

UNIVERSIDAD NACIONAL DE TRUJILLO
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS



ALUMNOS:

- **PARIMANGO GOMEZ KEVIN ESTEEVEN**
- **VILLACORTA CARRANZA, RONALD DAVID**

ASIGNATURA:

ANALÍTICA DE NEGOCIOS

DOCENTE:

Ing. Ricardo Mendoza Rivera

TRUJILLO – PERÚ

2026

1. Problemática Central

La institución educativa cuenta con una población de 1,000 estudiantes con rendimientos muy diversos en las áreas de matemáticas y lectura. Actualmente, se aplica la misma estrategia pedagógica para todos, lo que genera:

- Bajo aprovechamiento en alumnos con potencial alto.
- Falta de apoyo específico para alumnos en riesgo de reprobación.
- Desperdicio de recursos en programas de tutoría generalizados.

Pregunta de investigación: ¿Cómo agrupar a los estudiantes en segmentos homogéneos según sus competencias académicas para diseñar planes de intervención personalizados?

2. Herramientas y Metodología Utilizada

Para la resolución de este caso, se utilizó el lenguaje de programación **Python** y las siguientes librerías especializadas:

- **Pandas / Numpy:** Para el procesamiento y limpieza de datos.
- **Matplotlib / Seaborn:** Para la visualización de distribuciones y correlaciones.
- **Scikit-Learn:** Para la implementación del algoritmo **K-Means** y el escalamiento de datos (**StandardScaler**).

3. Desarrollo del Proyecto (Paso a Paso)

A) Calidad de Datos y Análisis Descriptivo

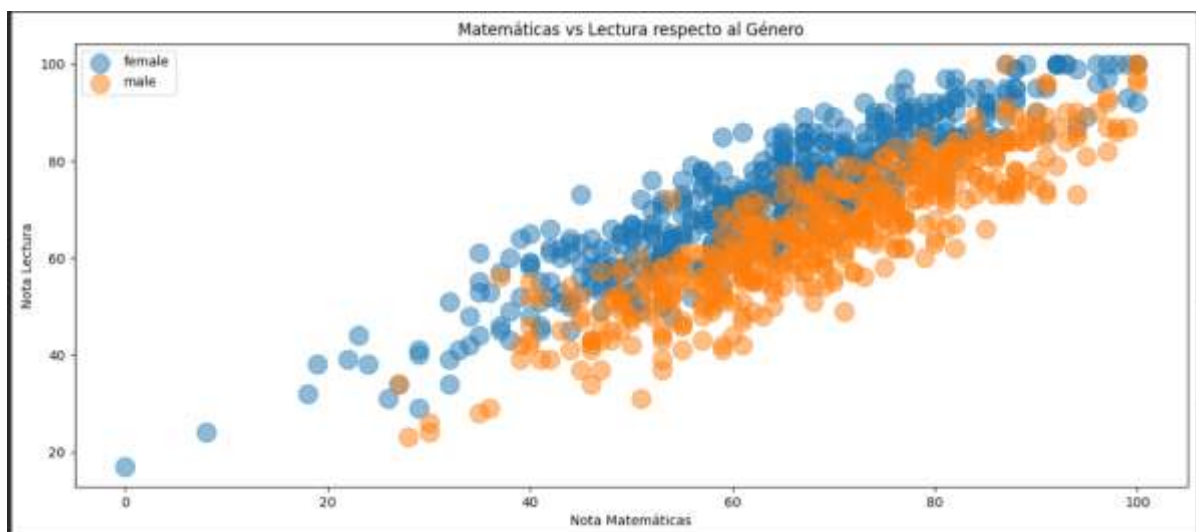
Se realizó una auditoría de la data para asegurar que no existieran valores nulos que sesgaran el modelo. Se calculó una nueva variable denominada mean score (promedio de notas).

	math score	reading score	writing score	mean_score	Grupo
count	1000.00000	1000.000000	1000.000000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000	67.762000	1.590000
std	15.16308	14.600192	15.195657	14.258354	1.146828
min	0.00000	17.000000	10.000000	9.000000	0.000000
25%	57.00000	59.000000	57.750000	58.000000	1.000000
50%	66.00000	70.000000	69.000000	68.000000	2.000000
75%	77.00000	79.000000	79.000000	78.000000	2.000000
max	100.00000	100.000000	100.000000	100.000000	4.000000

	0
gender	0
race/ethnicity	0
parental level of education	0
lunch	0
test preparation course	0
math score	0
reading score	0
writing score	0
mean_score	0

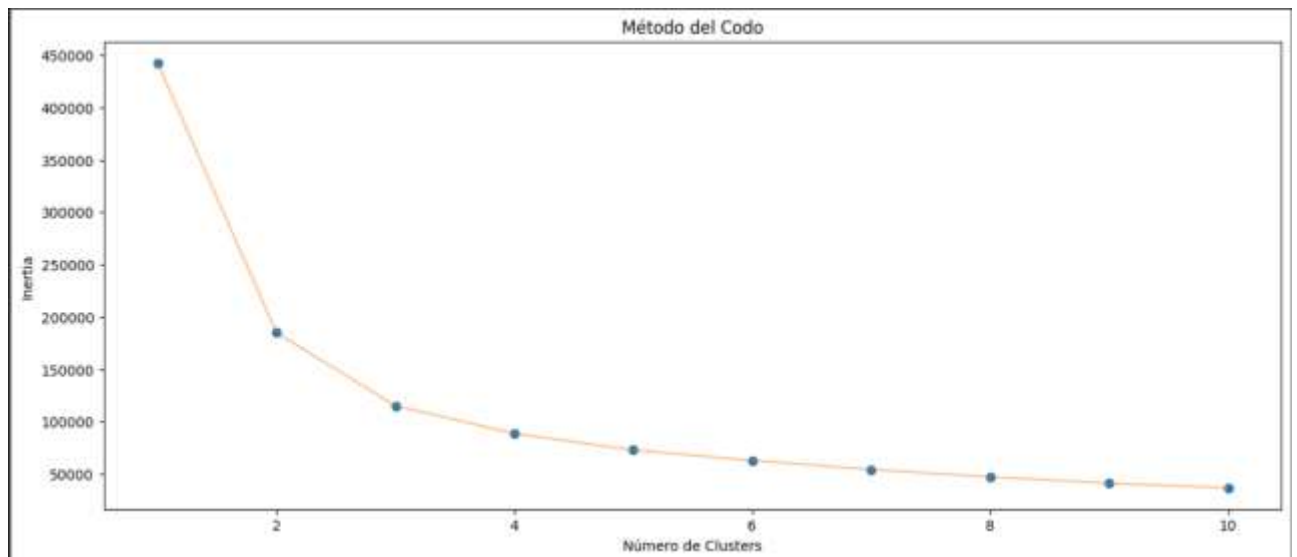
B) Análisis Exploratorio (EDA)

Se analizaron las distribuciones de las notas y la relación entre Matemáticas y Lectura, segmentando visualmente por género para observar si existían patrones previos.



