UNIVERSIDAD SERGIO ARBOLEDA

Ingeniería en Ciencias de la Computación e I.A.
Informe Parcial III - Regresión Lineal
Curso: Computación de Alto Rendimiento 03

Modelo Regresión Lineal Python vs. C++

Autor: Carlos Alberto Arévalo Martínez

Resumen: Comprobar la efectividad de los modelos de Regresión Lineal implementados en los dos lenguajes de programación, mediante una métrica de rendimiento.

I. MARCO TEÓRICO

I.I Regresión Lineal

La regresión lineal, según IBM (s.f.), "es una forma de análisis que se utiliza para predecir el valor de una variable según el valor de otra" y "ayuda a comprender y predecir el comportamiento de sistemas complejos" (MathWorks, s.f.).

La regresión lineal permite crear un modelo que describe la relación entre una variable dependiente y ("target") como una función de una o más variables independientes Xi ("features").

La ecuación general que corresponde a un modelo de regresión lineal es:

$$Y = \beta_0 + \sum_{i=1}^{n} \beta_i X_i + \epsilon_i$$

donde β representa las estimaciones de parámetros lineales que se deben calcular y ϵ representa los términos de error.

1.1 Tipos de regresión lineal

Existen 4 tipos de regresión lineal: simple, múltiple, multivariante y múltiple multivariante.

Para el presente ejercicio, se utilizará el tipo de **regresión lineal simple**, que permite crear modelos que utilizan un único predictor. La ecuación general es:

$$Y = \beta_0 + \beta_i X + \epsilon_i$$

La anterior ecuación se puede demostrar de la siguiente forma:

$$y = mX + b + \epsilon_i$$

donde b representa la constante del modelo (también llamada intercepto), que es el punto donde corta el eje de coordenadas y cuando el valor de la variable X es cero.

Por otra parte, m, representa la pendiente (inclinación) de la recta de regresión. Este coeficiente significa el incremento de unidades de la variable y que se produce por cada incremento de una unidad de la variable X.

I.II Método de los Mínimos Cuadrados Ordinarios OLS

El método de los mínimos cuadrados ordinarios se utiliza para calcular la recta de regresión lineal que minimiza los residuos.

El componente estocástico de la ecuación, ϵ_i, representa el error en la estimación; la diferencia entre el valor real de y en la nube de puntos y el valor estimado, representado como $\hat{y}$. Se puede representar matemáticamente como:

$$ei = y_i - \hat{y}_i$$

Los valores de las variables X e y ya se encuentran en la nube de puntos para la que se quiere calcular la recta. Lo que varía en la ecuación de la recta son los coeficientes del modelo, ***b*** y ***m***, aquellos con los se espera que el valor de la suma de residuos sea el menor posible.

Partiendo de la ecuación anterior de cada residuo, se puede representar la suma de residuos de la siguiente forma, donde n es el número de pares de valores de X e y que se dispone:

$$\sum_{i=1}^{n} e_i = \sum y_i - \hat{y}_i$$

El resultado de la diferencia de la fórmula anterior puede arrojar valores tanto positivos como negativos. Por este motivo, se deben calcular estas diferencias elevadas al cuadrado, según la siguiente fórmula:

$$\sum_{i=1}^{n} e_i^2 = \sum (y_i - \hat{y}_i)^2$$

Según la fórmula anterior, definimos el error cuadrático como ***Se***:

$$Se = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

I.III Función de Costo

Para calcular los coeficientes de la recta de regresión se debe sustituir el valor estimado $\hat{y}$, por los términos de la ecuación de la recta de regresión, y así encontrar los valores de ***b*** y ***m*** que minimicen la **función de costo**:

UNIVERSIDAD SERGIO ARBOLEDA

Ingeniería en Ciencias de la Computación e I.A.
Informe Parcial III - Regresión Lineal
Curso: Computación de Alto Rendimiento 03

$$Se = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - (mX_i + b))^2$$

La anterior fórmula se puede resolver como:

$$Se = \sum_{i=1}^{n} (y_i - mX_i - b)^2$$

Al realizar la sumatoria de la anterior fórmula, la función de costo se puede simplificar en:

$$Se = n\bar{y}^2 - 2mn\bar{X}\bar{y} - 2bn\bar{y} + m^2 n\bar{X}^2 + 2mbn\bar{X} + nb^2$$

I.IV Método de Optimización del Gradiente Descendiente

El Gradiente Descendiente es un algoritmo de optimización que permite determinar el mínimo de una función matemática. El gradiente es sinónimo de pendiente o inclinación, y matemáticamente recibe el nombre de derivada.

