```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression

from sklearn.model_selection import train_test_split

df=pd.ExcelFile("estudio_nota.xlsx")

df1 = pd.read_excel(df,'study_logistic_2d')

estudio promedio nota

0 2.0 4.0 1.0

1 5.0 3.0 3.2

2 7.0 4.0 4.5

3 9.0 7.0 6.0

4 10.0 4.0 4.0

df1.head()

estudio 0

promedio 0

nota 2

dtype: int64

df1.isnull().sum()

 df1.dropna(subset=["nota"], inplace=True)

df1

estudio promedio nota

0 2.0 4.0 1.0

1 5.0 3.0 3.2

2 7.0 4.0 4.5

3 9.0 7.0 6.0

4 10.0 4.0 4.0

5 11.0 3.0 4.5

7 14.0 3.0 3.0

8 15.0 6.0 5.0

9 17.0 4.0 5.0
```

```
10 19.0 8.0 6.0
11 22.0 8.0 7.0
13 29.0 9.0 9.1
14 31.0 5.0 5.4
15 33.0 6.0 7.0
16 35.0 9.0 10.0
17 40.0 8.0 10.0
18 42.0 7.0 9.1
from sklearn import preprocessing as pp
X = df1.drop(["nota"], axis=1)
y = df1.nota
 estudio promedio
0 2.0 4.0
1 5.0 3.0
2 7.0 4.0
3 9.0 7.0
4 10.0 4.0
0 1.0
1 3.2
2 4.5
3 6.0
4 4.0
Name: nota, dtype: float64
print(X.head())
print(y.head())
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.20,random_state = 10)
5 4.5
3 6.0
16 10.0
8 5.0
```

```
Name: nota, dtype: float64

y_test.head()


X_train.head()

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

from sklearn.linear_model import LinearRegression

reg = LinearRegression()

reg.fit(X_train, y_train)

y_pred = reg.predict(X_test)

[3.12049296 5.0508561 9.24216616 5.22543963]

print(y_pred)

Prediction accuracy:

 r2 score = 0.82

Mean Squared Error:

 MSE = 0.86

Error absoluto medio:

 MAE = 0.83

from sklearn.linear_model import LinearRegression

from sklearn.preprocessing import RobustScaler

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error

from sklearn.model_selection import train_test_split

print("Prediction accuracy: \n r2 score = {:.2f}".format(r2_score(y_test, y_pred)))

print("Mean Squared Error: \n MSE = {:.2f}".format(mean_squared_error(y_test, y_pred)))

print("Error absoluto medio: \n MAE = {:.2f}".format(mean_absolute_error(y_test,y_pred)))

5 4.5

3 6.0

16 10.0

8 5.0

Name: nota, dtype: float64

print(y_test)

reg.coef_
```

```
array([0.11948624, 0.54233391])
0.17914259232206842
reg.intercept_
<matplotlib.axes._subplots.AxesSubplot at 0x7f8fe70b8c50>
import matplotlib.pyplot as plt
import seaborn as sns
corr = df1.corr()
plt.figure(figsize=(10,10))
sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=corr.columns.values )
import seaborn as sb
import matplotlib.pyplot as pyplot
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
estudio = df1['estudio'].values
promedio = df1['promedio'].values
nota = df1['nota'] values
nota = df1[ nota ].values
fig = pyplot.figure(figsize=(16, 9))
ax = Axes3D(fig)
x1 = estudio
x2 = promedio
z = nota
ax.scatter(x1, x2, z, marker='*', c='b')
ax.set_xlabel('estudio')
ax.set_ylabel('promedio')
ax.set_zlabel('nota');
inference_df = pd.DataFrame({"Actual": y_test,
 "Predictions": y_pred,
"Abs Error": np.abs(y_test-y_pred)})

inference_df


5 4.5 3.120493 1.379507

3 6.0 5.050856 0.949144

16 10.0 9.242166 0.757834

8 5.0 5.225440 0.225440

[8.75492955]

new_dato = np.array([[40,7]])

print(reg.predict(new_dato))

df1.to_csv("examen_teorema2.csv")
```