Modeling Gold Close Using CPI and Nasdaq Close: A Linear Regression and LWLR Approach

Comparative analysis of predictive models for estimating the gold price using linear regression and LWLR on historical financial data

Hernández Pérez Erick Fernando

Figure 1: Gold Close

Since time immemorial, gold has been a fundamental asset in the global economy, serving as a store of value, a refuge from crises, and a reference for monetary systems. Its importance has remained unwavering since the adoption of the Gold Standard in the 19th century until its subsequent abandonment in the 1970s, when the global financial system migrated to a fiat money economy. However, despite this transition, the price of gold remains a key indicator of economic stability and confidence in financial markets.

Between 2010 and 2014, the gold market experienced a period of high volatility motivated by various macroeconomic and geopolitical factors. Following the 2008 Financial Crisis, central banks implemented monetary expansion strategies, such as quantitative easing (QE), which encouraged investors to seek refuge in safe assets, such as gold. In 2011, the price of the metal reached its highest point in history up to that point, exceeding USD 1,900 per ounce, driven by uncertainty surrounding sovereign debt in Europe and slowing growth in advanced economies.

However, from 2012 onwards, the picture changed dramatically. The progressive normalisation of monetary policy in the US, together with signs of economic recovery and contained inflation, caused the price of gold to fall, marking a downward trend that persisted until 2014. During this period, factors such as the evolution of the Consumer Price Index (CPI) and the performance of the stock markets played a crucial role in the dynamics of the gold market.

Since gold is an asset strongly influenced by macroeconomic indicators, this study focuses on analyzing the relationship between the Consumer Price Index (CPI) and the Nasdaq Close with the gold price, using Multiple Linear Regression (MLR) and Locally Weighted Linear Regression (LWLR). The focus of this paper is to perform a regression of the CPI and the Nasdaq Close to evaluate their ability as qualitative indicators of the gold price, providing an analytical tool.

For a deeper understanding of these techniques, it is recommended to see the previous entry: 5 Applying Regression in TensorFlow 2: Methods and Techniques.

- Multiple Linear Regression (MLR): Allows the joint influence of the CPI and the Nasdaq to be assessed on the price of gold, providing an interpretable and efficient model.

- Locally Weighted Linear Regression (LWLR): A technique that adjusts local models depending on the proximity of the data, allowing for better capture of non-linear patterns in the relationship between economic variables and the closing price of gold.

Evolution and Influence of the Gold Price

The price of gold experienced significant fluctuations between 2010 and 2014, reflecting the interaction of various macroeconomic and financial factors. According to data from IndexMundi, in September 2011, gold reached an all-time high of approximately 1,772.14 USD per ounce, driven by global economic uncertainty and expansionary monetary policies. However, in 2013, the price suffered a notable correction, closing December at around 1,221.51 USD per ounce. This fall was influenced by the improvement in economic conditions and expectations of a reduction in stimuli by the US Federal Reserve.

One of the key factors affecting the price of gold is monetary policy and interest rates. Decisions by central banks, especially the Federal Reserve, directly impact the price of the precious metal. The quantitative easing policies implemented after the 2008 financial crisis increased the money supply, leading investors to seek refuge in gold for fear of possible inflation. In turn, investment demand and gold's role as a safe haven asset have been decisive in its behavior. During periods of economic or geopolitical uncertainty, such as the sovereign debt crisis in Europe in 2011, investors increased their exposure to gold, raising its price significantly.

Another fundamental aspect is the relationship between gold and the US dollar, since both assets usually show an inverse correlation. When the dollar weakens, the price of gold tends to rise, as it becomes more accessible to investors operating in other currencies. This dynamic has been observed on multiple occasions throughout history.

Gold Price Regression

The dataset available on Kaggle, titled Gold Price Regression and published by FranciscoGCC, brings together a wide collection of financial information relevant to market analysis. This dataset includes key economic indicators, such as the Consumer Price Index (CPI), the Nasdaq closing price, the price of gold, among others.

Starting the project

```
import kagglehub
import matplotlib.pyplot as plt
import seaborn as sns
import os
import pandas as pd


# Download latest version
path = kagglehub.dataset_download("franciscogcc/financial-data")
csv_file_path = os.path.join(path, "financial_regression.csv")
df = pd.read_csv(csv_file_path)

print(df.head())
```

In order to model the behavior of the gold price, it is essential to consider variables that reflect both macroeconomic factors and financial market dynamics. In this study, the following indicators have been selected, each with a key relationship with gold:

The S&P 500 Close and the Nasdaq Close represent the performance of the US stock market, which has historically shown an inverse relationship with gold. In periods of economic uncertainty, investors tend to abandon equity assets and take refuge in gold, increasing its price.