Para minimizar la función de costo, se procede a utilizar el método de optimización del gradiente descendiente, que permite encontrar el mejor valor para los coeficientes ***m*** y ***b***. Con base en esto, se igualan a cero las derivadas parciales con respecto a ***m*** y respecto a ***b***, de la siguiente manera:

$$\frac{\partial Se}{\partial m} = 0$$
$$\frac{\partial Se}{\partial b} = 0$$

I.V Cálculo de coeficientes

Al despejar ***m*** y ***b*** de las derivadas parciales, se obtienen las siguientes ecuaciones para hallar sus valores:

$$m = \frac{\bar{X}\bar{y} - \bar{X}\bar{y}}{\bar{X}^2 - (\bar{X})^2}$$
$$b = \bar{y} - m\bar{X}$$

II. RECURSOS UTILIZADOS

Se utilizaron los lenguajes de programación C++, donde se implementó una clase artesanal de Regresión Lineal y se verificaron los datos en un cuaderno de Python en Google Colaboratory.

A. Software: Qt Creator, Google Colaboratory.

B. Componentes: Bibliotecas para Aprendizaje de Máquina como SkLearn..

C. Equipos: Para Qt Creator se utilizó una memoria flash con sistema operativo persistente Ubuntu 20.4. Para la parte de Python se utilizó el servidor de Google Colab.

III. PROCEDIMIENTO

Se realizaron los siguientes pasos para cumplir con el objetivo:

A. Se importan las bibliotecas necesarias para poder realizar el procedimiento.
B. Se importan los resultados obtenidos en el modelo implementado en C++.
C. Se importa el dataset California House.
D. Se realiza el análisis exploratorio de los datos EDA, en el que se decide imputar los datos faltantes con los vecinos más cercanos KNN, de las columnas **latitude, housing_median_age, total_rooms, total_bedrooms.** También se decide implementar una transformación de potencia para volver los datos más gaussianos.
E. Se separa en 2 grupos el dataset: variable dependiente y variables independientes.
F. Se dividen los datos en 2 grupos: prueba y entrenamiento mediante la función ***train_test_split***.
G. Se crea un Pipeline donde se implementan los métodos del EDA (incluyendo el escalado Z) y el modelo de Regresión Lineal. Posteriormente se entrena el pipeline (Ver Anexo 3).
H. Con el resultado del modelo de RL del Pipeline se muestra el vector de coeficientes y el punto de corte.
I. Se crean los vectores de predicción del modelo tanto para entrenamiento como para prueba.
J. Se evalúa el rendimiento de los 2 modelos (Python y C++) para entrenamiento y prueba con el método R2 score.
K. Se realiza el gráfico de la función de costo del modelo C++.
L. Se realiza el gráfico comparativo de predicciones de entrenamiento del modelo SkLearn vs predicciones de entrenamiento del modelo C++.
M. Se realiza el gráfico comparativo de predicciones de prueba del modelo SkLearn vs predicciones de prueba del modelo C++.

UNIVERSIDAD SERGIO ARBOLEDA

Ingeniería en Ciencias de la Computación e I.A.
Informe Parcial III - Regresión Lineal
Curso: Computación de Alto Rendimiento 03

IV. RESULTADOS

Al realizar paso a paso el procedimiento, se muestra la matriz de correlación (Ver Anexo 1), en la que se puede presumir que no hay una relación inversamente proporcional entre la variable dependiente e independientes, por lo tanto, se evidencia que presenta una relación significativamente baja.

Al presentar la distribución de las variables (Ver Anexo 2) se puede apreciar que los resultados no presentan una distribución 'normal', por lo tanto, se decide aplicar una transformación de potencia que hace que los datos sean más gaussianos.

Al realizar el entrenamiento del Pipeline (Ver Anexo 3), se ubica paso del modelo de regresión lineal y se muestran el vector de coeficientes y el punto de corte (Ver Anexo 4).