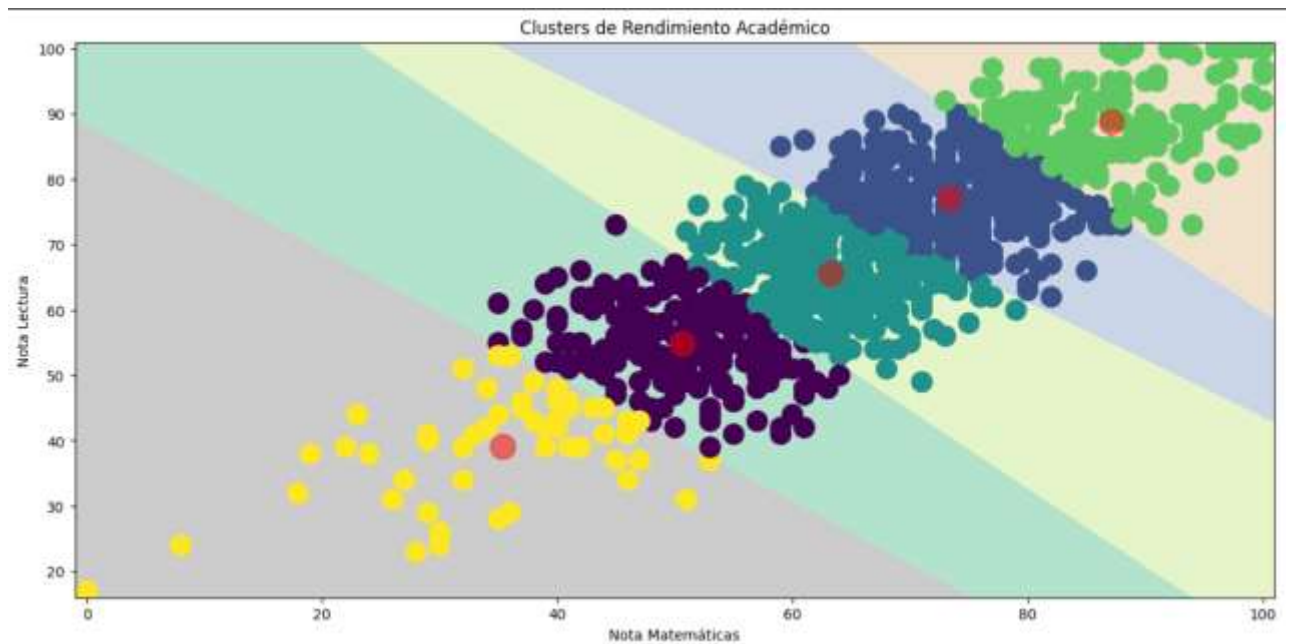
C) Determinación del Número Óptimo de Grupos (Método del Codo)

Siguiendo la metodología del Dr. Mendoza, se ejecutó el **Método del Codo (Elbow Method)** para calcular la inercia y determinar que 5 clústeres es la cantidad adecuada para segmentar a los alumnos sin sobreajustar el modelo.



4. Resultados del Modelo (Clusterización)

Se aplicó el algoritmo **K-Means** con 5 centroides inicializados con k-means++. El resultado visual muestra zonas de influencia claramente definidas.



5. Interpretación y Toma de Decisiones

Basándonos en el modelo resultante, hemos identificado los siguientes perfiles:

1. **Grupo Premium (Excelencia):** Alumnos con notas sobresalientes en ambas áreas. **Acción:** Integrarlos a programas de monitores o becas de excelencia.
2. **Hábiles en Letras:** Alumnos con alta nota en lectura pero baja en matemáticas. **Acción:** Refuerzo específico en razonamiento lógico-matemático.
3. **Hábiles en Números:** Alumnos con alta nota en matemáticas pero baja en lectura. **Acción:** Talleres de comprensión lectora y redacción.
4. **Grupo Promedio:** Rendimiento estable. **Acción:** Seguimiento preventivo semestral.
5. **Grupo de Riesgo Crítico:** Alumnos con promedios muy bajos en ambas áreas. **Acción:** Intervención pedagógica urgente y tutorías personalizadas.

6. Conclusión

El modelo de Machine Learning permitió convertir datos crudos en **conocimiento accionable**. Al asignar a cada alumno un "Grupo" (Cluster ID), la institución ahora puede automatizar la asignación de recursos, asegurando que cada estudiante reciba el apoyo que realmente necesita según su perfil académico.