US interest rates (us_rates_%) are a determining factor in the valuation of gold. When rates rise, the opportunity cost of holding gold (which does not generate returns) increases, which reduces its demand and, consequently, its price. Conversely, low rates favor investment in this precious metal.

The Consumer Price Index (CPI) is a key indicator of inflation, and gold has historically been used as a hedge against the loss of value of money. An increase in the CPI could imply a higher demand for gold as a safe haven asset.

The USD/CHF and EUR/USD exchange rates allow us to assess the strength of the dollar against other currencies. Because gold is priced in dollars, a weaker dollar tends to make gold more accessible and attractive to international investors, driving up its price.

Finally, the silver close has been included in the analysis due to the correlation between the two precious metals. Since silver shares similar investment characteristics to gold, its behavior can provide valuable information on commodity market trends.

Starting the project

```
columns_of_interest = ['sp500 close', 'nasdaq close', 'us_rates_%', 'CPI', 'usd_chf',
↪   'eur_usd', 'silver close', 'gold close']
df_selected = df[columns_of_interest]

sns.pairplot(df_selected)
```

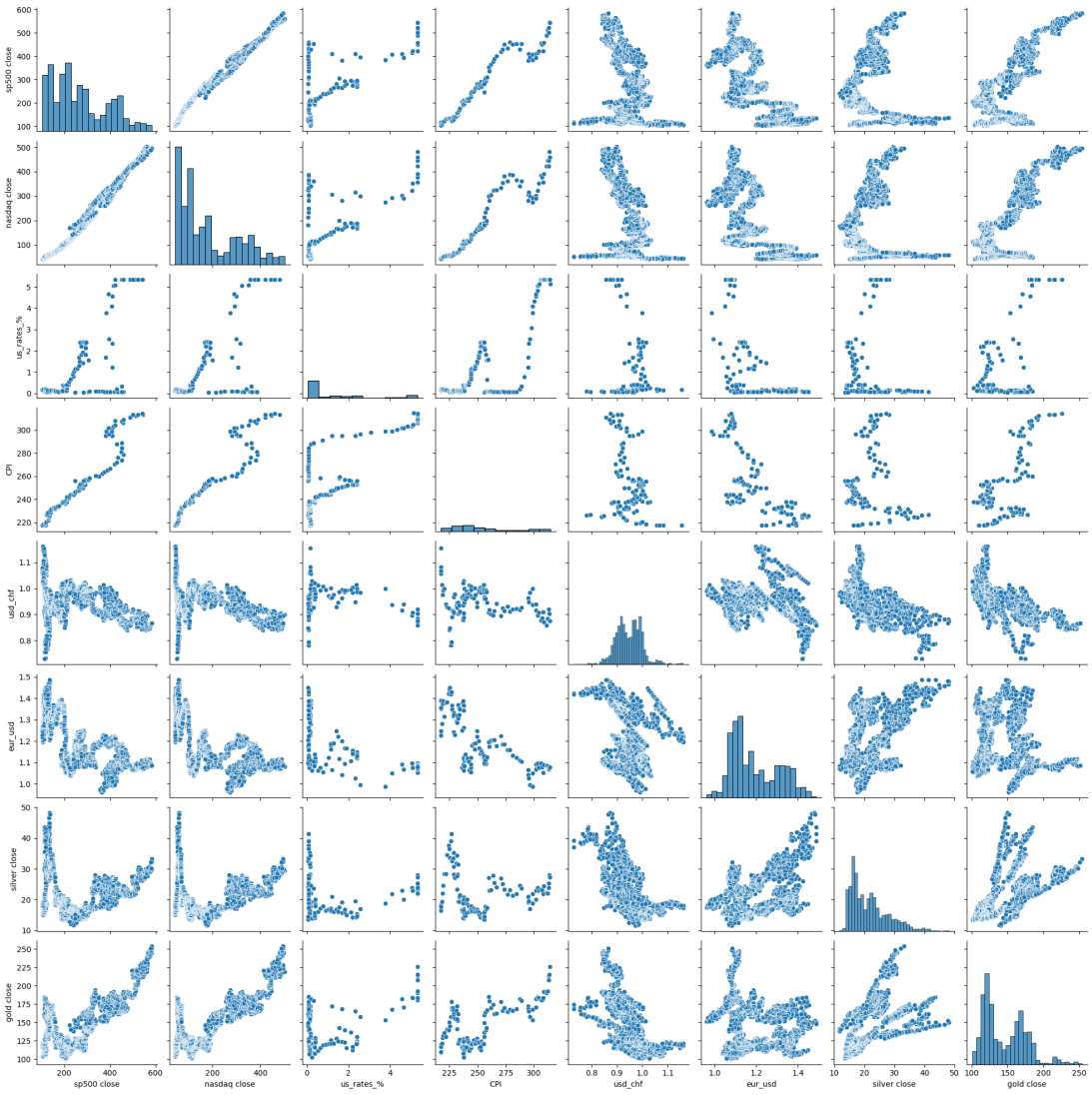

Figure 2: Gold Pairplot

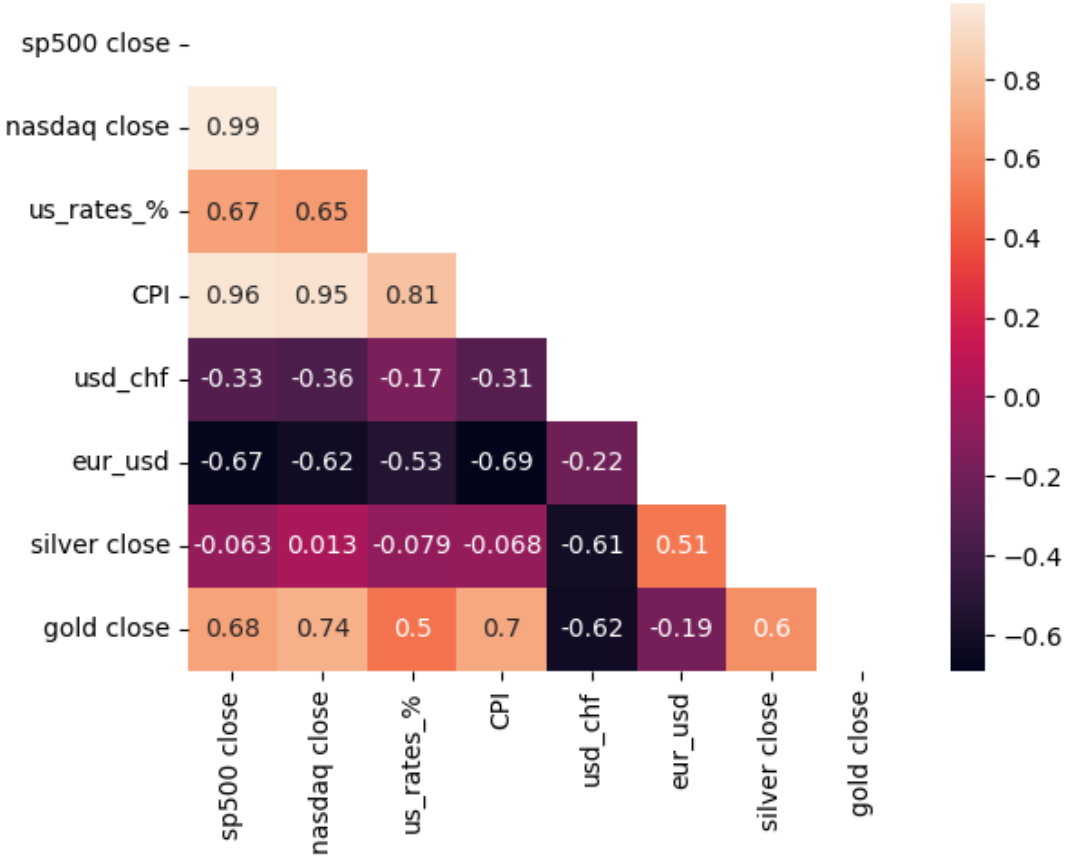

Figure 3: Heatmap of correlation matrix

A strong positive correlation is observed between the CPI and the Nasdaq Close (0.95), suggesting that when the price level in the economy increases, the value of technology companies tends to do so as well. This could be explained by the relationship between inflation and growth in financial markets: when prices rise, monetary policy and access to credit can impact investor expectations, boosting the performance of the Nasdaq.

A significant correlation is also identified between the CPI and the gold close (0.70), indicating that the increase in the price level generally coincides with an increase in the value of gold. This is consistent with the idea that gold is a safe haven asset in periods of inflation.

