


Machine Learning Models

Unidad 1 Preprocesado

Transformación de variables

- ❑ 1. Normalización de variables numéricas
 - ❑ 2. Conversión de variables categóricas a numéricas
- 

1. Normalización de variables numéricas



Normalización

Introducción

- La **normalización** es una técnica que se utiliza para cambiar los valores de las columnas numéricas en el conjunto de datos a una escala común, sin distorsionar las diferencias en los rangos de valores
- En **Machine Learning**, a menudo se normalizan las características (o variables de entrada) para asegurarse de que todas ellas sean evaluadas en una escala similar
- El **objetivo** es evitar que las características con rangos de valores más grandes influyan desproporcionadamente en el modelo en comparación con las características con rangos de valores más pequeños
- Hay varios tipos de normalización, nos centraremos en dos: **estandarización** y Min-Max Scaling (**escalado 0-1**)
- **Ejemplo**: supón que tienes un conjunto de datos con dos características: la edad (que varía de 0 a 100) y el ingreso anual (que varía de 0 a 100.000). Un modelo de aprendizaje automático podría considerar erróneamente que el ingreso es una característica más importante simplemente debido a su mayor magnitud



Normalización

Min-Max scaling

- Una forma común de normalización es la llamada "**Min-Max Scaling**", que escala todos los valores de una variable / columna para que estén entre 0 y 1
- La fórmula es: $X_{esc} = \frac{X - X_{min}}{X_{max} - X_{min}}$ donde X_{min} es el valor mínimo de los datos de esa variable y X_{max} es el valor máximo
- Esta fórmula se aplica a cada valor de la variable y el resultado es que todos los datos de la variable están entre 0 y 1 (y ambos, son tomados, lógicamente)

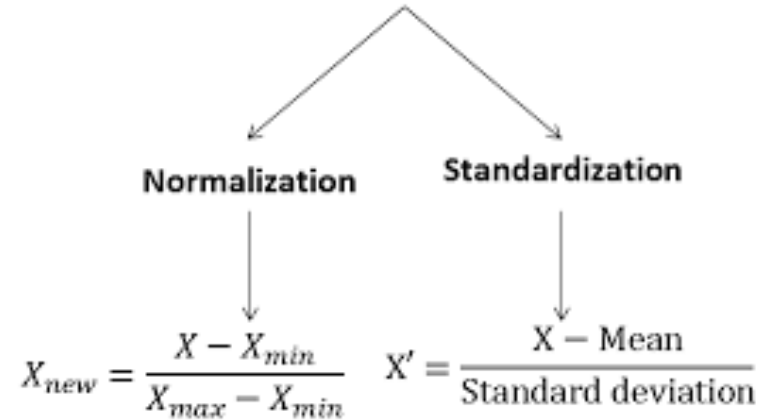



Normalización Estandarización

- Otra forma también muy común (como sabéis) de normalizar una variable es estandarizarla. Es decir, convertirla en una variable de media cero y desviación estándar 1
- La fórmula es: $X_{est} = \frac{X - \bar{X}}{S_X}$ donde \bar{X} es la media de la variable X y S_X es la desviación estándar de la variable X
- Esta fórmula se aplica a cada valor de la variable y el resultado es que esa variable tiene media 0 y desviación estándar 1
- En este caso la media puede no ser un dato existente y por lo tanto en los datos estandarizados puede no existir ningún valor de cero
- Recordemos que la interpretación de un valor (puntuación) estandarizada se obtiene evaluando si está por encima de la media (valor mayor a 0) o por debajo de la media (menor a cero) o igual que la media (igual a cero). Además, lo que te separa de la media está en unidades de su desviación estándar
- Ejemplo de interpretación: Una puntuación estandarizada de $X=2$ es un valor por encima de la media de X en 2 desviaciones estándar de esa variable X

Normalización– Resumen


Feature scaling





Normalización EN MACHINE LEARNING

- **IMPORTANTE**
- Hay que distinguir dos conceptos en la aplicación de la normalización en MACHINE LEARNING:
 - **Construcción** del normalizador en base a unos datos
 - Tomamos unos datos
 - Construimos el normalizador con esos datos
- SÓLO Calculamos lo que necesitamos para normalizar
 - Mínimo y máximo para el Min-Max Scaling
 - Media y Desviación estándar para la estandarización
- **Aplicación** del normalizador ya creado a unos datos
- Tomamos unos datos (los que han servido para construirlo u otros datos nuevos que no han servido para construir la formula, el normalizador)
- Aplicamos el normalizador ya construido sobre esos datos (los mismos u otros)



Normalización

PYTHON I


Crear objeto

- **CREACIÓN** del objeto capaz de normalizar
 - Min-Max Scaler:
 - `from sklearn import preprocessing`
 - `escalador = preprocessing.MinMaxScaler()`
 - Estandarización
 - `from sklearn import preprocessing`
 - `estandarizador = preprocessing.StandardScaler()`



