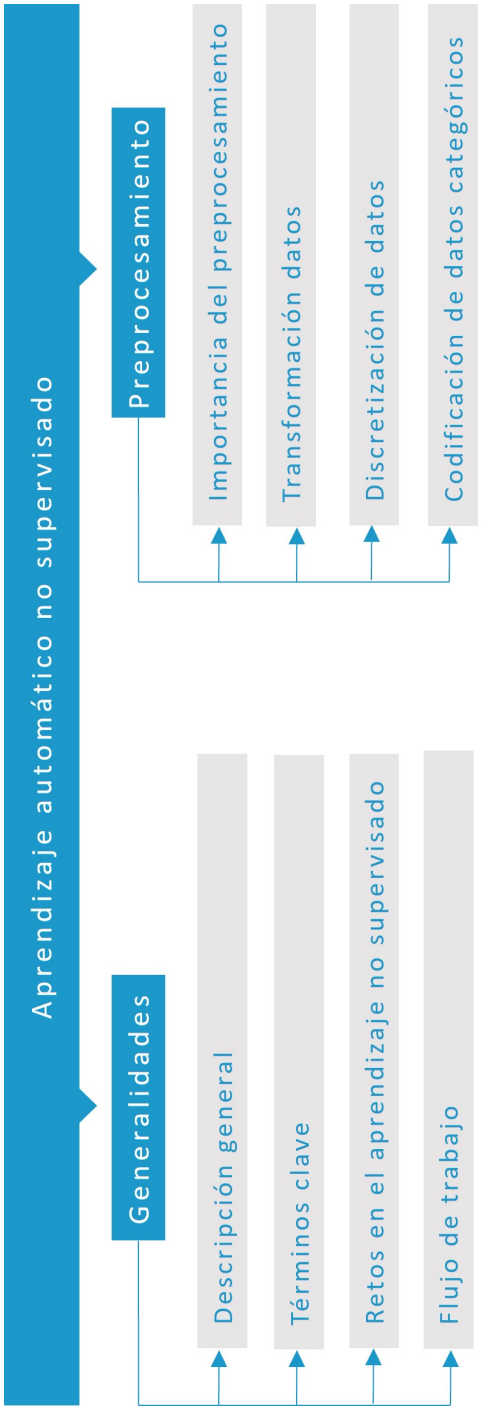
Aprendizaje Automático No Supervisado

Tema 1. Introducción al aprendizaje automático no supervisado

Índice

Esquema



1.1. Introducción y objetivos

E l **aprendizaje automático** ha revolucionado la manera en que las máquinas interactúan con el mundo, permitiendo que los sistemas de información aprendan de los datos y apoyen la toma de decisiones con base en la identificación de patrones y experiencia previas. Dentro de este campo, el **aprendizaje no supervisado** y el **aprendizaje por refuerzo** son dos áreas que permiten abordar problemas complejos sin la necesidad de datos etiquetados de antemano o con la capacidad de aprender a través de la interacción con el entorno.

En esta asignatura se verán diferentes algoritmos de aprendizaje no supervisado, fundamentos, aplicaciones y diferentes implementaciones del K-Means, algoritmos jerárquicos, algoritmos agrupamiento basado en densidad, técnicas lineales y no lineales para reducción de dimensionalidad y técnicas para detección de anomalías. Además, se introduce el aprendizaje por refuerzo detallado en el algoritmo Q-learning.

El aprendizaje automático no supervisado se denomina así porque no hay un supervisor externo proporcionando retroalimentación sobre cómo debe ser clasificado un dato.

En este tema se describen términos clave, retos y flujos de trabajo en el aprendizaje no supervisado. Además de detallar algunas técnicas de preprocesamiento que deberían tenerse en cuenta antes de intentar modelar con aprendizaje no supervisado.

1.2. Descripción general

El **aprendizaje automático no supervisado** parte de la premisa de que incluso en ausencia de un supervisor que indique cómo debe ser clasificado un dato, se puede extraer conocimiento a partir de los datos sin procesar. El corazón del aprendizaje no supervisado es la detección de patrones y estructuras de conjuntos de datos sin etiquetas. Algunos autores expresan que este tipo de aprendizaje se puede comparar con el proceso de examinar un cofre lleno de piedras preciosas variadas, cada una tiene su brillo y su forma (Van Der Post y Smith, 2023).

Los algoritmos actúan como clasificadores de joyas, categorizando y entendiendo cada gema o cada instancia del *dataset* basándose en sus características. El proceso no sigue una guía explicita, la única brújula son los mismos datos.

Es una de las técnicas más conocidas; busca **reunir instancias** similares dentro de clústeres o grupos, con base en la estructura de los datos.

Otra área es la **reducción de dimensionalidad;** simplifica la complejidad de los datos y elige los componentes más importantes o aquellos que generan mayor conocimiento sobre la estructura de los datos, haciendo que el conjunto de datos sea más manejable y comprensible.