On the other hand, the Nasdaq Close and the Gold Close present a positive correlation of 0.74, suggesting that the stock market and gold do not always have an inverse relationship, as is often thought. In this case, both can grow simultaneously in certain periods, reflecting a dynamic in which economic expansion drives both the stock market and the demand for gold.

Figure 4 presents a scatter plot comparing the Consumer Price Index (CPI) with the Nasdaq Close, incorporating a color scale based on the Gold Close. This type of visualization allows for the identification of key patterns in the relationship between these variables and their influence on the behavior of the gold market.

One of the most notable aspects of Figure 4 is the clear trend in which, as both the CPI and the Nasdaq Close increase, the Gold Close values also tend to rise.

The use of the color scale in the graph reinforces this observation: the most intense shades, corresponding to higher Gold Close values, are concentrated in the region where both the CPI and the Nasdaq Close reach high levels. This finding is consistent with the correlation analysis, which showed a positive relationship between these variables.

From an economic perspective, this behaviour can be explained by the impact of inflation on financial markets and the demand for safe haven assets such as gold. When the CPI rises, investors seek to protect their capital against the loss of purchasing power, which drives the purchase of gold and, consequently, its price. On the other hand, the growth of the Nasdaq may be reflecting a phase of economic expansion, where liquidity in the markets favours both investment in technological stocks and in commodities such as gold.

