Actividad | Selección y limpieza de los Datos en Python

Integrantes

Alumno: **Erick de Jesus Hernández Cerecedo**
Matricula: **A01066428**

Alumno: **Francisco Javier Hernández Camarillo**
Matricula: **A00998083**

Información del Curso

Nombre: **Ciencia y analítica de datos**
Profesor: **Jobish Vallikavungal Devassia**

Fechas: **Martes 4 de octubre de 2022**

Parte 1: Fundamentos de bases de datos

1. Fundamentos de bases de datos y para ciencia de datos.

Una **base de datos** es una colección o lista de información organizada de datos relacionados, estas pueden estar compuestas de una o varias listas que reciben por nombre tablas de datos; Las **tablas de datos** a su vez organizan la información en columnas y filas conocidas como campos y registros respectivamente, cada uno de los **campos** cuenta con un nombre que identifica el tipo de información o categoría que se está almacenando en dicha columna, por ejemplo Nombre, Apellido, dirección, ciudad, código postal, teléfono etc., por lo regular el *nombre del campo* está ubicado en la fila superior de una base de datos. Las **filas** en las bases de datos están compuestas por registros, cada una de ellas puede ser llamada **"Ttupla** y todos los registros de una tupla pertenecen a un mismo objeto.

Los **registros** son la información relacionada que está separada por columnas o campos, es importante diferenciar que un nombre y dirección son registros diferentes en la base de datos.

2. Fundamentos de almacenes de datos (Data Warehouse) para ciencia de datos.

Un **almacén de datos** es una base de datos gigante o depósito que contiene información de diversas fuentes que están unidas y disponibles en un solo lugar para poder ser analizadas y empleadas como respaldo en la toma de decisiones más informadas.

Un **desafío importante de los almacenes de datos (Data Warehouse)** es realizar una *extracción* de datos de diferentes fuentes de información, *transformarlos* en un esquema que permita consultar las fuentes simultáneamente y *obtener* información a través del análisis de los datos, los pasos mencionados anteriormente es conocido **ETL** *(Extraer, Transformar, Cargar)* para posteriormente terminar con el análisis de consulta, la problemática está en que el proceso *ELT* conlleva un **80%** del trabajo de un científico de datos, debido a que si no se hace de manera adecuada el resultado esperado en la obtención de un modelo de Machine Learning resultaría poco confiable o inservible.

Actualmente existe un concepto innovador para optimizar el proceso ETL este es llamado **ELT**, que consiste en el uso de bases de datos distribuidas que son bases de datos lógicamente relacionadas en diferentes puntos lógicos y geográficos por ejemplo un servidor con 2 o más máquinas conectadas por la red, en fines prácticos es usar el poder de procesamiento de estas bases distribuidas para cargar la información directa a nuestra colección de datos.

Fuentes

- Teate, R. M. P. (s. f.). SQL for Data Scientists. O'Reilly Online Learning. Recuperado 4 de octubre de 2022, de https://www.oreilly.com/library/view/sql-for-data/9781119669364/
- Kane, F. (s. f.). Hands-On Data Science and Python Machine Learning. O'Reilly Online Learning. Recuperado 4 de octubre de 2022, de https://www.oreilly.com/library/view/hands-on-data-science/9781787280748/