Resultado Final de la Segmentación:				
	gender	math score	reading score	Grupo
0	female	72	72	1
1	female	69	90	1
2	female	90	95	3
3	male	47	57	0
4	male	76	78	1

ANEXOS:

Algoritmo: K-Means Clustering (Aprendizaje No Supervisado)

Dataset: <https://www.kaggle.com/datasets/spscientist/students-performance-in-exams/data>

Código:

```
# =====  
  
# a) Importando Librerias  
# =====  
import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
from sklearn.cluster import KMeans  
import seaborn as sns  
import plotly as py  
import plotly.graph_objs as go  
import warnings  
import os  
warnings.filterwarnings("ignore")  
  
# b) Leyendo datos  
# =====  
# Usamos el dataset de Kaggle que elegiste  
df = pd.read_csv('StudentsPerformance.csv')  
  
# Pre-procesamiento inicial: Creamos el promedio (EDA)  
df["mean_score"] = ((df["math score"] + df["reading score"] +  
df["writing score"]) / 3).round()  
  
# c) Datos descriptivos  
# =====  
# valores promedios de las notas  
df.describe()  
  
# d) Calidad de data (Nulos)  
# =====  
# Verificando si hay información faltante  
df.isnull().sum()  
  
# e) Analizando datos por variable  
# =====
```

```

plt.figure(1 , figsize = (15 , 6))
n = 0
# Adaptamos a las variables de notas
for x in ['math score' , 'reading score' , 'writing score']:
    n += 1
    plt.subplot(1 , 3 , n)
    plt.subplots_adjust(hspace =0.5 , wspace = 0.5)
    sns.distplot(df[x] , bins = 20)
    plt.title('Distplot de {}'.format(x))
plt.show()

# Analizando el género

plt.figure(1 , figsize = (15 , 5))
sns.countplot(y = 'gender' , data = df)
plt.show()

# f) Análisis de variables en conjunto

# =====
# Note donde los puntos se entrecruzan (Matemáticas vs Lectura)
plt.figure(1 , figsize = (15 , 6))
for gender in ['female' , 'male']:
    plt.scatter(x = 'math score' , y = 'reading score' ,
                data = df[df['gender'] == gender] ,
                s = 200 , alpha = 0.5 , label = gender)

plt.xlabel('Nota Matemáticas') , plt.ylabel('Nota Lectura')
plt.title('Matemáticas vs Lectura respecto al Género')
plt.legend()
plt.show()

# g) Encontrando número de clusters (Método del Codo)

# =====
# Seleccionamos Math Score y Reading Score para agrupar
X2 = df[['math score' , 'reading score']].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n , init='k-means++' , n_init =
10 , max_iter=300,
                                tol=0.0001, random_state= 111 ,
algorithm='elkan') )
    algorithm.fit(X2)
    inertia.append(algorithm.inertia_)

```

```

# Graficamos la inercia

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Número de Clusters') , plt.ylabel('Inertia')
plt.title('Método del Codo')
plt.show()

# h) Aplicando clusters de acuerdo a análisis anterior

# =====
# Se trabaja con 5 clusters
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10
,max_iter=300,
                    tol=0.0001, random_state= 111 ,
algorithm='elkan') )
algorithm.fit(X2)
labels2 = algorithm.labels_
centroids2 = algorithm.cluster_centers_

# i) Datos para graficas clusters determinados por modelo

# =====
h = 0.02
x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1
y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
y_max, h))
Z2 = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

# Graficando grupos con el fondo de color (Paso i del profe)

plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z2 = Z2.reshape(xx.shape)
plt.imshow(Z2 , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'math score' , y = 'reading score' , data = df , c
= labels2 , s = 200 )
plt.scatter(x = centroids2[:, 0] , y = centroids2[:, 1] , s = 300
, c = 'red' , alpha = 0.5)
plt.ylabel('Nota Lectura') , plt.xlabel('Nota Matemáticas')

