Importar librerias

```
# Instalar librerias si es necesario
#!pip install pandas numpy matplotlib seaborn scikit-learn statsmodels

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import LabelEncoder
import statsmodels.api as sm
```

Cargar los datos

```
# Cambia el nombre del archivo por el tuyo
df = pd.read_csv('student_lifestyle_dataset.csv')

# Ver primeros datos
print("Primeras filas del dataset:")
print(df.head())
```

```
Primeras filas del dataset:
   Student_ID  Study_Hours_Per_Day  Extracurricular_Hours_Per_Day  \
0           1                  6.9                            3.8
1           2                  5.3                            3.5
2           3                  5.1                            3.9
3           4                  6.5                            2.1
4           5                  8.1                            0.6

   Sleep_Hours_Per_Day  Social_Hours_Per_Day
Physical_Activity_Hours_Per_Day  \
0                  8.7                   2.8
1.8
1                  8.0                   4.2
3.0
2                  9.2                   1.2
4.6
3                  7.2                   1.7
6.5
4                  6.5                   2.2
6.6

    GPA Stress_Level
```

```
0  2.99     Moderate
1  2.75          Low
2  2.67          Low
3  2.88     Moderate
4  3.51         High
```

Informacion basica del dataset

```
print("Informacion del dataset:")
print(df.info())

print("\nEstadisticas basicas:")
print(df.describe())
```

```
Informacion del dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 8 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Student_ID                      2000 non-null   int64  
 1   Study_Hours_Per_Day             2000 non-null   float64
 2   Extracurricular_Hours_Per_Day   2000 non-null   float64
 3   Sleep_Hours_Per_Day             2000 non-null   float64
 4   Social_Hours_Per_Day            2000 non-null   float64
 5   Physical_Activity_Hours_Per_Day  2000 non-null   float64
 6   GPA                             2000 non-null   float64
 7   Stress_Level                    2000 non-null   object 
dtypes: float64(6), int64(1), object(1)
memory usage: 125.1+ KB
None

Estadisticas basicas:
        Student_ID  Study_Hours_Per_Day  Extracurricular_Hours_Per_Day
\
count  2000.000000          2000.000000                    2000.000000

mean   1000.500000             7.475800                       1.990100

std     577.494589             1.423888                       1.155855

min       1.000000             5.000000                       0.000000

25%     500.750000             6.300000                       1.000000

50%    1000.500000             7.400000                       2.000000

75%    1500.250000             8.700000                       3.000000
```

```
max    2000.000000             10.000000                              4.000000


       Sleep_Hours_Per_Day  Social_Hours_Per_Day  \
count          2000.000000           2000.000000
mean              7.501250              2.704550
std               1.460949              1.688514
min               5.000000              0.000000
25%               6.200000              1.200000
50%               7.500000              2.600000
75%               8.800000              4.100000
max              10.000000              6.000000

       Physical_Activity_Hours_Per_Day          GPA
count                       2000.00000  2000.000000
mean                           4.32830     3.115960
std                            2.51411     0.298674
min                            0.00000     2.240000
25%                            2.40000     2.900000
50%                            4.10000     3.110000
75%                            6.10000     3.330000
max                           13.00000     4.000000
```