Generar **reglas de asociación** también es un problema no supervisado donde encontrar reglas que describan una gran porción de datos, tales como clientes que compran un producto y que, a su vez, tienden a comprar otro producto. Es muy útil, en el análisis de la cesta de la compra, descubrir relaciones entre ítems en una gran cantidad de datos, lo cual permite a las empresas diseñar estrategias de marketing con promociones cruzadas.

Detectar anomalías es otra área del aprendizaje no supervisado. El algoritmo detecta patrones en los datos; si estos son inusuales o atípicos puede indicar que se

trata de una anomalía. Se utilizan algoritmos de agrupación para encontrar agrupaciones entre los datos y luego identificar instancias que no encajan en ninguno de los grupos o incluso se pueden encontrar grupos con pocas instancias que pueden pertenecer a datos anómalos. Es muy útil en aplicaciones como detección de fraude, detección de fallas en redes o en sistemas de producción, donde no siempre se cuenta con etiquetas de las instancias fallidas.

En los últimos años, en la bioinformática, se ha aplicado el aprendizaje no supervisado, el cual juega un papel fundamental en la secuencia genética. Los algoritmos de agrupamiento ayudan a categorizar genes con expresiones o patrones similares, los cuales pueden revelar variaciones genéticas, potenciales marcadores para enfermedades. Esto puede permitir un avance en la medicina y tratamientos personalizados.

En el **procesamiento del lenguaje natural** (PLN), los algoritmos no supervisados también han sido utilizados. Por ejemplo, el algoritmo Latent Dirichlet Allocation (LDA) puede descubrir temas latentes dentro de un gran conjunto de documentos o textos. Esto permite acciones como la organización de bibliotecas, mejorar los motores de búsqueda e incluso monitorizar análisis de sentimientos.

En el reconocimiento de imágenes, el aprendizaje no supervisado ha tenido un profundo impacto. Aprende a representar y categorizar imágenes teniendo en cuenta patrones existentes dentro de la propia imagen. Por ejemplo, en astronomía se pueden clasificar objetos celestes y analizar una gran cantidad de datos recolectadas por telescopios. Los clústeres les permiten a los astrónomos identificar grupos de galaxias, estrellas y otros fenómenos astronómicos.

La reducción de dimensionalidad utilizando técnicas como PCA, t-SNE, son muy utilizadas para la visualización de datos con una alta dimensionalidad y hacerlos más interpretables. Permite a los científicos de datos descubrir patrones y correlaciones que pueden no verse a simple vista (Van Der Post y Smith, 2023).

Por otro lado, el **aprendizaje por refuerzo** se basa en la interacción de un agente con su entorno para maximizar una recompensa acumulativa. Aquí, el agente toma decisiones secuenciales, aprendiendo de las recompensas o penalizaciones recibidas como resultado de sus acciones. Este tipo de aprendizaje es particularmente útil en entornos dinámicos donde las decisiones deben adaptarse constantemente a cambios en el entorno.

1.3. Términos clave en aprendizaje no supervisado

Al familiarizarse con los términos que se aplican en esta rama del aprendizaje automático, se puede entender de manera profunda los algoritmos. Además de mejorar la comunicación efectiva al querer intercambiar conocimientos en esta área.

Dentro de los **términos más comunes** se encuentran:

- **Característica:** es un concepto ya visto en aprendizaje supervisado. En esta asignatura nos referimos a una dimensión en el conjunto de datos. La selección de características influye en el rendimiento de los algoritmos no supervisados.
- **Clúster o grupo:** grupo de datos que comparten características similares. En aprendizaje automático no supervisado se crean grupos, con el objetivo de maximizar la distancia intra-clúster y minimizar la distancia inter-clúster.
- **Dimensionalidad:** es el número de características del conjunto de datos. La alta dimensionalidad es un problema y un reto. Generalmente se utilizan técnicas para reducir la dimensionalidad.
- **Métrica de distancia:** es una medida que indica qué tan similares son dos puntos; fundamental para crear grupos. Las distancias más comunes son Euclidiana, Manhattan y la del coseno.
- ***Outliers* o datos atípicos:** como su nombre lo dice, son datos que se desvían del resto del conjunto de datos. Pueden ser eliminados después de un proceso de limpieza.
- **Centroide:** el centro de un conjunto de datos.
- **Vectores y valores propios:** conceptos heredados del algebra lineal. Representan la dirección de la máxima variación en los datos y son utilizados para pasar de una alta dimensionalidad a un espacio de características menor.

Ideas clave

- ***Overfitting* o sobreajuste:** es un término que se utiliza para decir que un modelo funciona muy bien con los datos de entrenamiento, pero no se puede aplicar a otros datos. No tiene la capacidad de generalizar.

- **Hiperparámetros:** parámetros que se establecen en el momento inicial del entrenamiento.

1.4. Retos en el aprendizaje no supervisado

Al no tener etiquetas o un camino claro a seguir para modelar, se enfrenta a muchos retos. Algunos de los **retos del aprendizaje no supervisado** son:

- **Determinar el número óptimo de grupos en un *dataset:*** un reto siempre presente. Dentro de los métodos que existen para intentar predecir el número ideal de grupos están la técnica del codo, análisis de silueta y el índice Davies-Bouldin. Sin embargo, las técnicas mencionadas anteriormente necesitan de la experiencia de un humano para tomar la decisión final.

