

✓ CURRENCY EXCHANGE RATE PREDICTION

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
file_path = '/content/Nifty 50 Historical Data.csv'
data = pd.read_csv(file_path)

# Data cleaning and preprocessing
# Convert 'Date' to datetime
data['Date'] = pd.to_datetime(data['Date'], format='%d/%m/%Y')

# Remove commas from numerical columns and convert to float
columns_to_clean = ['Price', 'Open', 'High', 'Low']
for col in columns_to_clean:
    data[col] = data[col].str.replace(',', '').astype(float)

# Convert 'Vol.' to numeric (remove suffixes like 'B')
data['Vol.'] = data['Vol.'].str.replace('B', '').astype(float) * 1e9 # Convert 'B' to billion


# Convert 'Change %' to numeric (remove '%')
data['Change %'] = data['Change %'].str.replace('%', '').astype(float)

# Sorting data by date
data = data.sort_values(by='Date')

# Features and target
X = data[['Open', 'High', 'Low', 'Vol.', 'Change %']]
y = data['Price']


# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

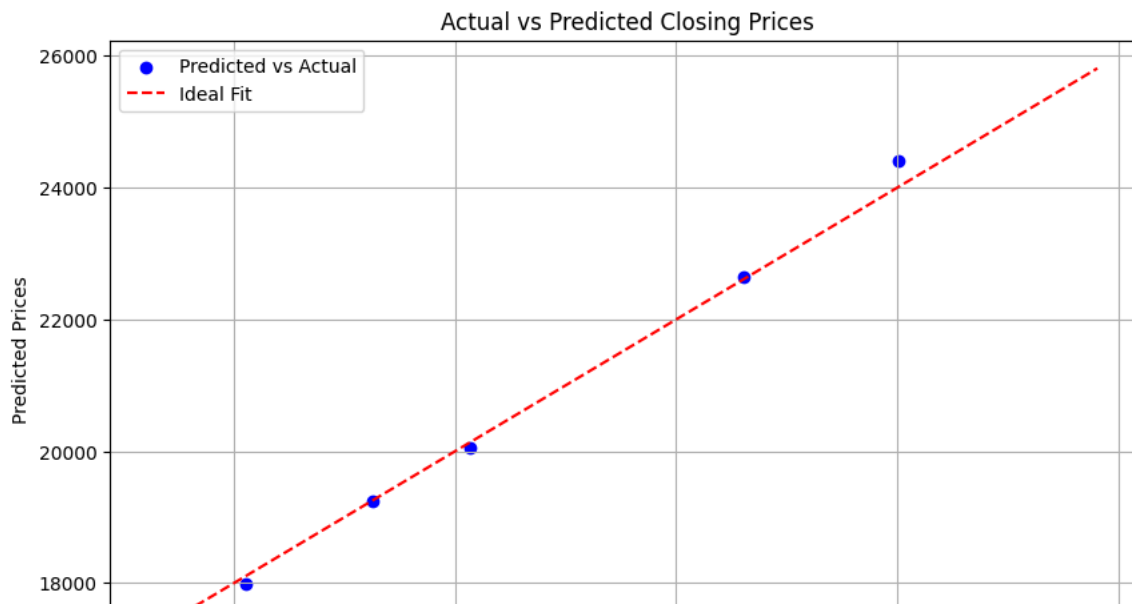


```
# Predictions
y_pred = model.predict(X_test)

# Evaluation metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(mse)
print(r2)
```



```
# Plotting actual vs predicted prices
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', label='Predicted vs Actual')
plt.plot([y.min(), y.max()], [y.min(), y.max()], color='red', linestyle='--', label='Ideal Fit')
plt.title('Actual vs Predicted Closing Prices')
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.legend()
plt.grid(True)
plt.show()
```



```
# Scatter plot with trend line
'''plt.figure(figsize=(10, 6))
sns.regplot(x=y_test, y=y_pred, ci=None, color='blue', line_kws={'color': 'red', 'lw': 2})
plt.title('Actual vs Predicted Closing Prices with Trend Line')
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.grid(True)
plt.show()'''
```



```
'plt.figure(figsize=(10, 6))\nsns.regplot(x=y_test, y=y_pred, ci=None, color='blue', line_kws={'color': 'red', 'lw': 2})\nplt.title('Actual vs Predicted Closing Prices with Trend Line')\nplt.xlabel('Actual Prices')\nplt.ylabel('Predicted Prices')\nplt.grid(True)\nplt.show()'
```

```
# Example of new data points for prediction
```

```
new_data = pd.DataFrame({
    'Open': [18500.0, 18700.0, 19000.0],
    'High': [18800.0, 19000.0, 20000.0],
    'Low': [18200.0, 18350.0, 19100.0],
    'Vol.': [5.5e9, 6.0e9, 7.0e9],
    'Change %': [0.5, -0.8, 1.2]
})
```

```
# Making predictions for the new data
new_predictions = model.predict(new_data)
```

```
# Displaying the predictions
new_data['Predicted Price'] = new_predictions
print(new_data)
```



	Open	High	Low	Vol.	Change %	Predicted Price
0	18500.0	18800.0	18200.0	5.500000e+09	0.5	18551.465204
1	18700.0	19000.0	18350.0	6.000000e+09	-0.8	18511.927047
2	19000.0	20000.0	19100.0	7.000000e+09	1.2	19296.876892