



Instituto Tecnológico y de Estudios Superiores de Monterrey

Proyecto Integrador:

Modelo Clasificador

Avance 1. Análisis exploratorio de datos

Profesor titular:

Dr. Horacio Martínez Alfaro

Alumnos:

José de Jesús Peña Rodríguez

A01794940

Kevin Dueñas Aguirre

A01283104

Juan Antonio Chiñas Mata

A01794191

Fecha: 2 de febrero de 2025

INTRODUCCIÓN AL PROYECTO

El presente proyecto, desarrollado bajo la asesoría del Dr. Horacio Martínez, profesor asignado al proyecto integrador, tiene como objetivo proporcionar una clasificación basada en ciertas características contenidas en una base de datos.

Dicha base recopila información extraída previamente por un conjunto de modelos encargados de analizar distintos tipos de documentos, como identificaciones oficiales (INE), recibos de pago (por ejemplo, de la CFE o servicios de cable) y estados de cuenta emitidos por el IMSS, entre otros. Los datos obtenidos se almacenan en un archivo CSV, que servirá como base para la construcción de nuestro modelo clasificador.

Al conjunto de modelos encargados de extraer y almacenar información en la base de datos los denominaremos, de manera general, "**Compiladores**". Aunque todos estos modelos extraen las mismas características, algunos se utilizan con mayor frecuencia que otros. Si bien se desconoce el criterio exacto para la selección de un modelo específico en función del archivo procesado, los resultados obtenidos hasta el momento sugieren que la estrategia aplicada es adecuada para la clasificación de documentos.

Los parámetros extraídos por los "**Compiladores**" conformarán las variables de entrada (X) para el desarrollo de nuestro modelo de salida, que será un clasificador. Este modelo proporcionará una respuesta categórica basada en las características extraídas, clasificando la información en tres niveles: **bueno, regular y malo**, con la posibilidad de expandir el número de niveles en el futuro. Algunos de los parámetros presentes en la base de datos incluyen:

- **Fecha del documento**
- **Tipo de documento del usuario**
- **Número de página**
- **Total de páginas**
- **Modelo aplicado**
- **Nivel de confianza del modelo**
- **Nivel de nitidez (sharpness)**
- **Nivel de contraste**
- **Resolución**

Todos los avances en el desarrollo del proyecto se llevarán a cabo siguiendo un modelo de ciclo de vida **Agile**, con el objetivo de realizar entregas periódicas en fechas establecidas, alineadas con las expectativas del cliente. El repositorio con el seguimiento del proyecto se encuentra en la siguiente liga:

https://github.com/Jesus2342/IntProject_Team8

ESTRUCTURA DE DATOS

Hasta la fecha, el archivo contiene **62,960 registros**. Sin embargo, este archivo es preliminar, y se espera la incorporación de más datos en los próximos días. A pesar de ello, el análisis realizado hasta el momento muestra resultados prometedores.

Para ofrecer una descripción general de la estructura y los tipos de datos, se adjunta las siguientes imágenes, donde se pueden observar las características principales del **dataframe**, la cantidad de valores faltantes por columna y, en el caso de las variables categóricas, la frecuencia de sus respectivas clases. Para más detalle se recomienda consultar la liga del repositorio.

```
df.info()
✓ 0.0s

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62960 entries, 0 to 62959
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Index                 62960 non-null  int64
1   Date                  62960 non-null  object
2   Model                 62960 non-null  object
3   ClassId               34140 non-null  object
4   Page                  62960 non-null  int64
5   TotalPages            62960 non-null  int64
6   PagesProcessed        62960 non-null  int64
7   Fields                62960 non-null  int64
8   EmptyFields           62960 non-null  int64
9   ModelApplied          62960 non-null  object
10  ModelReported         44333 non-null  object
11  Details                62960 non-null  object
12  ConfR                  62960 non-null  float64
13  ConfA                  62960 non-null  float64
14  ConfW                  62960 non-null  float64
15  Sharpness              62960 non-null  float64
16  Contrast               62960 non-null  float64
17  Clarity                62960 non-null  float64
18  Resolution             62960 non-null  float64
19  DocType                62960 non-null  object
dtypes: float64(7), int64(6), object(7)
memory usage: 9.6+ MB
```

ANÁLISIS EXPLORATORIO DE DATOS(EDA)

Summary Statistics for all columns										
df.describe(include="all")										
✓ 0.1s										
	Index	Date	Model	ClassId	Page	TotalPages	PagesProcessed	Fields	EmptyFields	ModelApplied
count	62960.000000	62960	62960	34140	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960
unique	NaN	27436	5	4	NaN	NaN	NaN	NaN	NaN	10
top	NaN	2025-01-18T13:04:43-08:00	estado_cuenta	TXN	NaN	NaN	NaN	NaN	NaN	Lynx_BankingTXN_v5
freq	NaN	49	34140	26556	NaN	NaN	NaN	NaN	NaN	26467
mean	47081.315025	NaN	NaN	NaN	2.934593	5.942186	3.326938	12.881210	3.495426	NaN
std	27193.866099	NaN	NaN	NaN	3.068428	4.682230	3.418311	13.533471	6.201645	NaN
min	0.000000	NaN	NaN	NaN	1.000000	1.000000	1.000000	0.000000	0.000000	NaN
25%	23511.500000	NaN	NaN	NaN	1.000000	2.000000	1.000000	0.000000	0.000000	NaN
50%	46975.000000	NaN	NaN	NaN	2.000000	6.000000	1.000000	12.000000	0.000000	NaN
75%	71010.250000	NaN	NaN	NaN	4.000000	8.000000	5.000000	27.000000	3.000000	NaN
max	94108.000000	NaN	NaN	NaN	48.000000	48.000000	15.000000	38.000000	21.000000	NaN

2 Estadísticas generales del df

numeric_df.describe()										
✓ 2.8s										
	Page	TotalPages	PagesProcessed	Fields	EmptyFields	ConfR	ConfA	ConfW	Sharpness	Contrast
count	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000	62960.000000
mean	2.934593	5.942186	3.326938	12.881210	3.495426	0.917741	0.791809	0.879958	224.199117	43.402370
std	3.068428	4.682230	3.418311	13.533471	6.201645	0.128916	0.149599	0.091764	271.111842	11.423269
min	1.000000	1.000000	1.000000	0.000000	0.000000	0.001000	0.288000	0.152000	0.000000	0.000000
25%	1.000000	2.000000	1.000000	0.000000	0.000000	0.914000	0.699000	0.849000	92.000000	36.000000
50%	2.000000	6.000000	1.000000	12.000000	0.000000	0.956000	0.841000	0.904000	153.000000	43.000000
75%	4.000000	8.000000	5.000000	27.000000	3.000000	0.993000	0.915000	0.930000	232.000000	50.000000
max	48.000000	48.000000	15.000000	38.000000	21.000000	0.999000	0.984000	0.993000	10209.000000	124.000000

