UNIVERSIDAD DE TARAPACÁ

FACULTAD DE INGENIERÍA

Departamento de Ingeniería en Computación e Informática

Tarea 2
Preprocesamiento de Fuentes de Datos

Autor: Fabián Justo Villalobos

Curso: Minería de Datos

Profesor: Roberto Espinoza Oliva

Arica, 25 de septiembre de 2024

Índice

1. Introducción **3**

2. Objetivos **4**
2.1. Objetivo general 4
2.2. Objetivos específicos 4

3. Dataset Diabetes **5**
3.1. Análisis Exploratorio Previo 5
3.2. Descripción Preprocesamiento 9
3.3. Resultados 12

4. Dataset Notebooks SoloTodo **13**
4.1. Análisis Exploratorio Previo 13
4.2. Descripción Preprocesamiento 20
4.3. Resultados 23

5. Conclusiones **24**

Índice de tablas

1. Descripción fuente de datos Diabetes 5
2. Análisis matemático de los atributos numéricos en DDS 7
3. Descripción atributos tabla SNDS 13
4. Análisis matemático de los atributos numéricos en SNDS 14
5. Cantidad de nulos en el dataset SNDS 15

Índice de figuras

1. Tabla inicial dataset DDS 7
2. Gráfico de distribución de tipos de diabetes 8
3. Tabla resultado para el dataset DDS 12
4. Distribución de precios de los equipos 16
5. Relación entre el precio y la capacidad de almacenamiento 17
6. Relación entre el precio y la cantidad de RAM 18
7. Frecuencia de equipos por cantidad de RAM 19
8. Frecuencia de sistemas operativos 20
9. Cantidad de filas con valores 0 en cada columna 21
10. Cantidad de filas con valores 0 en cada columna 23

1. Introducción

La minería de datos es un proceso crucial dentro de la ciencia de datos que busca identificar patrones y relaciones significativas en grandes conjuntos de datos. Entre los diferentes pasos que se tienen que seguir, la etapa de análisis y preprocesamiento es una de las más laboriosas y duraderas de este proceso, puesto que es donde se analiza en profundidad una fuente de información, estudiar los rangos en que oscilan sus datos, para luego en el preprocesamiento prepararlos para ser utilizados en algoritmos de minería de datos, como regresión lineal, clasificación, reglas de asociación, entre otros.

Este trabajo se centra en las dos fases mencionadas, aplicadas a dos fuentes de datos: una que trata de pacientes que padecen de distintos tipos de diabetes y sus rasgos, y la segunda sobre los aspectos técnicos de portátiles. La primera fuente proveniente del sitio de Kaggle, mientras que la segunda fue realizado por medio de web scraping del sitio web de SoloTodo.

2. Objetivos

2.1. Objetivo general

Aplicar técnicas de Análisis Exploratorio de Datos y Preprocesamiento en fuentes de datos con distintos comportamientos para prepararlos a la etapa de Minería de Datos

2.2. Objetivos específicos

O.E.1: Recopilar información sobre fuentes de datos viables

O.E.2: Realizar análisis exploratorio a cada fuente de dato elegida

O.E.3: Realizar etapa de preprocesamiento para prepararlos a la siguiente etapa

O.E.4: Sacar conclusiones respecto a la etapa de preprocesamiento

3. Dataset Diabetes

3.1. Análisis Exploratorio Previo

El dataset consta de una amplia descripción de varios tipos de diabetes y una amplia gama de atributos médicos, genéticos y de estilo de vida que son determinantes para comprender los factores que contribuyen al padecimiento de diabetes en las personas. Desde ahora referenciado como dataset de Diabetes (DDS), consta de:

- **Cantidad de Ejemplos:** 70000
- **Cantidad de Atributos:** 34
- **Cantidad de Clases:** 13

La descripción de los atributos del dataset DDS están descritos en la Tabla 1, además de un análisis de la media, desviación estándar, mínimos y máximos de los valores númericos del mismo (Tabla 2). En el dataset no hubo valores nulos, por lo que no se muestra una tabla al respecto.

Tabla 1: Descripción fuente de datos Diabetes

N°	Atributo	Tipo de dato	Descripción
1	Target	*str*	El tipo de diabetes o prediabetes
2	Genetic Markers	*str*	Indicadores de predisposición genética a la diabetes.
3	Autoantibodies	*str*	Presencia de autoanticuerpos comúnmente asociados con la diabetes autoinmune
4	Family History	*bool*	Información sobre si existen antecedentes familiares conocidos de diabetes
5	Environmental Factors	*str*	Detalles sobre las influencias ambientales que pueden contribuir a la diabetes
6	Insulin Levels	*int*	Niveles de insulina medidos en los pacientes
7	Age	*int*	Edad del paciente
8	BMI	*int*	El índice de masa corporal del paciente
9	Physical Activity	*str*	Nivel de acividad física del paciente
10	Dietary Habits	*str*	Hábitos alimenticios del paciente
11	Blood Pressure	*int*	Presión de la sangre
12	Cholesterol Levels	*int*	Niveles de colesterol

Continúa en la siguiente página

Tabla 1: Descripción fuente de datos Diabetes (Continuación)

N°	Atributo	Tipo de dato	Descripción
13	Waist Circumference	*int*	Medida de cintura
14	Blood Glucose Levels	*int*	Niveles de glucosa en la sangre
15	Ethnicity	*str*	Etnia
16	Socioeconomic Factors	*str*	Factores socioeconómicos
17	Smoking Status	*str*	Si el paciente es fumador o no
18	Alcohol Consumption	*str*	Nivel de consumo de alcohol
19	Glucose Tolerance Test	*str*	Resultado de prueba de tolerancia a la glucosa
20	History of PCOS	*bool*	Antecedentes de síndrome del ovario poliquístico
21	Previous Gestational Diabetes	*bool*	Diabetes gestacional previo
22	Pregnancy History	*str*	Antecedentes de embarazo
23	Weight Gain During Pregnancy	*int*	Aumento de peso durante el embarazo
24	Pancreatic Health	*int*	Salud pancreática
25	Pulmonary Function	*int*	Función pulmonar
26	Cystic Fibrosis Diagnosis	*bool*	Diagnóstico de fibrosis quística
27	Steroid Use History	*bool*	Antecedentes de uso de esteroides
28	Genetic Testing	*str*	Pruebas genéticas
29	Neurological Assessments	*int*	Evaluaciones nerológicas
30	Liver Function Tests	*str*	Pruebas de función hepática
31	Digestive Enzyme Levels	*int*	Niveles de enzimas digestivas
32	Urine Test	*str*	Análisis de orina
33	Birth Weight	*int*	Peso al nacer
34	Early Onset Symptoms	*bool*	Síntomas de aparición precoz

