**Universidad De Cundinamarca**

# TALLER REGRESION LINEAL Y POLINOMIAL

**Juan David Echeverria García**

**Lukas David Davila Alzate**

**Profundización Disciplinar Ciencia De Datos**

**Arley Alexander Rodriguez Pascuas**

**Séptimo Semestre**

**Universidad De Cundinamarca**

**Septiembre, 2025**

**Ejercicio 1 – Regresión Lineal Simple**

Contexto: Una empresa quiere predecir las ventas en función de la inversión en publicidad.

**Actividades:** Graficar, ajustar el modelo con LinearRegression, obtener coeficientes, predecir ventas con 7 miles USD.

```
# importar librerias
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

# leer valores
data = {
   "publicidad": [1,2,3,4,5,6],
   "ventas": [3,4.5,6,7.5,9,10]
}
df = pd.DataFrame(data)

# variables dependientes e independientes
x = df[["publicidad"]]
y = df["ventas"]

# dividir los datos para entrenar y de prueba
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.5,
random_state=42)

# 1. Ajustar modelo con LinearRegression para entrenar
model = LinearRegression()
model.fit(x_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
# 2. graficar
plt.scatter(x, y, color='blue', label='Datos Reales')
plt.plot(x, model.predict(x), color="red", label="Regresión Lineal")
plt.xlabel("Publicidad (miles de USD)")
```

```
plt.ylabel("Ventas (miles de USD)")
plt.title("Ventas segun la publicidad")
plt.legend()
plt.show()
```


```
# 3. obtener el coeficiente
R2 = model.score(x, y)
print(f"Coeficiente de determinacion : {R2:.3f}")
```

```
Coeficiente de determinacion : 0.993
```

```
# 4. predecir ventas con 7 miles USD
nuevo = pd.DataFrame({
    "publicidad":[7]
})
prediccion = model.predict(nuevo)
print(f"La prediccion con 7 miles USD es {prediccion} miles USD")
```

```
La prediccion con 7 miles USD es [12.] miles USD
```

**Ejercicio 2 – Regresión Lineal Múltiple**

Contexto: Una empresa quiere predecir las ventas con base en publicidad digital y televisión.

**Actividades:** Ajustar el modelo múltiple, interpretar coeficientes, calcular $R^2$ y predecir ventas para-Digital=6 y TV=10.

```
# importar librerias necesarias
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
# leer datos
data = {
   "pub_digital": [2,3,4,5,6,7],
   "pub_TV": [4,5,6,7,8,9],
   "ventas": [8,10,12,13,15,16]
}
df = pd.DataFrame(data)
# dividir en datos de prueba y entrenamiento
x = df[["pub_digital", "pub_TV"]]
y = df["ventas"]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.4,
random_state=42)
# 1. ajustar modelo con regresion
model = LinearRegression()
model.fit(x_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
# 2. interpretar coeficientes
coef = model.coef_
print("Coeficientes:", coef)
```

```
Coeficientes: [0.75 0.75]
```

```
# 3. calcular R2
R2 = model.score(x_test, y_test)
print(f"Coeficiente de determinacion : {R2:.3f}")
```

```
Coeficiente de determinacion : 0.974
```

```
# 4. predecir ventas para digital = 5 y TV = 10
nuevo = pd.DataFrame({
    "pub_digital": [7],
    "pub_TV": [10]
})
prediccion = model.predict(nuevo)
print(f"La prediccion con digital: 7 y Tv: 10 es de: {prediccion[0]:.2f} mil USD")
```

```
La prediccion con digital: 7 y Tv: 10 es de: 17.08 mil USD
```

**Ejercicio 3 – Regresión Polinomial**

Contexto: El crecimiento de usuarios de una aplicación no sigue una tendencia lineal.