3 Estadísticas de las variables numéricas

```

for i in categorical_columns:
    print(df[i].value_counts())
    print("-----")

```

[60] ✓ 0.4s

```

...
Model
estado_cuenta      34140
recibo_nomina      10193
ine                 9346
comprobante_domicilio 5780
carta_libranza      3501
Name: count, dtype: int64
-----
ClassId
TXN      26556
APP       4261
SUC       2581
ATM        742
Name: count, dtype: int64
-----
ModelApplied
Lynx_BankingTXN_v5      26467
Lynx_ProofPaysheet_v5    9722
Lynx_IdProof_v1         9346
Lynx_ATM_APP_v1         7568
Lynx_ProofAddress_v6     5780
Lynx_CLibranza_v2        3501
Lynx_IMSS3_v2           469
Lynx_BankingTXN_v6        89
Lynx_ATM_APP_v2          16
...
GAS         61
PASS        10
Name: count, dtype: int64

```

4 Niveles de variables categóricas

```

missing_values = df.isnull().sum()
missing_values

```

✓ 0.0s

```

Index      0
Date       0
Model      0
ClassId    28820
Page       0
TotalPages 0
PagesProcessed 0
Fields     0
EmptyFields 0
ModelApplied 0
ModelReported 18627
Details    0
ConfR      0
ConfA      0
ConfW      0
Sharpness  0
Contrast   0
Clarity    0
Resolution 0
DocType    0
dtype: int64

```

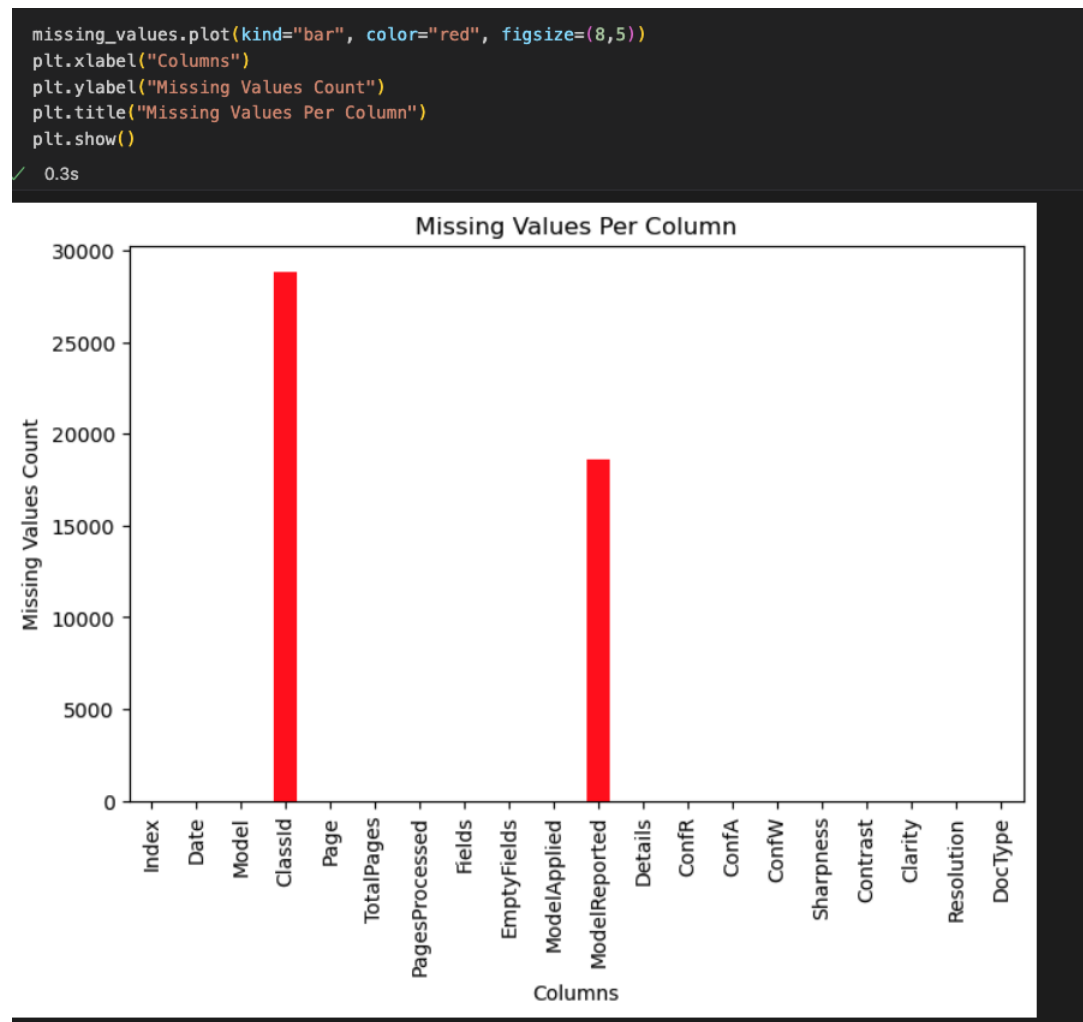
5 Valores faltantes en el df

¿Existen valores faltantes en el conjunto de datos? ¿Se pueden identificar patrones de ausencia?

Al analizar el conjunto de datos, se identificó que solo dos columnas presentan valores faltantes:

- **ClassID** con **28,820** entradas faltantes.
- **ModelReported** con **18,627** entradas faltantes.

Esta evaluación corresponde a un análisis preliminar del conjunto de datos, por lo que, en esta etapa, no se tomarán medidas como la eliminación de datos. Sin embargo, será necesario monitorear estos valores y determinar si su ausencia sigue algún patrón específico que pueda afectar el desempeño del modelo.



6 Detección de valores faltantes

¿Existen valores atípicos en el conjunto de datos?