Tabla 2: Análisis matemático de los atributos numéricos en DDS

Atributo	Media aritmética	Desviación estándar	Mínimo	Máximo
Insulin Levels	21.607443	10.785852	5.0	49.0
Age	32.020700	21.043173	0.0	79.0
BMI	24.782943	6.014236	12.0	39.0
Blood Pressure	111.339543	19.945000	60.0	149.0
Cholesterol Levels	194.867200	44.532466	100.0	299.0
Waist Circumference	35.051657	6.803461	20.0	54.0
Blood Glucose Levels	160.701657	48.165547	80.0	299.0
Weight Gain During Pregnancy	15.496414	9.633096	0.0	39.0
Pancreatic Health	47.564243	19.984683	10.0	99.0
Pulmonary Function	70.264671	11.965600	30.0	89.0
Neurological Assessments	1.804157	0.680154	1.0	3.0
Digestive Enzyme Levels	46.420529	19.391089	10.0	99.0
Birth Weight	3097.061071	713.837300	1500.0	4499.0

```
diabetes_dataset
[3]  ✓ 0.0s
```

	Target	Genetic Markers	Autoantibodies	Family History	Environmental Factors	Insulin Levels	Age	BMI	Physical Activity	Dietary Habits	...	Pulmonary Function	Cystic Fibrosis Diagnosis	Steroid Use History	Genetic Testing	Neurological Assessments	Liver Function Tests	Digestive Enzyme Levels
0	Steroid-Induced Diabetes	Positive	Negative	No	Present	40	44	38	High	Healthy	...	76	No	No	Positive	3	Normal	56
1	Neonatal Diabetes Mellitus (NDM)	Positive	Negative	No	Present	13	1	17	High	Healthy	...	60	Yes	No	Negative	1	Normal	28
2	Prediabetic	Positive	Positive	Yes	Present	27	36	24	High	Unhealthy	...	80	Yes	No	Negative	1	Abnormal	55
3	Type 1 Diabetes	Negative	Positive	No	Present	8	7	16	Low	Unhealthy	...	89	Yes	No	Positive	2	Abnormal	60
4	Wolfram Syndrome	Negative	Negative	Yes	Present	17	10	17	High	Healthy	...	41	No	No	Positive	1	Normal	24
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
69995	Steroid-Induced Diabetes	Negative	Negative	Yes	Present	24	38	35	High	Unhealthy	...	71	Yes	Yes	Positive	2	Abnormal	34
69996	LADA	Positive	Positive	No	Absent	21	51	31	Low	Unhealthy	...	82	Yes	Yes	Negative	1	Abnormal	58
69997	Type 1 Diabetes	Positive	Negative	No	Absent	18	11	15	Low	Unhealthy	...	77	Yes	Yes	Positive	2	Normal	53
69998	Cystic Fibrosis-Related Diabetes (CFRD)	Positive	Negative	No	Absent	32	30	24	High	Healthy	...	70	No	No	Positive	1	Abnormal	35
69999	LADA	Positive	Positive	Yes	Absent	27	41	28	Moderate	Healthy	...	84	No	Yes	Negative	2	Abnormal	64

70000 rows × 34 columns

Figura 1: Tabla inicial dataset DDS

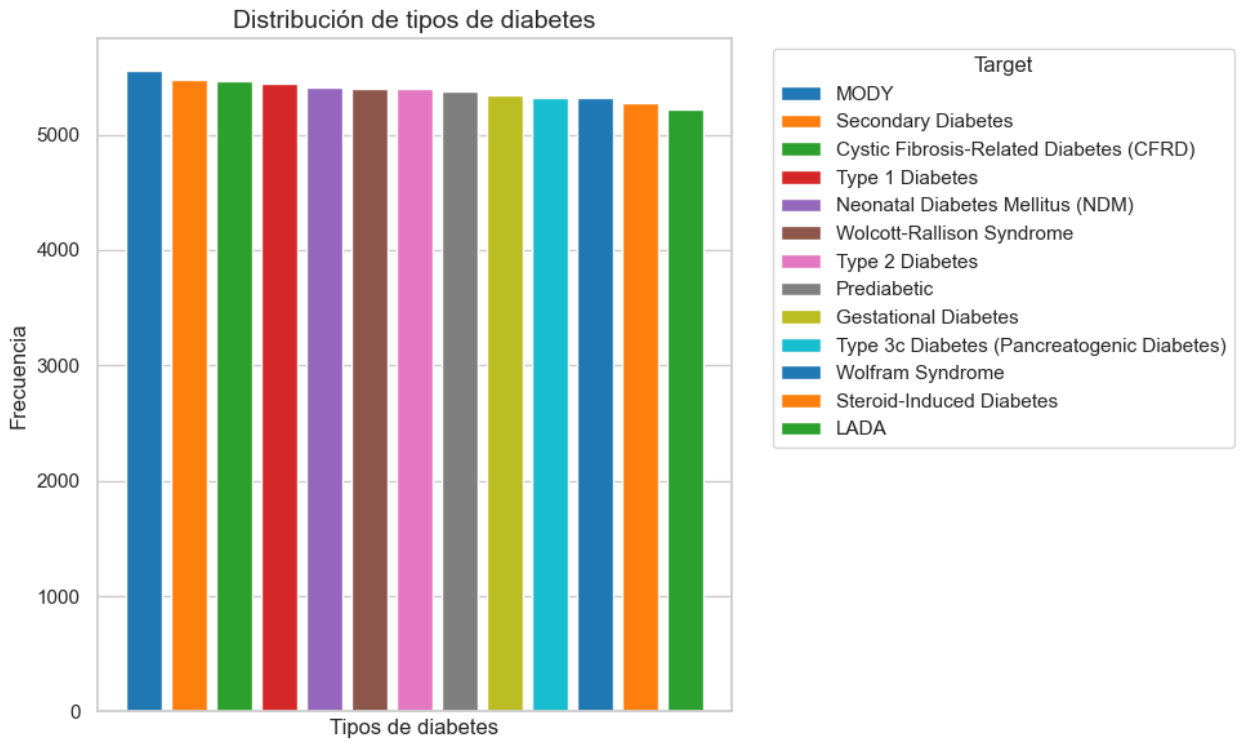


Figura 2: Gráfico de distribución de tipos de diabetes

3.2. Descripción Preprocesamiento

Para esta etapa de preprocesamiento, se utilizaron todos los atributos del dataset, por lo que se hizo un mapeo de las columnas con valores nominales (texto) a valores que sean más fácil de usar por algún modelo de minería de datos. En este caso, las *one_hot_columns* corresponden a columnas con dos valores nominales, se hizo uso de la función integrada con Python Pandas *get_dummies* para cambiar a valores booleanos (equivalentes a 0 y 1).

```
import pandas as pd
diabetes_dataset = pd.read_csv('diabetes\diabetes_dataset00.csv')

one_hot_columns = [
  'Genetic Markers',
  'Autoantibodies',
  'Environmental Factors',
  'Dietary Habits',
  'Ethnicity',
  'Smoking Status',
  'Glucose Tolerance Test',
  'Pregnancy History',
  'Genetic Testing',
  'Liver Function Tests',
]
diabetes_dataset_encoded = pd.get_dummies(diabetes_dataset.drop('Target', axis=1),
↪  columns=one_hot_columns, drop_first=True)
```

