



FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

Programación para Inteligencia Artificial

Grupo: 001 | Equipo: 09

Gestión y Normalización de Datos con Python

Actividad Fundamental 4

Fecha: 10/10/2025

Profesor: Dr. Erick de Jesús Ordaz Rivas

Integrantes

Nombre	Matrícula	Hora clase
Orlando Alvarado Vargas	2226968	V1
Diego Alonso Carrillo Castillo	2144556	V1

Índice

1. Introducción	3
2. Desarrollo	3
3. Solución	3
4. Resultados Obtenidos	8
5. Conclusiones	9

1. Introducción

Normalización de un archivo CSV, lo cual implica ajustar y estructurar la información para garantizar que los datos sean uniformes, precisos y sencillos de procesar y analizar. Esto permite que los datos sean más claros y útiles para el procesamiento.

2. Desarrollo

El archivo de datos **Airbnb_site_hotel_new.csv** presenta diversos problemas que impiden analizar la información de manera eficiente.

- Datos numéricos almacenados como texto o con símbolos.
- Filas con valores nulos o incompletos.
- Columnas irrelevantes o no útiles para el análisis.
- Variables categóricas sin codificar.

3. Solución

3.1. Composición del programa

- main.py
- gestion.py
- normalizacion.py
- train.py

Archivo "main.py":

```
1 import pandas as pd
2 import numpy as np
3
4 import gestion as gt
5 import normalizacion as nz
6 import train as tr
7
8 print("\n##### INICIANDO PROGRAMA #####\n")
9
10 ndf = gt.load_clean_data()      # Funcion para cargar y limpiar la base de
11      datos
12
13 df_normalizado = nz.normalizar(ndf)      # Funcion para normalizar la base
14      de datos
15
16 tr.entrenar(df_normalizado)      # Funcion para entrenar el modelo
```

```
16 gt.exportar(df_normalizado) # Funcion para exportar el DataFrame  
17     normalizado a un archivo CSV  
18 print("\n##### PROGRAMA TERMINADO #####\n")
```

Archivo "normalización.py":

```
1 import pandas as pd  
2 import numpy as np  
3  
4 from sklearn.preprocessing import LabelEncoder, OneHotEncoder, MinMaxScaler  
5 from sklearn.model_selection import train_test_split  
6  
7 def normalizar(ndf):  
8     print("\n*****NORMALIZANDO BASE DE DATOS*****\n")  
9     print("...Generando Dummies...\n")  
10    # Se utilizo la tecnica de One-Hot Encoding para las columnas  
11        categoricas "city" y "area" y que se puedan normalizar sin problema  
12    # Esto crea nuevas columnas binarias (0 y 1) para cada categoria en  
13        las columnas originales donde si es 1 significa que pertenece a esa  
14        categoria y 0 que no  
15    dumisc = pd.get_dummies(ndf['city']).astype(int)  
16    dumisa = pd.get_dummies(ndf['area']).astype(int)  
17  
18    print("...Integrando Dummies...\n")  
19    # Al utilizar One-Hot Encoding, se crean muchas columnas nuevas, por  
20        lo que es necesario agregarlas al data set original y eliminar las  
21        columnas originales  
22    # Se agregan los Dummies en el data set  
23    newndf = ndf.join([dumisc, dumisa])  
24  
25    # Se eliminan las columnas originales de "city" y "area" porque ya no  
26        nos sirven  
27    newndf = newndf.drop(columns=['city', 'area'])  
28  
29    print("...Normalizando datos numericos...\n")  
30    # Se normalizan todos los datos numericos para que esten en un rango  
31        de 0 a 1  
32    scaler = MinMaxScaler()  
33    df_normalizado = pd.DataFrame(scaler.fit_transform(newndf), columns=  
34        newndf.columns)  
35    print("-----BASE DE DATOS NORMALIZADA-----")  
36  
37    return df_normalizado
```