Al evaluar los modelos de C++ y Python para entrenamiento y prueba (Ver Anexo 5) se obtiene un rendimiento bastante bajo para ambos. Aunque se evidencia que el modelo realizado en Python es más efectivo que el de C++, un posible factor se debe a que al modelo de Python se le implementa los métodos de EDA para mejorar su efectividad.

Al graficar la función de costo del modelo C++, se evidencia como el método de optimización del gradiente descendiente va encontrando el valor mínimo en cada salto (tasa de aprendizaje). El valor mínimo encontrado después de 1000 épocas es 0.326177 (Ver Anexo 6).

En las gráficas comparativas de las predicciones de los modelos para entrenamiento y prueba se evidencia una gran diferencia entre los datos predecidos, esto demuestra el bajo rendimiento en los resultados de las métricas (Ver Anexos 7 y 8).

V. CONCLUSIONES

Al realizar artesanalmente la clase en C++ sobre Regresión Lineal, no se implementaron tantos métodos para el análisis exploratorio de los datos y poder jugar más con ellos. Para mejorar su rendimiento se podría integrar manualmente los métodos utilizados en Python, como el de la transformación de potencia e imputación de datos nulos, entre otros.

Se recomienda utilizar distintas métricas de rendimiento como Error cuadrático medio (MSE) o validación cruzada (cross_validate) para tener varios resultados y tener una mejor perspectiva del modelo.

VI. REFERENCIAS

IBM. (s.f.). Regresión lineal. Obtenido de IBM: https://www.ibm.com/co-es/analytics/learn/linear-regression

The MathWorks, Inc. (s.f.). ¿Qué es la regresión lineal? Obtenido de MathWorks: https://la.mathworks.com/discovery/linear-regression.html

Molina, M. (17 de 6 de 2020). La distancia más corta. El método de los mínimos cuadrados. Obtenido de Anestesiar: https://anestesiar.org/2020/la-distancia-mas-corta-el-metodo-de-los-minimos-cuadrados/

Sotaquirá, M. (2 de 7 de 2018). ¿Qué es el Gradiente Descendente? Obtenido de CodificandoBits: https://www.codificandobits.com/blog/el-gradiente-descendente/

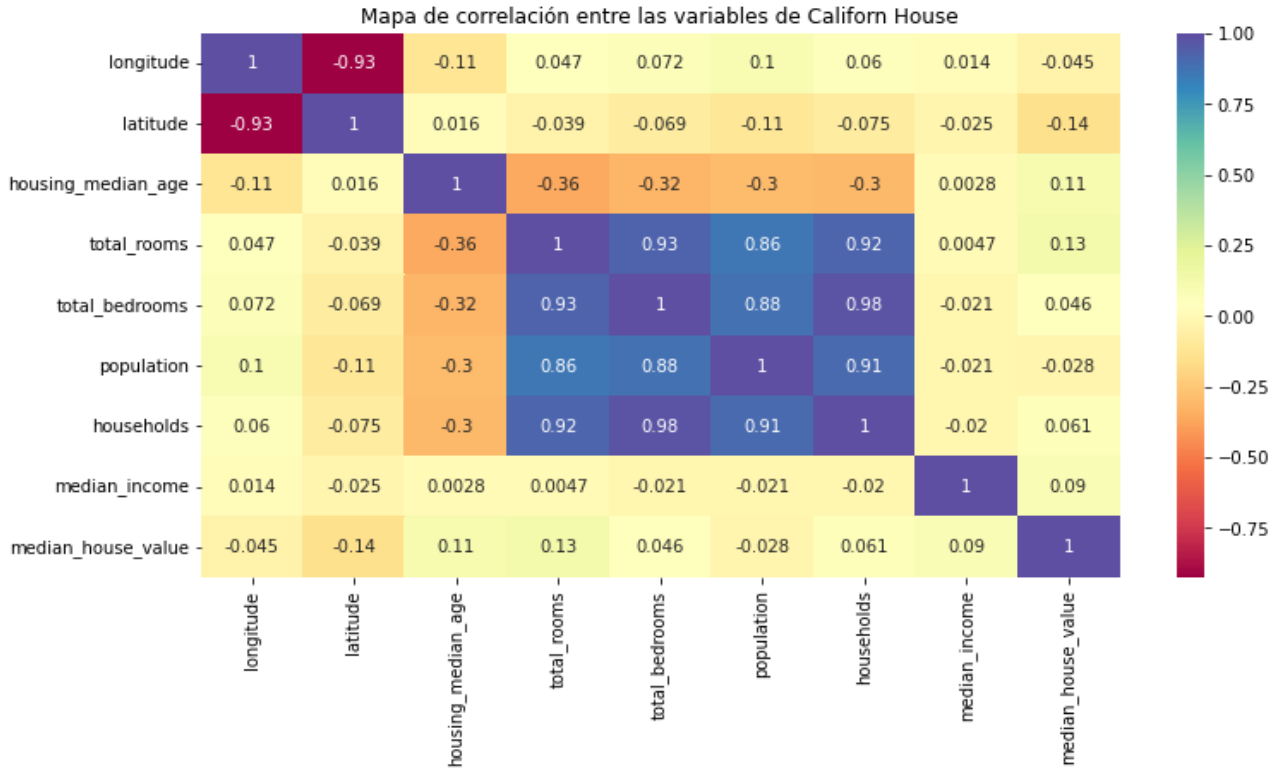
InteractiveChaos. (s.f.). R2. Obtenido de InteractiveChaos: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/r2

UNIVERSIDAD SERGIO ARBOLEDA

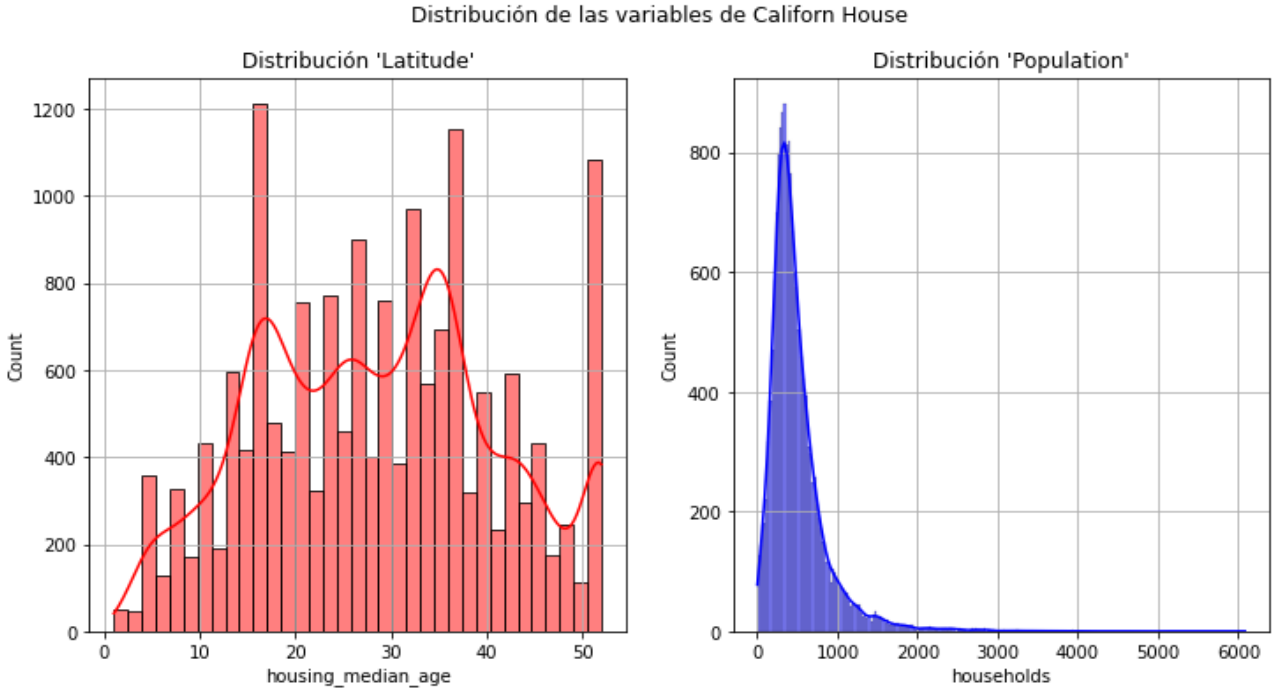
Ingeniería en Ciencias de la Computación e I.A.
Informe Parcial III - Regresión Lineal
Curso: Computación de Alto Rendimiento 03