**Actividades:** Ajustar un modelo lineal y polinomial de grado 2, comparar resultados, predecir usuarios en el mes 7.

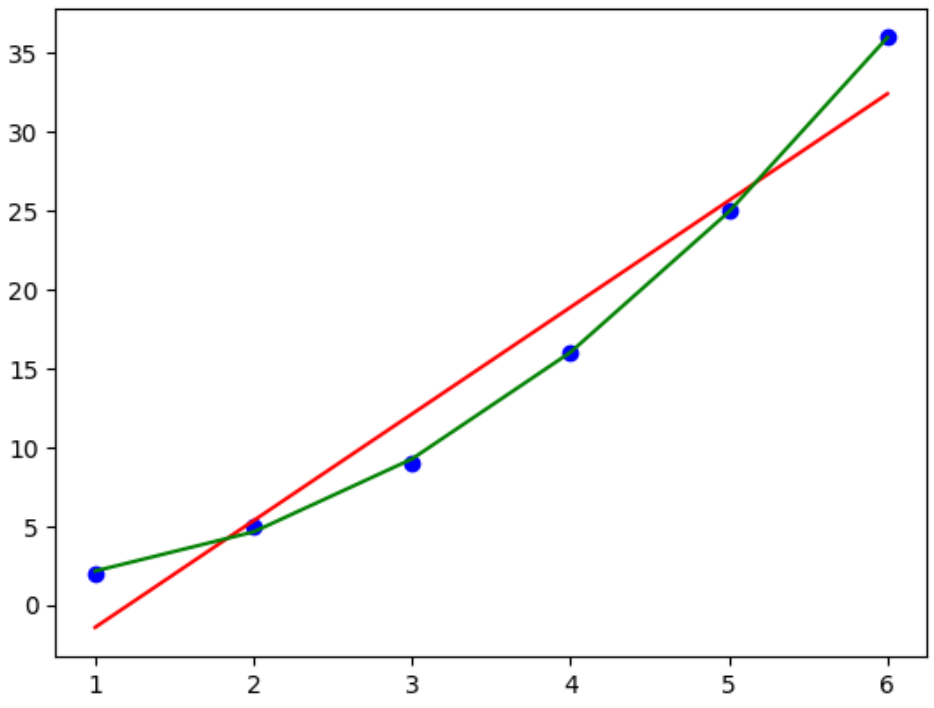
```
# importar libresias necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

# leer datos
data = {
    "tiempo": [1,2,3,4,5,6],
    "usuarios": [2,5,9,16,25,36]
}
df = pd.DataFrame(data)

# 1.1 ajustar modelo lineal
x = df[["tiempo"]]
y = df["usuarios"]
model = LinearRegression()
model.fit(x, y)
```

```
LinearRegression
LinearRegression()
```

```
# 1.2 ajustar modelo polinomial
poli = PolynomialFeatures(degree=2)
x_poli = poli.fit_transform(x)
model_poli = LinearRegression()
model_poli.fit(x_poli, y)
```

```
LinearRegression
LinearRegression()
```

```
# 2.1 comparar resultados (grafico)
plt.scatter(x, y, color='blue', label='Datos Reales')
plt.plot(x, model.predict(x), color="red", label="Regresión Lineal")
plt.plot(x, model_poli.predict(x_poli), color="green", label="Regresión Polinomial")
```




```
# 2.2 compara resultados
r2_lineal = model.score(x, y)
r2_poli = model_poli.score(x_poli, y)
print(f"Lineal: {r2_lineal}")
```

```
print(f"Polinomial: {r2_poli}")
```

```
Lineal: 0.9490411421812959
Polinomial: 0.999729661231731
```

```
# 3. predecir usuarios en el mes 7
nuevo = pd.DataFrame({
   "tiempo": [7]
})
pred_lineal = model.predict(nuevo)
print(f"La prediccion lineal en el mes 7 es de {pred_lineal[0]:.3f}")
x_poli = poli.fit_transform(nuevo)
pred_poli = model_poli.predict(x_poli)
print(f"La prediccion polinomial en el mes 7 es de {pred_poli[0]:.3f}")
```

```
La prediccion lineal en el mes 7 es de 39.200
La prediccion polinomial en el mes 7 es de 49.200
```

**Ejercicio 4 – Regresión Lineal Simple**
Contexto: Una universidad quiere predecir el rendimiento académico según horas de estudio.

