EJEMPLO MÉTRICAS DE REGRESIÓN

In [51]:
```
import pandas as pd
import os
import numpy as np
import sklearn.metrics
import matplotlib.pyplot as plt
import scipy.stats as stats

#Importing the dataset
data = pd.read_csv("Advertising.csv")
data.head()
```

Out[51]:

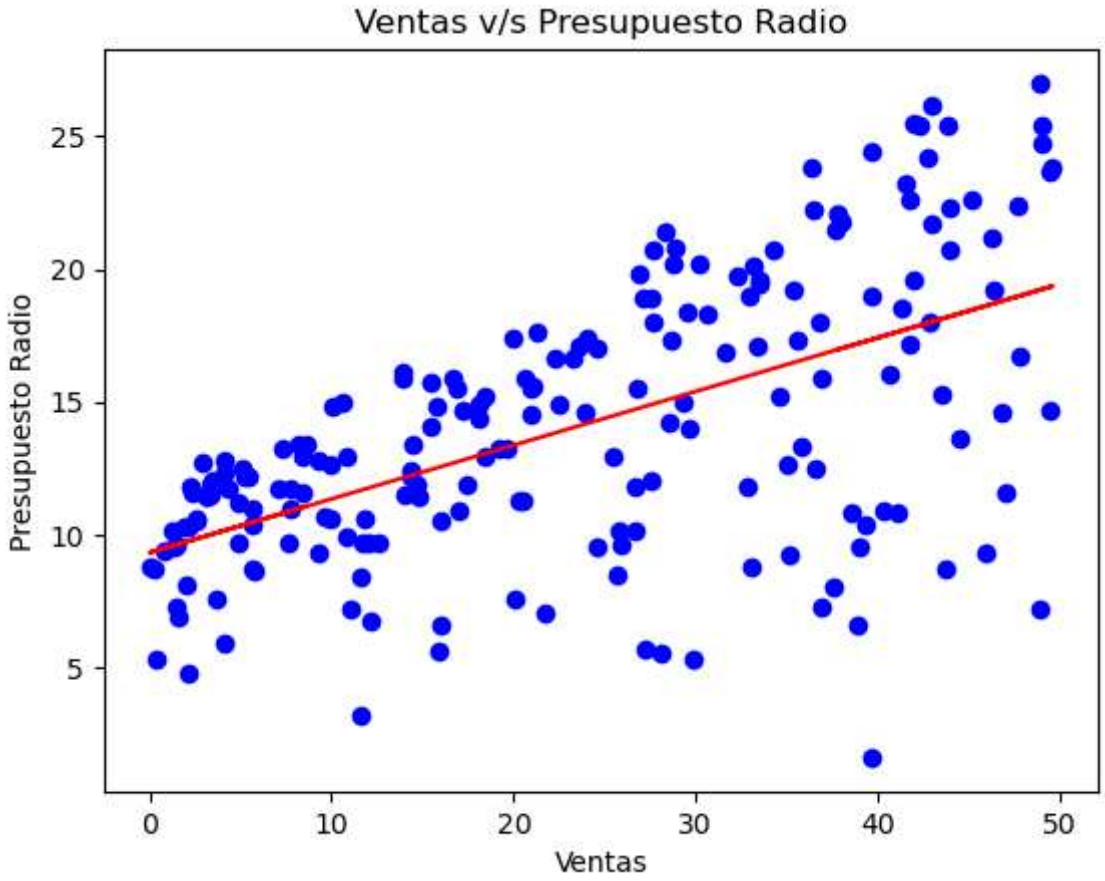
	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

In [52]:
```
x = data.iloc[:,2].values #Indexación basada en ubicaciones enteras para selección la
y = data.iloc[:,4].values
```

In [53]:
```
y=y.reshape(-1, 1) #cambiar la forma de una matriz. Vector columna.
x=x.reshape(-1, 1)
print(x.shape, y.shape)
```

```
(200, 1) (200, 1)
```

In [54]:
```
# Importar LinearRegression.

from sklearn.linear_model import LinearRegression

# modelo de regresión lineal vacío
radio_model = LinearRegression()

# Para crear el modelo, usamos fit(x,y)
radio_model.fit(x,y)


y_pred = radio_model.predict(x)
plt.scatter(x,y,color = 'b')
plt.plot(x,y_pred,color = 'r')
plt.title('Ventas v/s Presupuesto Radio')
plt.xlabel('Ventas')
plt.ylabel('Presupuesto Radio')
plt.show()
```



In [55]:
```
from sklearn.metrics import r2_score
r2_score(y, y_pred)
```

Out[55]:
```
0.33203245544529525
```

Fuente: https://machinelearningmastery.com/regression-metrics-for-machine-learning/

Mean square error MSE

In [56]:
```
# example of calculate the mean squared error
from sklearn.metrics import mean_squared_error
# real value
expected = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
# predicted value
predicted = [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0]
# calculate errors
#errors = mean_squared_error(expected, predicted, squared=False)
errors = mean_squared_error(expected, predicted, squared=True)
# report error
print(errors)
```

```
0.35000000000000003
```