- **La maldición de la dimensionalidad:** en conjuntos de datos con alta dimensionalidad existe un alto riesgo de no encontrar grupos con características comunes; por tal motivo, si se reduce la dimensionalidad se puede mitigar este riesgo. Sin embargo, el proceso de reducción de la dimensionalidad debe hacerse de manera muy estructurada para preservar la estructura de los datos.

- **La escalabilidad de los algoritmos existentes:** hoy en día los conjuntos de datos son cada vez más grandes y el desempeño de los algoritmos puede no ser el óptimo para grandes conjuntos de datos. Es necesario desarrollar algoritmos de aprendizaje no supervisado escalables que puedan manejar datos masivos sin comprometer la calidad de las agrupaciones obtenidas.

- **Delimitar el ruido y los *outliers:*** los datos ruidosos pueden distorsionar la verdadera estructura de los datos y los grupos encontrados pueden no representar el conjunto de datos analizado. Métodos robustos de *clustering* y filtro de ruido son fundamentales.

- **La interpretabilidad de los resultados:** se puede estar frente a interpretaciones subjetivas. Al no tener categorías o etiquetas predefinidas, asignar significado a cada grupo creado va a depender del contexto y de la habilidad del experto o la experticia en el dominio.

Ideas clave

- **No contar con métricas muy fuertes:** las métricas sirven para validar el éxito o fracaso de un algoritmo. Se necesitan métricas definidas y robustas para medir la calidad y el desempeño de un algoritmo.

1.5. Flujo de trabajo en el aprendizaje no supervisado

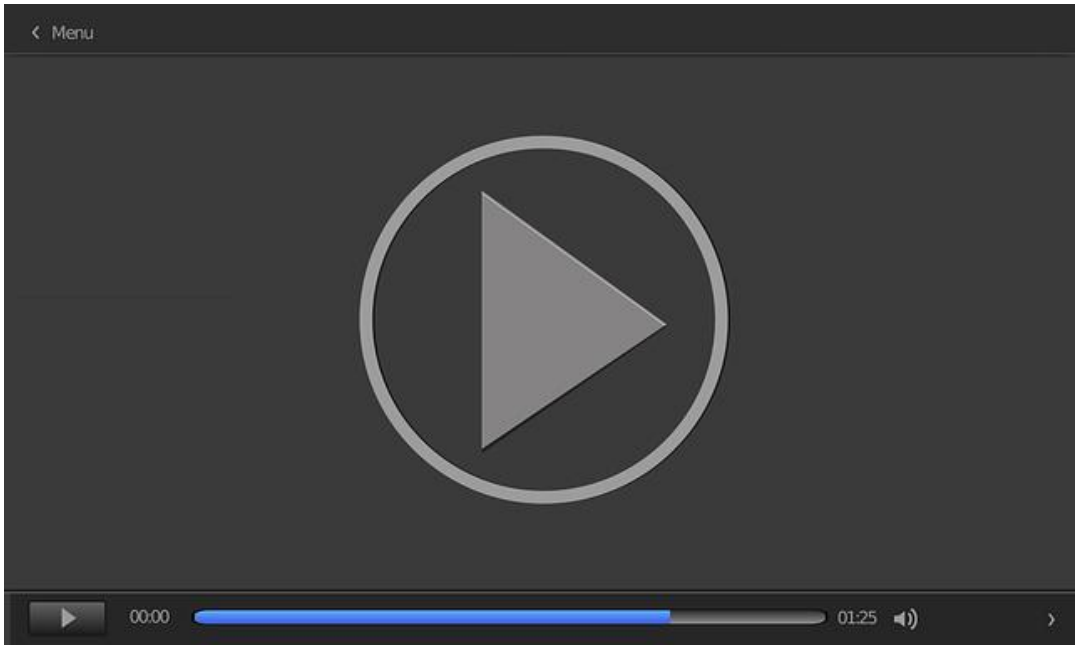
Los **principales pasos u hoja de ruta** que deberíamos seguir en un proyecto de aprendizaje no supervisado son los siguientes:

- Recolección de datos.
- Limpieza y preprocesamiento de datos.
- Preparación de los datos.
- Elegir y aplicar un algoritmo no supervisado.
- Entrenar y parametrizar el modelo.
- Interpretación de los resultados.
- Evaluación de los resultados.

El proceso inicia con la recolección; puede provenir de diferentes fuentes y en diferentes formatos. En este paso se debe consolidar el *dataset* y asegurar la consistencia. Una vez se tengan los datos, el siguiente paso es limpiarlos y preprocesarlos. En esta etapa se eliminan o corrigen anomalías, se pueden incluir técnicas de normalización o reescalado. Siguiendo con el flujo, se seleccionan las características más importantes, reduciendo de esta manera el número de variables. Una vez se tenga el *dataset* listo, se puede proceder con elegir y aplicar un algoritmo, el cual se parametriza se entrena; una vez entrenado se interpretan los resultados y si es satisfactorio se evalúan los resultados utilizando alguna métrica o se repite el proceso de elegir, parametrizar y entrenar otro algoritmo.