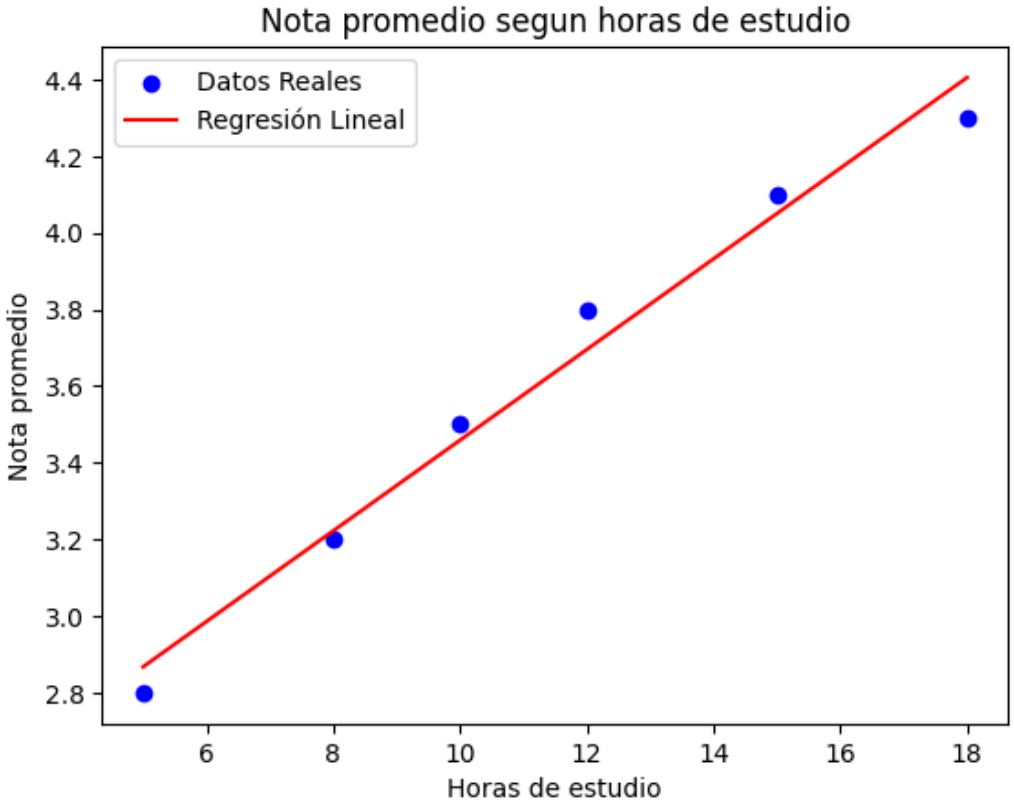
**Actividades:** Graficar, entrenar modelo, evaluar $R^2$, predecir nota promedio para 14 horas.

```
# importar librerias necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# leer datos
data = {
   "horas_est": [5,8,10,12,15,18],
   "nota_prom": [2.8,3.2,3.5,3.8,4.1,4.3]
}
df = pd.DataFrame(data)

# 1. entrenar modelo
x = df[["horas_est"]]
y = df["nota_prom"]
model = LinearRegression()
```

```
model.fit(x, y)
```

```
LinearRegression
LinearRegression()
```

```
#2. graficar
plt.scatter(x, y, color='blue', label='Datos Reales')
plt.plot(x, model.predict(x), color="red", label="Regresión Lineal")
plt.xlabel("Horas de estudio")
plt.ylabel("Nota promedio")
plt.title("Nota promedio segun horas de estudio")
plt.legend()
plt.show()
```




```
# 3. evaluar R2
R2 = model.score(x, y)
print(f"Coeficiente de determinacion : {R2:.3f}")
```

```
Coeficiente de determinacion : 0.980
```

```
# 4. predecir nota para 14 horas de estudio
nuevo = pd.DataFrame({
    "horas_est": [14]
})
prediccion = model.predict(nuevo)
print(f"La prediccion con 14 horas de estudio es {prediccion[0]:.2f}")
```

```
La prediccion con 14 horas de estudio es 3.93
```