Parte 2: Selección y limpieza de los Datos en Python

Lectura de datos

In [37]:
```
# Importar la librerias necesarias para la actividad
import pandas as pd
import numpy as np
import requests
import os

# URL de la base de datos
url='https://raw.githubusercontent.com/PosgradoMNA/Actividades_Aprendizaje-/main/default

# Peticion de los datos
r=requests.get(url)

# Guardamos los datos en un archivo local con el mismo nombre
path=os.path.join(os.getcwd(),'default of credit card clients.csv')  # Obtenemos la dire
with open(path,'wb') as f:                                           # Crear arc
    f.write(r.content)                                               # Escribir

# Lectura de los datos obtenios CSV
data = pd.read_csv(path, index_col=0)
print("Información de la base de datos: \n")
print(data.info())
print("\nVisualización de datos: \n")
print(data.head())
```

```
Información de la base de datos:

<class 'pandas.core.frame.DataFrame'>
Int64Index: 30000 entries, 1 to 30000
Data columns (total 24 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   X1      30000 non-null  int64  
 1   X2      29999 non-null  float64
 2   X3      29998 non-null  float64
 3   X4      29998 non-null  float64
 4   X5      29995 non-null  float64
 5   X6      29997 non-null  float64
 6   X7      29995 non-null  float64
 7   X8      29993 non-null  float64
 8   X9      29991 non-null  float64
 9   X10     29984 non-null  float64
 10  X11     29986 non-null  float64
```

```
 11  X12     29989 non-null  float64
 12  X13     29989 non-null  float64
 13  X14     29987 non-null  float64
 14  X15     29985 non-null  float64
 15  X16     29983 non-null  float64
 16  X17     29990 non-null  float64
 17  X18     29992 non-null  float64
 18  X19     29991 non-null  float64
 19  X20     29992 non-null  float64
 20  X21     29989 non-null  float64
 21  X22     29989 non-null  float64
 22  X23     29995 non-null  float64
 23  Y       29997 non-null  float64
dtypes: float64(23), int64(1)
memory usage: 5.7 MB
None

Visualización de datos:

        X1   X2   X3   X4    X5   X6   X7   X8   X9  X10  ...       X15  \
ID                                                        ...
1    20000  2.0  2.0  1.0  24.0  2.0  2.0 -1.0 -1.0 -2.0  ...       0.0
2   120000  2.0  2.0  2.0  26.0 -1.0  2.0  0.0  0.0  0.0  ...    3272.0
3    90000  2.0  2.0  2.0  34.0  0.0  0.0  0.0  0.0  0.0  ...   14331.0
4    50000  2.0  2.0  1.0  37.0  0.0  0.0  0.0  0.0  0.0  ...   28314.0
5    50000  1.0  2.0  1.0  57.0 -1.0  0.0 -1.0  0.0  0.0  ...   20940.0

        X16      X17     X18      X19      X20     X21     X22     X23    Y
ID
1       0.0      0.0     0.0    689.0      0.0     0.0     0.0     0.0  1.0
2    3455.0   3261.0     0.0   1000.0   1000.0  1000.0     0.0  2000.0  1.0
3   14948.0  15549.0  1518.0   1500.0   1000.0  1000.0  1000.0  5000.0  0.0
4   28959.0  29547.0  2000.0   2019.0   1200.0  1100.0  1069.0  1000.0  0.0
5   19146.0  19131.0  2000.0  36681.0  10000.0  9000.0   689.0   679.0  0.0

[5 rows x 24 columns]
```

In [43]:

```python
# Verificaciónes de la base de datos

# Existen valores vacios en la base?
print("Existen null en bd: ", data.isnull().values.any())

# Existen valores Null en cada na de las columnas?
print("Existen null en cada campo: \n", data.isnull().any())

# Existen valores NaN en la bd?
print("Existen NaN en bd: ", data.isna().values.any())
```

```
Existen null en bd:  True
Existen null en cada campo:
 X1     False
X2      True
X3      True
X4      True
X5      True
X6      True
X7      True
X8      True
X9      True
X10     True
X11     True
X12     True
X13     True
X14     True
X15     True
X16     True
```

```
X17     True
X18     True
X19     True
X20     True
X21     True
X22     True
X23     True
Y       True
dtype: bool
Existen NaN en bd:  True
```

Descartar las observaciones con valores faltantes

In [53]:
```
# Descartar los valores NaN
data.dropna(inplace = True)

# Existen valores NaN en la bd?
print("Existen NaN en bd: ", data.isna().values.any())
```

```
Existen NaN en bd:  False
```