Archivo "gestion.py":

```
1 import pandas as pd  
2 import numpy as np  
3  
4 # Funcion que sirve para cargar y limpiar los datos de la base de datos  
5 def load_clean_data(file_path = "airbnb.csv"):
```

```
6     print("Cargando la base de datos...\n")
7
8     datos = pd.read_csv("airbnb.csv", low_memory=False)      # Leer el
9         archivo .csv
10
11    print("...Limpiando la base de datos...\n")
12    # Toda la tabla utiliza "," en lugar de "."
13    # Ademas de que muchas columnas que deberian ser numericas estan como
14        objetos (strings)
15
16    # La columna "price" Se tiene que cambiar primero a string para poder
17        eliminar los caracteres "$" y "," y finalmente a float
18    datos['price'] = datos['price'].astype(str)
19    datos['price'] = datos['price'].str.replace('$', '')
20    datos['price'] = datos['price'].str.replace(',', '')
21    datos['price'] = datos['price'].astype(float)
22
23    # Las columnas "consumer", "bathrooms", "host response rate" y "host
24        acceptance rate" solo necesitan cambiar "," por "." y luego a float
25
26    datos['consumer'] = datos['consumer'].str.replace(',', '.')
27    datos['consumer'] = datos['consumer'].astype(float)
28
29    datos['bathrooms'] = datos['bathrooms'].str.replace(',', '.')
30    datos['bathrooms'] = datos['bathrooms'].astype(float)
31
32    datos['host response rate'] = datos['host response rate'].str.replace
33        (',', '.')
34    datos['host response rate'] = datos['host response rate'].astype(float)
35
36    datos['host acceptance rate'] = datos['host acceptance rate'].str.
37        replace(',', '.')
38    datos['host acceptance rate'] = datos['host acceptance rate'].astype(
39        float)
40
41    ndf = datos.dropna()      # Esta funcion Quita las filas que tienen
42        valores nulos
43
44
45
46    def exportar(df_normalizado):
47        print("\n***** EXPORTANDO BASE DE DATOS NORMALIZADA
48            *****\n")
```

```

48     df_normalizado.to_csv('airbnb_normalizado.csv', index=False)
49     print("DataFrame guardado exitosamente como 'airbnb_normalizado.csv'")

```

Archivo "train.py":

```

1 import pandas as pd
2 import numpy as np
3
4 from sklearn.preprocessing import LabelEncoder, OneHotEncoder, MinMaxScaler
5 from sklearn.model_selection import train_test_split
6
7 def entrenar(df_normalizado):
8     print("\n*****ENTRENANDO MODELO*****\n")
9     columnas = ['reply time', 'guest favourite', 'host since', 'host
10         Certification', 'room_type', 'host total listings count', 'consumer',
11         'total reviewers number', 'accommodates', 'bathrooms', 'bedrooms', 'beds
12         ', 'listing number', 'host response rate', 'sales', 'NewYork', 'Toronto',
13         'sydney', 'North America']
14
15     # Se definen X y Y
16     x = df_normalizado[columnas].values
17     y = df_normalizado['price'].values
18
19     # Se dividen los datos en Training Set (60%), Validation Set (20%) y
20     # Testing Set (20%)
21     x_train, x_temp, y_train, y_temp = train_test_split(x, y, test_size=0.2,
22             random_state=42)
23     x_val, x_test, y_val, y_test = train_test_split(x_temp, y_temp,
24             test_size=0.5, random_state=42)
25
26     print("...Dividiendo datos...\n")
27     print("...Entrenando modelo...\n")
28
29     print(f"Tamaño de Training Set: {len(x_train)}")
30     print(f"Tamaño de Validation Set: {len(x_val)}")
31     print(f"Tamaño de Testing Set: {len(x_test)}")
32
33     print("\nX_train (primeras filas):\n", x_train[:3])
34     print("y_train:", y_train)
35
36     print("----- Modelo entrenado exitosamente -----")

```

3.2. Solución de problemas

- **Problema 1:** Datos numéricos almacenados como texto o con símbolos

```

1 # Toda la tabla utiliza "," en lugar de "."
2 # Ademas de que muchas columnas que deberian ser numericas estan como
3     objetos (strings)