Tras analizar los valores atípicos en las distintas columnas del conjunto de datos, se han identificado algunas conclusiones clave para el proyecto de clasificación:

- **La columna con el mayor número de valores atípicos es "Claridad" de la página.** Sin embargo, al calcular el porcentaje de estos valores, se determinó que representan solo el **8.70%** del total de los datos. Esto indica que, aunque existen valores atípicos en esta variable, su proporción no es lo suficientemente significativa como para afectar negativamente el desempeño del modelo de clasificación.
- **La única columna que no presenta valores atípicos es "Fields".** No obstante, esta columna no aporta información relevante sobre el contenido real de las imágenes. Para evaluar mejor su importancia en la tarea de clasificación, será necesario obtener más detalles sobre su significado y su impacto en el conjunto de datos.

```
numeric_df = df.select_dtypes(include=["number"])
numeric_df = numeric_df.drop("Index", axis=1)

Q1 = numeric_df.quantile(0.25)
Q3 = numeric_df.quantile(0.75)
IQR = Q3 - Q1

outliers = ((numeric_df < (Q1 - 1.5 * IQR)) | (numeric_df > (Q3 + 1.5 * IQR)))
outliers.sum()
```

✓ 0.0s

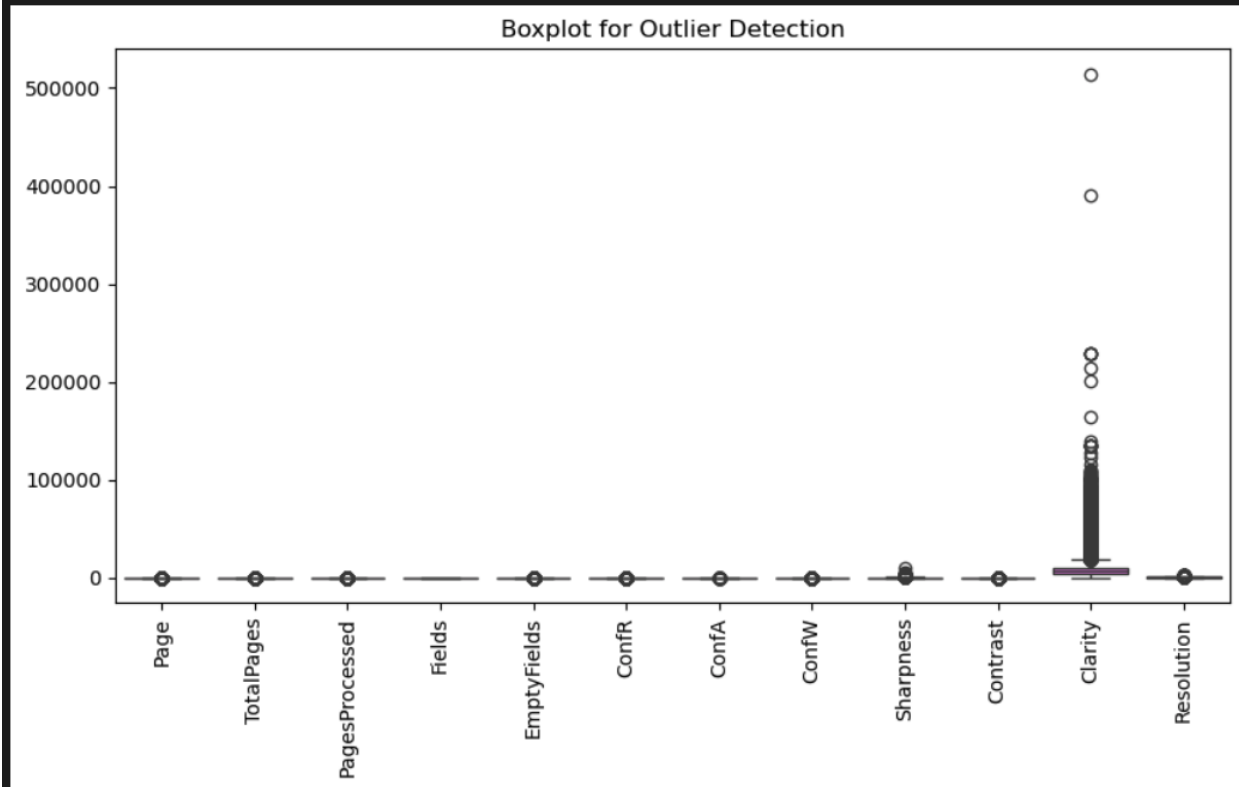
Page	4005
TotalPages	231
PagesProcessed	3017
Fields	0
EmptyFields	11194
ConfR	5641
ConfA	57
ConfW	3067
Sharpness	5005
Contrast	1218
Clarity	5472
Resolution	1275

dtype: int64

7 Detección de outliers

```
plt.figure(figsize=(10, 5))
sns.boxplot(data=numeric_df)
plt.xticks(rotation=90)
plt.title("Boxplot for Outlier Detection")
plt.show()
```