Root Mean square error RMSE

In [57]:
```
# example of calculate the root mean squared error
from sklearn.metrics import mean_squared_error
```

```
# calculate errors
#errors = mean_squared_error(expected, predicted)
errors = mean_squared_error(expected, predicted, squared=False)
# report error
print(errors)
```

```
0.5916079783099616
```

Mean Absolute Error MAE

In [58]:
```
# example of calculate the mean absolute error
from sklearn.metrics import mean_absolute_error
# calculate errors
errors = mean_absolute_error(expected, predicted)
# report error
print(errors)
```

```
0.5
```

Root Mean Square Logarithmic Error (RMSLE)

In [59]:
```
 # example of calculate the mean absolute error
from sklearn.metrics import mean_squared_log_error
# calculate errors
errors = mean_squared_log_error(expected, predicted)
# report error
print(errors)
```

```
0.14402673725223544
```

R2_Score

In [60]:
```
 # example of calculate the mean absolute error
from sklearn.metrics import r2_score
# real value
expected = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
# predicted value
#predicted = [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0]
predicted = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
# calculate errors
errors = r2_score(expected, predicted)
# report error
print(errors)
```

```
1.0
```

Hallar el R cuadrado o el coeficiente de determinación:

A continuación se muestra la solución de la Actividad Solicitada

In [61]:
```
#Funciones
from sklearn.metrics import r2_score
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [62]:
```
# Creando el DataFrame
data = {
    'TV': [230.1, 44.5, 17.2, 151.5, 180.8],
    'Radio': [37.8, 39.3, 45.9, 41.3, 10.8],
    'Newspaper': [69.2, 45.1, 69.3, 58.5, 58.4],
    'Sales': [22.1, 10.4, 9.3, 18.5, 12.9]
}
df = pd.DataFrame(data)