```

```

4 # La columna "price" Se tiene que cambiar primero a string para poder
   eliminar los caracteres "$" y "," y finalmente a float
5
6 datos['price'] = datos['price'].astype(str)
7 datos['price'] = datos['price'].str.replace('$', '')
8 datos['price'] = datos['price'].str.replace(',', '')
9 datos['price'] = datos['price'].astype(float)
10
11 # Las columnas "consumer", "bathrooms", "host response rate" y "host
    acceptance rate" solo necesitan cambiar "," por "." y luego a float
12
13 datos['consumer'] = datos['consumer'].str.replace(',', '.')
14 datos['consumer'] = datos['consumer'].astype(float)
15
16 datos['bathrooms'] = datos['bathrooms'].str.replace(',', '.')
17 datos['bathrooms'] = datos['bathrooms'].astype(float)
18
19 datos['host response rate'] = datos['host response rate'].str.replace
    (',,', '..')
20 datos['host response rate'] = datos['host response rate'].astype(float)
21
22 datos['host acceptance rate'] = datos['host acceptance rate'].str.
    replace(',,', '..')
23 datos['host acceptance rate'] = datos['host acceptance rate'].astype(
    float)

```

■ **Problema 2:** Filas con valores nulos.

```

1 # Al ver la informacion de la base de datos, notamos que hay columnas
   que tienen valores nulos, con la siguiente funcion quitamos esas
   filas
2 # Guardamos el resultado en una nueva variable: ndf(New data frame)
3
4 ndf = datos.dropna()

```

■ **Problema 3:** Columnas no relevantes para el análisis

```

1 # Finalmente, eliminamos las columnas que no aportan informacion
   relevante para el analisis o que no se pueden normalizar
2 ndf = ndf.drop(columns=['id', 'name', 'host_id', 'host_name'])
3
4 print(f"Numero de filas originales: {len(datos)}")
5 print(f"Numero de filas DESPUES de eliminar nulos: {len(ndf)}")

```

■ **Problema 4:** Variables categóricas sin codificar.

```

1 # Se utilizo la tecnica de One-Hot Encoding para las columnas
   categoricas "city" y "area" y que se puedan normalizar sin problema
2 # Esto crea nuevas columnas binarias (0 y 1) para cada categoria en
   las columnas originales donde si es 1 significa que pertenece a esa
   categoria y 0 que no
3 dumisc = pd.get_dummies(ndf['city']).astype(int)

```

```

4 | dumisa = pd.get_dummies(ndf['area']).astype(int)
5 | # Al utilizar One-Hot Encoding, se crean muchas columnas nuevas, por
|   lo que es necesario agregarlas al data set original y eliminar las
|   columnas originales
6 | # Se agregan los Dummies en el data set
7 | newndf = ndf.join([dumisc, dumisa])
8 |
9 | # Se eliminan las columnas originales de "city" y "area" porque ya no
|   nos sirven
10| newndf = newndf.drop(columns=['city', 'area'])

