Informe de Regresión Lineal Múltiple en Python

Tu Nombre

30 de marzo de 2025

1. Introducción

La **Regresión Lineal** es un algoritmo de *Machine Learning* supervisado utilizado para modelar la relación entre una variable dependiente (objetivo) y una o más variables independientes (predictoras). Su objetivo es encontrar una función lineal que mejor se ajuste a los datos, permitiendo predecir valores futuros.

En este ejercicio, se extiende el modelo de **Regresión Lineal Simple** a **Regresión Lineal Múltiple**, incorporando dos variables predictoras:

- **Word count** (cantidad de palabras)
- **Suma de enlaces, comentarios e imágenes** (como segunda variable)

La ecuación general para n variables predictoras es:

$$Y = b + m_1X_1 + m_2X_2 + \cdots + m_nX_n \tag{1}$$

donde:

- Y: Variable objetivo (# Shares)
- b: Término de intersección (bias)
- m_i: Coeficientes de las variables predictoras X_i

2. Metodología

2.1. Preparación de Datos

Se utilizó un conjunto de datos que incluye:

- **Variables predictoras**:
  - `Word count` (palabras)
  - Suma de `# of Links`, `# of comments` (rellenados con 0 si eran NaN) y `# Images video`

- **Variable objetivo**: `# Shares` (compartidas en redes sociales)

```
import pandas as pd
import numpy as np
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score

# Creaci n de la segunda variable predictiva
suma = filtered_data["# of Links"] + filtered_data['# of comments'
    ].fillna(0) + filtered_data['# Images video']

# DataFrame con las 2 variables predictoras
dataX2 = pd.DataFrame()
dataX2["Word count"] = filtered_data["Word count"]
dataX2["suma"] = suma

# Conversi n a arrays para el modelo
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

Listing 1: Preparación de datos

2.2. Entrenamiento del Modelo

Se aplicó la regresión lineal múltiple utilizando `sklearn.linear_model.LinearRegression()`.

```
# Creaci n y entrenamiento del modelo
regr2 = linear_model.LinearRegression()
regr2.fit(XY_train, z_train)

# Predicciones
z_pred = regr2.predict(XY_train)

# M tricas
print('Coefficients: \n', regr2.coef_)
print("Mean squared error: %.2f" % mean_squared_error(z_train,
    z_pred))
print('Variance score: %.2f' % r2_score(z_train, z_pred))
```

Listing 2: Entrenamiento del modelo

3. Resultados

3.1. Coeficientes del Modelo

- **Coeficientes**:
  - `Word count`: 6,63 (impacto positivo en las comparticiones)
  - `suma`: $-483{,}41$ (impacto negativo, posiblemente por ruido en los datos)

3.2. Métricas de Evaluación

- **Error Cuadrático Medio (MSE)**: 352 122 816,48 (alto, indica grandes discrepancias)
- **Puntaje de Varianza (R^2)**: 0,11 (muy bajo; el modelo explica solo el 11 % de la variabilidad)

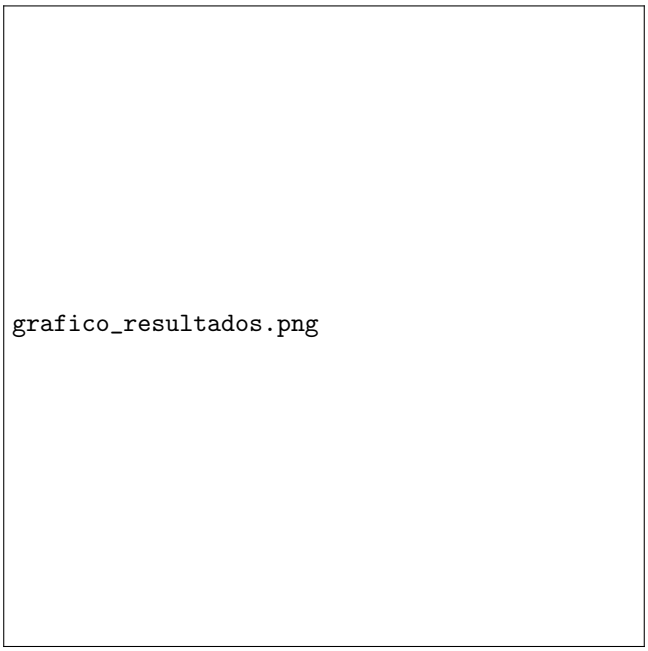

Figura 1: Relación entre variables predictoras y objetivo

Interpretación:

- El modelo tiene un **ajuste pobre** debido a la baja correlación entre las variables predictoras y el target (ver Figura 1)
- La segunda variable (`suma`) podría introducir ruido o no ser relevante para predecir `# Shares`

4. Conclusión

1. **Limitaciones del Modelo**:
    - El bajo R^2 (0,11) sugiere que las variables predictoras no explican adecuadamente las variaciones

- La alta magnitud del MSE (352 122 816,48) indica que las predicciones son inexactas

2. **Recomendaciones**:
   - **Reducción de dimensionalidad**: Aplicar técnicas como **PCA** para extraer características más significativas
   - **Ingeniería de características**: Probar otras combinaciones de variables o transformaciones (ej.: logarítmicas)
   - **Modelos alternativos**: Evaluar algoritmos no lineales (ej.: Random Forest, Gradient Boosting) si la relación no es lineal
3. **Lección aprendida**: La regresión lineal múltiple es útil para explorar relaciones multivariadas, pero su efectividad depende críticamente de la **calidad y relevancia** de los datos de entrada, como se demostró en este análisis.