# Preparando los datos para el modelo
X = df[['TV', 'Radio', 'Newspaper']]
y = df['Sales']

df.head()
```

Out[62]:

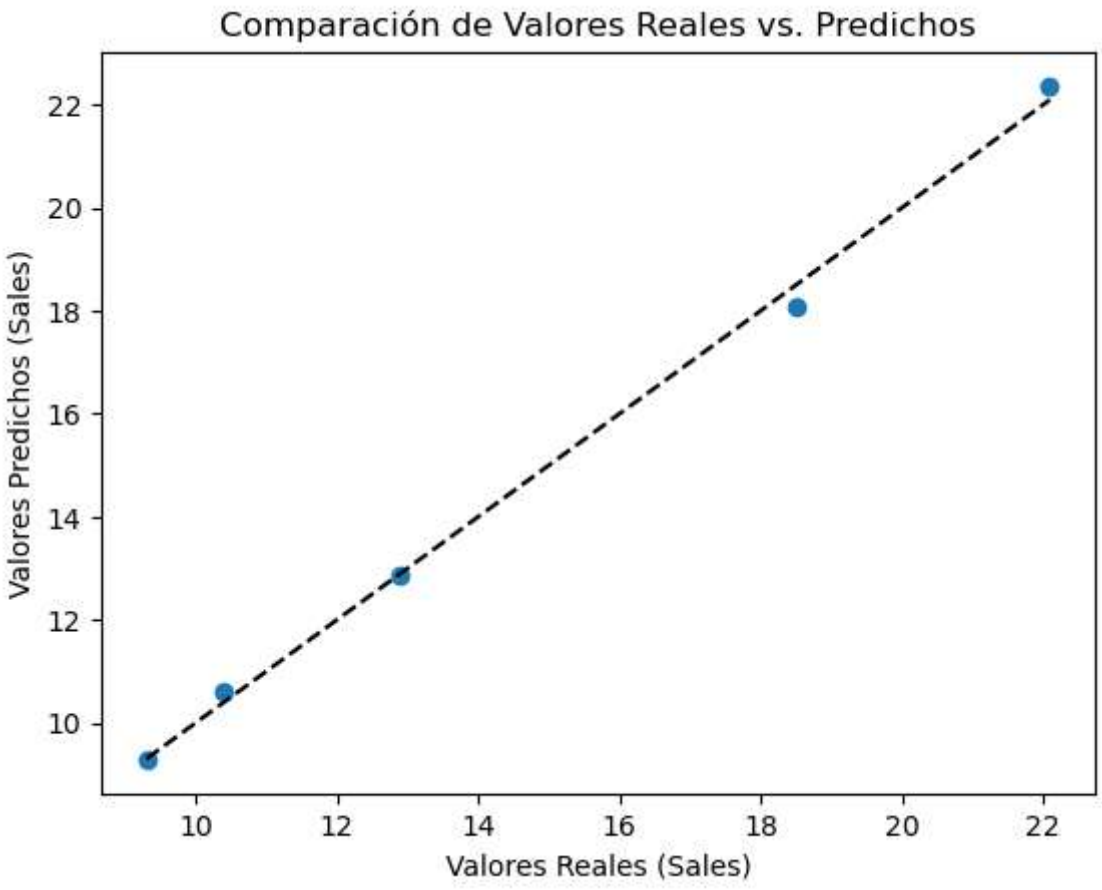
	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

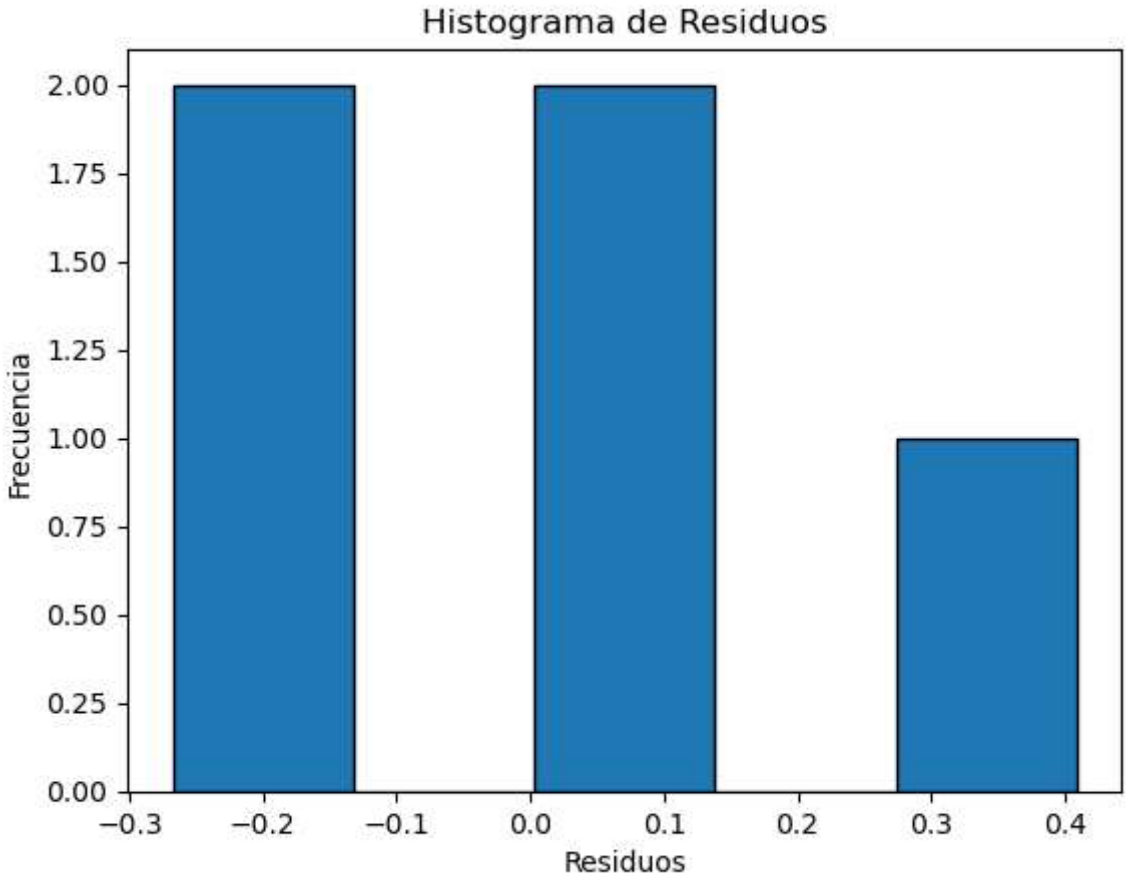
In [80]:
```
# Ajustando el modelo de regresión lineal
model = LinearRegression()
model.fit(X, y)

# Realizando predicciones
y_pred = model.predict(X)

# calculando errors MAE
mae = mean_absolute_error(y, y_pred)

# calculndo errors MSE
mse = mean_squared_error(y, y_pred, squared=True)

# Calculando el Root Mean Squared Logarithmic Error (RMSLE)
rmsle = mean_squared_log_error(y, y_pred_adjusted, squared=False)

# Calcular el R2 Score
# y_true son los valores reales
# y_pred son los valores predichos por tu modelo
r2 = r2_score(y, y_pred)

print("Mean square error (MSE):", mse)
print("Mean Absolute Error (MAE):", mae)
print("Root Mean Squared Logarithmic Error (RMSLE):", rmsle)
print("R2 Score:", r2)
```

```
Mean square error (MSE): 0.05636428314127091
Mean Absolute Error (MAE): 0.1878129769362708
Root Mean Squared Logarithmic Error (RMSLE): 0.013440870045646465
R2 Score: 0.9976528964645683
```

Gráficas:

In [81]:
```
# Gráfico de Predicción vs Realidad
plt.scatter(y, y_pred)
plt.xlabel('Valores Reales (Sales)')
plt.ylabel('Valores Predichos (Sales)')
plt.title('Comparación de Valores Reales vs. Predichos')
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'k--')  # Línea de perfecta predicció
plt.show()

# Histograma de Residuos
residuos = y - y_pred
plt.hist(residuos, bins=5, edgecolor='black')
plt.xlabel('Residuos')
plt.ylabel('Frecuencia')
plt.title('Histograma de Residuos')
plt.show()
```




Histograma de Residuos
Frecuencia
Residuos
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4

In [ ]: