna *housing type*. Estos valores serán reemplazados por el tipo de vivienda correspondiente en el vecindario correspondiente.

  El Row Splitter tiene dos **Table Views**:

- **14.500 registros** sin valores "Unknown",
  que permanecen como el conjunto de datos principal

- **757 registros** con valores "Unknown", que serán reemplazados
  en función de la información del vecindario correspondiente

4.2 Agregación por vecindario

- Se utiliza un nodo **GroupBy** para calcular el tipo de vivienda más frecuente (Mode) dentro de cada vecindario. Este valor se utilizará para reemplazar las entradas con "Unknown" en la columna housing type.

4.3 Unión con los valores de referencia

- Se utiliza un **Joiner** para asignar el tipo de vivienda predominante a los registros con valores "Unknown" en la columna *housing type*.
  La unión se realiza a través de la columna *neighborhood*, asegurando que cada valor desconocido sea reemplazado por el más frecuente en su vecindario.
  Se conservan todas las filas de la tabla principal, incluso aquellas sin coincidencias en la tabla de referencia.

4.4 Aplicación de reglas para la imputación de valores

- Se emplea un nodo **Rule Engine** para reemplazar los valores "Unknown" en la columna *housing type* con el valor más frecuente determinado en el paso anterior. La regla aplicada garantiza que, si el valor en *housing type* es "Unknown", se sustituya por el valor agregado correspondiente.
  De este modo, todos los valores desconocidos se completan de manera lógica sin afectar los registros ya existentes.

4.5 Eliminación de columna duplicada

- Un nodo **Column Filter** elimina la columna duplicada *housing type* creada en pasos anteriores, garantizando la consistencia de los datos y evitando redundancias.

4.6 Integración del conjunto de datos final

- Un nodo **Concatenate** fusiona el conjunto de datos principal con los **757 valores editados**, produciendo el conjunto de datos final limpio con **15,257 filas**.
  Un nodo **Table View** se utiliza para verificar la corrección del resultado final

4.7: Almacenamiento del conjunto de datos final

- Un nodo **CSV Writer** guarda el conjunto de datos limpio y consolidado en un archivo CSV. Este archivo puede ser utilizado directamente en procesos de Machine Learning. El formato CSV se mantiene para asegurar compatibilidad con diferentes herramientas y plataformas.