A continuación, veremos el vídeo ***Aprendizaje no supervisado: conceptos básicos y generalidades*** con un ejemplo sencillo en Python.

Ideas clave



Accede al vídeo:

https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=5ab637da-b76d-4ca8-ab42-b1ca00dd2f52

1.6. Importancia del preprocesamiento

L a **calidad del preprocesamiento** influye directamente en la efectividad de las siguientes tareas dentro del proceso, datos bien preprocesados permiten que los algoritmos detecten mejor los patrones y las relaciones, generando un conocimiento más preciso sobre el conjunto de datos.

Generalmente se encuentran datos inconsistentes, irrelevantes, que pueden sesgar cualquier algoritmo de aprendizaje automático supervisado y no supervisado.

Uno de los primeros pasos es el manejo de datos faltantes. Si existen instancias incompletas pueden introducir fallos en el modelo. Aquí se pueden utilizar estrategias como imputación de datos o eliminación de faltantes.

Otro paso importante en el preprocesamiento es el escalado de características. Fundamental en aprendizaje no supervisado, ya que algoritmos como K-Means son muy sensibles a la escala de los datos. Se pueden llevar a cabo procesos de estandarización o normalización para que todos los atributos contribuyan de la misma forma. Previniendo así que una característica pese más que otra influenciando al modelo.

Utilizar técnicas de transformación de datos, como transformación logarítmica, puede ser importante. Esto ayuda a estabilizar la varianza, especialmente en características que no siguen una distribución conocida.

Otro paso interesante y muy recomendado es la reducción de dimensionalidad, la cual ayuda a eliminar ruido y mejorar el cálculo computacional.

Codificación de variables categóricas. Básicamente es convertir una variable no numérica en un formato que entienda un algoritmo, transformar atributos en valores numéricos.

Y, finalmente, limpiar el *dataset* de datos ruidosos como duplicados o corregir errores de digitación o de ingreso inadecuado de datos.

En este tema veremos algunas técnicas de preprocesamiento que no se han visto en la asignatura de Técnicas de Aprendizaje Automático Supervisado.

1.7. Técnicas de transformación de datos

Se pueden aplicar técnicas lineales o no lineales con el objetivo de reducir asimetría, convertir relaciones no lineales en lineales y mejorar la predicción.

Por ejemplo, utilizar una transformación logarítmica a cada punto puede reducir la asimetría y estabilizar la varianza. Se utiliza este tipo de transformación cuando los datos siguen una distribución de ley de potencia y se van a aplicar algoritmos que asumen normalidad en los datos.

La transformación logarítmica es una extensión de la transformación de Box-Cox, introduce un parámetro lambda (λ), el cual permite identificar la mejor transformación de los datos para que se parezca más a una campana de Gauss. Esto lo hace suavizando las distribuciones inusuales y mitigando los efectos de los valores atípicos.

A continuación, se puede ver un ejemplo en Python, utilizando la librería de Scikit-learn e importando la clase PowerTransformer:

```
import pandas as pd
from sklearn.preprocessing import PowerTransformer

# Leer el conjunto de datos desde un archivo CSV
data = pd.read_csv("tu_archivo.csv")

# Selecciona las columnas numéricas para transformar
numeric_cols = data.select_dtypes(include=['float64', 'int64']).columns

# Crear y ajustar el transformador de potencia
pt = PowerTransformer(method='yeo-johnson') # Selecciona 'box-cox' o 'yeo-
johnson' para el método de transformación
pt.fit(data[numeric_cols])

# Aplicar la transformación a las columnas numéricas
data[numeric_cols] = pt.transform(data[numeric_cols])
```

En el código, primero se importa pandas para leer el archivo .Csv. Después se

identifican las columnas numéricas, en caso de que sean todas las columnas numéricas las que se deseen transformar. Y, por último, se crea un objeto PowerTransformer, para aplicar la transformación y actualizar las columnas con los valores transformados.

1.8. Técnicas de discretización de datos

L a s **técnicas de discretización** convierten variables continuas en variables discretas, permitiendo ver los datos desde una perspectiva diferente. Este enfoque divide en rangos las variables continúas generando intervalos que tienen una etiqueta. De esta forma, el valor numérico pasa a una categoría o *bins.* En aprendizaje no supervisado, la discretización simplifica relaciones complejas al etiquetar con una categoría o *bin* a un valor numérico.

Al discretizar se divide la variable en pequeños contenedores, en donde teniendo en cuenta la distribución de los datos se asegura que cada *bin* sea representativo del conjunto de datos.

Se puede utilizar los **árboles de decisión** para este proceso de discretización, aprovechando el uso de la ganancia de información para delimitar los límites óptimos de cada grupo de datos. Es un método supervisado que requiere una variable objetivo, pero se puede adaptar utilizando ceros como etiquetas.