Preparar los datos

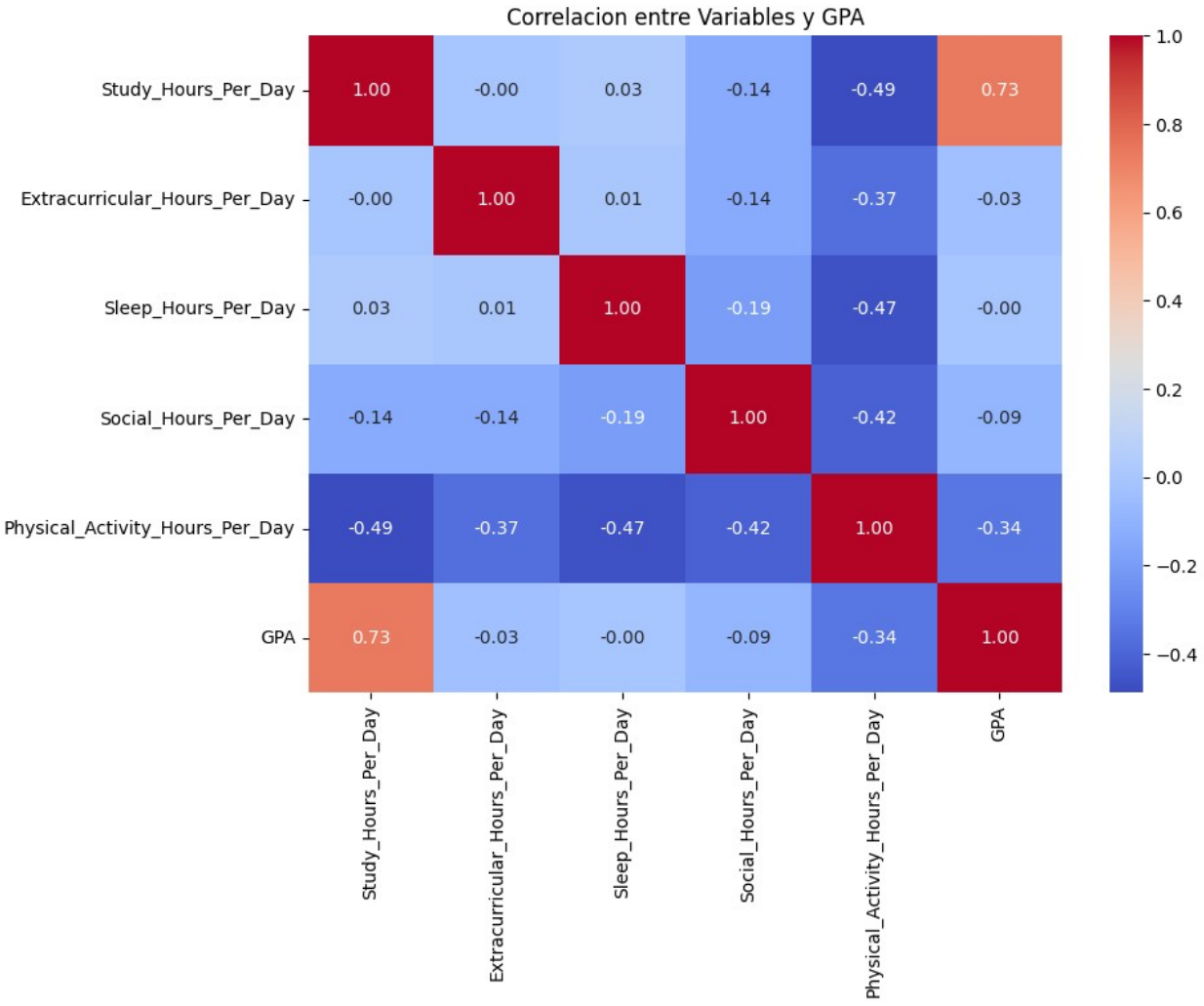
```
# Codificar nivel de estres si es texto
if 'Stress_Level' in df.columns:
    le = LabelEncoder()
    df['Stress_Level_Num'] = le.fit_transform(df['Stress_Level'])
    print("Niveles de estres codificados:", dict(zip(le.classes_,
le.transform(le.classes_))))

# Crear transformaciones no lineales para horas de sueño
df['Sleep_Hours_Squared'] = df['Sleep_Hours_Per_Day'] ** 2
df['Sleep_Hours_Cubed'] = df['Sleep_Hours_Per_Day'] ** 3
```

```
Niveles de estres codificados: {'High': np.int64(0), 'Low':
np.int64(1), 'Moderate': np.int64(2)}
```

Matriz de correlacion

```
plt.figure(figsize=(10, 8))
numeric_cols = ['Study_Hours_Per_Day',
'Extracurricular_Hours_Per_Day',
                'Sleep_Hours_Per_Day', 'Social_Hours_Per_Day',
                'Physical_Activity_Hours_Per_Day', 'GPA']
```