✓ 3.5s

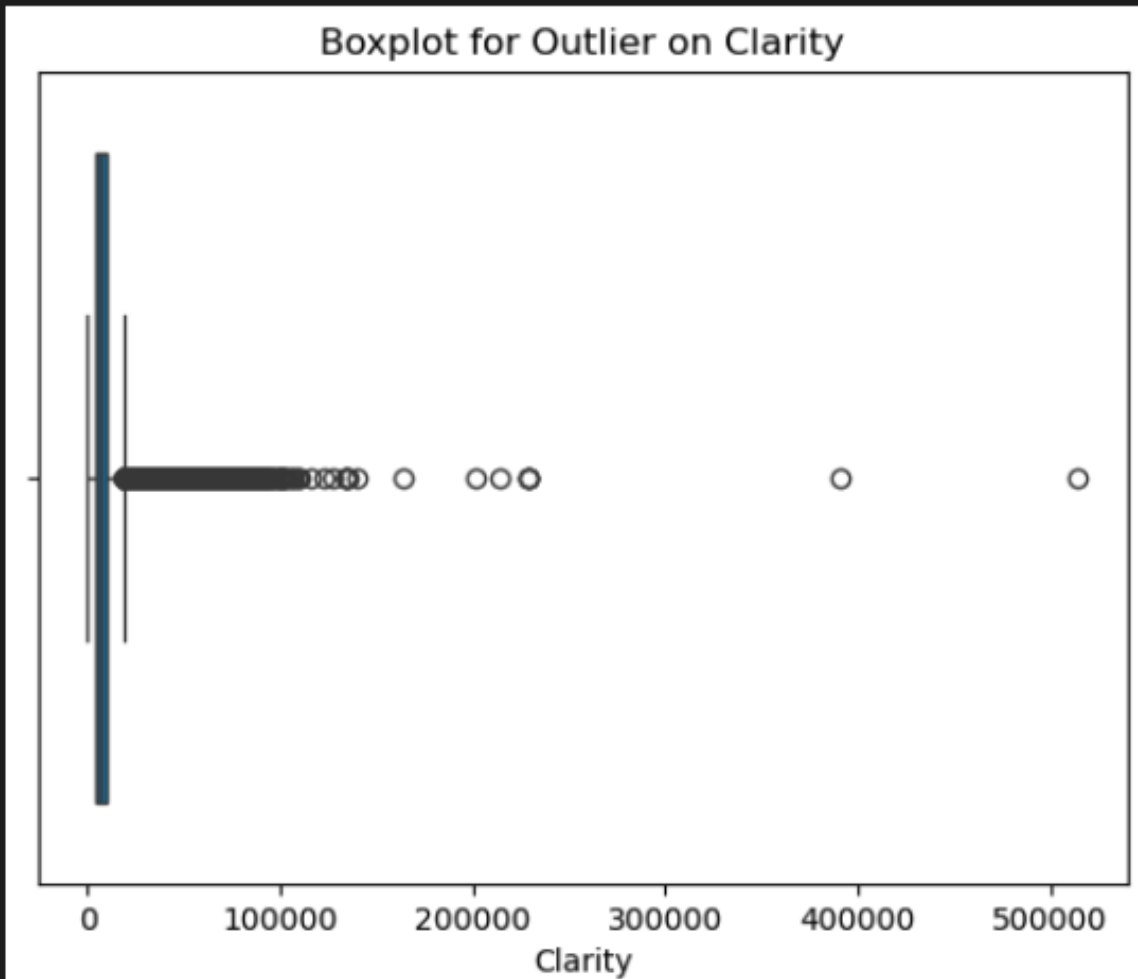


8 Gráficos de outliers


```
sns.boxplot(x=df["Clarity"])
plt.title("Boxplot for Outlier on Clarity")
plt.show()
```

✓ 0.4s

Python



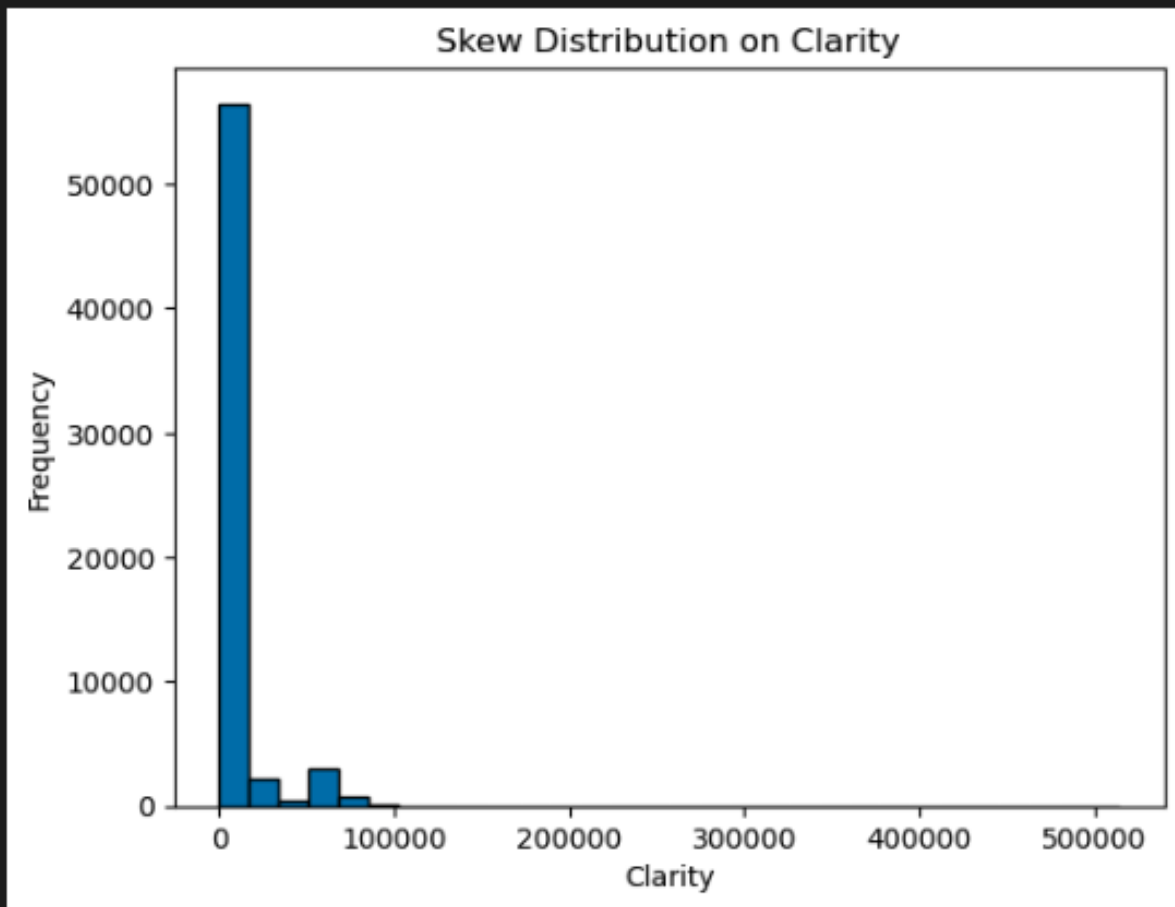
9 Outliers de Clarity

Para determinar el porcentaje adecuado de valores atípicos, es necesario analizar la distribución de los datos en la columna "**Clarity**". Este análisis permitirá evaluar si el **método de detección por cuartiles** es el más apropiado para identificar los valores atípicos dentro del conjunto de datos.

```
column = "Clarity"
plt.hist(df[column], bins=30, edgecolor="black")
plt.xlabel(column)
plt.ylabel("Frequency")
plt.title(f"Skew Distribution on {column}")
plt.show()
```

✓ 0.2s

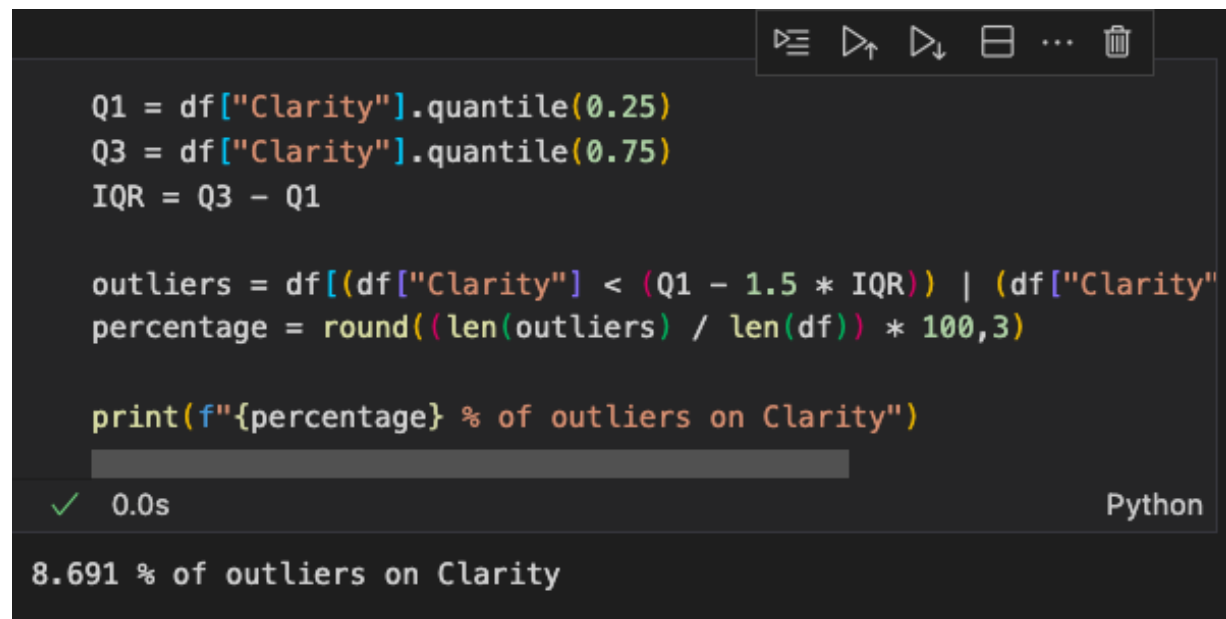
Pyth



10 Distribución de Clarity [Skew positiva]

Dado que la distribución de los datos **no es normal**, es recomendable utilizar el **método de cuartiles (IQR - Rango Inter cuartílico)** para la detección de valores atípicos. Este método es más adecuado para distribuciones asimétricas, ya que no asume normalidad y se basa en la dispersión de los datos.

