LSTM Tweaks

```
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train, y_train, epochs=50, batch_size=32)
```

Epochs

- Tweak: Increase or decrease (e.g., 30, 70).
  More epochs can lead to better learning but also to overfitting.

Batch Size

- Tweak: Try different sizes like 16, 64, or 128.
  Larger batch sizes can speed up training but might impact the model's ability to generalize.

```
LSTM(50, return_sequences=True, input_shape=(X_train.shape[1], 1)),
Dropout(0.2),
LSTM(50, return_sequences=False),
Dropout(0.2),

Dense(1)
```

Dropout Rates

- Tweak: Try different rates like 0.1, 0.3, or 0.5.
  Dropout prevents overfitting by randomly dropping units, but too much dropout can lead to underfitting.

LSTM Layers

- Tweak: Experiment with more layers or different numbers of units (e.g., 100 units or 3 layers).
  More layers/units can capture complex patterns but may also lead to overfitting.

```
if X.size > 0 and y.size > 0:
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Test Size

- Tweak: Experiment with different ratios like 0.3 or 0.1.
  A larger test set gives a better idea of how the model performs on unseen data, but reduces the data available for training.

```
def prepare_data_for_lstm(prices, time_steps=90):
```

Time Steps

- Tweak: Try different time steps like 30, 90, or 120.
  This defines how many past hours the model looks at to make a prediction. A larger window might
  capture more trends but could also introduce noise.