Normalización PYTHON II Construir

- **CONSTRUCCIÓN** del normalizador en base a unos datos
 - Min-Max Scaler:
 - `escalador.fit(df[['Ingreso']])`
 - `escalador.fit(df[['Ingreso', 'gasto']])`
 - Estandarización
 - `estandarizador.fit(df[['Ingreso']])`
 - `estandarizador.fit(df[['Ingreso', 'gastos']])`




Normalización

PYTHON III

Aplicar

- APLICACIÓN del normalizador a unos datos
 - Min-Max Scaler:
 - `escalador.transform(df2[['Ingreso']])`
 - `escalador.transform(df2[['Ingreso', 'gasto']])`
 - Estandarización
 - `estandarizador.transform(df2[['Ingreso']])`
 - `estandarizador.transform(df2[['Ingreso', 'gasto']])`



Normalización

PYTHON IIII

Construir + Aplicar

- **CONSTRUCCIÓN + APLICACIÓN**
- Podemos crear y aplicar sobre unos mismos datos todo a la vez
 - Min-Max Scaler:
 - `escalador.fit_transform(df2[['Ingreso']])`
 - `escalador.fit_transform(df2[['Ingreso', 'gasto']])`
 - Estandarización
 - `estandarizador.fit_transform(df2[['Ingreso', 'gasto']])`

1. Conversión de variables categóricas a numéricas



Conversión a numérica Introducción

- La mayoría de algoritmos de ML necesitan trabajar con **variables numéricas**. Por ello habremos de convertir las variables categóricas a variables numéricas **de forma inteligible** para los algoritmos
- Hay dos formas principales de convertir variables categóricas en numéricas:
 - **One-Hot encoding**. Esta técnica genera tantas variables dicotómicas como categorías tiene la variable a convertir
 - **Ordinal encoding**. Esta técnica asigna un valor numérico a cada categoría.

Conversión a numérica One-Hot encoding

One-Hot encoding. Esta técnica genera tantas **variables dicotómicas** como categorías tiene la variable a convertir

SECTOR	Automoción	Turismo	Editorial
Automoción	1	0	0
Turismo	0	1	0
Turismo	0	1	0
Editorial	0	0	1
Turismo	0	1	0
Automoción	1	0	0

Conversión a numérica

One-Hot encoding

PYTHON

```
import pandas as pd
from sklearn.preprocessing import OneHotEncoder
dic={'Sector':["Automocion","Turismo","Turismo","Editorial","Turismo","Automocion"],
    "Ingresos": [1000,2000,300,150,2000,900]}
df=pd.DataFrame(dic)
df
```

	Sector	Ingresos
0	Automocion	1000
1	Turismo	2000
2	Turismo	300
3	Editorial	150
4	Turismo	2000
5	Automocion	900

Conversión a numérica

One-Hot encoding PYTHON

```
[38] one_hot=pd.get_dummies(df, columns=['Sector'])  
one_hot
```

	Ingresos	Sector_Automocion	Sector_Editorial	Sector_Turismo
0	1000	1	0	0
1	2000	0	0	1
2	300	0	0	1
3	150	0	1	0
4	2000	0	0	1
5	900	1	0	0

Conversión a numérica Ordinal encoding

Ordinal encoding. Esta técnica asigna un número a cada categoría. SOLO EN CASOS DONDE EL ORDEN TENGA SENTIDO y hay que mantenerlo

NIVEL ESTUDIOS		NIVEL ESTUDIOS
GRADO		3
ESO		1
ESO	→	1
BATX		2
ESO		1
GRADO		3

Conversión a numérica Ordinal encoding PYTHON

```
import pandas as pd  
dic={'NivelEstudios': ["ESO", "GRADO", "ESO", "ESO", "BATX", "ESO", "GRADO"],  
     "Ingresos": [1000, 2000, 300, 150, 2000, 900, 2000]}  
df=pd.DataFrame(dic)  
df
```



	NivelEstudios	Ingresos
0	ESO	1000
1	GRADO	2000
2	ESO	300
3	ESO	150
4	BATX	2000
5	ESO	900
6	GRADO	2000



Conversión a numérica Ordinal encoding PYTHON



```
from sklearn.preprocessing import OrdinalEncoder  
le = OrdinalEncoder(categories=[['ESO', 'BATX', 'GRADO']], dtype=int)  
a=le.fit_transform(df[['NivelEstudios']])  
df['NivelEstudios_codificado']=a  
df
```

	NivelEstudios	Ingresos	NE	NivelEstudios_codificado
0	ESO	1000	0	0
1	GRADO	2000	2	2
2	ESO	300	0	0
3	ESO	150	0	0
4	BATX	2000	1	1
5	ESO	900	0	0
6	GRADO	2000	2	2



Nota: estamos fitando y aplicando a la vez (fit_transform)