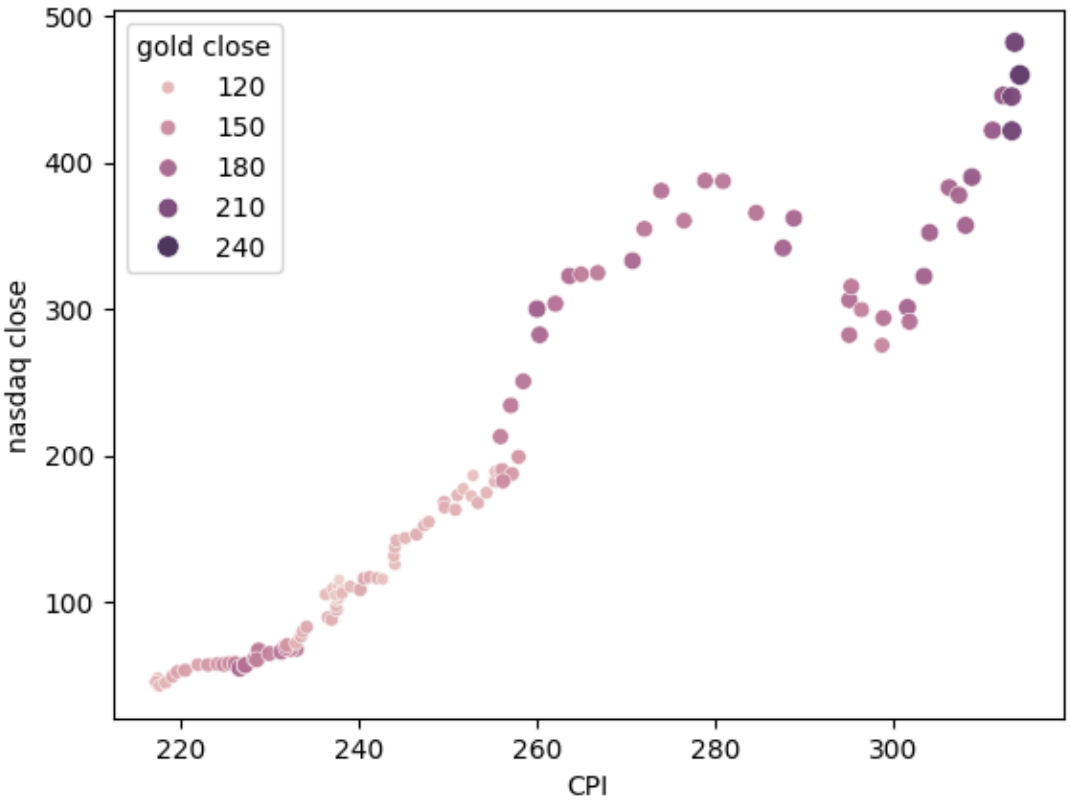

Figure 4: The CPI compared to the Nasdaq Close with a Gold Close hue

Linear Regression

Scaled Model

```
import tensorflow as tf
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt

df_new = df[['CPI', 'nasdaq close', 'gold close']].dropna()
df_new.head()

# Hyperparameter configuration
learning_rate = 0.01
training_epochs = 500
num_coeffs = 2
batch_size = 32

# Vectorized polynomial model
def model(X, w):
    X = tf.expand_dims(X, axis=1)  # [batch_size, 1]
    exponents = tf.range(num_coeffs, dtype=tf.float32)  # [num_coeffs]
    terms = w * tf.pow(X, exponents)  # [batch_size, num_coeffs]
    return tf.reduce_sum(terms, axis=1)  # [batch_size]
```

```python
# Data preparation
X = df_new['CPI'].to_numpy().astype(np.float32)
y = df_new['nasdaq close'].to_numpy().astype(np.float32)

# Normalization
scaler_X = MinMaxScaler()
X_scaled = scaler_X.fit_transform(X.reshape(-1, 1)).flatten()

scaler_y = MinMaxScaler()
y_scaled = scaler_y.fit_transform(y.reshape(-1, 1)).flatten()

# Dataset splitting
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y_scaled, test_size=0.15, random_state=42)

# TensorFlow dataset creation
train_dataset = tf.data.Dataset.from_tensor_slices((X_train, y_train))
train_dataset = train_dataset.shuffle(buffer_size=1024).batch(batch_size)

# Variable initialization and optimizer
w = tf.Variable(tf.zeros([num_coeffs], dtype=tf.float32), name="parameters")
optimizer = tf.optimizers.SGD(learning_rate=learning_rate)

# Compiled training function
@tf.function
def train_step(x_batch, y_batch):
    with tf.GradientTape() as tape:
        y_pred = model(x_batch, w)
        loss = tf.reduce_mean(tf.square(y_batch - y_pred))
    gradients = tape.gradient(loss, [w])
    optimizer.apply_gradients(zip(gradients, [w]))
    return loss

# Training loop
for epoch in range(training_epochs):
    epoch_loss = 0
    for x_batch, y_batch in train_dataset:
        loss = train_step(x_batch, y_batch)
        epoch_loss += loss.numpy()
    if (epoch + 1) % 100 == 0:
        print(f"Epoch {epoch + 1}, Average loss: {epoch_loss / len(train_dataset):.6f}")

# Visualization of the learned model
plt.scatter(X_train, y_train, label='Scaled dataset', color="blue", alpha=0.5)
x_vals = np.linspace(0, 1, 100, dtype=np.float32)
y_vals = model(x_vals, w).numpy()
plt.plot(x_vals, y_vals, 'r-', label='Learned model')
plt.xlabel('CPI (scaled)')
plt.ylabel('Nasdaq Close (scaled)')
plt.legend()
plt.title('Linear Regression (scaled)')
plt.show()
```