```

```
plt.title('Clusters de Rendimiento Académico')
plt.show()

# j) Asignando modelo a cada alumno

# =====
df['Grupo'] = labels2

## Verifique el grupo creado y asignado a cada alumno.

print("Resultado Final de la Segmentación:")
df[['gender', 'math score', 'reading score', 'Grupo']].head()
```

ALGORITMO 2: - Logistic regression

1. Problemática Central

La institución desea predecir si un estudiante aprobará o no el examen final basándose en factores socio-académicos (género, nivel educativo de los padres, almuerzo y curso de preparación). El objetivo es identificar a los estudiantes en riesgo **antes** de que tomen el examen para asignarles apoyo preventivo.

Variable Objetivo: Aprobado (1) / No Aprobado (0). (Se define aprobación con una nota promedio ≥ 60).

2. Código del Modelo :

```
# =====

# a) Importando Librerías

# =====

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split
```

```

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import classification_report, confusion_matrix

from sklearn.preprocessing import LabelEncoder

import warnings

warnings.filterwarnings("ignore")

# =====

# b) Leyendo datos

# =====

df = pd.read_csv('StudentsPerformance.csv')

# Pre-procesamiento: Definir la variable objetivo 'Aprobado'

df["mean_score"] = ((df["math score"] + df["reading score"] + df["writing score"]) /
3).round()

# Si la nota es mayor o igual a 60, aprueba (1), de lo contrario reprueba (0)

df['Result'] = df['mean_score'].apply(lambda x: 1 if x >= 60 else 0)

# =====

# c) Datos descriptivos

# =====

# Note el balance de la clase 'Result'

print(df['Result'].value_counts())

df.describe()

# =====

# d) Calidad de data (Nulos)

# =====

df.isnull().sum()

# =====

# e) Analizando datos por variable

```

```

# =====

# Visualizamos cuántos aprobaron vs cuántos reprobaron

plt.figure(figsize=(8, 5))

sns.countplot(x='Result', data=df, palette='Set1')

plt.title('Distribución de Estudiantes Aprobados (1) vs Reprobados (0)')

plt.show()


# =====

# f) Análisis de variables en conjunto (Correlación)

# =====

# Transformamos variables categóricas a numéricas para el análisis

le = LabelEncoder()

df_encoded = df.copy()

for col in ['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation
course']:

    df_encoded[col] = le.fit_transform(df[col])


plt.figure(figsize=(10, 8))

sns.heatmap(df_encoded.corr(), annot=True, cmap='RdYlGn')

plt.title('Matriz de Correlación de Variables')

plt.show()


# =====

# g) Preparando datos para el entrenamiento (Split)

# =====

# Seleccionamos variables predictoras y la variable objetivo

X = df_encoded[['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test
preparation course']]

y = df_encoded['Result']


# Dividimos en 80% entrenamiento y 20% pruebas (Testing)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=111)