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier

# Leer el conjunto de datos desde un archivo CSV
data = pd.read_csv("tu_archivo.csv")

# Selecciona las columnas numéricas que deseas discretizar
numeric_cols = data.select_dtypes(include=['float64', 'int64']).columns

# Entrenar un árbol de decisión para discretizar las variables
dt = DecisionTreeClassifier(max_leaf_nodes=4)

# Ajusta este valor según sea necesario
dt.fit(data[numeric_cols], np.zeros(len(data)))

# Utiliza una matriz de ceros como etiquetas ya que no tenemos etiquetas
reales

# Discretizar las variables utilizando las divisiones del árbol
```

```
discretized_values = dt.apply(data[numeric_cols])

# Agregar las variables discretizadas al conjunto de datos original
for i, col in enumerate(numeric_cols):
    data[col + '_discretized'] = discretized_values[:, i]
```

En el anterior ejemplo entrenamos un árbol con las columnas numéricas de un *dataset* y utilizamos una matriz de ceros como etiquetas para adaptarlo a un problema no supervisado. Después entrenamos el árbol de decisión para obtener las divisiones, obteniendo la transformación de una variable continua a una variable discreta.

Otra técnica es el uso de cuantiles. Los valores continuos se segmentan en cuatro cuantiles, donde cada cuantil encapsula una proporción igual de puntos de datos. Es muy útil si la variable tiene una distribución asimétrica.

```
import pandas as pd

# Cargar el conjunto de datos desde un archivo CSV
data = pd.read_csv("tu_archivo.csv")

# Seleccionar las columnas numéricas que deseas discretizar
numeric_cols = data.select_dtypes(include=['float64', 'int64']).columns

# Discretizar cada columna numérica por cuantiles
for col in numeric_cols:
    data[col + '_discretized'] = pd.qcut(data[col], labels=False,
duplicates='drop')
```

Iteramos sobre cada columna numérica seleccionada y aplicamos la función `pd.qcut()` de pandas para discretizar los datos en cuantiles. Establecemos `labels=False` para obtener números enteros que representan los cuantiles y `duplicates='drop'` para manejar duplicados en los cuantiles si los hubiera.

1.9. Técnicas de codificación de datos categóricos

Debido a que la mayoría de los algoritmos de aprendizaje automático utiliza números y no texto, se hace necesario convertir los datos textuales en formato numérico.

Existen dos **tipos de datos categóricos:** datos ordinales y dantos nominales.

Las categorías de **datos ordinales** tienen un orden intrínseco. Esto significa que las categorías se pueden ordenar de mayor a menor o viceversa. Por ejemplo, el grado más alto en estudios que tiene una persona es una variable ordinal: secundaria, bachillerato, grado, máster y doctorado.

Mientras que los **datos nominales** no tienen un orden inherente. Por ejemplo, ciudades de España: Málaga, Salamanca, Granada, Cádiz… No se clasifican o se ordenan.

La codificación es el proceso de convertir datos categóricos o textuales a formato numérico, de modo que puedan usarse como entrada para que los procesen los algoritmos.

- Esto permite que el modelo identifique patrones y haga predicciones basadas en esos patrones.
- La codificación también ayuda a evitar sesgos en el modelo al garantizar que todas las características tengan el mismo peso.
- La elección del método de codificación puede tener un impacto significativo en el rendimiento del modelo, por lo que es importante elegir una técnica de codificación adecuada según la naturaleza de los datos y los requisitos específicos del modelo (Jarapala, 2023).

Existen diferentes **métodos para llevar a cabo la codificación de variables:**

Codificación One-Hot

Es la forma más común de codificar una variable. Una columna binaria se crea por cada categoría de la variable. Si la variable está presente se le asigna el valor de 1, si no se le asigna 0.

```
import pandas as pd

# Crear un DataFrame de ejemplo
data = pd.DataFrame({'Color': ['Rojo', 'Verde', 'Azul', 'Rojo',
'Amarillo']})

Color
0    Rojo
1    Verde
2    Azul
3    Rojo
4    Amarillo

# Aplicar one-hot encoding
one_hot_encoded = pd.get_dummies(data['Color'])

# Concatenar los datos originales con las variables codificadas one-hot
data_encoded = pd.concat([data, one_hot_encoded], axis=1)

# Eliminar la columna original 'Color'
data_encoded.drop('Color', axis=1, inplace=True)

data_encoded

Amarillo Azul Rojo Verde
0       0      0     1      0
1       0      0     0      1
2       0      1     0      0
3       0      0     1      0
4       1      0     0      0
```

Cada columna representa una categoría única presente en la columna original 'Color' y el valor en cada celda indica si la observación corresponde o no a esa categoría. Si la observación tiene ese color, el valor será 1; de lo contrario, será 0.

Codificación ordinal

A cada categoría única se le asigna un valor entero único.

Este es un método de codificación más simple, pero tiene el inconveniente de que el algoritmo de aprendizaje automático puede malinterpretar los números enteros asignados como si tuvieran una relación ordenada cuando en realidad no es así.

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Crear un DataFrame de ejemplo
data = pd.DataFrame({'Color': ['Rojo', 'Verde', 'Azul', 'Rojo',
'Amarillo']})

# Inicializar el LabelEncoder
label_encoder = LabelEncoder()

# Aplicar el LabelEncoder a la columna 'Color'
data['Color_encoded'] = label_encoder.fit_transform(data['Color'])

# Imprimir el DataFrame resultante
print(data)

Color Color_encoded
0     Rojo              2
1    Verde             3
2     Azul              0
3     Rojo              2
4     Amarillo         1
```

Aquí, cada categoría única en la columna `'Color'` se ha asignado a un valor numérico único por el `LabelEncoder`. Por ejemplo, `'Azul'` se codifica como 0, `'Amarillo'` como 1, `'Rojo'` como 2 y `'Verde'` como 3.

Codificación binaria

Las categorías se representan como dígitos binarios. Por ejemplo, si una variable tiene cuatro categorías `'Bueno'`, `'Malo'`, `'Regular'` y `'Malo'`, se pueden representar como 0001, 0010, 0100 y 1000, respectivamente.

```
# Binary Encoding:
```

```
import pandas as pd

# Crear un dataframe con una variable categorica
df = pd.DataFrame({‘atencionCliente’: [‘Excelente’, ‘Regular’, ‘Malo’]})
print(f"Antes de codificar:\n\n{df}\n")

# aplicar la codificación binaria
servicioAlCliente_map = {'Excelente': 0, 'Regular': 1, 'Malo': 2}
df[‘atencionCliente’] = df[‘atencionCliente’].map(servicioAlCliente _map)
df[‘atencionCliente’] = df[‘atencionCliente’].apply(lambda x: format(x,
'b'))
```

Aquí, a cada categoría se le asigna un valor binario que la representa.

Codificación target

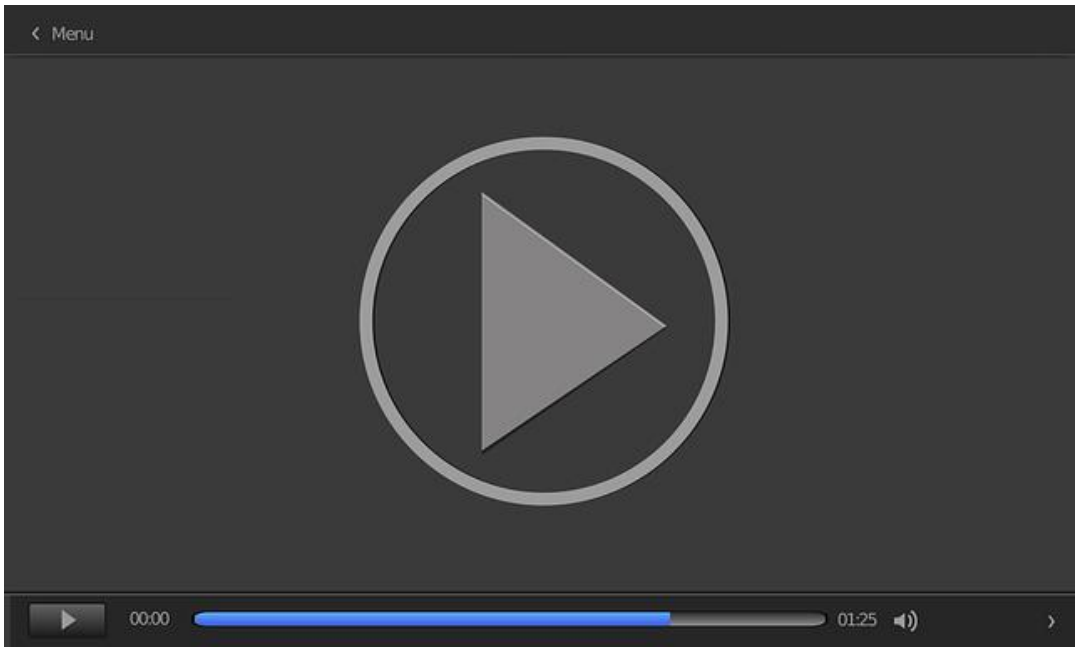
Se utiliza con variables categóricas de alta cardinalidad; es decir, características con muchas categorías únicas. Se calcula el valor objetivo promedio para cada categoría y este valor promedio es utilizado para reemplazar la característica categórica. Aunque toca utilizarla con precaución porque puede conducir a un sobreajuste.

```
# Binary Encoding:

import pandas as pd

# Crear un DataFrame de ejemplo
data = {'Categoría': ['A', 'B', 'A', 'C', 'B', 'C', 'A', 'A', 'B', 'C'],
        'Valor': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]}
df = pd.DataFrame(data)

# Definir las categorías y asignarles etiquetas
categorias = {'A': 'Grupo A', 'B': 'Grupo B', 'C': 'Grupo C'}

# Categorizar la columna 'Categoría'
df['Grupo'] = df['Categoría'].map(categorias)

# Agrupar por la nueva columna 'Grupo' y calcular la media de los valores
resultados = df.groupby('Grupo').mean()
```

Usamos el método `map()` para categorizar la columna `'Categoría'`, luego creamos una nueva columna llamada `'Grupo'`. Utilizamos `groupby` para agrupar por la nueva columna `'Grupo'` y calculamos la media de los valores en cada grupo utilizando el

método mean() .

A continuación, veremos el vídeo ***Preprocesamiento y transformación de datos en aprendizaje no supervisado*** con un ejemplo sencillo en python.



Accede al vídeo:

https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=ff372be3-ca92-4b2c-98ee-b1bd017637a2

1.10. Cuaderno de ejercicios

Explora y describe la estructura subyacente de los datos del conjunto de datos de precios de viviendas de Kaggle. Identifica y describe cualquier patrón o estructura que encuentres.

Yasser, M. (2022). *Housing Prices Dataset.* Kaggle. https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

Ejercicio 1: exploración y descripción de datos

Explora y describe la estructura que tienen los datos del conjunto de datos de precios de viviendas de Kaggle.

Solución

- Carga de datos y descripción general:

```
import pandas as pd
data = pd.read_csv('housing-prices-dataset.csv')
print(data.info())
print(data.describe())
```

El conjunto de datos contiene varias características como el precio, tamaño, número de habitaciones, etc.

Describe estadísticas básicas: medias, medianas, desviaciones estándar.

- Identificación de patrones:
  - Distribución de precios: utiliza histogramas para observar la distribución de precios.
  - Correlaciones: utiliza un mapa de calor *(heatmap)* para visualizar las correlaciones entre variables.

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.show()
```

Se pueden identificar variables altamente correlacionadas con el precio, como el tamaño de la vivienda y la ubicación.

Los precios tienden a agruparse en ciertas áreas geográficas, indicando patrones de agrupamiento.

Ejercicio 2: flujo de trabajo y retos

Diseña un flujo de trabajo paso a paso para abordar el análisis del conjunto de datos mencionado en el ejercicio 1. Identifica al menos uno de los retos del aprendizaje no supervisado.

Solución

Flujo de trabajo:

- Recolección de datos: obtener el conjunto de datos y consolidarlo.
- Preprocesamiento.
- Limpieza de datos (manejo de valores faltantes).
- Selección de características.
- Escalado de características.
- Aplicar algoritmos de agrupamiento: aplicar K-Means, DBSCAN o *clustering* jerárquico.
- Evaluación de resultados: usar métricas como el índice de la silueta.
- Reto.

- Maldición de la dimensionalidad: con conjuntos de datos de alta dimensionalidad es difícil encontrar grupos con características comunes. La reducción de dimensionalidad o selección de característcas puede ayudar, pero debe hacerse cuidadosamente para preservar la estructura de los datos.

Ejercicio 3: preprocesamiento de datos

Realiza un preprocesamiento completo del conjunto de datos mencionado en el ejercicio 1, incluyendo la limpieza de datos, eliminación de *outliers* y normalización.

Solución

- Cargar los datos:

```
import pandas as pd

# Cargar los datos
file_path = 'house_prices.csv' # Cambia esto por la ruta real de tu archivo
data = pd.read_csv(file_path)

# Mostrar las primeras filas del dataframe para verificar que se ha cargado
correctamente
print(data.head())
```

- Limpieza de datos:

```
# Manejo de valores faltantes
data = data.dropna()
```

- Identificación de *outliers:*

```
# Calcular el IQR para la columna 'price'
Q1 = data['price'].quantile(0.25)
Q3 = data['price'].quantile(0.75)
IQR = Q3 - Q1

# Definir los límites para identificar outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Identificar outliers
```

```
outliers = data[(data['price'] < lower_bound) | (data['price'] >
upper_bound)]

# Mostrar los outliers identificados
print(outliers)
```

- Eliminación de *otuliers:*

```
# Eliminar outliers
data_cleaned = data[(data['price'] >= lower_bound) &amp;amp; (data['price'] <=
upper_bound)]

# Mostrar las primeras filas del dataframe limpio
print(data_cleaned.head())

# Guardar el dataframe limpio en un nuevo archivo CSV
data_cleaned.to_csv('house_prices_cleaned.csv', index=False)
```

- Normalización:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data.select_dtypes(include=['float64',
'int64']))
```

Ejercicio 4: discretización y codificación de datos categóricos

Aplica técnicas de discretización y codificación de datos categóricos del conjunto de datos de precio de inmuebles de Kaggle para preparar los datos para su análisis con algoritmos de aprendizaje no supervisado.

```
data['price_discretized'] = pd.qcut(data['price'], q=4, labels=False)
#Codificación de datos categóricos:
data_encoded = pd.get_dummies(data, columns=['categorical_feature'])
```

1.11. Referencias bibliográficas

Jarapala, K. N. (2023, marzo 13). *Categorical Data Encoding Techniques.* Medium. https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f

Van Der Post, H. y Smith, M. (2023). *Unsupervised Machine Learning: with Python.* Reactive Publishing.

