Limpieza y Estandarización de Datos Estudiantiles Usando Python.

Vargas Duque Cindy Liliana, Chavarro Arias Oscar Eduardo, González Bonilla Jesús Ariel, Vargas Narváez Luis Ángel

Resumen—**Este informe detalla el proceso de carga, limpieza y estandarización de un conjunto de datos proporcionado en formato Excel, utilizando herramientas en Python como Pandas, Numpy, y expresiones regulares. Se explican los pasos implementados para garantizar la calidad de los datos, desde la eliminación de registros inválidos hasta la corrección de formatos y estandarización de columnas.**

***Abstract*--This report outlines the process of loading, cleaning, and standardizing a student dataset provided in Excel format. Using Python libraries such as Pandas, Numpy, and regular expressions, various steps were implemented to ensure data quality—from eliminating invalid records to correcting formatting issues and standardizing column structures**

I. Introducción

La calidad de los datos es un factor crítico en los procesos de análisis y modelado. Este informe describe el procedimiento técnico seguido para limpiar y estandarizar una base de datos de estudiantes, con el objetivo de garantizar su consistencia y validez para futuros análisis.

II. HERRAMIENTAS UTILIZADAS

Las bibliotecas de Python empleadas fueron:

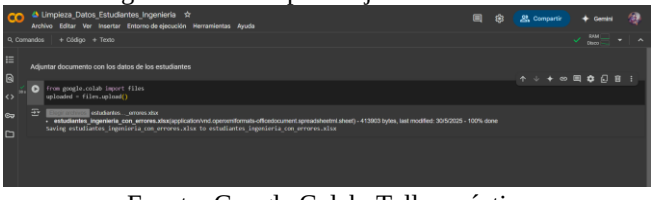
```
import pandas as pd
import numpy as np
import unicodedata
import re
```

Estas herramientas permiten manipular datos estructurados, realizar operaciones numéricas, limpiar caracteres especiales y aplicar expresiones regulares para estandarización.

III. CARGA DE DATOS

El archivo Excel proporcionado fue cargado utilizando la biblioteca Pandas. Se realizó una vista preliminar de las primeras filas para conocer la estructura general y el contenido de las columnas.

Imagen 1. Función para adjuntar documento



Fuente: Google Colab, Taller práctico

Se cargó el archivo Datos_Estudiantes.xlsx usando pd.read_excel() y se realizó una vista inicial para evaluar la estructura.

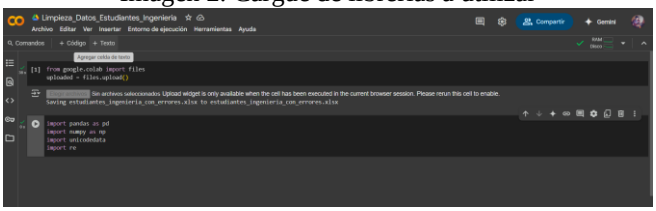
```
df = pd.read_excel("Datos_Estudiantes.xlsx")
df.head()
```

IV. CARGUE DE LIBRERÍAS A UTILIZADAR

Se emplearon las siguientes librerías:

- **Pandas**: para análisis y manipulación de datos en tablas.
- **Numpy**: para cálculos numéricos.
- **unicodedata**: para normalizar caracteres especiales.
- **re**: para aplicar expresiones regulares.

Imagen 2. Cargue de librerías a utilizar

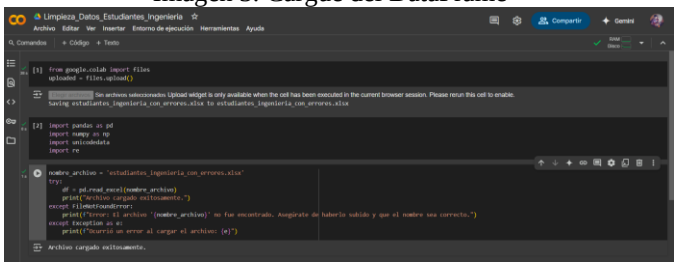


Fuente: Google Colab, Taller práctico

V. CREACIÓN DEL DATAFRAME

Se implementó un procedimiento para validar el nombre del archivo y cargar el contenido en un DataFrame. Si el archivo no se encontraba o tenía errores, el sistema lo notificaba y se evitaba su procesamiento.