VII. ANEXOS

1. Matriz de correlación



2. Distribuciones



3. Implementación del EDA mediante un Pipeline y posterior entrenamiento.

```
pipe = Pipeline([('Imputer', KNNImputer(n_neighbors=5)),
                 ('Scaler', StandardScaler()),
                 ('Gaussian', PowerTransformer(method='yeo-johnson', standardize=True)),
                 ('ModelLR', LinearRegression())])

# Se entrena el pipe
pipe.fit(X_train, y_train)
```

```
Pipeline(steps=[('Imputer', KNNImputer()), ('Scaler', StandardScaler()),
                ('Gaussian', PowerTransformer()),
                ('ModelLR', LinearRegression())])
```

4. Vector de coeficientes y punto de corte.

```
print(f"Vector de coeficientes: {pipe.steps[3][1].coef_}")
print(f"Punto intercepto: {pipe.steps[3][1].intercept_}")
```

```
Vector de coeficientes: [ -85311.45107139 -107125.62095851    8454.41334786  120174.2323868
  -71836.48331268  -70857.69884878   30959.09441453    2006.04430643]
Punto intercepto: 197507.07625
```

5. Métrica de rendimiento.

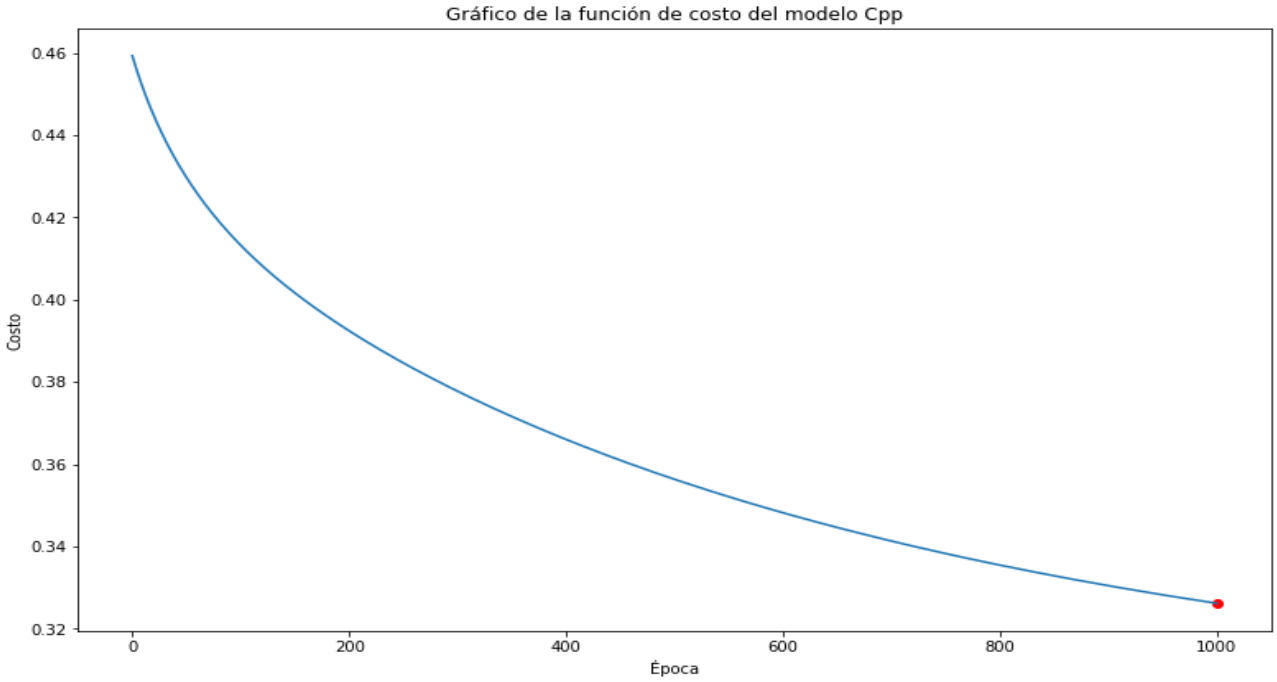
```
# Entrenamiento
r2Sk_train = r2_score(y_train, y_train_hat_sk)
r2Cpp_train = r2_score(y_train, y_hatCpp_train)
print(f'Métrica de rendimiento SK train (r2_score) {r2Sk_train*100:.2f}%')
print(f'Métrica de rendimiento CPP train (r2_score) {r2Cpp_train*100:.2f}%')
```

```
Métrica de rendimiento SK train (r2_score) 39.38%
Métrica de rendimiento CPP train (r2_score) 28.43%
```