```

4. Resultados Obtenidos

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	
1	price	reply_time	guest_favourit	host_since	host	Certific	room_type	host_total	list_consumer	total_reviews	accommodations	bedrooms	beds	listing_number	host_respons	host_accepts	sales	NewYork	Toronto	sydney	North America
2	0.012006	0	0	0.79467085	0	0.66666667	0	0.97285714	0.0406982	0.33333333	0.13043478	0.11111111	0.11111111	0	0.9	0.42	0.27945205	0	1	0	0
3	0.00710355	0	0	0.89167537	0	0.66666667	0.00478469	0.95285714	0.03580563	0.2	0.08956562	0.05555556	0.07407407	0.00083039	0.9	0.42	0.45959452	0	1	0	0
4	0.00550275	0	0	0.78962034	0	0.33333333	0.00478469	0.95285714	0	0	0.08956562	0.05555556	0.07407407	0.00083039	1	0.65	0.45959452	0	1	0	0
5	0.00325163	1	1	0.89010798	0	0.66666667	0	0.98	0.02472293	0	0.08956562	0	0.03703704	0	1	1	1	0	1	0	0
6	0.004002	0	0	0.87652386	0	0.66666667	0	0.93857143	0.0059676	0.06666667	0.05555556	0.053703704	0.0062279	1	0.8	0.89863014	0	1	0	0	
7	0.0046023	1	1	0.7830024	0	0.33333333	0.00119617	0.96857143	0.10315431	0	0.08956562	0.05555556	0.03703704	0.0004152	1	1	0.39452055	0	1	0	0
8	0.00250125	0	0	0.80651341	0	0.33333333	0.00857143	0.90857143	0.07246377	0.06666667	0.08956562	0.05555556	0	0	0	0	0	1	0	0	0
9	0.00508029	0	0	0.87338906	0	0.66666667	0.01196172	0.89	0.05626598	0.06666667	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0.16963031	0	1	0	0
10	0.01205603	1	1	0.77446863	0	0.66666667	0.00239234	0.93775142	0.05456995	0.4	0.13043478	0.16666667	0.14814818	0.0004152	1	0.9	0.24383562	0	1	0	0
11	0.00470235	0	0	0.87338906	0	0.33333333	0.01196172	0.87571429	0.0682012	0.06666667	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0	0	1	0	0
12	0.01755878	0	0	0.80703588	0	0.66666667	0	0.92	0.00767268	0.06666667	0.08956562	0.05555556	0.03703704	0	0	0	0.01917808	0	1	0	0
13	0.0070353	0	0	0.87338906	0	0.66666667	0.01196172	0.88714286	0.07331628	0.13333333	0.08956562	0	0.03703704	0.00207598	1	0.71	0.24109589	0	1	0	0
14	0.01755878	0	0	0.77338906	0	0.33333333	0.01196172	0.93	0.0767268	0.06666667	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0.44931507	0	1	0	0
15	0.00400245	1	1	0.77230365	0	0.33333333	0.01196172	0.98571429	0.10315431	0.06666667	0.08956562	0.05555556	0.03703704	0	1	0.94	0.16963031	0	1	0	0
16	0.004002	0	0	0.87338906	0	0.33333333	0.01196172	0.88714286	0.01875533	0	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0.47122388	0	1	0	0
17	0.00250125	1	1	0.85370951	0	0.33333333	0.00837321	0.98428571	0.00341096	0	0.08956562	0.05555556	0.03703704	0.00207598	1	0.97	0.0630137	0	1	0	0
18	0.00765383	0	0	0.87338906	0	0.66666667	0.01196172	0.93142857	0.05029838	0.13333333	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0.28493151	0	1	0	0
19	0.00250125	0	0	0.77098572	0	0.33333333	0	0.98285714	0.070673487	0.06666667	0.08956562	0.05555556	0.03703704	0.0062279	0	0	0.99726027	0	1	0	0
20	0.00525263	0	0	0.77307558	0	0.66666667	0	0.98	0.01790281	0.06666667	0.08956562	0.05555556	0.03703704	0	0	0	0.01917808	0	1	0	0
21	0.002001	1	1	0.85370951	0	0.33333333	0.00837321	0.96428571	0.01960784	0	0.04347826	0.05555556	0.03703704	0.00207598	1	0.97	0.56438356	0	1	0	0
22	0.00350175	1	1	0.85370951	0	0.66666667	0.00837321	0.92857143	0.00767268	0	0.04347826	0.05555556	0.03703704	0.00207598	1	0.97	0.47945205	0	1	0	0
23	0.002001	1	1	0.85370951	0	0.33333333	0.00837321	0.91571429	0.01449275	0	0.04347826	0.05555556	0.03703704	0.00207598	1	0.97	0.30684932	0	1	0	0
24	0.002001	1	1	0.85370951	0	0.33333333	0.00837321	0.94428571	0.02046036	0	0.04347826	0.05555556	0.03703704	0.00207598	1	0.97	0.61917808	0	1	0	0
25	0.004002	0	0	0.87338906	0	0.33333333	0.01196172	0.94285714	0.00426257	0.06666667	0.08956562	0.05555556	0.03703704	0.00207598	1	0.71	0.4739726	0	1	0	0
26	0.002001	1	1	0.85370951	0	0.33333333	0.00837321	1	0.00341096	0	0.08956562	0.05555556	0.03703704	0.00207598	1	0.97	0.31506849	0	1	0	0
27	0.01495748	1	1	0.82965023	0	0.66666667	0	0.99	0.01312673	0.02666667	0.1791304	0.16666667	0.11111111	0	1	0.88	0.91232877	0	1	0	0
28	0.0010005	1	1	0.77342389	0	0.33333333	0.00478469	0.91428571	0.03665814	0	0.13043478	0.05555556	0.03703704	0.0083039	1	0.86	0.1506849	0	1	0	0
29	0.00950475	1	1	0.87669801	0	0.66666667	0.00558852	0.96857143	0.07161125	0.06666667	0.08956562	0.05555556	0.03703704	0.0062279	1	1	0.39178082	0	1	0	0
30	0.0026013	1	1	0.76765092	0	0.33333333	0	0.92428571	0.01656434	0.06666667	0.08956562	0.05555556	0	0	1	0.19726027	0	1	0	0	
31	0.00295148	0	0	0.84186695	0	0.66666667	0.02033493	0.95142857	0.0404682	0.06666667	0.08956562	0.05555556	0.03703704	0.00373677	1	0.74	0.39726027	0	1	0	0
32	0.01345673	1	1	0.88214907	0	0.66666667	0	0.97571429	0.05370844	0.2	0.08956562	0.11111111	0.07407407	0.0083039	1	0.96	0.14520548	0	1	0	0
33	0.00225113	0	0	0.76262626	0	0.33333333	0	0.93714286	0.02898551	0.06666667	0.08956562	0.05555556	0.03703704	0	0	0.75890411	0	1	0	0	