**Ejercicio 5 – Regresión Lineal Múltiple**

Contexto: Una constructora quiere predecir el costo de un proyecto considerando trabajadores y semanas.

**Actividades:** Ajustar modelo múltiple, interpretar coeficientes, evaluar $R^2$ y MSE, predecir costo para 14 trabajadores y 12 semanas.

```
# importar librerias necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# leer datos
data = {
    "trabajadores": [5,8,10,12,15,18],
    "semanas": [10,8,12,9,11,10],
    "costo": [50,60,85,95,120,135]
}
df = pd.DataFrame(data)

# 1. ajustar modelo multiple
x = df[["trabajadores", "semanas"]]
y = df["costo"]
model = LinearRegression()
model.fit(x, y)
```

```
LinearRegression
LinearRegression()
```

```
# 2. interpretar coeficientes
coeficientes = model.coef_
print(f"Los coeficientes son {coeficientes[0]:.4f} y {coeficientes[1]:.4f}")
```

```
Los coeficientes son 6.7726 y 2.7592
```

```
# 3. evaluar R2
r2 = model.score(x,y)
print(f"Coeficiente de determinacion : {r2:.3f}")
```

```
Coeficiente de determinacion : 0.995
```

```
# 4. evaluar MSE (error cuadratico medio)
y_true = y
y_pred = model.predict(x)
mse = mean_squared_error(y_true, y_pred)
print(f"El error cuadrático medio es {mse}")
```

```
El error cuadratico medio es 4.399598079131439
```

```
# 5. predecir el costo para 14 trabajadores y 12 semanas
nuevo = pd.DataFrame({
    "trabajadores": [14],
    "semanas": [12]
})
prediccion = model.predict(nuevo)
print(f"La prediccion con 14 trabajadores y 12 semanas es {prediccion[0]:.2f} mil
USD")
```

```
La prediccion con 14 trabajadores y 12 semanas es 114.41 mil USD
```

### Ejercicio 6 – Comparación de Modelos (Lineal vs Polinomial)

Contexto: El crecimiento de la población de una ciudad no es perfectamente lineal. Se registró el número de habitantes (en miles) cada 2 años.

Actividades: Ajustar regresión lineal, polinomial de grado 2 y 3, comparar con $R^2$, graficar y predecir población en 2012.

[2]
✓ 5 s

```python
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.preprocessing import PolynomialFeatures
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.model_selection import train_test_split
```

EJERCICIO 6 – Comparación de Modelos (Lineal vs Polinomial)

```python
print("="*50)
print("EJERCICIO 6: COMPARACIÓN DE MODELOS (POBLACIÓN)")
print("="*50)

# 1. Crear el DataFrame de datos
data6 = {
    'Año': [2000, 2002, 2004, 2006, 2008, 2010],
    'Población (miles)': [120, 130, 150, 180, 220, 270]
}
df6 = pd.DataFrame(data6)
```

```
==================================================
EJERCICIO 6: COMPARACIÓN DE MODELOS (POBLACIÓN)
==================================================
```

```
# Normalizar el Año para usar Años desde el 2000 (mejora la estabilidad polinomial)
df6['Año_Norm'] = df6['Año'] - 2000
X6 = df6[['Año_Norm']] # DataFrame de features (necesario para evitar warnings)
y6 = df6['Población (miles)']

modelos_r2 = {}
modelos = {}
```

```
# --- A. Regresión Lineal (Grado 1) ---
model6_lin = LinearRegression().fit(X6, y6)
modelos_r2['Lineal (Grado 1)'] = r2_score(y6, model6_lin.predict(X6))
model = LinearRegression()
```

```
# --- B. Regresión Polinomial Grado 2 ---
poly_features2 = PolynomialFeatures(degree=2)
X6_poly2 = poly_features2.fit_transform(X6)
model6_poly2 = LinearRegression().fit(X6_poly2, y6)
modelos_r2['Polinomial (Grado 2)'] = r2_score(y6, model6_poly2.predict(X6_poly2))
```

```
# --- C. Regresión Polinomial Grado 3 ---
poly_features3 = PolynomialFeatures(degree=3)
X6_poly3 = poly_features3.fit_transform(X6)
model6_poly3 = LinearRegression().fit(X6_poly3, y6)
modelos_r2['Polinomial (Grado 3)'] = r2_score(y6, model6_poly3.predict(X6_poly3))
```

