Regresión Lineal Múltiple

August 20, 2025

1 Regresión Lineal Múltiple Sergio Alejandro Zamora Dávila

Importo el archivo Nasa.csv, el cual contiene las variables relacionadas con el comportamiento atmosférico. Primero reviso el tamaño de la base de datos, los nombres de las variables y una muestra de las primeras filas, con esto identifico las características principales de la base de datos. Después, separo los datos en entrenamiento (70% de los datos totales) y de prueba (30% de los datos totales) para poder ajustar y validar el modelo de regresión múltiple.

[87]:
```
import pandas as pd

# Leemos el csv
df = pd.read_csv("Nasa.csv")
print("Dimensiones del data frame: ")
# Esto imprimira tanto la forma de la base de datos como los nombres de las␣
 ↪columnas
print(df.shape)
print(df.columns)
print("\n")
print("Primeras filas de datos: ")
print(df.head(15))
print("\n")
print("\n")

# Se definen las muestras a utilizar para entrenar y para testear
train = df.sample(frac = 0.7)
test = df.drop(train.index)
# Se imprimen dichas cantidades
print("Cantidad de datos a usar para Train:", train.shape)
print("Cantidad de datos a usar para Test:",test.shape)
print("La cantidad de datos que se usaran para Entrenar (1052) más la cantidad␣
 ↪de datos que se usaran para Testear (451) da el total de datos (1503)")

print("\n")
print("Ejemplo de los datos seleccionados para entrenar: ")
print(train.head())
```

```
Dimensiones del data frame:
(1503, 6)
```

```
Index(['frecuencia', 'angulo', 'longitud', 'velocidad', 'espesor', 'presion'],
dtype='object')
```

```
Primeras filas de datos:
    frecuencia  angulo  longitud  velocidad   espesor  presion
0          800     0.0    0.3048       71.3  0.002663  126.201
1         1000     0.0    0.3048       71.3  0.002663  125.201
2         1250     0.0    0.3048       71.3  0.002663  125.951
3         1600     0.0    0.3048       71.3  0.002663  127.591
4         2000     0.0    0.3048       71.3  0.002663  127.461
5         2500     0.0    0.3048       71.3  0.002663  125.571
6         3150     0.0    0.3048       71.3  0.002663  125.201
7         4000     0.0    0.3048       71.3  0.002663  123.061
8         5000     0.0    0.3048       71.3  0.002663  121.301
9         6300     0.0    0.3048       71.3  0.002663  119.541
10        8000     0.0    0.3048       71.3  0.002663  117.151
11       10000     0.0    0.3048       71.3  0.002663  115.391
12       12500     0.0    0.3048       71.3  0.002663  112.241
13       16000     0.0    0.3048       71.3  0.002663  108.721
14         500     0.0    0.3048       55.5  0.002831  126.416
```

```
Cantidad de datos a usar para Train: (1052, 6)
Cantidad de datos a usar para Test: (451, 6)
La cantidad de datos que se usaran para Entrenar (1052) más la cantidad de datos
que se usaran para Testear (451) da el total de datos (1503)
```

```
Ejemplo de los datos seleccionados para entrenar:
      frecuencia  angulo  longitud  velocidad   espesor  presion
1277        5000     0.0    0.1016       39.6  0.001463  126.021
345          250     4.0    0.2286       31.7  0.005091  120.189
1420        2500    12.3    0.1016       71.3  0.033779  121.318
1479         800    15.6    0.1016       71.3  0.043726  124.188
226         4000     0.0    0.2286       39.6  0.002535  123.465
```

La variable de interés, que trataremos de predecir es la presión. Lo que significa que trataremos de predecir el aerodinamismo, medido como la presión sonora detectada. Las demás variables se utilizan como predictoras. Ajusto un modelo de regresión lineal múltiple utilizando la librería statsmodels.api, y reviso un resumen completo con los coeficientes, sus errores estándar, t-statistics y p-values.

[88]:
```
import statsmodels.api as sm
# Se "Dropea" Presion, ya que este dato sera el que vamos a predecir
X = train.drop('presion', axis = 1)
```

```
Y = train.presion
model = sm.OLS(Y, sm.add_constant(X))
results = model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                presion   R-squared:                       0.506
Model:                            OLS   Adj. R-squared:                  0.503
Method:                 Least Squares   F-statistic:                     213.9
Date:                Wed, 20 Aug 2025   Prob (F-statistic):          3.60e-157
Time:                        20:10:41   Log-Likelihood:                -3138.5
No. Observations:                1052   AIC:                             6289.
Df Residuals:                    1046   BIC:                             6319.
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        132.5916      0.642    206.684      0.000     131.333     133.850
frecuencia    -0.0013   5.07e-05    -25.861      0.000      -0.001      -0.001
angulo        -0.4274      0.047     -9.101      0.000      -0.520      -0.335
longitud     -33.7863      1.931    -17.500      0.000     -37.575     -29.998
velocidad      0.1003      0.010     10.373      0.000       0.081       0.119
espesor     -134.3779     18.084     -7.431      0.000    -169.863     -98.893
==============================================================================
Omnibus:                        5.614   Durbin-Watson:                   2.062
Prob(Omnibus):                  0.060   Jarque-Bera (JB):                7.088
Skew:                          -0.008   Prob(JB):                       0.0289
Kurtosis:                       3.402   Cond. No.                     5.18e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 5.18e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

En este caso, en el resumen los p-values aparecen como 0.000, lo cual significa que son muy pequeños. Para poder observar su valor real con mayor precisión, utilizo el atributo .pvalues. Lo que permite confirmar que las variables del modelo son estadísticamente significativas en la predicción de la presión.

[89]:
```
# P-values individuales de cada variable
print("P-values de cada variable:")
print(results.pvalues)

significativas = results.pvalues[results.pvalues < 0.05]
```

```
print("\n")
print("Variables P value menor a 0.05:")
print(significativas)
```

```
P-values de cada variable:
const          0.000000e+00
frecuencia    2.067638e-114
angulo         4.429749e-19
longitud       2.434519e-60
velocidad      4.595152e-24
espesor        2.238851e-13
dtype: float64


Variables P value menor a 0.05:
const          0.000000e+00
frecuencia    2.067638e-114
angulo         4.429749e-19
longitud       2.434519e-60
velocidad      4.595152e-24
espesor        2.238851e-13
dtype: float64
```

[90]:
```
print("Valores absolutos en el estadistico de t: ")
print(results.tvalues.abs())
```

```
Valores absolutos en el estadistico de t:
const         206.684279
frecuencia     25.861435
angulo          9.100605
longitud       17.499832
velocidad      10.373316
espesor         7.430802
dtype: float64
```

La variable más influyente para la presión es frecuencia, ya que presenta el mayor valor absoluto en el estadístico t, lo que indica que su asociación lineal con la presión es más fuerte a comparación de las demás variables del modelo.

Ahora calculo el Residual Standard Error (RSE) y el R^2 tanto para los datos de entrenamiento como los de validación. Para entrenamiento, uso directamente los atributos results.scale**0.5 y results.rsquared. Para validación, realizo las predicciones sobre el conjunto de prueba y calculo manualmente ambas métricas.

[91]:
```
# Datos de entrenamiento
print("Datos de entrenamiento: \n")

# 'scale' guarda el MSE (Mean Squared Error) del modelo
# El Residual Standard Error (RSE) se calcula con la raíz cuadrada
```

```
RSE_train = results.scale**0.5

# 'rsquared' devuelve directamente el R² del modelo en entrenamiento
# R² indica la proporción de variabilidad de Y explicada por las variables␣
 ↪predictoras
R2_train = results.rsquared

print("RSE (train) =", RSE_train)
print("R^2 (train) =", R2_train)

print("\nDatos de validacion: \n")
# Se hace un drop a presion, ya que esta variable era la que vamos a predecir
XTest = test.drop('presion', axis = 1)
YTest = test.presion
# Se predicen los valores de Y
yhatNew = results.predict(sm.add_constant(XTest))

# mide la suma de los errores al cuadrado en validación
RSS_val = sum((YTest - yhatNew)**2)

# mide la variabilidad total de Y respecto a su media
TSS_val = sum((YTest - np.mean(YTest))**2)

# Se obtiene el número de observaciones (nTest) y el número de predictores␣
 ↪(mTest)
nTest = XTest.shape[0]
mTest = XTest.shape[1]

# Se calcula el error estándar residual en validación
RSETest = np.sqrt(RSS_val/(nTest-mTest-1))
# Se calcula la proporción de variabilidad explicada por el modelo
R2Test = 1 - RSS_val/TSS_val

print("RSE (test) =", RSETest)
print("R^2 (test) =", R2Test)
```

```
Datos de entrenamiento:

RSE (train) = 4.793813626603953
R^2 (train) = 0.5055110009624317

Datos de validacion:

RSE (test) = 4.89790851461986
R^2 (test) = 0.5310380536123369
```

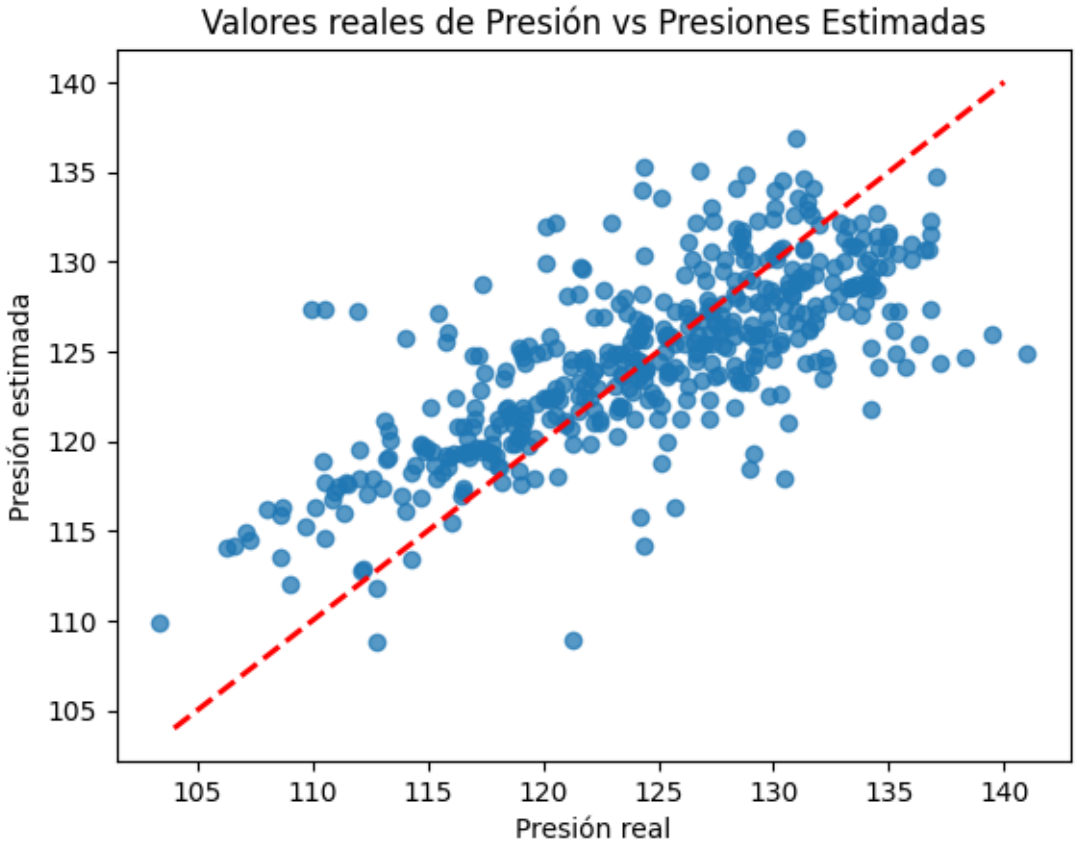
Por último, genero una gráfica de dispersión con los valores reales de presión en el eje X y los valores estimados por el modelo en el eje Y. Si el modelo fuera perfecto, todos los puntos deberían caer

sobre la diagonal de 45°. Mientras más cerca estén los puntos de esa línea, mejor será el desempeño del modelo.

[92]:
```
%matplotlib inline
import matplotlib.pyplot as plt

x_line = np.linspace(104, 140)
y_line = x_line

plt.scatter(YTest, yhatNew, alpha=0.75)
plt.plot(x_line,y_line, 'r--', linewidth=2)
plt.xlabel("Presión real")
plt.ylabel("Presión estimada")
plt.title("Valores reales de Presión vs Presiones Estimadas")
plt.show()
```



1.0.1 Conclusión

El modelo de regresión múltiple tiene valores cercanos a los reales en la presión. Los valores de R^2 obtenidos fueron de 0.50 en entrenamiento y 0.53 en los datos de validación, lo que indica que

el modelo se comporta de manera similar a lo esperado.

La gráfica de dispersión muestra que las predicciones se aproximan a la linea diagonal de 45 grados. Se puede deducir que el modelo es útil para aproximar valores de la presión, pero cuenta con variabilidad.