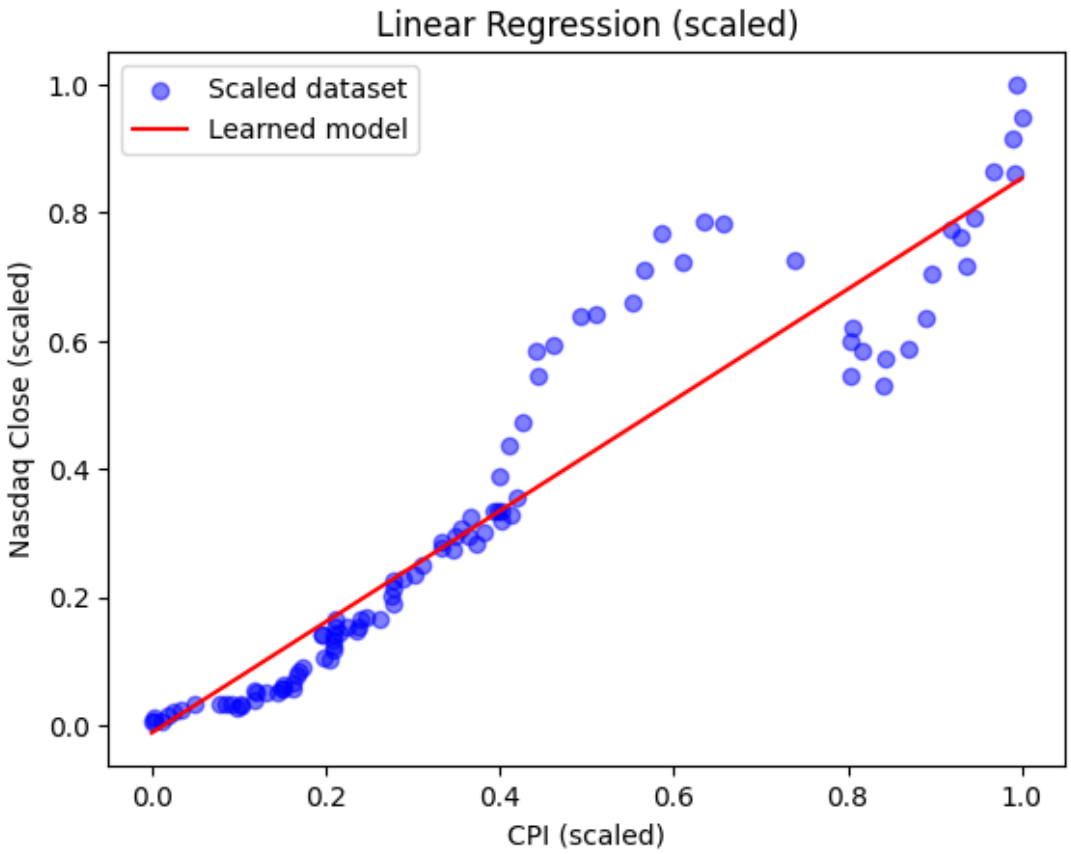

Figure 5: Scaled Linear Regression

And when evaluating the model we have:

- Mean Squared Error (MSE): 0.007302
- Mean Absolute Error (MAE): 0.060765
- R-squared (R2): 0.910129

Unscaled Model

```
# Unscaling X and y
X_train_original = scaler_X.inverse_transform(X_train.reshape(-1, 1)).flatten()
y_train_original = scaler_y.inverse_transform(y_train.reshape(-1, 1)).flatten()

# Generate predictions in the original scale
x_vals_scaled = np.linspace(0, 1, 100, dtype=np.float32)  # Value range in normalized
↪  scale
y_vals_scaled = model(x_vals_scaled, w).numpy()  # Predictions in normalized scale

# Unscale the predicted values
x_vals_original = scaler_X.inverse_transform(x_vals_scaled.reshape(-1, 1)).flatten()
y_vals_original = scaler_y.inverse_transform(y_vals_scaled.reshape(-1, 1)).flatten()

# Generate predictions in the normalized scale using X_train
y_pred_scaled = model(X_train, w).numpy()

# Unscale the predictions
y_pred_original = scaler_y.inverse_transform(y_pred_scaled.reshape(-1, 1)).flatten()
```

```
# Compute metrics correctly using original data
mse = mean_squared_error(y_train_original, y_pred_original)
mae = mean_absolute_error(y_train_original, y_pred_original)
r2 = r2_score(y_train_original, y_pred_original)

# Display metrics
print(f"Mean Squared Error (MSE): {mse:.6f}")
print(f"Mean Absolute Error (MAE): {mae:.6f}")
print(f"Coefficient of Determination (R²): {r2:.6f}")

# Visualization in the original scale
plt.scatter(X_train_original, y_train_original, label='Original data', color="blue",
↪  alpha=0.5)
plt.plot(x_vals_original, y_vals_original, 'r-', label='Learned model')
plt.xlabel('CPI')
plt.ylabel('Nasdaq Close')
plt.legend()
plt.title('Linear Regression (Original Scale)')
plt.show()
```