[12]
0 s

```
# 2. Comparar R²
print("\nComparación de Modelos por R²:")
r2_df = pd.Series(modelos_r2).sort_values(ascending=False).to_frame('R² Score')
print(r2_df)

mejor_modelo = r2_df.index[0]
print(f"\nEl modelo con mejor ajuste (mayor R²) es: {mejor_modelo}")
```

```
Comparación de Modelos por R²:
                       R² Score
Polinomial (Grado 2)   1.000000
Polinomial (Grado 3)   1.000000
Lineal (Grado 1)       0.944056

El modelo con mejor ajuste (mayor R²) es: Polinomial (Grado 2)
```

[17] ✓ 0 s

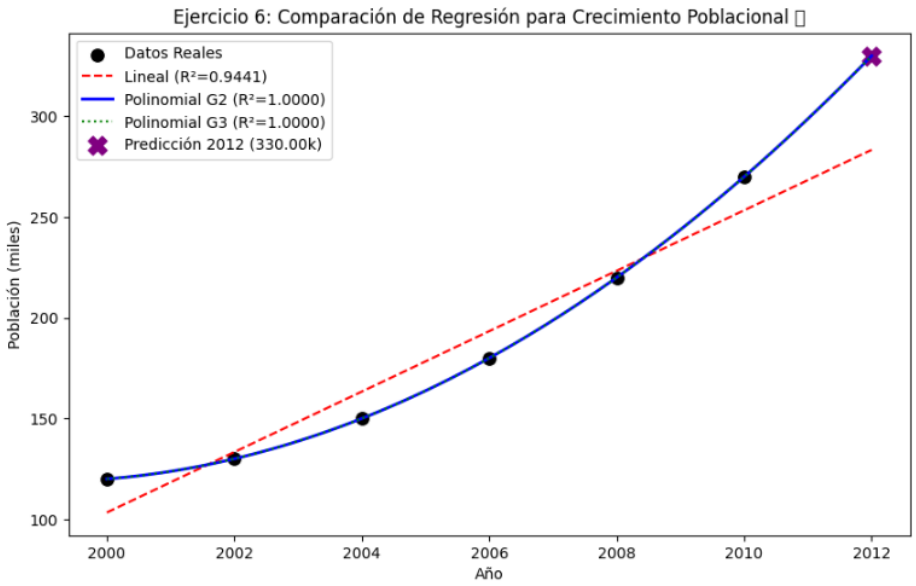
```
# 3. Predecir población en 2012 (Año_Norm = 12)
año_prediccion_val = 12
# CREACIÓN DEL DATAFRAME para la predicción (evita warnings y mantiene la estructura)
año_prediccion_df = pd.DataFrame({'Año_Norm': [año_prediccion_val]})
```

