Preprocesamiento de Datos

Profesor: Juan Gamarra Moreno

Introducción al Preprocesamiento de Datos

Importancia del Preprocesamiento en ML

- El preprocesamiento de datos es una fase crítica en el desarrollo de cualquier modelo de *Machine Learning* porque los algoritmos de aprendizaje automático funcionan de manera más eficiente y efectiva cuando los datos están en una forma adecuada.

Importancia del Preprocesamiento en ML

1. **Mejora de la Calidad del Modelo**:
   Datos sin preprocesar pueden llevar a un mal ajuste del modelo (*overfitting* o *underfitting*) debido a inconsistencias, valores faltantes o escalas desbalanceadas en las variables.

Importancia del Preprocesamiento en ML

2. **Reducción de Sesgo y Ruido**:
   Eliminar o corregir datos erróneos, duplicados o valores atípicos ayuda a reducir el ruido que puede influir negativamente en el rendimiento del modelo.

Importancia del Preprocesamiento en ML

3. **Facilita la Eficiencia del Algoritmo**:
   Modelos como regresión lineal o redes neuronales se benefician de datos escalados y normalizados, mejorando la convergencia durante el entrenamiento.

Importancia del Preprocesamiento en ML

4. **Ajuste a Requisitos de los Algoritmos**:
   Muchos algoritmos de *Machine Learning* requieren que los datos estén en un formato específico, por ejemplo, variables numéricas o normalizadas. El preprocesamiento garantiza que los datos cumplan con estos requisitos.

Importancia del Preprocesamiento en ML

5. **Impacto en el Tiempo de Ejecución**:
   Datos más limpios y bien estructurados pueden reducir considerablemente el tiempo de entrenamiento y evaluación del modelo.

Impacto de los Datos Sucios

- Datos sucios, es decir, aquellos que contienen valores erróneos, incompletos, duplicados o fuera de rango, pueden tener un impacto negativo significativo en el rendimiento de un modelo de *Machine Learning*

Impacto de los Datos Sucios

1. **Modelos Inexactos o Engañosos**:
   Los algoritmos pueden aprender patrones incorrectos o sobreajustarse a datos atípicos. Por ejemplo, valores atípicos extremos pueden sesgar los resultados, conduciendo a predicciones poco precisas.

Impacto de los Datos Sucios

2. **Menor Capacidad de Generalización**:
   Si un modelo está entrenado en datos con ruido o errores, su capacidad para generalizar correctamente en nuevos conjuntos de datos se reduce considerablemente, lo que afecta su rendimiento en producción.

Impacto de los Datos Sucios

3. **Problemas de Convergencia**:
   Los datos que no están normalizados o que contienen escalas desbalanceadas pueden dificultar que los modelos, especialmente los basados en gradientes (como redes neuronales), converjan de manera eficiente.

Impacto de los Datos Sucios

4. **Resultados Sesgados**:
   Variables con valores faltantes no tratados pueden llevar a sesgos en el modelo, pues el algoritmo puede aprender patrones distorsionados, lo que afecta la equidad y la representatividad de las predicciones.

Impacto de los Datos Sucios

5. **Mayor Complejidad en la Interpretación**:
   Datos desorganizados o inconsistentes hacen que sea difícil interpretar los resultados del modelo, lo que puede complicar la toma de decisiones basadas en las predicciones.

Limpieza de Datos

Manejo de Datos Faltantes

Los datos faltantes son un problema común en cualquier conjunto de datos y, si no se manejan adecuadamente, pueden reducir la precisión de los modelos de *Machine Learning*. Existen varias estrategias para tratar los valores faltantes:

Manejo de Datos Faltantes

Eliminación de datos faltantes:
Si la cantidad de datos faltantes es pequeña, una opción puede ser eliminar las filas o columnas que contienen estos valores. Por ejemplo, si una columna tiene un 2% de valores faltantes, se puede optar por eliminar esas filas sin un gran impacto negativo en los resultados.

- *Ejemplo*:
  Imagina un dataset sobre el peso y la altura de un grupo de personas. Si algunas entradas no tienen valor para la altura (1-2%), podrías eliminarlas para evitar el sesgo que causaría el análisis con esos datos incompletos.

Manejo de Datos Faltantes

Imputación:
Cuando no se pueden eliminar datos faltantes, se pueden imputar los valores utilizando la media, mediana o moda de la columna correspondiente.

- *Ejemplo*:
  Si un conjunto de datos sobre el ingreso de personas tiene varios valores faltantes, puedes reemplazarlos con la media de los ingresos existentes. Si trabajas con datos categóricos, puedes imputar los valores faltantes usando la moda (la categoría más frecuente).

Manejo de Datos Faltantes

- **Técnicas Avanzadas (modelos predictivos)**:
  Si la distribución de los datos es compleja, es posible utilizar modelos predictivos, como regresión, árboles de decisión o algoritmos de *k-nearest neighbors* para predecir los valores faltantes.
- *Ejemplo*:
  En un conjunto de datos sobre ventas de una tienda, donde faltan algunos valores de ventas diarias, se puede utilizar un modelo predictivo basado en otras características (por ejemplo, el número de clientes o la temporada) para estimar las ventas faltantes.

Eliminación de Duplicados

Los datos duplicados pueden causar distorsiones significativas en los resultados del modelo, especialmente si se incluyen en el entrenamiento. La eliminación de duplicados es esencial para evitar que el modelo sea engañado por patrones repetitivos que no existen en los datos reales.

Eliminación de Duplicados

- **Identificación de duplicados**:
  Generalmente, los duplicados se identifican por la coincidencia exacta de los valores en todas las columnas. Sin embargo, puede ser necesario revisar solo ciertas columnas clave.
- *Ejemplo*:
  En un conjunto de datos de clientes de un banco, puede que algunas filas tengan la misma información para un cliente registrado dos veces por error. Estos registros duplicados podrían hacer que el banco sobrestime su número de clientes, por lo que deben ser eliminados.

Eliminación de Duplicados

- **Eliminación de duplicados**:
  Una vez identificados los duplicados, puedes optar por eliminarlos, manteniendo solo una entrada para cada observación.
- *Ejemplo*:
  En un estudio médico con pacientes registrados múltiples veces, si todos los datos coinciden, puedes eliminar los registros adicionales. Sin embargo, si hay diferencias sutiles, debes analizar si las diferencias representan una evolución en el tiempo o un error.

Transformación de datos

Los datos pueden venir en formatos inconsistentes o incorrectos, lo que afecta el análisis y la creación de modelos. La transformación de datos asegura que los datos sean coherentes y útiles para los modelos.

Transformación de datos

- **Corrección de formatos**:
  A veces, los datos pueden estar en formatos inadecuados, como fechas representadas de diferentes maneras o números almacenados como cadenas de texto. Estos datos deben ser convertidos a un formato adecuado.
- *Ejemplo*:
  En un conjunto de datos de transacciones de ventas, las fechas pueden estar registradas de formas diferentes, como "01/01/2023", "January 1, 2023", o "2023-01-01". Es necesario unificar estos formatos a un estándar, como "YYYY-MM-DD", para evitar confusiones durante el análisis.

Transformación de datos

- **Conversión de tipos de datos**:
  Algunas veces, variables numéricas pueden estar almacenadas como texto o viceversa, lo que puede causar problemas al aplicar modelos. Convertir los tipos de datos a su formato adecuado es una transformación esencial.
- *Ejemplo*:
  Si una columna que contiene la cantidad de productos vendidos está guardada como texto en lugar de números, debes convertirla a valores numéricos para que las operaciones matemáticas puedan realizarse sin errores.

Transformación de datos

- **Manejo de Outliers**:
  Los outliers (valores atípicos) son puntos de datos que difieren significativamente del resto y pueden afectar negativamente a los modelos. Se pueden eliminar, ajustar o transformar los outliers para que no distorsionen los resultados.
- *Ejemplo*:
  En un conjunto de datos de ingresos de empleados, puedes tener un pequeño grupo de ejecutivos con ingresos significativamente mayores que la mayoría de los empleados. Estos outliers pueden sesgar los análisis de distribución, por lo que podrías decidir eliminarlos o transformarlos para obtener un análisis más representativo.

Normalización y Estandarización

Normalización

La normalización es el proceso de reescalar las características de un conjunto de datos para que se ajusten dentro de un rango específico, comúnmente entre 0 y 1. Este proceso es útil cuando las diferentes características tienen diferentes unidades o rangos, y se desea que todas contribuyan equitativamente al modelo de *Machine Learning*.

Normalización

- **Cuándo usar normalización**:
  Es especialmente útil cuando se emplean algoritmos basados en distancias, como *k-nearest neighbors* (KNN) o *Support Vector Machines* (SVM), donde las diferencias de escala pueden influir en el cálculo de distancias y, por lo tanto, en el rendimiento del modelo.

Normalización

Ejemplo descriptivo:
Considera un conjunto de datos que tiene dos características: la edad de las personas (rango de 18 a 80 años) y sus ingresos anuales (rango de $20,000 a $500,000). Debido a las diferencias significativas entre estos rangos, la característica "ingresos" tendría un peso desproporcionado en cualquier modelo que utilice distancias. Al normalizar ambas características entre 0 y 1, la edad y los ingresos tendrían un impacto proporcional en el modelo.

- **Edad antes de normalizar**: 18 - 80
- **Ingresos antes de normalizar**: 20,000 - 500,000

Después de la normalización:

- **Edad**: 0 - 1
- **Ingresos**: 0 - 1

De esta forma, ambas características tienen la misma escala y el modelo puede procesarlas de manera equitativa.

Estandarización

- La estandarización es otra técnica de escalado que transforma los datos para que tengan una media de 0 y una desviación estándar de 1. A diferencia de la normalización, que ajusta los valores dentro de un rango fijo, la estandarización se utiliza cuando se quiere mantener la variabilidad de los datos, pero ajustarlos a una escala común.

Estandarización

Ejemplo descriptivo:
Supón que tienes un conjunto de datos con el peso y la altura de varias personas. El peso tiene una media de 70 kg y una desviación estándar de 15 kg, mientras que la altura tiene una media de 1.70 metros y una desviación estándar de 0.15 metros. Estas características tienen diferentes escalas, lo que podría afectar el modelo.

Al estandarizar los datos:

- **Peso antes de estandarizar**: 50 kg - 100 kg (con una media de 70 kg)
- **Altura antes de estandarizar**: 1.50 m - 1.90 m (con una media de 1.70 m)

Después de la estandarización:

- **Peso**: los valores estarán alrededor de 0 (con una desviación estándar de 1).
- **Altura**: de manera similar, los valores estarán centrados en 0, con una desviación estándar de 1.

Esto significa que el peso y la altura están en la misma escala y tendrán un impacto similar en el modelo, lo que ayuda a algoritmos que dependen de la varianza y la correlación entre características.

Diferencias clave entre normalización y estandarización

- **Rango**:
  - Normalización: ajusta los valores dentro de un rango específico, como [0, 1].
  - Estandarización: centra los valores alrededor de 0 con una desviación estándar de 1.
- **Uso en algoritmos**:
  - Normalización: es ideal para algoritmos basados en distancias, como *k-nearest neighbors* o *Support Vector Machines*.
  - Estandarización: es preferida en algoritmos que suponen que los datos tienen una distribución normal o donde la varianza es importante, como la regresión lineal o PCA.

Codificación de Variables

Introducción

- En *Machine Learning*, la mayoría de los algoritmos requieren que las variables sean numéricas, lo que significa que las variables categóricas deben ser convertidas en números para poder ser utilizadas por los modelos. Existen varias técnicas para realizar esta conversión, dependiendo del tipo de variable categórica.

One-Hot Encoding

Descripción:
One-Hot Encoding transforma cada categoría única en una nueva columna binaria (0 o 1). Esta técnica es útil cuando no hay ningún tipo de relación de orden entre las categorías y todas deben ser tratadas de manera independiente. Cada categoría se representa con una columna adicional, y solo una de esas columnas tendrá un valor de 1 para cada fila del dataset.

One-Hot Encoding

Ejemplo:
Imagina que tienes una columna de "Color" en tu dataset con tres valores: Rojo, Azul y Verde. Al aplicar One-Hot Encoding, se crean tres columnas adicionales: "Color_Rojo", "Color_Azul", "Color_Verde", y para cada fila solo una de esas columnas tendrá el valor 1.

Antes del One-Hot Encoding:

- Color: Rojo, Azul, Verde

Después del One-Hot Encoding:

- Color_Rojo: 1, 0, 0
- Color_Azul: 0, 1, 0
- Color_Verde: 0, 0, 1

One-Hot Encoding

Impacto en los modelos:

- Ventajas: El One-Hot Encoding es ideal cuando no hay relación de orden entre las categorías, ya que evita introducir relaciones incorrectas en el modelo.
- Desventajas: Si tienes muchas categorías, puede aumentar considerablemente el número de columnas (dimensionalidad), lo que podría afectar el rendimiento del modelo.

Label Encoding

Descripción:
Label Encoding asigna un número entero único a cada categoría. En lugar de crear nuevas columnas, simplemente se reemplaza cada categoría por un valor numérico. Es una técnica eficiente en términos de espacio, ya que no añade dimensionalidad extra.

Label Encoding

Ejemplo descriptivo:
En lugar de crear columnas adicionales como en el One-Hot Encoding, la columna "Color" simplemente sería reemplazada con números:

Antes del Label Encoding:

- Color: Rojo, Azul, Verde

Después del Label Encoding:

- Color: 1, 2, 3

Label Encoding

Impacto en los modelos:

- Ventajas: El Label Encoding es simple y eficiente, ya que no aumenta la dimensionalidad del dataset.
- Desventajas: Esta técnica puede inducir un sesgo de orden en los modelos que utilizan los valores numéricos, ya que los algoritmos podrían interpretar que hay una relación de magnitud entre los números asignados a las categorías (por ejemplo, que "Rojo" es menor que "Azul"). Por esta razón, es más adecuada cuando las categorías tienen algún tipo de orden.

Codificación Ordinal

Las variables ordinales son aquellas que tienen un orden natural, como niveles de educación (Primaria, Secundaria, Universitaria) o calificaciones (Bajo, Medio, Alto). A diferencia de las variables categóricas, el orden de estas categorías tiene un significado y debe ser conservado al codificarlas.

Técnicas para Variables Ordinales

- En este caso, se utiliza una codificación ordinal simple, asignando un número que refleje el orden de las categorías.

Codificación Ordinal

Ejemplo:
Imagina que tienes una columna "Nivel de Educación" con tres valores: Primaria, Secundaria y Universitaria. Dado que existe un orden natural en estos niveles, podrías asignar los números 1, 2 y 3 de acuerdo con el nivel jerárquico:

Antes de la codificación:

- Nivel de Educación: Primaria, Secundaria, Universitaria

Después de la codificación:

- Nivel de Educación: 1, 2, 3

Codificación Ordinal

Impacto en los modelos:

- Ventajas: Al preservar el orden natural de las categorías, los modelos podrán aprovechar esta información para realizar predicciones más precisas.
- Desventajas: Esta técnica no es adecuada para variables que no tienen un orden natural, ya que podría inducir un sesgo incorrecto en el modelo.

Codificación

Resumen:

- **One-Hot Encoding**: Se utiliza cuando las variables no tienen un orden y se desea crear columnas adicionales que representen cada categoría de manera binaria.
- **Label Encoding**: Asigna un número entero a cada categoría, pero debe ser usado con precaución en variables sin orden natural para evitar introducir relaciones incorrectas.
- **Codificación Ordinal**: Se utiliza para variables con un orden natural, preservando la jerarquía entre las categorías.

Cada técnica tiene su contexto de aplicación, y es importante seleccionar la más adecuada para evitar que los modelos de *Machine Learning* hagan suposiciones incorrectas sobre los datos.