Tras analizar el conjunto de datos proporcionado, se ha determinado que el **8.691%** de las entradas son valores atípicos. Este porcentaje, aunque no es insignificante, no necesariamente implica un impacto negativo en el modelo de clasificación.



```
Q1 = df["Clarity"].quantile(0.25)
Q3 = df["Clarity"].quantile(0.75)
IQR = Q3 - Q1

outliers = df[(df["Clarity"] < (Q1 - 1.5 * IQR)) | (df["Clarity"] > (Q3 + 1.5 * IQR))]
percentage = round((len(outliers) / len(df)) * 100, 3)

print(f"{percentage} % of outliers on Clarity")
```

✓ 0.0s Python

8.691 % of outliers on Clarity

¿Cuál es la cardinalidad de las variables categóricas?

Para este conjunto de datos, hay 4 variables categóricas.

- Model – 5 valores únicos
- ClassId – 4 valores únicos
- ModelApplied – 10 valores únicos
- DocType – 18 valores únicos

```
#unique values
unique_counts = df.nunique()

print(unique_counts)
```

```
Unamed: 0      62960
Date          27436
Model           5
ClassId        4
Page          48
TotalPages     27
PagesProcessed 15
Fields         7
EmptyFields    22
ModelApplied   10
ModelReported 1639
Details       1905
ConfR          97
ConfA         605
ConfW         641
Sharpness     2297
Contrast      1324
Clarity       18561
Resolution    2556
DocType        18
dtype: int64
```

```
[ ] # show unique values per categorical variable
unique_values_list = df[['DocType', 'Model', 'ModelApplied', 'ClassId']].apply(lambda x: list(x.unique()))

#show results
for column, unique_vals in unique_values_list.items():
    print(f'Valores únicos de {column}:')
    print(unique_vals)
    print("-" * 50)
```

```
Valores únicos de DocType:
['CFE', 'IMSS', 'INE', 'EC', 'CLIB', 'IMSS3', 'CABLE', 'APP', 'ATM', 'TEL', 'ISSSTE', 'CONST', 'SEP', 'SUC', 'PEMEX', 'AGUA', 'GAS', 'PASS']
-----
Valores únicos de Model:
['comprobante_domicilio', 'recibo_nomina', 'ine', 'estado_cuenta', 'carta_libranza']
-----
Valores únicos de ModelApplied:
['Lymx_ProofAddress_v6', 'Lymx_ProofPaysheet_v5', 'Lymx_IdProof_v1', 'Lymx_BankingTXN_v6', 'Lymx_CLibranza_v2', 'Lymx_IMSS3_v3', 'Lymx_ATM_APP_v2', 'Lymx_BankingTXN_v5', 'Lymx_ATM_APP_v1', 'Lymx_IMSS3_v2']
-----
Valores únicos de ClassId:
[nan, 'TXN', 'APP', 'ATM', 'SUC']
-----
```

¿Existen distribuciones sesgadas en el conjunto de datos? ¿Necesitamos aplicar alguna transformación no lineal?

Inicialmente, visualizamos los datos a través de histogramas y diagramas de caja para evaluar los patrones de distribución y detectar cualquier posible asimetría o valores atípicos. Esto nos permite obtener información sobre la distribución de datos subyacente y determinar si se requiere alguna transformación (como normalización o transformaciones logarítmicas) para cumplir con los supuestos de los modelos analíticos posteriores.

```

# Select only the numeric columns
numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns.difference(['Unnamed', 'Date', 'Index'])
#numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns

# Visualize distributions and calculate skewness
fig, axes = plt.subplots(nrows=len(numeric_columns), ncols=2, figsize=(12, len(numeric_columns)*5))

for i, column in enumerate(numeric_columns):
    # Get the data from the column
    data = df[column]

    # Histogram and KDE for the distribution
    sns.histplot(data, kde=True, ax=axes[i, 0], color='skyblue')
    axes[i, 0].set_title(f"Distribution of {column}")

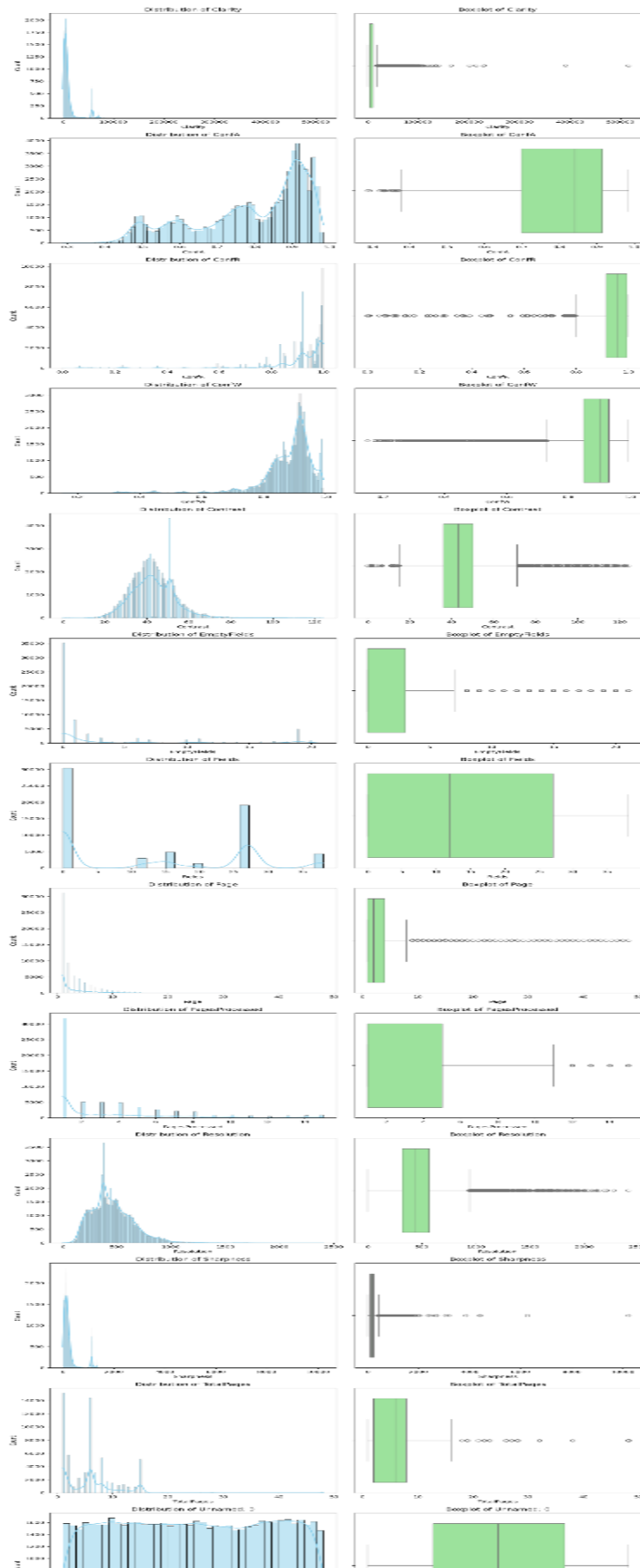
    # Boxplot to observe symmetry and possible outliers
    sns.boxplot(x=data, ax=axes[i, 1], color='lightgreen')
    axes[i, 1].set_title(f"Boxplot of {column}")

    # Calculate the skewness of the distribution
    skewness_value = skew(data.dropna()) # .dropna() to remove null values
    print(f"Skewness of the distribution of {column}: {skewness_value:.2f}")

# Adjust the layout of the subplots
plt.tight_layout()
plt.show()