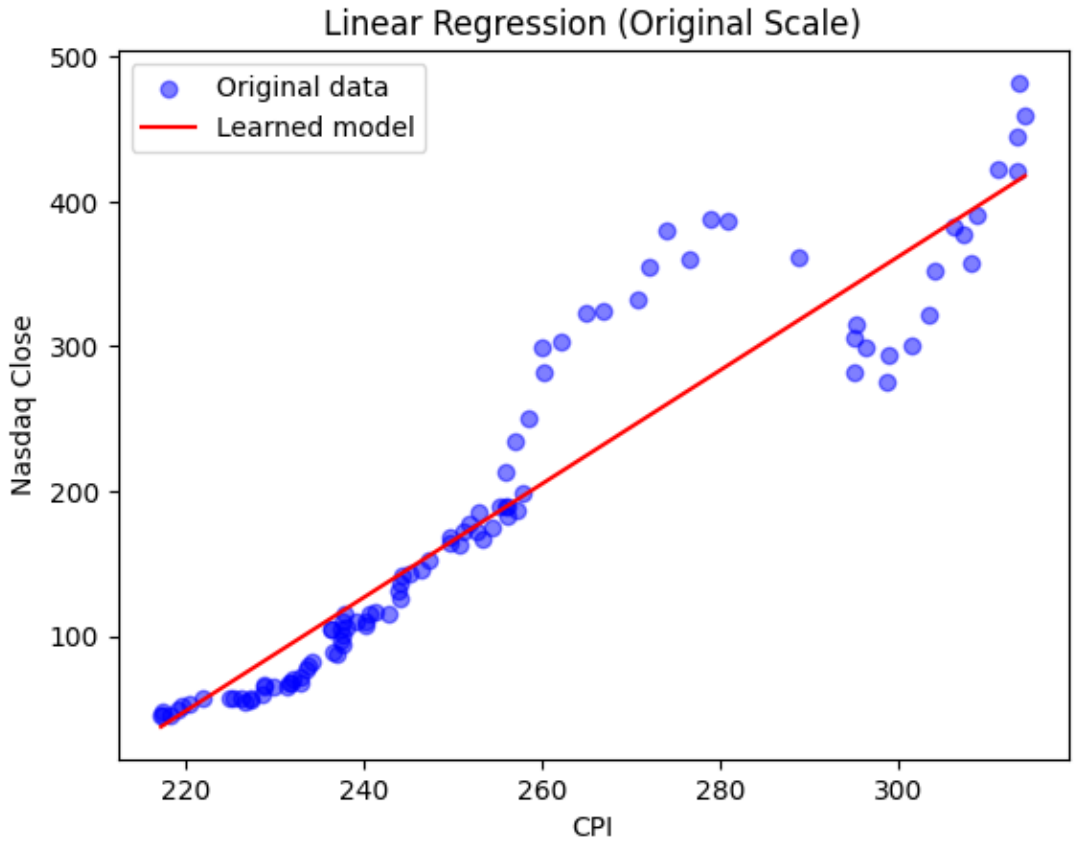

Figure 6: Unscaled Linear Regression

- Mean Squared Error (MSE): 1630.772217
- Mean Absolute Error (MAE): 27.603405

- R-squared (R2): 0.891319

LWLR

LWLR Model

```
import tensorflow as tf
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Compute weights using a Gaussian kernel
def get_weights(X, x_query, tau):
    distances = tf.square(X - x_query)
    weights = tf.exp(-distances / (2 * tau**2))
    return tf.linalg.diag(tf.squeeze(weights))

# LWLR implementation in TensorFlow 2
def locally_weighted_regression(X, y, x_query, tau):
    m = tf.shape(X)[0]
    X_bias = tf.concat([tf.ones((m, 1)), X], axis=1)
    W = get_weights(X, x_query, tau)

    # Solve weighted normal equation:  = (X^T W X)^(-1) X^T W y
    XTWX = tf.matmul(tf.matmul(tf.transpose(X_bias), W), X_bias)
    XTWy = tf.matmul(tf.matmul(tf.transpose(X_bias), W), y)
    theta = tf.linalg.solve(XTWX, XTWy)

    # Ensure that x_query has the correct shape
    x_query_bias = tf.concat([tf.ones((1, 1)), tf.reshape(x_query, (1, 1))], axis=1)
    y_pred = tf.matmul(x_query_bias, theta)

    return tf.squeeze(y_pred)

X_tensor = tf.cast(tf.reshape(df_new['CPI'].to_numpy(), (-1, 1)), tf.float32)
y_tensor = tf.cast(tf.reshape(df_new['nasdaq close'].to_numpy(), (-1, 1)), tf.float32)

tau = 5# Smoothing parameter

# Predict using LWLR
x_test = tf.reshape(tf.linspace(df_new['CPI'].min(), df_new['CPI'].max(), 100), (-1, 1))
y_pred = tf.map_fn(lambda x_q: locally_weighted_regression(X_tensor, y_tensor, x_q, tau),
↪   x_test)

# Plot results
plt.scatter(X_tensor.numpy(), y_tensor.numpy(), label="Data", color="blue", alpha=0.5)
plt.plot(x_test.numpy(), y_pred.numpy(), label="LWLR", color="red", linewidth=2)
plt.title("Locally Weighted Linear Regression")
plt.legend()
plt.show()

X_test = tf.cast(tf.reshape(df_new['CPI'].to_numpy(), (-1, 1)), tf.float32)
y_test = tf.cast(tf.reshape(df_new['nasdaq close'].to_numpy(), (-1, 1)), tf.float32)

# Apply LWLR to each point in X_test
y_pred_test = tf.map_fn(lambda x_q: locally_weighted_regression(X_test, y_test, x_q,
↪   tau), X_test)
```

```
# Convert predictions to NumPy
y_pred_test = y_pred_test.numpy()
y_test_np = y_test.numpy()

# Compute metrics
mse = mean_squared_error(y_test_np, y_pred_test)
mae = mean_absolute_error(y_test_np, y_pred_test)
rmse = np.sqrt(mse)
r2 = r2_score(y_test_np, y_pred_test)

# Display metrics
print(f"Mean Squared Error (MSE): {mse:.6f}")
print(f"Mean Absolute Error (MAE): {mae:.6f}")
print(f"R-squared (R2): {r2:.6f}")
```