Luego, para las columnas con más de dos valores nominales, se realizó un proceso aparte de mapear cada una de estas instancias y hacer el reemplazo correspondiente:

```
yes_no_mapping = {
  'No': 0,
  'Yes': 1
}
physical_activity_mapping = {
  'Low': 0,
  'Moderate': 1,
  'High': 2
}
socioeconomic_factors_mapping = {
  'Low': 0,
  'Medium': 1,
  'High': 2
}
alcohol_consumption_mapping = {
  'Low': 0,
  'Moderate': 1,
  'High': 2
}
urine_test_mapping = {
  'Normal': 0,
  'Protein Present': 1,
  'Ketones Present': 2,
  'Glucose Present': 3
}

yes_no_columns = ['Family History', 'History of PCOS', 'Cystic Fibrosis Diagnosis',
↪  'Steroid Use History', 'Previous Gestational Diabetes', 'Early Onset Symptoms']

for column in yes_no_columns:
  diabetes_dataset_encoded[column] =
  ↪  diabetes_dataset_encoded[column].map(yes_no_mapping)

diabetes_dataset_encoded['Physical Activity'] = diabetes_dataset_encoded['Physical
↪  Activity'].map(physical_activity_mapping)
diabetes_dataset_encoded['Socioeconomic Factors'] =
↪  diabetes_dataset_encoded['Socioeconomic
↪  Factors'].map(socioeconomic_factors_mapping)
diabetes_dataset_encoded['Alcohol Consumption'] = diabetes_dataset_encoded['Alcohol
↪  Consumption'].map(alcohol_consumption_mapping)
diabetes_dataset_encoded['Urine Test'] = diabetes_dataset_encoded['Urine
↪  Test'].map(urine_test_mapping)
```

Por último, para poder utilizar de forma correcta las columnas con valores numéricos que oscilan entre varios rangos y unidades de medida, se hizo una estandarización de estos valores por medio de la herramiento ofrecida por scikit-learn, *StandardScaler* el cual calcula el nuevo valor siguiente la fórmula:

$$z = (x - u)/s$$

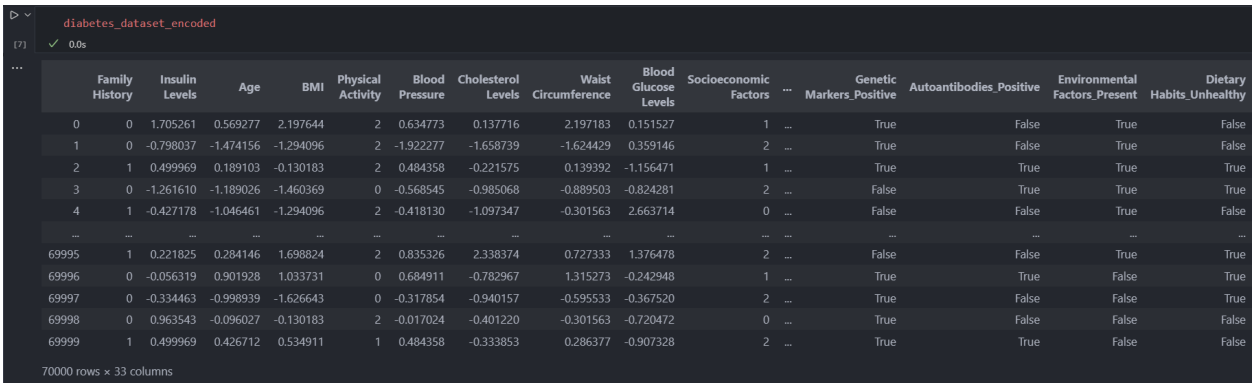
Donde:

- z: Nuevo valor calculado
- x: Valor original
- u: Media aritmética del atributo
- s: Desviación estándar del atributo

```
from sklearn.preprocessing import StandardScaler
numerical_columns = [
 'Insulin Levels',
 'Age',
 'BMI',
 'Blood Pressure',
 'Cholesterol Levels',
 'Waist Circumference',
 'Blood Glucose Levels',
 'Weight Gain During Pregnancy',
 'Pancreatic Health',
 'Pulmonary Function',
 'Neurological Assessments',
 'Digestive Enzyme Levels',
 'Birth Weight'
]

scaler = StandardScaler()
diabetes_dataset_encoded[numerical_columns] =
↪  scaler.fit_transform(diabetes_dataset_encoded[numerical_columns])
```

3.3. Resultados

Para finalizar, se logra apreciar en la Figura 3 la tabla resultante después de haber aplicado el preprocesamiento redactado con anterioridad. En este, a comparación de la tabla inicial en la Figura 1, todas las columnas han sido transformadas a valores numéricos (en el caso de True y False, 0's y 1's respectivamente), con tal que este nuevo set de datos esté preparado para ser utilizado en algún modelo predictivo, ya sea usando un Árbol de Decisión, red neuronal, etc.



```
diabetes_dataset_encoded
[7] ✓ 0.0s
```