Descartar las columnas que no consideramos

In [52]:
```
# Eliminación de campos no considerados
df = data.drop(columns=['X2', 'X9', 'X10', 'X11', 'X15', 'X16', 'X17', 'X21', 'X22', 'X2

print("Base de datos despues de eliminar las columnas: ")

df.info()

# Existen valores vacios en la base?
print("Existen null en bd: ", df.isnull().values.any())

# Existen valores Null en cada na de las columnas?
print("Existen null en cada campo: \n", df.isnull().any())

# Existen valores NaN en la bd?
print("Existen NaN en bd: ", df.isna().values.any())
```

```
Base de datos despues de eliminar las columnas:
<class 'pandas.core.frame.DataFrame'>
Int64Index: 29958 entries, 1 to 30000
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   X1      29958 non-null  int64
 1   X3      29958 non-null  float64
 2   X4      29958 non-null  float64
 3   X5      29958 non-null  float64
 4   X6      29958 non-null  float64
 5   X7      29958 non-null  float64
 6   X8      29958 non-null  float64
 7   X12     29958 non-null  float64
 8   X13     29958 non-null  float64
 9   X14     29958 non-null  float64
 10  X18     29958 non-null  float64
 11  X19     29958 non-null  float64
 12  X20     29958 non-null  float64
 13  Y       29958 non-null  float64
dtypes: float64(13), int64(1)
memory usage: 3.4 MB
Existen null en bd:  False
Existen null en cada campo:
 X1     False
X3     False
```

```
X4     False
X5     False
X6     False
X7     False
X8     False
X12    False
X13    False
X14    False
X18    False
X19    False
X20    False
Y      False
dtype: bool
Existen NaN en bd:  False
```

Parte 3: Preparación de los datos

1. ¿Qué datos considero mas importantes? ¿Por qué?

Respuesta:
Se analizó la información anexa del la base de datos y se seleccionaron los siguientes campos de la base de datos:

- **X1** = Monto del crédito dato que esta dado en unidates NT Dollar (Nuevo Dólar Taiwanés), el tipo de dato es de numeros enteros.
- **X3-X5** = Porque son datos que identifican un conjunto de la población de clientes, estos son:

    1. **X3:** Nivel de educación.
    2. **X4:** Estado Marital.
    3. **X5:** Edad.
- **X6-X8** = Historial del estado de los pagos previos, en el que se reastrea la confiabilidad del cliente para realizar sus pagos a tiempo y forma durante los meses Julio - Agosto y Septiembre (2005)

    1. -1 Pay duly.
    2. 1 Payment delay for one month
    3. 2 Payment delay for two months; . . .; 9 Payment delay for nine months and above.
- **X12-X14** = Cuenta pendiente de los últimos 3 meses
- **X18-X20** = Monto de pagos previos de los últimos 3 meses
- **Y** = Valor booleano si se otorgó crédito o no

2. ¿Se eliminaron o reemplazaron datos nulos? ¿Qué se hizo y por qué?

Respuesta:
Se eliminaron los datos NaN y Null, ya que el numero de tuplas con datos nulos corresponde a un 0.14 % del contenido total, a lo cual consideramos despreciable por ser menor a 5% del volumen total. Las columnas eliminadas fueron:

- X2: Genero del cliente.
- X9 - X11 = Registros de estados de pagos de meses antiguos.
- X15 - X17 y X21 - X23 = Balance del monto de credito en meses antiguos.

Otra de las razones por las que decidimos eliminar y no reemplazar fue que el valor promedio resulta diferente cuando los valores NaN se reemplazan con 0.

3. ¿Es necesario limpiar los datos para el análisis? Sí / No / ¿Por qué?

Respuesta:
Si, se realíza para corregir o eliminar los datos corruptos, irrelevantes, incorrectos o inexactos en una base de datos con el fin de:

- Evitar que el modelo a entrenar aprenda del ruido de la información actual.
- Evitar que los resultados y algoritmos no sean confiables.
- Lograr que la relación entre los datos sea coherentes y libre de error.

4. ¿Existen problemas de formato que deban solucionar antes del proceso de modelado? Sí / No / Por qué.

Respuesta:
Sí, los problemas de formato existentes estan relacionados con tipos de dato Null y NaN, estos fueron eliminados con el fin de evitar realizar la modificacion a 0 ya que esto afectaba al valor de la medida estadistica promedio, ademas que los datos con problemas de formato representaban solo un 0.14 % que resulta de poco impacto, fue considerada irrelevante.

También fue limpiada la primera columna que es el índice por defecto de los datos consultados.

5. ¿Qué ajustes se realizaron en el proceso de limpieza de datos (agregar, integrar, eliminar, modificar registros (filas), cambiar atributos (columnas)?

Respuesta:
El cambio de atributos no fue necesario, ya que todas las columnas cuenta con datos "int" o tipo entero. Las acciones realizadas eliminación de columnas y eliminación de algunos registro