[18] ✓ 0 s

```
# Usamos el modelo Polinomial Grado 2/3 (el mejor R² o el más simple de los mejores)
X_pred_poly2 = poly_features2.transform(año_prediccion_df)
poblacion_predicha = model6_poly2.predict(X_pred_poly2)

print(f"\nPredicción (usando Polinomial Grado 2) para el año 2012: {poblacion_predicha
```

```
Predicción (usando Polinomial Grado 2) para el año 2012: 330.00 miles de habitantes
```

```
# 4. Graficar
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Año', y='Población (miles)', data=df6, color='black', s=100, label='Datos Reales')

# Puntos para la curva suave (extender hasta 2012 para la predicción)
x_range_norm = np.linspace(df6['Año_Norm'].min(), año_prediccion_val, 100).reshape(-1, 1)
x_range_df = pd.DataFrame(x_range_norm, columns=['Año_Norm'])
x_range_year = x_range_df['Año_Norm'].values + 2000 # Convertir a Año Real

# Predicciones para la curva
y_curve_lin = model6_lin.predict(x_range_df)
y_curve_poly2 = model6_poly2.predict(poly_features2.transform(x_range_df))
y_curve_poly3 = model6_poly3.predict(poly_features3.transform(x_range_df))

plt.plot(x_range_year, y_curve_lin, color='red', linestyle='--', label=f'Lineal (R²={modelos_r2["Lineal (Grado 1)"]:.4f})')
plt.plot(x_range_year, y_curve_poly2, color='blue', linewidth=2, label=f'Polinomial G2 (R²={modelos_r2["Polinomial (Grado 2)"]:.4f})')
plt.plot(x_range_year, y_curve_poly3, color='green', linewidth=1.5, linestyle=':',
         label=f'Polinomial G3 (R²={modelos_r2["Polinomial (Grado 3)"]:.4f})')

# Marcar la predicción
plt.scatter(2012, poblacion_predicha, color='purple', s=150, marker='X', label=f'Predicción 2012 ({poblacion_predicha[0]:.2f}k)')

plt.title('Ejercicio 6: Comparación de Regresión para Crecimiento Poblacional ')
plt.xlabel('Año')
plt.ylabel('Población (miles)')
plt.legend()
plt.show()
```




## Ejercicio 7 – Regresión Múltiple con Variables Sintéticas

Contexto: Una empresa de transporte quiere predecir el consumo de combustible de sus buses considerando la distancia recorrida y la cantidad de pasajeros.