```

```

# =====

# h) Aplicando algoritmo de ML (Regresión Logística)

# =====

model = LogisticRegression(solver='liblinear')
model.fit(X_train, y_train)

# =====

# i) Evaluación del Modelo (Métricas)

# =====

predictions = model.predict(X_test)

# Generamos la matriz de confusión
print("Matriz de Confusión:")
cm = confusion_matrix(y_test, predictions)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicción')
plt.ylabel('Real')
plt.show()

# Reporte de Clasificación
print("\nReporte de Clasificación:")
print(classification_report(y_test, predictions))

# =====

# j) Asignando modelo/Predicción Final

# =====

# Aplicamos la predicción a todo el dataset original para ver quiénes "deberían" aprobar
df['Prediccion_Final'] = model.predict(X)

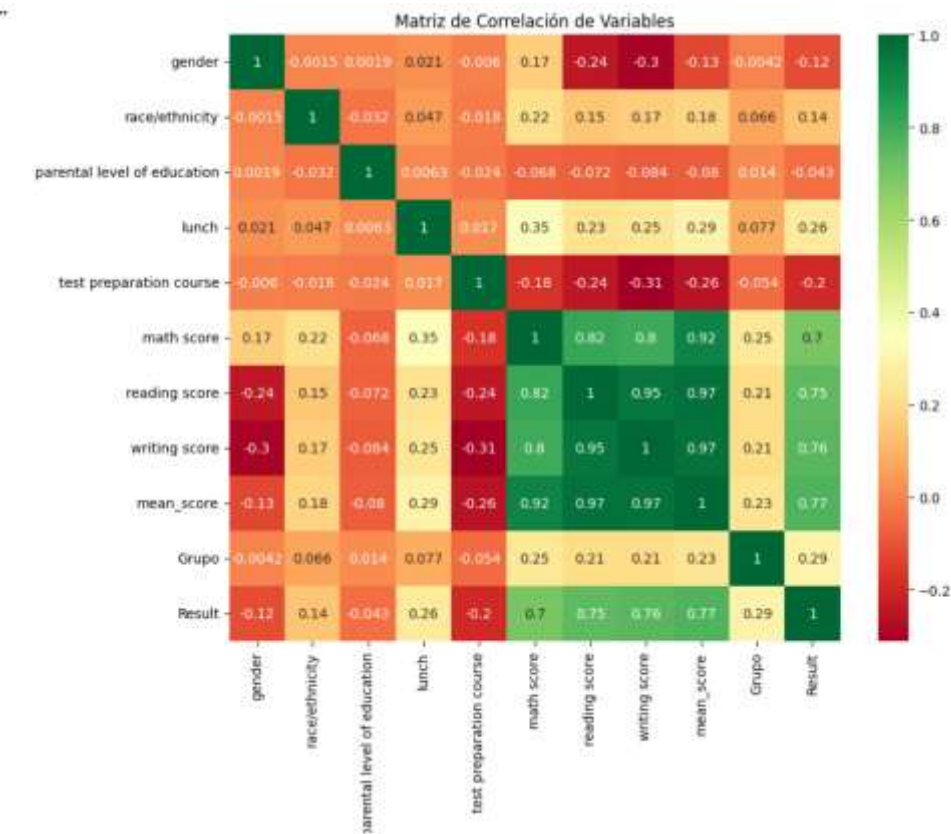
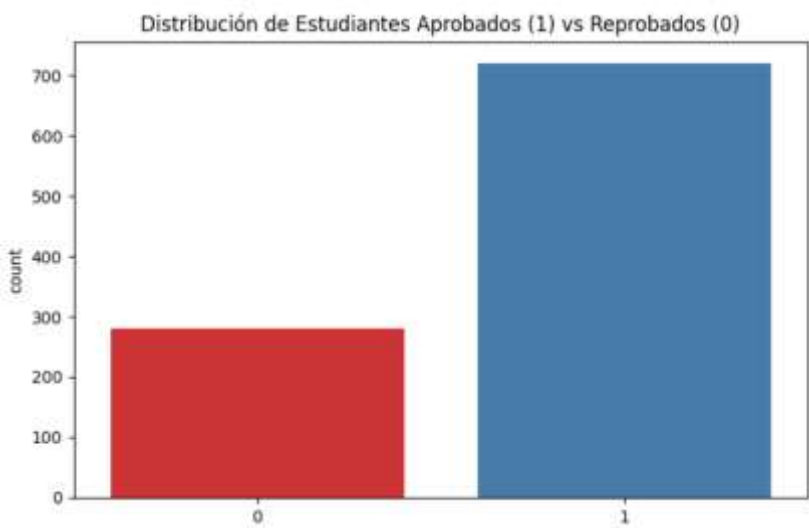