Imagen 3. Cargue del DataFrame



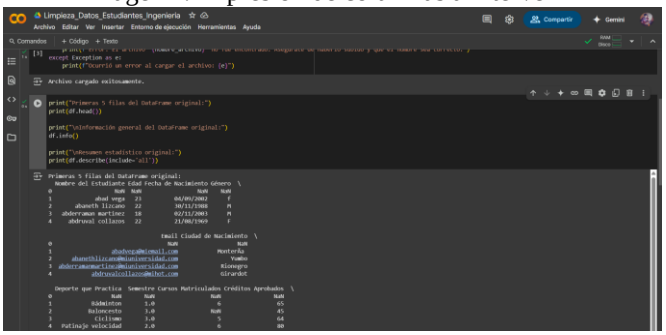
Fuente: Google Colab, Taller práctico

VI. IDENTIFICACIÓN Y ELIMINACIÓN DE REGISTROS NO VÁLIDOS

Se procede a realizar la evisión de la estructura del documento mediante Python para identificar las columnas (cabeceras) y que información contienen.

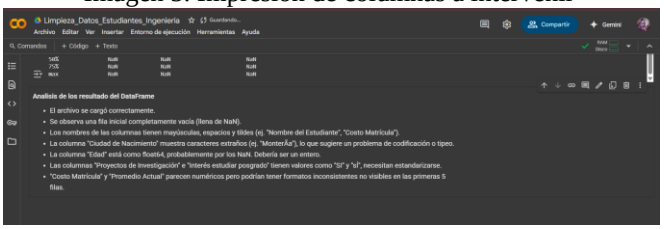
* Revista Argentina de Trabajos Estudiantiles. Patrocinada por la IEEE.

Imagen 4. Impresión de columnas a intervenir



Fuente: Google Colab, Taller práctico

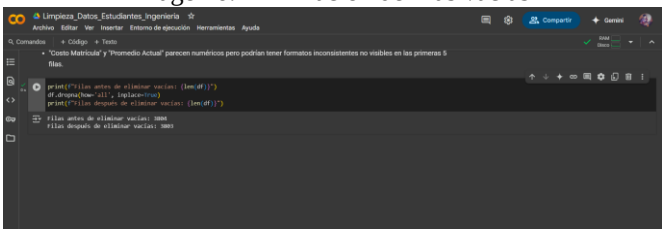
Imagen 5. Impresión de columnas a intervenir



Fuente: Google Colab, Taller práctico

Se eliminaron filas completamente vacías y se detectaron valores erróneos como asteriscos, textos en campos numéricos (e.g. “cuarenta”), y formatos no estándar (e.g. “2.300.000” o “tres punto siete”).
Se usaron funciones como len(df) y df.dropna(how='all') para validar y eliminar estos registros.

Imagen 6. Eliminación de filas vacías



Fuente: Google Colab, Taller práctico

Se eliminaron las filas completamente vacías usando:

df.dropna(how='all', inplace=True)
df['edad'] = df['edad'].replace(r'\D', '', regex=True)

Inicio depuración de (.) (,) (caracteres especiales) en columnas de valores numéricos para identificar y hacer la operación.

Imagen 7. Depuración de caracteres especiales

Fuente: Google Colab, Taller práctico

Imagen 8. Depuración de caracteres especiales

Fuente: Google Colab, Taller práctico

VII. Estandarización de Columnas

Se aplicaron procedimientos para:

- Uniformar el formato de nombres de columna.
- Separar nombres y apellidos en columnas distintas.
- Ajustar los tipos de datos (int, float, string).
- Eliminar tildes y reemplazar espacios por guiones bajos (“_”).

Se utilizó la función col_name para llamar cada columna y aplicar estos cambios.

```
def clean_column(col):
    col = col.strip().lower().replace(" ", "_")
    col = unicodedata.normalize("NFKD",
col).encode("ascii", "ignore").decode("utf-8")
    return col

df.columns = [clean_column(c) for c in df.columns]
```

Se deben encontrar los nombres de las columnas originales, para eso uso Python para llamar cada columna con la función col_name, y comienzo a definicar que los nombres sean tratados como textos, valido temas de mayúscula, minúscula, y además quitar tildes como caracteres especiales. De esta manera convierto texto a formato ASCII, para después convertirlo en formato UTF-8, reemplazo todos los espacios por (_) “Guion bajo”.