Actividades: Ajustar modelo de regresión múltiple, interpretar coeficientes, calcular $R^2$ Y predecir consumo para 28 km y 32 pasajeros.

```
print("\n" + "="*50)
print("EJERCICIO 7: REGRESIÓN MÚLTIPLE (CONSUMO COMBUSTIBLE)")
print("="*50)

# 1. Crear el DataFrame de datos
data7 = {
    'Distancia (km)': [10, 15, 20, 25, 30, 35],
    'Pasajeros': [20, 30, 25, 35, 40, 45],
    'Consumo Combustible (litros)': [15, 25, 28, 40, 48, 55]
}
df7 = pd.DataFrame(data7)
print("Datos de Consumo:\n", df7)
```

```
==================================================
EJERCICIO 7: REGRESIÓN MÚLTIPLE (CONSUMO COMBUSTIBLE)
==================================================
Datos de Consumo:
    Distancia (km)  Pasajeros  Consumo Combustible (litros)
0              10         20                            15
1              15         30                            25
2              20         25                            28
3              25         35                            40
4              30         40                            48
5              35         45                            55
```

```python
# 2. Definir Variables
X7 = df7[['Distancia (km)', 'Pasajeros']]
y7 = df7['Consumo Combustible (litros)']

# 3. Ajustar Modelo
model7 = LinearRegression().fit(X7, y7)

# 4. Interpretar Coeficientes y Ecuación
beta0_7 = model7.intercept_
beta_coefs_7 = model7.coef_

print(f"\nEcuación del modelo:")
print(f"Consumo = {beta0_7:.2f} + {beta_coefs_7[0]:.2f} * Distancia + {beta_coefs_7[1]:.2f} * Pasajeros")

print("\nInterpretación de Coeficientes:")
print(f" - **Coef. Distancia (β1 = {beta_coefs_7[0]:.2f}):** Por cada km adicional, el Consumo aumenta en **{beta_coefs_7[0]:.2f} litros** (Pasajeros constantes).")
print(f" - **Coef. Pasajeros (β2 = {beta_coefs_7[1]:.2f}):** Por cada pasajero adicional, el Consumo aumenta en **{beta_coefs_7[1]:.2f} litros** (Distancia constante).")
```

```
Ecuación del modelo:
Consumo = -6.80 + 1.12 * Distancia + 0.52 * Pasajeros

Interpretación de Coeficientes:
 - **Coef. Distancia (β1 = 1.12):** Por cada km adicional, el Consumo aumenta en **1.12 litros** (Pasajeros constantes).
 - **Coef. Pasajeros (β2 = 0.52):** Por cada pasajero adicional, el Consumo aumenta en **0.52 litros** (Distancia constante).
```

```python
# 5. Calcular R²
r_squared_7 = r2_score(y7, model7.predict(X7))

print(f"\nCoeficiente de Determinación (R²): {r_squared_7:.4f}")
print(f"El {r_squared_7 * 100:.2f}% de la variación en el consumo es explicada por la Distancia y los Pasajeros.")

# 6. Predecir consumo para 28 km y 32 pasajeros
prediccion_7_df = pd.DataFrame({'Distancia (km)': [28], 'Pasajeros': [32]}) # Usamos DataFrame
consumo_predicho = model7.predict(prediccion_7_df)

print(f"\nPredicción para 28 km y 32 pasajeros: {consumo_predicho[0]:.2f} litros")
```

```
Coeficiente de Determinación (R²): 0.9984
El 99.84% de la variación en el consumo es explicada por la Distancia y los Pasajeros.

Predicción para 28 km y 32 pasajeros: 41.05 litros
```

**Finalmente responde en el notebook:**

- Diferencias entre regresión simple, múltiple y polinomial.

**1. Diferencias entre Regresión Simple, Múltiple y Polinomial**

| Tipo de Regresión | Ecuación General (Forma) | Variables Independientes ($X$) | Tipo de Relación Modelada |
|---|---|---|---|
| **Simple** | $Y = \beta_0 + \beta_1 X_1$ | **Una** sola variable predictora ( $X_1$). | Estrictamente **lineal** entre $Y$ y $X_1$. |
| **Múltiple** | $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots$ | **Dos o más** variables predictoras ($X_1, X_2, \dots$). | **Lineal** con respecto a cada variable (plano o hiperplano). |
| **Polinomial** | $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \dots$ | **Una** variable ($X_1$) con términos elevados a potencias. | **No lineal** o **curvilínea**. |

- ¿Qué ventajas y limitaciones tiene cada una?

```
### 2. Ventajas y Limitaciones

#### Regresión Simple:
* **Ventaja:** Fácil de interpretar.
* **Limitación:** Rara vez modela fenómenos complejos.

#### Regresión Múltiple:
* **Ventaja:** Mayor precisión predictiva al incluir múltiples factores.
* **Limitación:** Susceptible a la **multicolinealidad** (correlación entre $X$'s) y requiere más datos.

#### Regresión Polinomial:
* **Ventaja:** Flexibilidad para capturar relaciones curvas.
* **Limitación:** Alto riesgo de **sobreajuste** (*overfitting*) con grados altos y coeficientes difíciles de interpretar.

***
```

- ¿En qué situaciones reales se aplicarían?

```
### 3. Aplicaciones en Situaciones Reales

* **Regresión Simple:** Predecir el **salario** de un empleado basándose *únicamente* en sus **años de experiencia**.
* **Regresión Múltiple:** Predecir el **precio de una vivienda** basándose en su **área**, el **número de habitaciones** y la **antigüedad**.
* **Regresión Polinomial:** Modelar la **curva de crecimiento** de las ventas de un producto tecnológico o de la población a largo plazo.
```