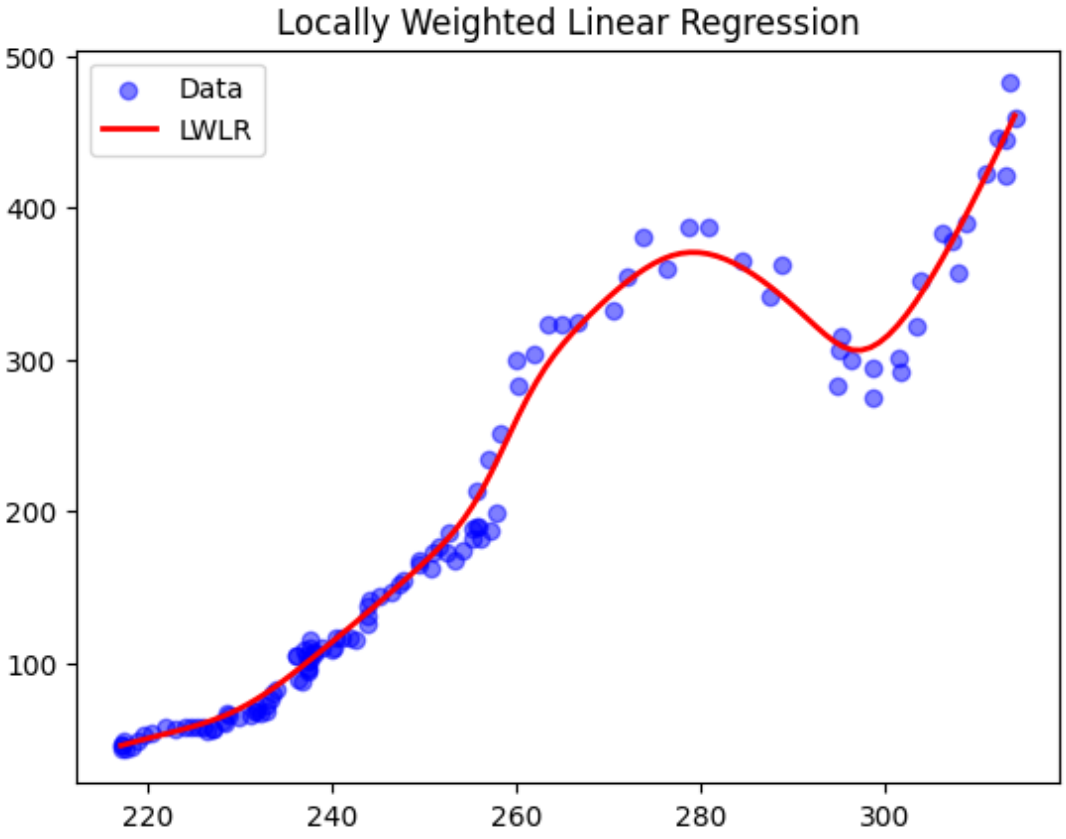

Figure 7: LWLR

- Mean Squared Error (MSE): 170.382996
- Mean Absolute Error (MAE): 9.254128
- R-squared (R2): 0.988766

Conclusion

This study presents a qualitative approach to modeling the Gold Close in relation to the CPI and Nasdaq Close. By employing polynomial regression and locally weighted regression, we aimed to explore the general trends and relationships between these variables rather than precisely quantifying their interactions. The results indicate that as the CPI and Nasdaq Close increase, the Gold Close follows a similar upward trend, reinforcing the idea of a strong correlation between these financial indicators.

However, for a more rigorous and quantitative analysis, a multiple regression approach would be necessary to account for additional influencing factors and isolate the independent effects of each variable on the Gold Close. Such an extension, which would involve feature engineering, interaction terms, and potentially more complex statistical modeling, falls beyond the scope of this text. Nonetheless, the insights gained here provide a foundation for future research and serve as an exploratory perspective on the dynamic interplay between these economic metrics.