	Family History	Insulin Levels	Age	BMI	Physical Activity	Blood Pressure	Cholesterol Levels	Waist Circumference	Blood Glucose Levels	Socioeconomic Factors	...	Genetic Markers_Positive	Autoantibodies_Positive	Environmental Factors_Present	Dietary Habits_Unhealthy
0	0	1.705261	0.569277	2.197644	2	0.634773	0.137716	2.197183	0.151527	1	...	True	False	True	False
1	0	-0.798037	-1.474156	-1.294096	2	-1.922277	-1.658739	-1.624429	0.359146	2	...	True	False	True	False
2	1	0.499969	0.189103	-0.130183	2	0.484358	-0.221575	0.139392	-1.156471	1	...	True	True	True	True
3	0	-1.261610	-1.189026	-1.460369	0	-0.568545	-0.985068	-0.889503	-0.824281	2	...	False	True	True	True
4	1	-0.427178	-1.046461	-1.294096	2	-0.418130	-1.097347	-0.301563	2.663714	0	...	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
69995	1	0.221825	0.284146	1.698824	2	0.835326	2.338374	0.727333	1.376478	2	...	False	False	True	True
69996	0	-0.056319	0.901928	1.033731	0	0.684911	-0.782967	1.315273	-0.242948	1	...	True	True	False	True
69997	0	-0.334463	-0.998939	-1.626643	0	-0.317854	-0.940157	-0.595533	-0.367520	2	...	True	False	False	True
69998	0	0.963543	-0.096027	-0.130183	2	-0.017024	-0.401220	-0.301563	-0.720472	0	...	True	False	False	False
69999	1	0.499969	0.426712	0.534911	1	0.484358	-0.333853	0.286377	-0.907328	2	...	True	True	False	False

70000 rows × 33 columns

Figura 3: Tabla resultado para el dataset DDS

4. Dataset Notebooks SoloTodo

4.1. Análisis Exploratorio Previo

El dataset consta del registro de todas las portátiles registradas en el sitio web de SoloTodo, con algunas especificaciones técnicas y sus valores puros, como también puntajes en rendimiento de procesador, tarjeta gráfica y evaluación de rendimiento en videojuegos, aplicaciones en general y movilidad. Desde ahora se referenciará al dataset como SNDS (SoloTodo Notebooks Data Set).

- **Cantidad de Ejemplos:** 1500
- **Cantidad de Atributos total:** 33
- **Cantidad de Atributos seleccionados:**

La descripción de los atributos del dataset SNDS están descritos en la Tabla 3, además de un análisis de la media, desviación estándar, mínimos y máximos de los valores númericos del mismo (Tabla 4), y el conteo de valores nulos en la Tabla 5.

Tabla 3: Descripción atributos tabla SNDS

N°	Atributo	Tipo de dato	Descripción
1	price	*int*	Precio en pesos chilenos
2	score_general	*int*	Puntaje de rendimiento en aplicaciones
3	score_games	*int*	Puntaje de rendimiento en juegos
4	score_mobility	*int*	Puntaje de rendimiento en movilidad
5	weight	*int*	Peso en gramos
6	operating_system_family_name	*str*	Nombre del sistema operativo
7	screen_size_value	*int*	Tamaño de la pantalla en pulgadas
8	screen_resolution_unicode	*str*	Resolución de la pantalla
9	screen_rr_value	*int*	Tasa de refreso de la pantalla
10	battery_mwh	*int*	Capacidad de la batería integrada en megavatios hora
11	processor_f_value	*int*	Frecuencia del procesador en MHz
12	processor_p_core_count	*int*	Número de núcleos físicos
13	processor_thread_count	*int*	Número de núcleos lógicos
14	processor_speed_score	*int*	Puntaje del procesador
15	main_gpu_speed_score	*int*	Puntaje del procesador gráfico principal
16	ram_quantity_value	*int*	Cantidad de RAM del equipo
17	ram_frequency_value	*int*	Frecuencia de la ram en MHz

Continúa en la siguiente página

Tabla 3: Descripción atributos tabla SNDS (Continuación)

N°	Atributo	Tipo de dato	Descripción
18	ram_type_name	*str*	Tipo de RAM
19	sd_capacity_value	*int*	Almacenamiento más grande del equipo
20	sd_drive_type_name	*str*	Tipo del almacenamiento más grande

Tabla 4: Análisis matemático de los atributos numéricos en SNDS

Atributo	Media aritmética	Desviación estándar	Mínimo	Máximo
price	1110340.105461	802208.236083	0	6294710
score_general	326.251726	163.877823	41	1000
score_games	251.394852	204.110915	24	1000
score_mobility	489.246704	160.197048	0	975
weight	1723.526679	519.917049	0	4282
screen_size_value	14.806591	1.068114	10	18
screen_rr_value	87.455744	50.536948	60	480
battery_mwh	51031.043942	24484.649538	0	100000
processor_f_value	2115.362210	916.266959	100	4300
processor_p_core_count	4.733208	2.378929	0	16
processor_thread_count	12.662272	6.113577	2	32
processor_speed_score	17510.682360	9805.010515	1202	61985
main_gpu_speed_score	8134.362210	9841.355721	0	46832
ram_quantity_value	15.509102	10.263737	4	128
ram_frequency_value	3686.951036	1657.028580	0	8448
sd_capacity_value	625.414940	357.215184	32	4000

Tabla 5: Cantidad de nulos en el dataset SNDS

Atributo	Cantidad de nulos
price	0
score_general	0
score_games	0
score_mobility	0
weight	0
operating_system_family_name	31
screen_size_value	0
screen_resolution_unicode	0
screen_rr_value	0
battery_mwh	0
processor_f_value	0
processor_p_core_count	0
processor_thread_count	0
processor_speed_score	0
main_gpu_speed_score	0
ram_quantity_value	0
ram_frequency_value	0
ram_type_name	0
sd_capacity_value	0
sd_drive_type_name	0

Se tuvo en cuenta analizar la distribución de los equipos en base a su precio, lo que resultó en que la mayor concentración se encuentra en el rango de 0 - 1.000.000 CLP, debido a que los costos de producción de equipos portátiles para uso hogareño no son elevados. A medida que sube el precio, las especificaciones del equipo son mejores, como procesadores más potentes, procesadores gráficos dedicados, mayor nitidez en las pantallas integradas, entre otros.