Imagen 9. Reemplazo de espacios

Fuente: Google Colab, Taller práctico

La columna nombre_del_estudiante (que originalmente era "Nombre del Estudiante") probablemente contiene tanto el nombre como el apellido. Por lo cual se hace la segmentación independiente de cada dato

Imagen 10. Segmentación de datos

Fuente: Google Colab, Taller práctico

Se hace una vista de las columnas que se han estandarizado como muestra de que los script fueron funcionales

Imagen 11. Impresión de datos

Fuente: Google Colab, Taller práctico

VIII. Limpieza Específica por Columna

- **Edad:** Convertida a valores numéricos y se eliminaron valores fuera de rango (mayores de 80).

```
df['edad'] = pd.to_numeric(df['edad'], errors='coerce')
df = df[df['edad'] <= 80]
```

Imagen 12. Ajuste de datos en edad

Fuente: Google Colab, Taller práctico

- **Género:** Solo se permitieron las categorías “M” o “F”, corrigiendo valores como “f”, “x”, “otro”.

```
df['genero'] = df['genero'].str.upper().replace({'FEMENINO': 'F', 'MASCULINO': 'M', 'X': np.nan})
```

Imagen 13. Ajuste de datos Género

Fuente: Google Colab, Taller práctico

- **Ciudad de nacimiento:** Se corrigieron errores ortográficos y caracteres no válidos.

```
df['ciudad'] = df['ciudad'].str.replace(r'[^A-Za-z\s]', '', regex=True).str.title()
```

Imagen 14. Ajuste de datos Ciudad de Nacimiento

Fuente: Google Colab, Taller práctico

- **Semestre:** Validado entre 1 y 10; se eliminaron valores inválidos como 13 o 20.

```
df['semestre'] = pd.to_numeric(df['semestre'], errors='coerce')
df = df[df['semestre'].between(1, 10)]
```

Imagen 15. Ajuste de datos Semestre

Fuente: Google Colab, Taller práctico

- **Promedio actual:** Convertido a formato decimal uniforme (con puntos).

```
df['promedio'] =
df['promedio'].astype(str).str.replace(",",
".").astype(float)
```

Imagen 16. Ajuste de datos en Promedio

Fuente: Google Colab, Taller práctico

- **Costo matrícula:** Se eliminaron separadores de miles y se convirtió a tipo numérico.

```
df['costo_matricula'] =
df['costo_matricula'].astype(str).str.replace(".",
"").astype(float)
```

Imagen 17. Ajuste de datos en Costo de Matrícula

Fuente: Google Colab, Taller práctico

Cada transformación fue justificada en función de su impacto sobre la calidad de los datos.

IX. Exportación de Datos Limpios

Tras verificar la calidad del DataFrame final y revisar los valores faltantes (NaN o NA), se exportó el archivo limpio con el nombre: estudiantes_ingenieria_limpios.xlsx.

```
df.to_excel("estudiantes_ingenieria_limpios.xlsx",
index=False)
```

Imagen 18. Se imprime y guarda la información limpia

Fuente: Google Colab, Taller práctico

Guardo toda la limpieza en un nuevo Excel, con el nombre recomendado según la guía

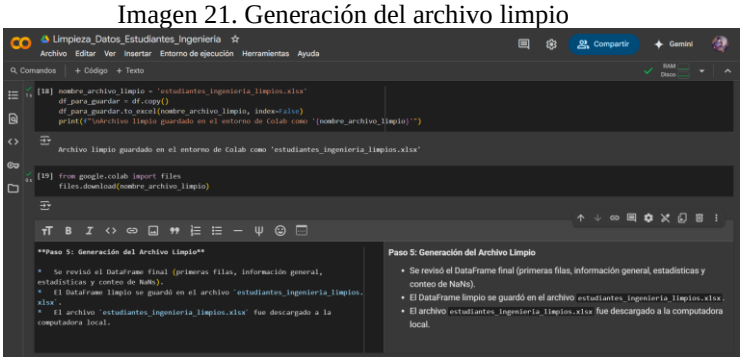
Imagen 19. Se carga la información limpia

Fuente: Google Colab, Taller práctico

Imagen 20. Guardado de la información limpia

Fuente: Google Colab, Taller práctico

Imagen 21. Generación del archivo limpio



Fuente: Google Colab, Taller práctico

X. Conclusión

El proceso de limpieza permitió transformar una base de datos cruda en un conjunto confiable y estandarizado. Este procedimiento es esencial para asegurar resultados válidos en análisis posteriores y garantizar la integridad de la información académica.