print("Comparación de Datos Reales vs Predicción:")

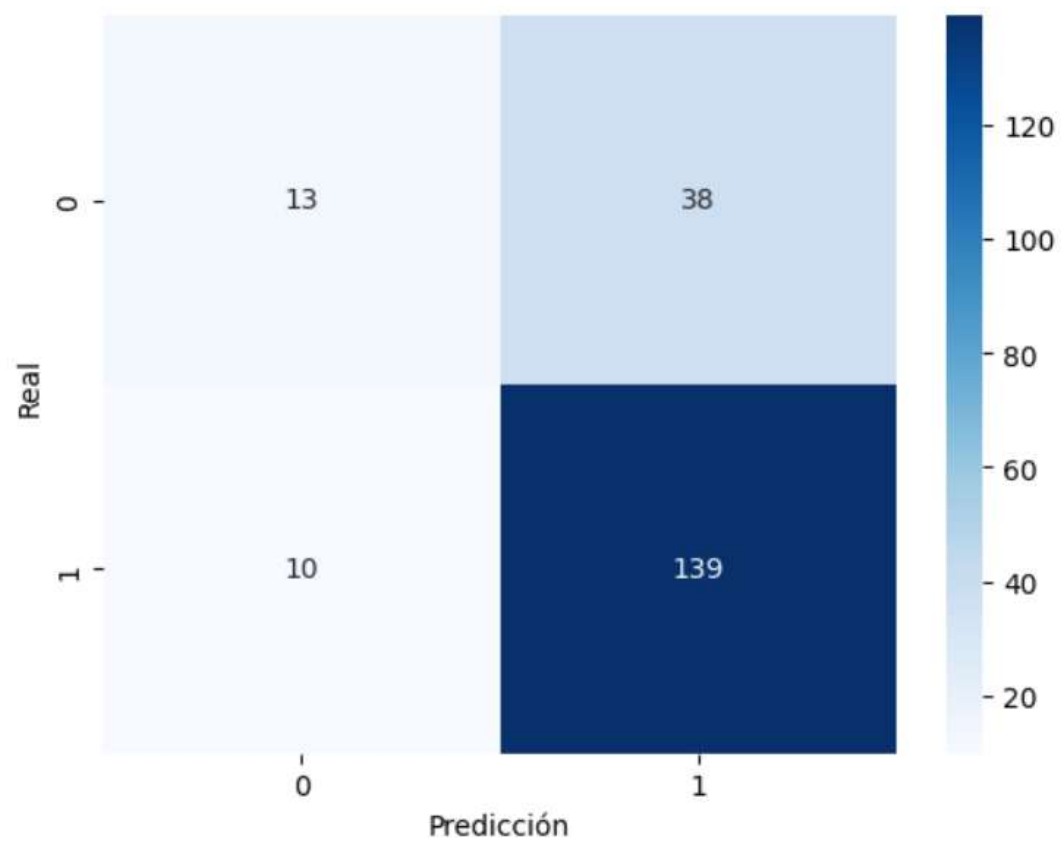
```

```
df[['gender', 'mean_score', 'Result', 'Prediccion_Final']].head(10)
```

3. Instrucciones para tu Presentación

Para que el informe se vea completo, sigue estas instrucciones al ejecutar el código:





Reporte de Clasificación:

	precision	recall	f1-score	support
0	0.57	0.25	0.35	51
1	0.79	0.93	0.85	149
accuracy			0.76	200
macro avg	0.68	0.59	0.60	200
weighted avg	0.73	0.76	0.72	200

*** Comparación de Datos Reales vs Predicción:

	gender	mean_score	Result	Prediccion_Final
0	female	73.0	1	1
1	female	82.0	1	1
2	female	93.0	1	1
3	male	49.0	0	0
4	male	76.0	1	1
5	female	77.0	1	1
6	female	92.0	1	1
7	male	41.0	0	0
8	male	65.0	1	1
9	female	49.0	0	0

Algoritmo: Logistic Regression(Supervisado)

Dataset: <https://www.kaggle.com/datasets/spscientist/students-performance-in-exams/data>