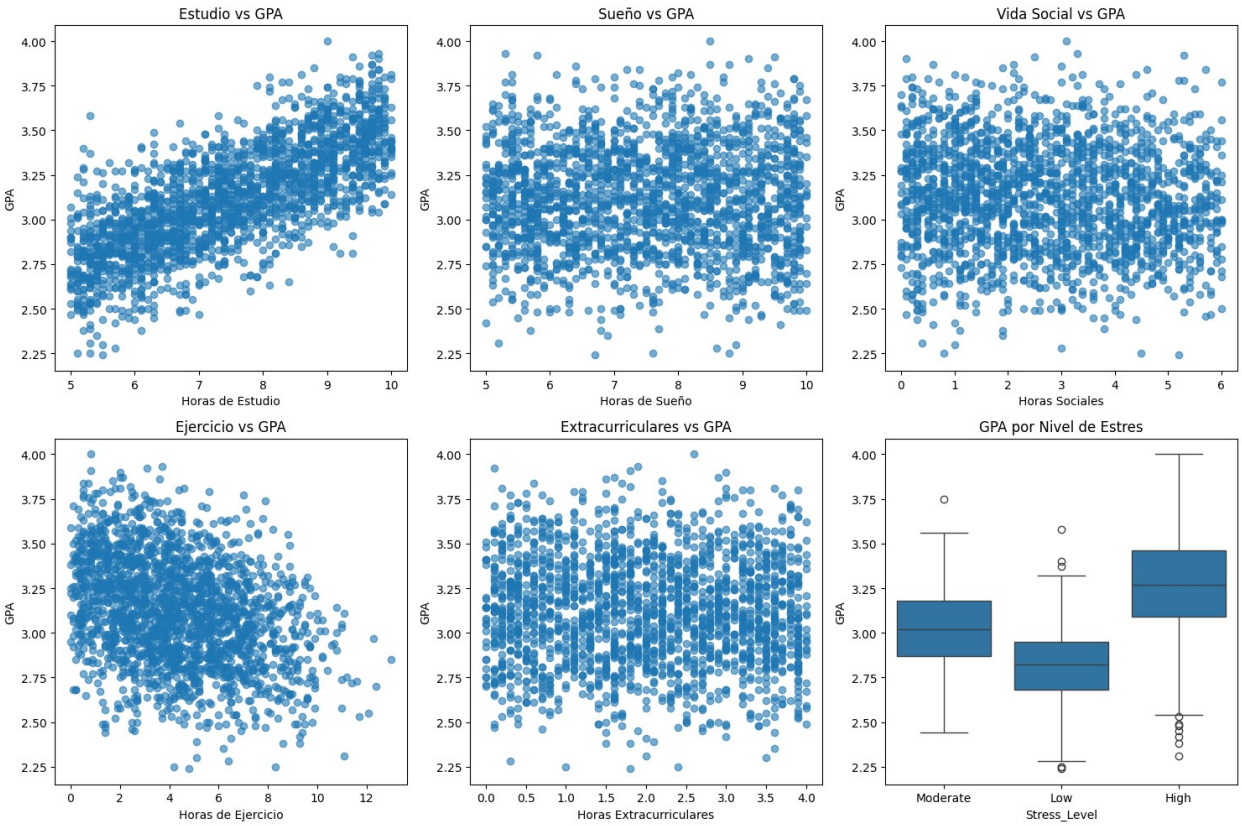
```
corr_matrix = df[numeric_cols].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlacion entre Variables y GPA')
plt.tight_layout()
plt.show()
```



Graficos de relaciones individuales

```
fig, axes = plt.subplots(2, 3, figsize=(15, 10))

# Estudio vs GPA
axes[0,0].scatter(df['Study_Hours_Per_Day'], df['GPA'], alpha=0.6)
axes[0,0].set_xlabel('Horas de Estudio')
axes[0,0].set_ylabel('GPA')
axes[0,0].set_title('Estudio vs GPA')
```