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.histplot(notebooks_df_cleaned['price'])
plt.xlabel('Precio')
plt.ylabel('Frecuencia')
plt.show()
```

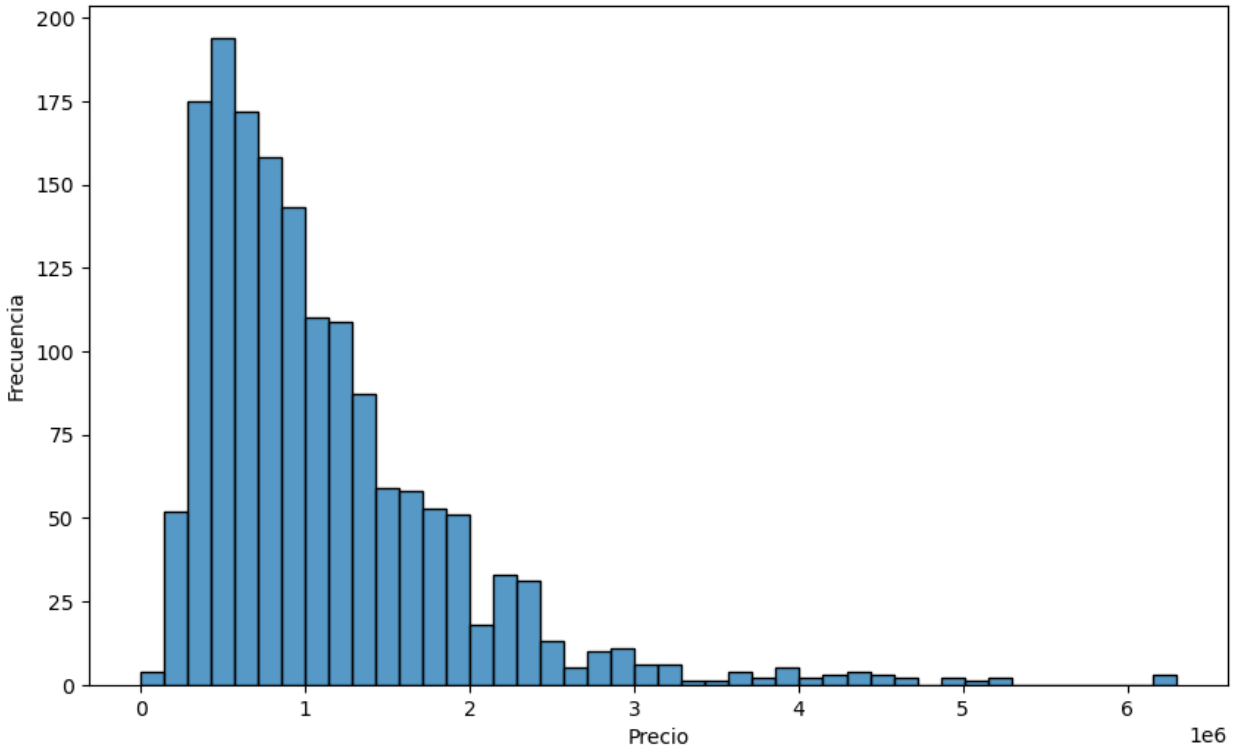


Figura 4: Distribución de precios de los equipos

Se aprecia en la Figura 5 la relación entre la cantidad de almacenamiento de los equipos y sus precios, el cual muestra una concentración entre 0 - 1000 GB. Los equipos localizados en la línea de los 4TB de almacenamiento son equipos de trabajo con otros aspectos técnicos también de potentes.

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.scatterplot(x='sd_capacity_value', y='price', data=notebooks_df_cleaned)
plt.xlabel('Capacidad de almacenamiento (GB)')
plt.ylabel('Precio')
plt.show()
```

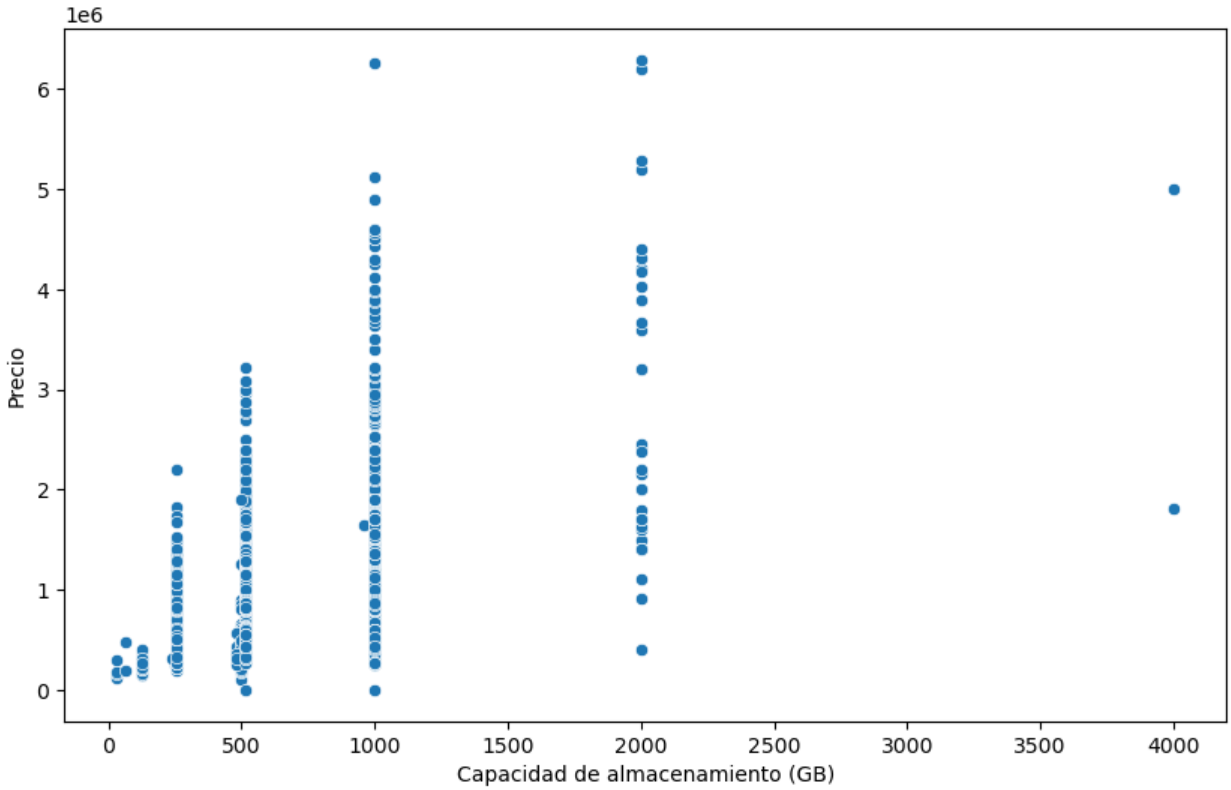


Figura 5: Relación entre el precio y la capacidad de almacenamiento

Otro vistazo muestra la relación entre el precio y la cantidad de RAM de los equipos por medio de un diagrama de cajas (Figura 6). Se puede apreciar varios valores atípicos, principalmente en los valores de 2^n: 4GB, 8GB, 16GB y 32GB, esto debido a que son las configuraciones más comunes que se venden en el mercado (Figura 7), por lo que mantenerlos en los registros sería la mejor opción.