```

Skewness of the distribution of Clarity: 4.13
 Skewness of the distribution of ConfA: -0.76
 Skewness of the distribution of ConfR: -3.80
 Skewness of the distribution of ConfW: -2.98
 Skewness of the distribution of Contrast: 1.26
 Skewness of the distribution of EmptyFields: 1.73
 Skewness of the distribution of Fields: 0.34
 Skewness of the distribution of Page: 2.87
 Skewness of the distribution of PagesProcessed: 1.71
 Skewness of the distribution of Resolution: 1.34
 Skewness of the distribution of Sharpness: 4.00
 Skewness of the distribution of TotalPages: 1.76
 Skewness of the distribution of Unnamed: 0: 0.00



Encontramos que la mayoría de las variables en el conjunto de datos muestran un grado de asimetría en las distribuciones.

Hallazgos

Sharpness (Nitidez): 4.00

El sesgo positivo indica que la mayoría de los valores de nitidez están concentrados en el rango bajo, pero hay algunos documentos con puntuaciones de nitidez inusualmente altas, lo que crea una larga cola hacia la derecha. Esto podría sugerir que los valores de nitidez son mayormente promedio o bajos, con algunos documentos teniendo una nitidez muy alta.

Contrast (Contraste): 1.26

Esta es una distribución moderadamente sesgada positivamente. Hay más valores en el extremo bajo de la escala de contraste, con algunos valores más altos creando una cola en el lado derecho. Esto sugiere que la mayoría de los documentos tienen un contraste moderado, con algunos teniendo un contraste mucho mayor.

Clarity (Claridad): 4.13

Similar a "Sharpness," esta es una distribución fuertemente sesgada positivamente. Esto sugiere que los valores de claridad están concentrados en el rango bajo, con algunos documentos teniendo claridad extremadamente alta, lo que tira la distribución hacia la derecha.

Resolution (Resolución): 1.34

Esta es una distribución sesgada positivamente con un valor moderado. Indica que los valores de resolución generalmente son bajos para la mayoría de los documentos, pero hay algunos documentos con alta resolución, creando una cola a la derecha en la distribución.

ConfR: -3.80

Esta es una distribución fuertemente sesgada negativamente. La gran asimetría negativa indica que la mayoría de los puntos de datos están concentrados hacia los valores más altos de "ConfR," con una cola larga de valores bajos. En este caso, "ConfR" parece tener valores mayormente concentrados en el extremo superior, con algunos valores extremadamente bajos que tiran la distribución hacia la izquierda.

ConfA: -0.76

Interpretación: Esta es una distribución moderadamente sesgada negativamente. La asimetría de -0.76 indica una distribución en la que la mayoría de los valores de "ConfA" están en el extremo superior de la escala, pero hay una cola notable hacia la izquierda (con algunos valores muy bajos que tiran la distribución hacia la izquierda).

Para abordar las distribuciones sesgadas en los datos, aplicamos transformaciones siguiendo estas pautas:

- **Transformación Logarítmica (para sesgo positivo):** Puede ayudar a normalizar los datos con sesgo positivo.
- **Transformación de raíz cuadrada o cúbica (para sesgo moderado):** También puede ayudar a reducir la asimetría.
- **Transformación Box-Cox o Yeo-Johnson:** Transformaciones más complejas que pueden manejar tanto sesgo positivo como negativo.

```
numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns.difference(['Unnamed', 'Date', 'Index'])

# Visualize original and transformed distributions
fig, axes = plt.subplots(nrows=len(numeric_columns), ncols=4, figsize=(16, len(numeric_columns)*5))

for i, column in enumerate(numeric_columns):
    # Get the original data
    data = df[column]

    # Plot original data (histogram and boxplot)
    sns.histplot(data, kde=True, ax=axes[i, 0], color='skyblue')
    axes[i, 0].set_title(f"Original Distribution of {column}")

    sns.boxplot(x=data, ax=axes[i, 1], color='lightgreen')
    axes[i, 1].set_title(f"Original Boxplot of {column}")

    # Initialize transformed data and transformation type
    transformed_data = data
    transformation_type = "None"

    # Apply transformations based on skewness
    skewness_value = skew(data.dropna())

    if skewness_value > 0.5: # Right-skewed data -> Log Transformation
        transformed_data = np.log1p(data) # log(1 + x) to handle zero or small values
        transformation_type = "Log"
    elif skewness_value > 0: # Slightly right-skewed data -> Square Root Transformation
        transformed_data = np.sqrt(data)
        transformation_type = "Sqrt"
    elif skewness_value < -0.5: # Left-skewed data -> Box-Cox Transformation (ensure positive values)
        # Ensure the data is positive, Box-Cox requires all values > 0
        data_pos = data - data.min() + 1 # Shift the data to make it positive
        transformed_data, _ = boxcox(data_pos) # Apply Box-Cox transformation
        transformation_type = "Box-Cox"

    # Convert transformed data back to a pandas Series if it's a NumPy array
    if isinstance(transformed_data, np.ndarray):
        transformed_data = pd.Series(transformed_data, index=data.index)

    # Plot transformed data (histogram and boxplot)
    sns.histplot(transformed_data, kde=True, ax=axes[i, 2], color='lightcoral')
    axes[i, 2].set_title(f"{transformation_type} Transformed Distribution of {column}")

    sns.boxplot(x=transformed_data, ax=axes[i, 3], color='lightblue')
    axes[i, 3].set_title(f"{transformation_type} Boxplot of {column}")

    # Print skewness of the transformed data
    transformed_skewness = skew(transformed_data.dropna()) # Now dropna() works since it's a Series
    print(f"Skewness of {column} after {transformation_type} transformation: {transformed_skewness:.2f}")
```




¿Se identifican tendencias temporales? (En caso de que el conjunto incluye una dimensión de tiempo).

Aunque existe un campo de "Fecha" (relacionado con la fecha en que se originó la imagen/archivo), no es relevante para el análisis por las siguientes razones:

- El campo "Fecha" no tiene un impacto claro en los datos.
- No hay variables dependientes del tiempo (tendencias o estacionalidades).
- La fecha no está asociada con eventos importantes (como promociones, temporadas altas, etc.).

¿Hay correlación entre las variables dependientes e independientes?

Debido al contexto del problema, no se ha definido una variable dependiente ni se cuenta con una variable objetivo. El problema se abordará desde una perspectiva de cluster. A pesar de esto, se mostrará una matriz de correlación para visualizar cómo todas las variables en el conjunto de datos están relacionadas entre sí.

```
[8] numeric_columns = df.select_dtypes(include=["float64", "int64"]).columns.difference(['Unnamed', 'Date', 'Index'])

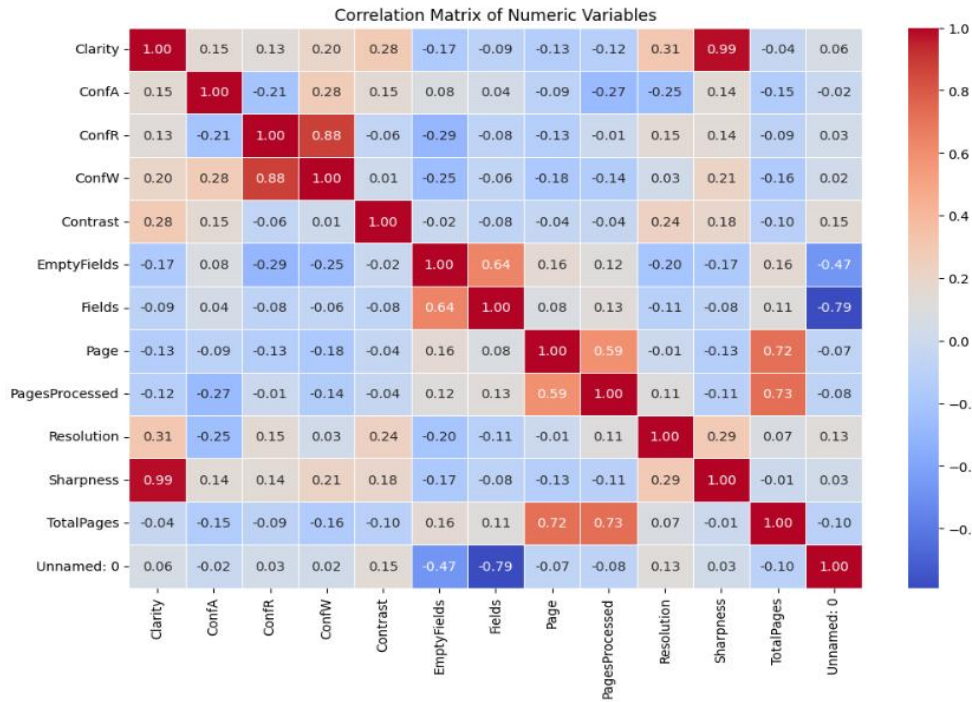
# Calcular la correlación entre las variables numéricas
correlation_matrix = df[numeric_columns].corr()

# Mostrar la matriz de correlación entre las variables numéricas
print(correlation_matrix)

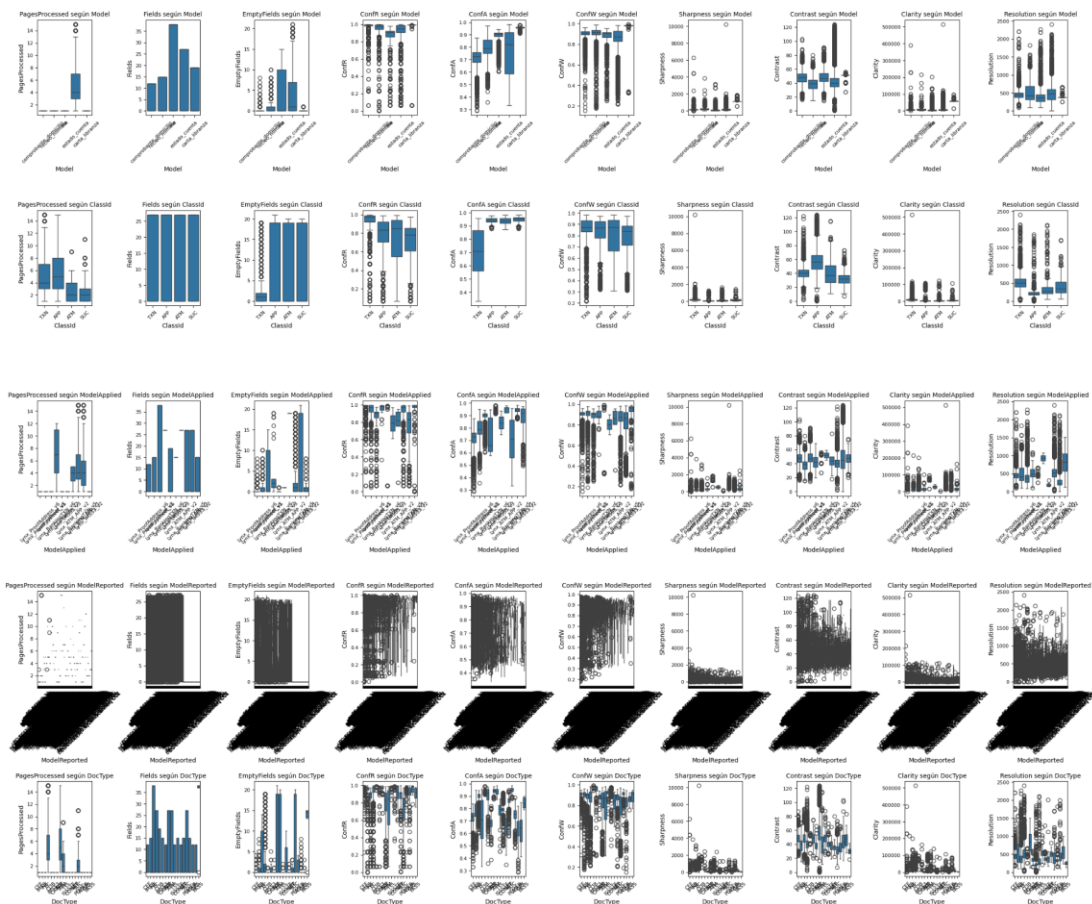
# Visualizar la matriz de correlación utilizando un heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt='.2f', linewidths=0.5)
plt.title("Correlation Matrix of Numeric Variables")
plt.show()
```

	Clarity	ConfA	ConfR	ConfW	Contrast	EmptyFields	Fields	Page	PagesProcessed	Resolution	Sharpness	TotalPages	Unnamed: 0
Clarity	1.000000	0.153853	0.127456	0.200587	0.282829	-0.165022	-0.092188	-0.134921	-0.118856	0.313468	0.985382	0.035669	0.058225
ConfA	0.153853	1.000000	-0.214491	0.278158	0.147393	0.077611	0.039716	-0.094785	-0.272182	-0.245563	0.144175	-0.035669	-0.019472
ConfR	0.127456	-0.214491	1.000000	0.878512	-0.061520	-0.294605	-0.080796	-0.130940	-0.008607	0.149684	0.143305	-0.086639	0.033116
ConfW	0.200587	0.278158	0.878512	1.000000	0.011576	-0.251788	-0.060057	-0.175170	-0.141605	0.027065	0.211438	-0.159878	0.023082
Contrast	0.282829	0.147393	-0.061520	0.011576	1.000000	-0.016318	-0.075965	-0.042542	-0.036712	0.235811	0.182808	-0.104766	0.145031
EmptyFields	-0.165022	0.077611	-0.294605	-0.251788	-0.016318	1.000000	0.636737	0.160590	0.116186	-0.204622	-0.168554	0.156180	-0.467925
Fields	-0.092188	0.039716	-0.080796	-0.060057	-0.075965	0.636737	1.000000	0.081662	0.126830	-0.114287	-0.076794	0.114830	-0.789383
Page	-0.134921	-0.094785	-0.130940	-0.175170	-0.042542	0.160590	0.081662	1.000000	0.590890	-0.009860	-0.125688	0.720107	-0.073270
PagesProcessed	-0.118856	-0.272182	-0.008607	-0.141605	-0.036712	0.116186	0.126830	0.590890	1.000000	0.107821	-0.105753	0.725317	-0.078377
Resolution	0.313468	-0.245563	0.149684	0.027065	0.235811	-0.204622	-0.114287	-0.009860	0.107821	1.000000	0.289668	0.072225	0.125893
Sharpness	0.985382	0.144175	0.143305	0.211438	0.182808	-0.168554	-0.076794	-0.125688	-0.105753	0.289668	1.000000	-0.007548	0.027506
TotalPages	-0.035669	-0.152626	-0.086639	-0.159878	-0.104766	0.156180	0.114830	0.720107	0.725317	0.072225	-0.007548	1.000000	0.027506
Unnamed: 0	0.058225	-0.019472	0.033116	0.023082	0.145031	-0.467925	-0.789383	-0.073270	-0.078377	0.125893	0.027506	0.027506	0.027506

1.



¿Cómo se distribuyen los datos en función de diferentes categorías (análisis bivariado)?



El análisis revela que la cantidad de páginas procesadas varía según **DocType**, con algunas categorías mostrando mayor procesamiento, mientras que otras presentan alta variabilidad en **Resolution** y **Clarity**, indicando posibles outliers. En cuanto a los modelos, **estado_cuenta** procesa más páginas en promedio, mientras que **carta_libranza** y **comprobante_domicilio** mantienen una distribución uniforme con solo una página. También se observa una mayor dispersión en **Resolution** y **Clarity**, lo que sugiere diferencias en la calidad de los documentos procesados.

Para **ClassId**, APP destaca por procesar más páginas en comparación con **ATM** y **SUC**, que tienen una menor cantidad. Asimismo, la variabilidad en **Resolution** y **Clarity** refuerza la idea de que ciertos documentos presentan diferencias en calidad y nitidez.

Los modelos aplicados muestran un comportamiento diferenciado, con **Lynx_BankingTXN_v5** liderando en páginas procesadas, mientras que **Lynx_ATM_APP_v1** y **Lynx_ATM_APP_v2** procesan menos. Se detectan valores máximos elevados en **Resolution**, lo que indica variaciones en la calidad del documento. Finalmente, los modelos reportados presentan valores constantes en **PagesProcessed**, mientras que **Resolution** y **Clarity** varían entre ellos, reflejando diferencias en la calidad de los documentos capturados.

¿Se deberían normalizar las imágenes para visualizarlas mejor?

Sí, normalizar los datos puede mejorar su visualización y análisis, especialmente en nuestro caso donde algunas variables tienen escalas muy diferentes (por ejemplo, **Resolution** varía en un rango amplio, mientras que **ConfR** y **ConfA** están entre 0.5 y 1.0). La normalización permite comparar mejor las distribuciones en gráficos y facilita la interpretación de patrones. Además, ayuda a reducir el impacto de valores extremos y distribuciones asimétricas, haciendo que los análisis bivariados sean más consistentes y visualmente claros.

```
# Normalizar los datos
scaler = MinMaxScaler()
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])
fig, axes = plt.subplots(len(categorical_columns), len(numerical_columns), figsize=(30, 25))
fig.subplots_adjust(hspace=1, wspace=1)
```

¿Hay desequilibrio en las clases de la variable objetivo?

No hay una variable objetivo en el sentido tradicional, ya que el modelo descubre patrones o agrupaciones en los datos sin etiquetas predefinidas.

CONCLUSIONES

1. **Clasificación del modelo**

El modelo clasificador actualmente está diseñado para realizar una clasificación en tres niveles: **bueno, regular y malo**.

2. **Valores faltantes en el conjunto de datos**

El **dataframe** contiene un total de **62,960 registros**, dentro de los cuales se identificaron valores faltantes en dos columnas clave:

- **ClassID**: 28,820 valores faltantes.
- **ModelReported**: 18,627 valores faltantes.

3. **Valores atípicos en los datos**

La columna con el mayor número de valores atípicos es "**Claridad**" de la página. Tras el análisis, se determinó que el **8.691%** de los registros corresponden a valores atípicos. Aunque este porcentaje no es despreciable, no implica necesariamente un impacto negativo en el desempeño del modelo de clasificación.

4. La normalización de datos mejora la comparación entre variables con escalas diferentes, facilitando la interpretación de patrones.
5. Se identifican variaciones significativas entre categorías en métricas clave como **Resolution** y **Clarity**, sugiriendo diferencias en la calidad del procesamiento.
6. No hay una variable objetivo, ya que el enfoque del trabajo es la clusterización