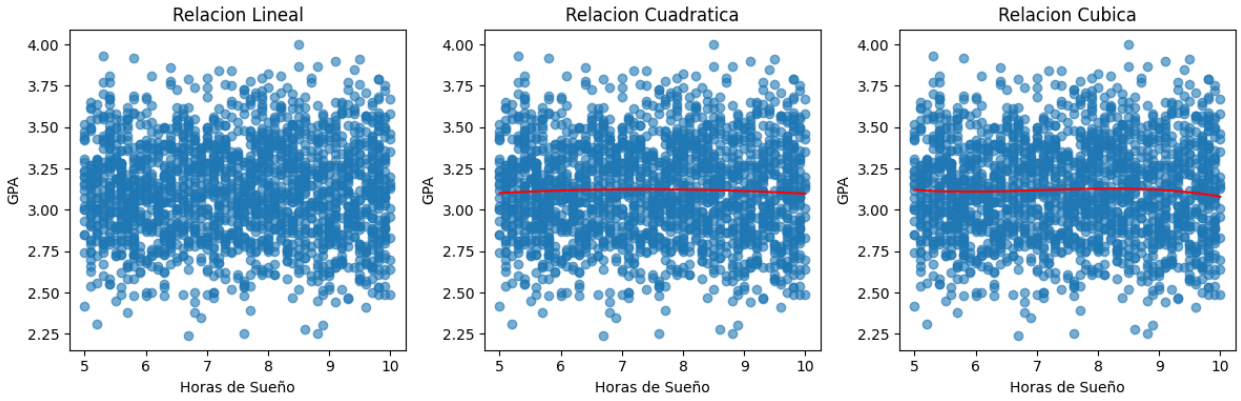
```python
# Sueño vs GPA
axes[0,1].scatter(df['Sleep_Hours_Per_Day'], df['GPA'], alpha=0.6)
axes[0,1].set_xlabel('Horas de Sueño')
axes[0,1].set_ylabel('GPA')
axes[0,1].set_title('Sueño vs GPA')

# Actividad Social vs GPA
axes[0,2].scatter(df['Social_Hours_Per_Day'], df['GPA'], alpha=0.6)
axes[0,2].set_xlabel('Horas Sociales')
axes[0,2].set_ylabel('GPA')
axes[0,2].set_title('Vida Social vs GPA')

# Actividad Fisica vs GPA
axes[1,0].scatter(df['Physical_Activity_Hours_Per_Day'], df['GPA'],
alpha=0.6)
axes[1,0].set_xlabel('Horas de Ejercicio')
axes[1,0].set_ylabel('GPA')
axes[1,0].set_title('Ejercicio vs GPA')

# Actividades Extracurriculares vs GPA
axes[1,1].scatter(df['Extracurricular_Hours_Per_Day'], df['GPA'],
alpha=0.6)
axes[1,1].set_xlabel('Horas Extracurriculares')
axes[1,1].set_ylabel('GPA')
axes[1,1].set_title('Extracurriculares vs GPA')

# Estres vs GPA
if 'Stress_Level' in df.columns:
    sns.boxplot(data=df, x='Stress_Level', y='GPA', ax=axes[1,2])
    axes[1,2].set_title('GPA por Nivel de Estres')

plt.tight_layout()
plt.show()
```



Analisis de sueño no lineal

```
plt.figure(figsize=(12, 4))

# Lineal
plt.subplot(1, 3, 1)
plt.scatter(df['Sleep_Hours_Per_Day'], df['GPA'], alpha=0.6)
plt.xlabel('Horas de Sueño')
plt.ylabel('GPA')
plt.title('Relacion Lineal')

# Cuadratica
plt.subplot(1, 3, 2)
sleep_sorted = df.sort_values('Sleep_Hours_Per_Day')
z = np.polyfit(sleep_sorted['Sleep_Hours_Per_Day'],
sleep_sorted['GPA'], 2)
p = np.poly1d(z)
plt.scatter(df['Sleep_Hours_Per_Day'], df['GPA'], alpha=0.6)
plt.plot(sleep_sorted['Sleep_Hours_Per_Day'],
p(sleep_sorted['Sleep_Hours_Per_Day']), 'r-')
plt.xlabel('Horas de Sueño')
plt.ylabel('GPA')
plt.title('Relacion Cuadratica')
```

```
# Cubica
plt.subplot(1, 3, 3)
z_cubic = np.polyfit(sleep_sorted['Sleep_Hours_Per_Day'],
sleep_sorted['GPA'], 3)
p_cubic = np.poly1d(z_cubic)
plt.scatter(df['Sleep_Hours_Per_Day'], df['GPA'], alpha=0.6)
plt.plot(sleep_sorted['Sleep_Hours_Per_Day'],
p_cubic(sleep_sorted['Sleep_Hours_Per_Day']), 'r-')
plt.xlabel('Horas de Sueño')
plt.ylabel('GPA')
plt.title('Relacion Cubica')

plt.tight_layout()
plt.show()
```



Preparar datos para regresion

```
# Variables predictoras
X = df[['Study_Hours_Per_Day', 'Extracurricular_Hours_Per_Day',
        'Sleep_Hours_Per_Day', 'Sleep_Hours_Squared',
'Sleep_Hours_Cubed',
        'Social_Hours_Per_Day', 'Physical_Activity_Hours_Per_Day']]

# Variable a predecir
y = df['GPA']

# Dividir en entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

Modelo de regresion lineal

```
model = LinearRegression()
model.fit(X_train, y_train)

# Predecir y evaluar
y_pred = model.predict(X_test)

print("RESULTADOS DEL MODELO:")
print(f"R² en entrenamiento: {model.score(X_train, y_train):.4f}")
print(f"R² en prueba: {model.score(X_test, y_test):.4f}")
print(f"Error cuadratico medio: {mean_squared_error(y_test,
y_pred):.4f}")
```