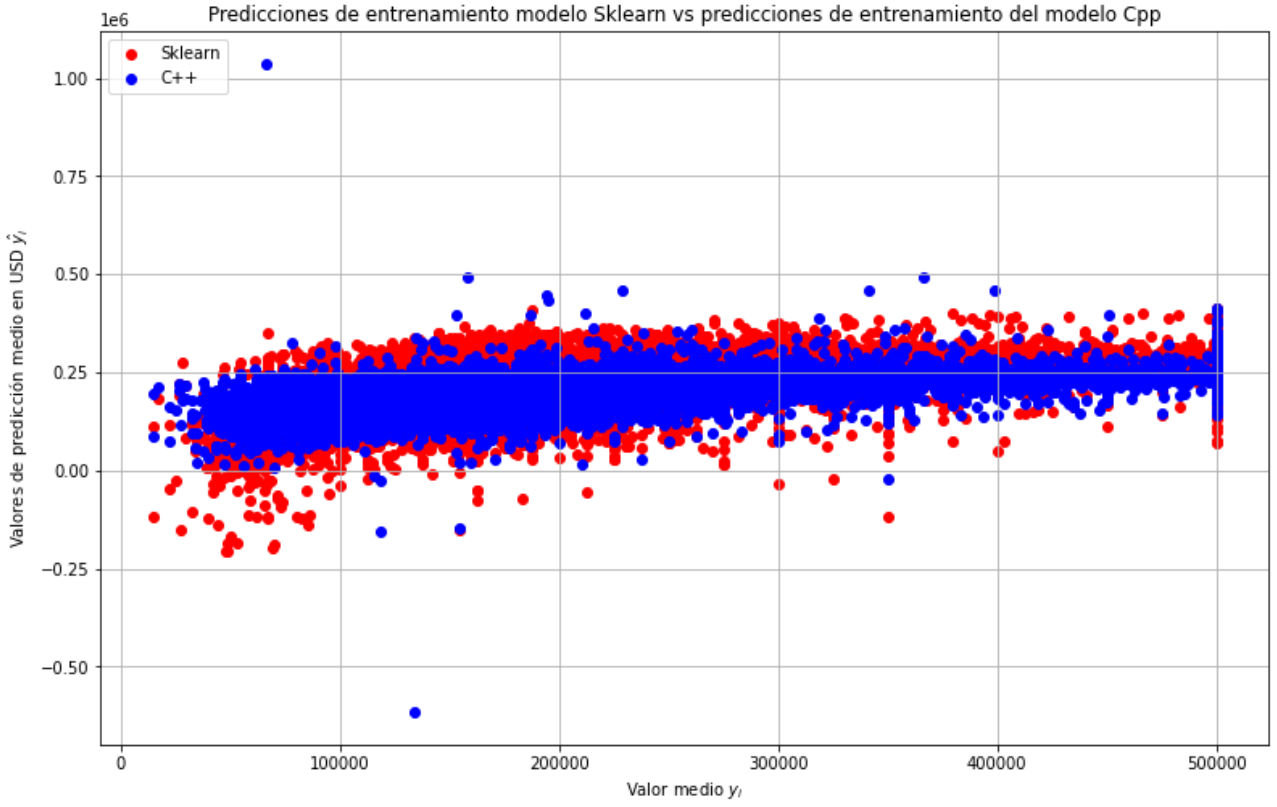
```
# Prueba
r2Sk_test = r2_score(y_test, y_test_hat_sk)
r2Cpp_test = r2_score(y_test, y_hatCpp_test)
print(f'Métrica de rendimiento SK test (r2_score) {r2Sk_test*100:.2f}%')
print(f'Métrica de rendimiento CPP test (r2_score) {r2Cpp_test*100:.2f}%')
```

```
Métrica de rendimiento SK test (r2_score) 24.24%
Métrica de rendimiento CPP test (r2_score) 33.70%
```

6. Gráfico de la función de costo del modelo C++.



7. Gráfico comparativo de las predicciones de entrenamiento del modelo Sklearn vs. las predicciones de entrenamiento del modelo C++.



8. Gráfico comparativo de las predicciones de prueba del modelo Sklearn vs. las predicciones de prueba del modelo C++.