airbnb_normalizado.csv

5. Conclusiones

El programa desarrollado permite realizar una limpieza y normalización del archivo **Airbnb_site_hotel_new.csv**, asegurando que los datos sean coherentes, completos y listos para su análisis. A través del uso de librerías como pandas y scikit-learn, se resolvieron problemas comunes en los conjuntos de datos, además, el código se estructuró en módulos, lo que mejora su legibilidad y mantenimiento.

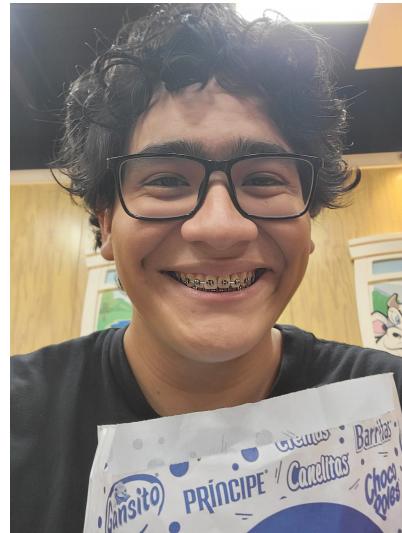
De igual, el proyecto podría optimizarse en algunos aspectos:

- Control de errores y validaciones, especialmente para detectar valores fuera de rango, como precios negativos o muy imposibles.
- Eficiencia de ejecución, el proceso de carga y escritura de archivos CSV puede mejorarse evitando operaciones repetidas.
- Visualización de resultados, donde se pueden incluir gráficos comparativos entre los datos originales y los normalizados para facilitar la interpretación de los cambios realizados.

Anexos

[Repository de GitHub](#)

Fotografía de los participantes



Orlando Alvarado Vargas



Diego Alonso Carrillo Castillo

Referencias

- [1] Oliveira, W. (2025). Página de bases de datos. Recuperado de: <https://www.kaggle.com/datasets/willianoliveiragibin/airbnb-site-hotel>. Consulta: 08/10/2025.
- [2] Ordaz, E. D. J. D. (2025). Página web sobre información para normalización de datos. Disponible en: <https://erickordazr.notion.site/Normalizaci-n-de-Datos-18f598df262b80e1a1c6d3d61284658d>. Consulta: 08/10/2025.