Unsupervised Learning Algorithms in Big Data: An Overview

Zhang, P. (2022). Unsupervised Learning Algorithms in Big Data: An Overview. En Holl, A., Chen, J. y Guan, G. (Eds.), *Proceedings of the 2022 5th International Conference on Humanities Education and Social Sciences (ICHESS 2022)* (pp. 910–931). https://www.atlantis-press.com/article/125983012.pdf

Este artículo proporciona una descripción general de los algoritmos no supervisados, sus ventajas y desventajas, además de las aplicaciones en los últimos años con el *big data.*

Study of Dimensionality Reduction Techniques and Interpretation of their Coefficients, and Influence on the Learned Models

García-Gutiérrez, M. A. (2023). *Study of Dimensionality Reduction Techniques and Interpretation of their Coefficients, and Influence on the Learned Models* [Trabajo fin de máster, Universidad Politécnica de Madrid]. ETSI Informáticos. https://oa.upm.es/75893/1/TFM_MIGUEL_ANGEL_GARCIA-GUTIERREZ_ESPINA.pdf

En este documento se analizan varias técnicas de reducción de dimensionalidad y su influencia en los modelos de aprendizaje automático.

Aprendizaje supervisado, no supervisado y por refuerzo

BitBoss. (2023, febrero 25). *Machine learning | aprendizaje supervisado, no supervisado y por refuerzo* [Vídeo]. YouTube. https://www.youtube.com/watch?v=FMkxSsfvMI4

Explica de forma muy dinámica la diferencia entre estos conceptos.

Test

1. ¿Qué es el aprendizaje no supervisado?

A. Un tipo de aprendizaje donde se proporcionan ejemplos de entrada junto con sus correspondientes salidas esperadas.

B. Un tipo de aprendizaje donde se utilizan datos no etiquetados para descubrir patrones o estructuras intrínsecas en los datos.

C. Un tipo de aprendizaje donde se utilizan datos etiquetados para entrenar un modelo.

D. Ninguna de las anteriores.

2. ¿Cuál de las siguientes afirmaciones es cierta sobre el aprendizaje no supervisado?

A. Se utiliza para predecir una variable de salida basada en variables de entrada.

B. No requiere la existencia de variables de entrada en los datos de entrenamiento.

C. Se utiliza para clasificar los datos en dos o más categorías predefinidas.

D. No requiere la existencia de variables de salida en los datos de entrenamiento, trabaja creando grupos para las instancias teniendo en cuenta las características.

3. ¿Cuál es uno de los principales objetivos del aprendizaje no supervisado?

A. Clasificación.

B. Regresión.

C. Exploración y descubrimiento de patrones.

D. Ninguna de las anteriores.

4. ¿Cuál de las siguientes técnicas se utiliza comúnmente en el aprendizaje no supervisado para reducir la dimensionalidad de los datos?

A. *Clustering.*

B. Regresión lineal.

C. Análisis de componentes principales (PCA).

D. Vectores y valores propios.

5. ¿Cuál de las siguientes afirmaciones describe mejor el proceso de *clustering* en el aprendizaje no supervisado?

A. Agrupar los datos en clústeres basados en similitudes entre ellos.

B. Dividir los datos en conjuntos de categorías predefinidas.

C. Asignar valores a las características de entrada para predecir una variable de salida.

D. Dividir los datos en conjuntos de igual tamaño ordenados por cualquier dimensión.

6. ¿Por qué es importante el preprocesamiento de datos en el aprendizaje no supervisado?

A. Para eliminar datos atípicos.

B. Para mejorar la calidad de los datos y facilitar el análisis.

C. Para aumentar la complejidad del modelo.

D. Ninguna de las anteriores.

7. ¿Qué tipo de datos se utilizan comúnmente en el aprendizaje no supervisado?

A. Datos no estructurados.

B. Datos categóricos.

C. Datos numéricos y categóricos.

D. Datos estructurados y no estructurados.

8. ¿Qué técnica se utiliza para convertir variables categóricas en una representación numérica en el aprendizaje no supervisado?

A. *Clustering.*

B. Transformación de datos.

C. Técnicas de visualización.

D. *Encoding* de datos categóricos.

9. ¿Cuál es uno de los retos del aprendizaje no supervisado?

A. La necesidad de etiquetas para cada ejemplo de entrenamiento.

B. La interpretación de los resultados obtenidos.

C. La falta de algoritmos disponibles.

D. El manejo de la maldición de la dimensionalidad.

10. ¿Cuál es uno de los beneficios del aprendizaje no supervisado?

A. La capacidad de predecir valores de salida precisos.

B. La capacidad de descubrir patrones y estructuras ocultas en los datos.

C. La capacidad de entrenar modelos con menor cantidad de datos.

D. Ninguna de las anteriores.