```
RESULTADOS DEL MODELO:
R² en entrenamiento: 0.5385
R² en prueba: 0.5499
Error cuadratico medio: 0.0420
```

Coeficientes del modelo

```
coeficientes = pd.DataFrame({
    'Variable': X.columns,
    'Coeficiente': model.coef_
}).sort_values('Coeficiente', key=abs, ascending=False)

print("\nIMPACTO DE CADA VARIABLE EN GPA:")
print(coeficientes)
```

```
IMPACTO DE CADA VARIABLE EN GPA:
                         Variable  Coeficiente
0              Study_Hours_Per_Day     0.149117
2              Sleep_Hours_Per_Day    -0.125797
3              Sleep_Hours_Squared     0.016686
1    Extracurricular_Hours_Per_Day    -0.014745
6  Physical_Activity_Hours_Per_Day    -0.004527
5             Social_Hours_Per_Day    -0.004048
4                Sleep_Hours_Cubed    -0.000768
```

Modelo estadistico detallado

```
X_with_const = sm.add_constant(X)
model_sm = sm.OLS(y, X_with_const).fit()

print("\nMODELO ESTADISTICO COMPLETO:")
print(model_sm.summary())
```

MODELO ESTADISTICO COMPLETO:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    GPA   R-squared:                       0.541
Model:                            OLS   Adj. R-squared:                  0.540
Method:                 Least Squares   F-statistic:                     391.7
Date:                Thu, 20 Nov 2025   Prob (F-statistic):               0.00
Time:                        04:13:55   Log-Likelihood:                358.38
No. Observations:                2000   AIC:                            -702.8
Df Residuals:                    1993   BIC:                            -663.6
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
const                            0.0189      0.007      2.892      0.004       0.006       0.032
Study_Hours_Per_Day              0.2561      0.032      7.970      0.000       0.193       0.319
Extracurricular_Hours_Per_Day    0.0943      0.032      2.925      0.003       0.031       0.158
Sleep_Hours_Per_Day             -0.1006      0.285     -0.353      0.724      -0.659       0.458
Sleep_Hours_Squared              0.0280      0.043      0.651      0.515      -0.056       0.112
Sleep_Hours_Cubed               -0.0013      0.002     -0.675      0.500      -0.005       0.002
Social_Hours_Per_Day             0.1031      0.032      3.205      0.001       0.040       0.166
Physical_Activity_Hours_Per_Day  0.1018      0.032      3.169      0.002       0.039       0.165
==============================================================================
Omnibus:                        0.813   Durbin-Watson:                   2.005
```

```
Prob(Omnibus):                  0.666   Jarque-Bera (JB):
0.725
Skew:                           0.025   Prob(JB):
0.696
Kurtosis:                       3.078   Cond. No.
2.38e+17
================================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The smallest eigenvalue is 1.02e-26. This might indicate that
there are
strong multicollinearity problems or that the design matrix is
singular.
```

Analizar nivel de estres

```
if 'Stress_Level_Num' in df.columns:
    print("\nANALISIS DE ESTRES:")

    X_stress = df[['Study_Hours_Per_Day', 'Sleep_Hours_Per_Day',
                   'Social_Hours_Per_Day',
'Physical_Activity_Hours_Per_Day', 'GPA']]
    y_stress = df['Stress_Level_Num']

    model_stress = LinearRegression()
    model_stress.fit(X_stress, y_stress)

    coef_stress = pd.DataFrame({
        'Variable': X_stress.columns,
        'Coeficiente': model_stress.coef_
    }).sort_values('Coeficiente', key=abs, ascending=False)

    print("Impacto en nivel de estres:")
    print(coef_stress)


ANALISIS DE ESTRES:
Impacto en nivel de estres:
                         Variable  Coeficiente
0             Study_Hours_Per_Day    -0.303929
1             Sleep_Hours_Per_Day     0.184197
4                             GPA    -0.134664
2            Social_Hours_Per_Day     0.007316
3  Physical_Activity_Hours_Per_Day    -0.006717
```

Grafico de importancia de variables

```
plt.figure(figsize=(10, 6))
coef_sorted = coeficientes.sort_values('Coeficiente', ascending=True)
plt.barh(coef_sorted['Variable'], coef_sorted['Coeficiente'])
plt.xlabel('Coeficiente (Impacto en GPA)')
plt.title('Variables que mas Afectan el GPA')
plt.tight_layout()
plt.show()
```