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.boxplot(x='ram_quantity_value', y='price', data=notebooks_df_cleaned)
plt.xlabel('Cantidad de RAM (GB)')
plt.ylabel('Precio')
plt.show()

plt.figure(figsize=(10, 6))
sns.countplot(x='ram_quantity_value', data=notebooks_df_cleaned)
plt.xlabel('Cantidad de RAM (GB)')
plt.ylabel('Frecuencia')
plt.show()
```

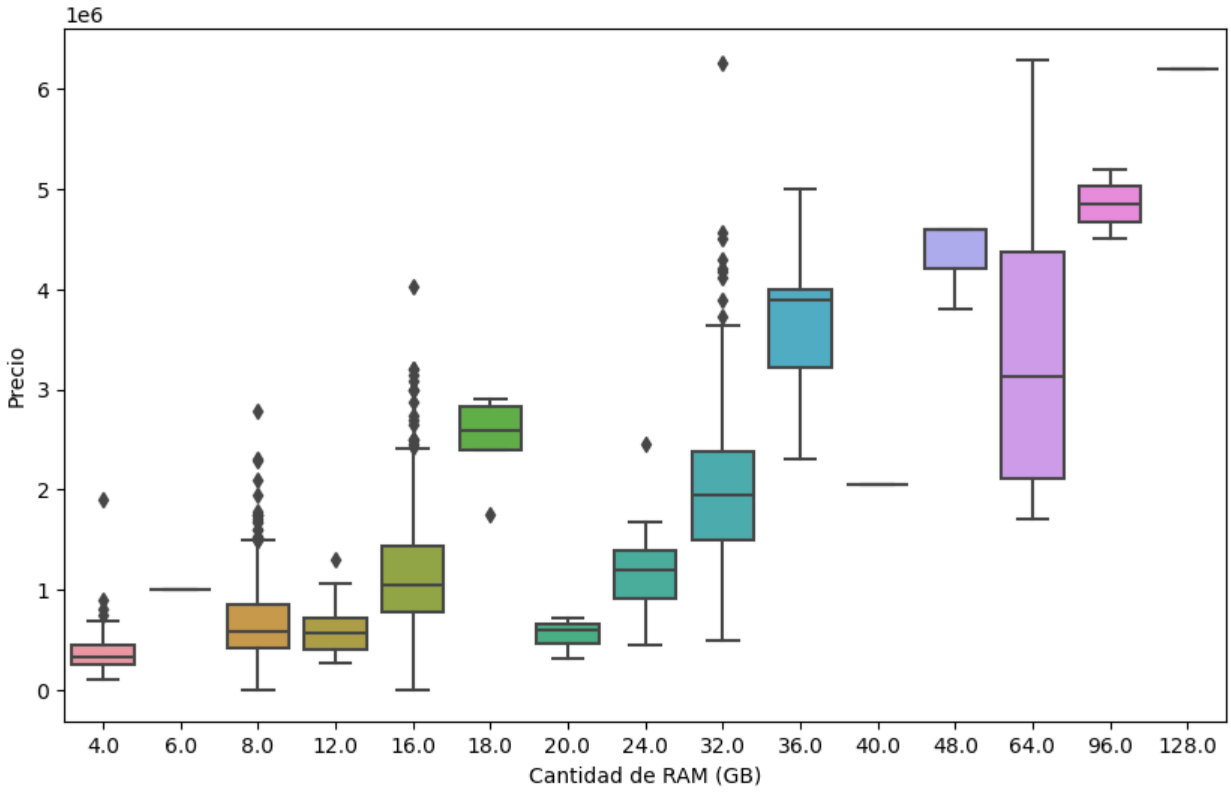


Figura 6: Relación entre el precio y la cantidad de RAM

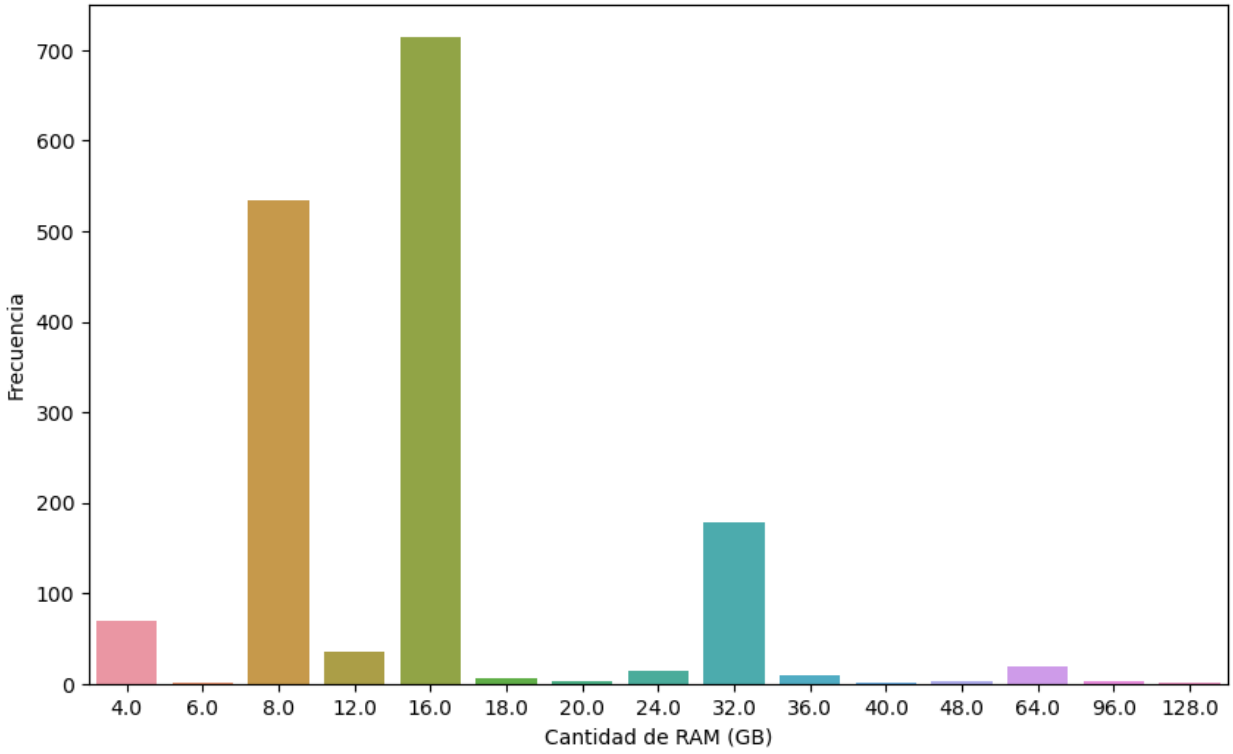


Figura 7: Frecuencia de equipos por cantidad de RAM

Por último, se analizó la frecuencia de los sistemas operativos en los equipos (Figura 8), donde se puede apreciar que la mayoría de los equipos tienen Windows. sin embargo, como se vio en la Tabla 5, hay 31 registros que no tienen sistema operativo, por lo que se deberá tratar para la etapa de preprocesamiento.

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.countplot(x='operating_system_family_name', data=notebooks_df_cleaned)
plt.xlabel('Sistema operativo')
plt.ylabel('Frecuencia')
plt.show()
```

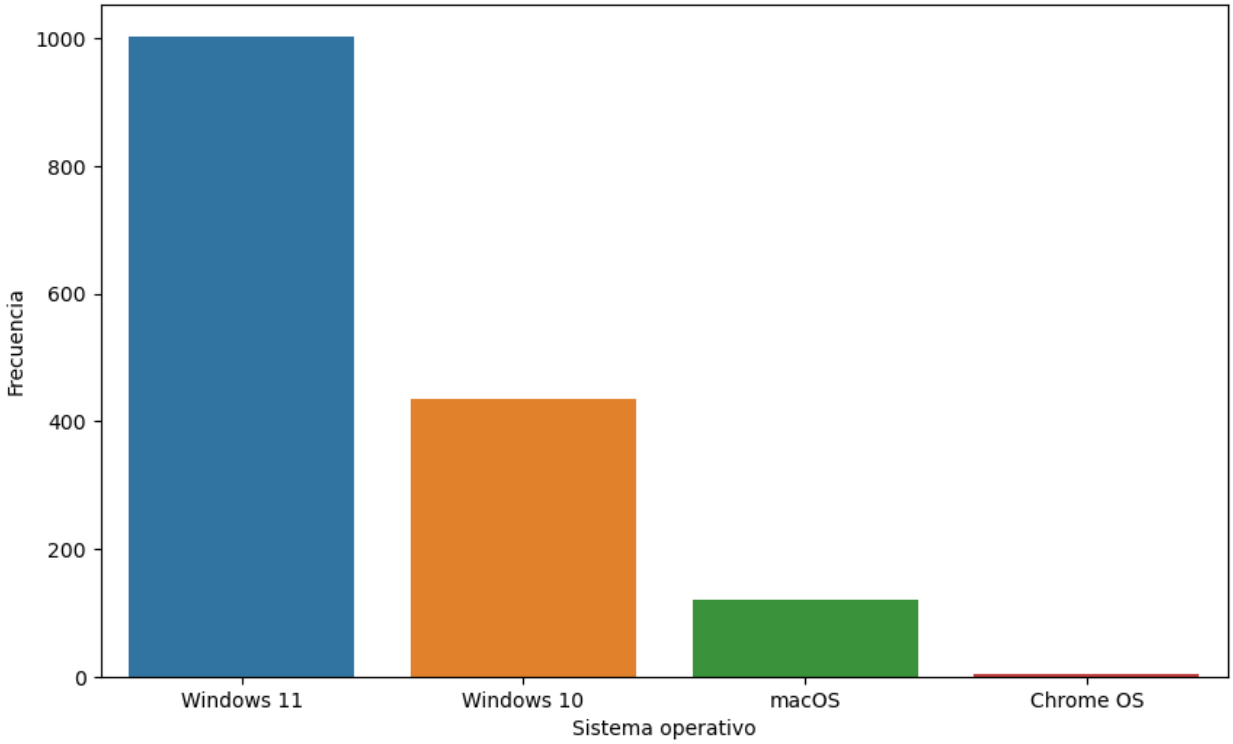


Figura 8: Frecuencia de sistemas operativos

4.2. Descripción Preprocesamiento

Previamente se descartaron 10 columnas del dataset original, ya que son columnas que no tienen influencia en el precio del equipo, como el nombre, ID, links, entre otros, por lo que quedan un restante de 20 columnas.

```
import pandas as pd
notebooks_df = pd.read_csv('notebook-dataset/notebook_data.tsv', sep='\t')
columns_to_drop = [
  'name',
  'url',
  'id',
  'slug',
  'picture_url',
  'default_bucket',
  'processor_unicode',
  'processor_thread_count_name',
  'main_gpu_name',
  'gpus_count',
  'processor_tdp',
  'sd_rpm_value',
  'storage_drives_count',
  'power_adapter_power',
]
notebooks_df_cleaned = notebooks_df.drop(columns=columns_to_drop)
```

En la Tabla 4 hay varias columnas con valores mínimos 0, y la cantidad de filas que contienen valores 0 es significativa (véase Figura 9), lo que no tiene sentido en la vida real, por lo que se procedió a cambiar estos valores por la media, mediana o moda en la columna correspondiente.

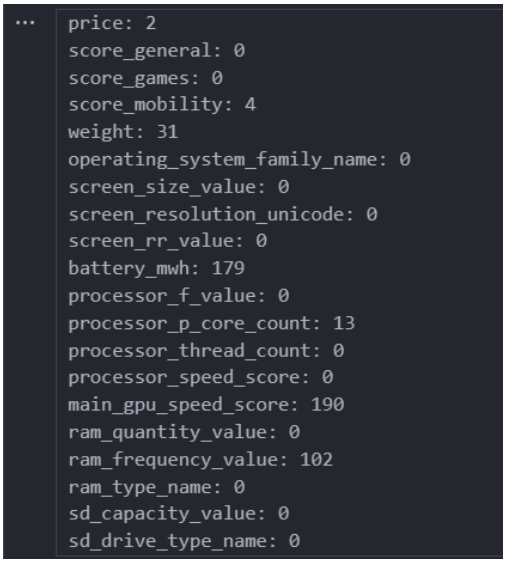


Figura 9: Cantidad de filas con valores 0 en cada columna

```
for col in notebooks_df_cleaned.columns:
    print(f'{col}: {notebooks_df_cleaned[notebooks_df_cleaned[col] ==
    ↪  0].shape[0]}')
```

```
notebooks_df_cleaned['score_mobility'].replace(0,
↪  int(notebooks_df_cleaned['score_mobility'].mean()), inplace=True)
notebooks_df_cleaned['weight'].replace(0,
↪  int(notebooks_df_cleaned['weight'].mean()), inplace=True)
notebooks_df_cleaned['processor_p_core_count'].replace(0, 4, inplace=True)
notebooks_df_cleaned['battery_mwh'].replace(0,
↪  int(notebooks_df_cleaned['battery_mwh'].mean()), inplace=True)
notebooks_df_cleaned['ram_frequency_value'].replace(0,
↪  int(notebooks_df_cleaned['ram_frequency_value'].mean()), inplace=True)
notebooks_df_cleaned['main_gpu_speed_score'].replace(0,
↪  int(notebooks_df_cleaned['main_gpu_speed_score'].median()))

so_more_frequent = notebooks_df_cleaned['operating_system_family_name'].mode()[0]
notebooks_df_cleaned['operating_system_family_name'].fillna(so_more_frequent,
↪  inplace=True)
```

Y se eliminaron los registros con valores 0 en la columna *price*, ya que no tiene sentido tener un equipo sin precio.

```
notebooks_df_cleaned = notebooks_df_cleaned[notebooks_df_cleaned['price'] > 0]
```

Para las columnas con valores tipo *str*, se procedió a codificarlas con *LabelEncoder* de *sklearn* y se guardó el encoder para su uso posterior, con el objetivo de que los algoritmos de aprendizaje automático puedan trabajar con ellas.

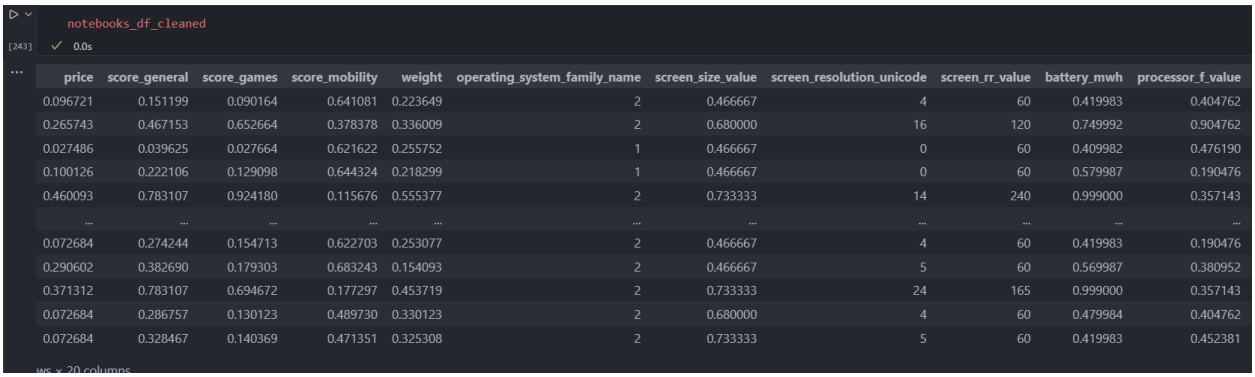
```
import pickle
from sklearn.preprocessing import LabelEncoder

categorical_columns = [
    'operating_system_family_name',
    'ram_type_name',
    'screen_resolution_unicode',
    'sd_drive_type_name',
]
label_encoders = {}

for col in categorical_columns:
    label_encoder = LabelEncoder()
    notebooks_df_cleaned[col] =
    ↪  label_encoder.fit_transform(notebooks_df_cleaned[col])
    label_encoders[col] = label_encoder

with open('label_encoders_snds.pkl', 'wb') as f:
    pickle.dump(label_encoders, f)
```

Como las columnas con valores numéricos están en distintas medidas, se recomienda realizar una normazliación de los datos. En este caso se usó *MinMaxScaler* de *sklearn* para que los valores estén en el rango de 0 a 1.

```
from sklearn.preprocessing import MinMaxScaler
numeric_columns = [
  'price',
  'score_general',
  'score_games',
  'score_mobility',
  'weight',
  'screen_size_value',
  'battery_mwh',
  'processor_f_value',
  'ram_quantity_value',
  'ram_frequency_value',
  'sd_capacity_value'
]
scaler = MinMaxScaler()
notebooks_df_cleaned[numeric_columns] =
↪  scaler.fit_transform(notebooks_df_cleaned[numeric_columns])
```

4.3. Resultados

El dataset SNDS resultado es un dataset limpio y listo para ser utilizado en algoritmos de aprendizaje automático. Se guardó en un archivo *notebooks_cleaned.csv*, junto al archivo *label_encoders_snds.pkl* para su uso posterior.



notebooks_df_cleaned

[243] ✓ 0.0s

price	score_general	score_games	score_mobility	weight	operating_system_family_name	screen_size_value	screen_resolution_unicode	screen_rr_value	battery_mwh	processor_f_value
0.096721	0.151199	0.090164	0.641081	0.223649	2	0.466667	4	60	0.419983	0.404762
0.265743	0.467153	0.652664	0.378378	0.336009	2	0.680000	16	120	0.749992	0.904762
0.027486	0.039625	0.027664	0.621622	0.255752	1	0.466667	0	60	0.409982	0.476190
0.100126	0.222106	0.129098	0.644324	0.218299	1	0.466667	0	60	0.579987	0.190476
0.460093	0.783107	0.924180	0.115676	0.555377	2	0.733333	14	240	0.999000	0.357143
...	...	...	...	...	...	...	...	...	...	...
0.072684	0.274244	0.154713	0.622703	0.253077	2	0.466667	4	60	0.419983	0.190476
0.290602	0.382690	0.179303	0.683243	0.154093	2	0.466667	5	60	0.569987	0.380952
0.371312	0.783107	0.694672	0.177297	0.453719	2	0.733333	24	165	0.999000	0.357143
0.072684	0.286757	0.130123	0.489730	0.330123	2	0.680000	4	60	0.479984	0.404762
0.072684	0.328467	0.140369	0.471351	0.325308	2	0.733333	5	60	0.419983	0.452381

ws x 20 columns

Figura 10: Cantidad de filas con valores 0 en cada columna

5. Conclusiones

El preprocesamiento de las fuentes de datos seleccionadas resultó en conjuntos de información de alta calidad, listos para ser utilizados en aplicaciones de minería de datos. Las técnicas aplicadas permitieron eliminar inconsistencias, mejorar la coherencia y preparar los datos para un análisis eficaz. Gracias a estas acciones, se generaron bases de datos estructuradas y confiables que podrán ser empleadas para construir modelos predictivos precisos y aplicar algoritmos como árboles de decisión. En resumen, el proceso de preprocesamiento permitió convertir los datos en un recurso valioso y preparado para la obtención de conocimiento mediante minería de datos, asegurando resultados más precisos y útiles en futuras fases del análisis.

Referencias

[1] "Diabetes Dataset." https://www.kaggle.com/datasets/ankitbatra1210/diabetes-dataset/data, 2024. Accedido (20-09-2024).

[2] "SoloTodo Public Api." https://publicapi.solotodo.com, 2024. Accedido (20-09-2024).