## Gold Price Prediction

## Algorithm

**Step 1**: **Start**

**Step 2: Load Data**

- Load the gold price data and economic indicators.

**Step 3: Preprocess Data**

- Clean the data (handle missing values, if any).
- Split the data into training and testing sets.

**Step 4**: **Engineer Features**

- Create features from the training data.

**Step 5: Train Model**

- Choose and train a machine learning model using the training data.

**Step 6: Make Predictions**

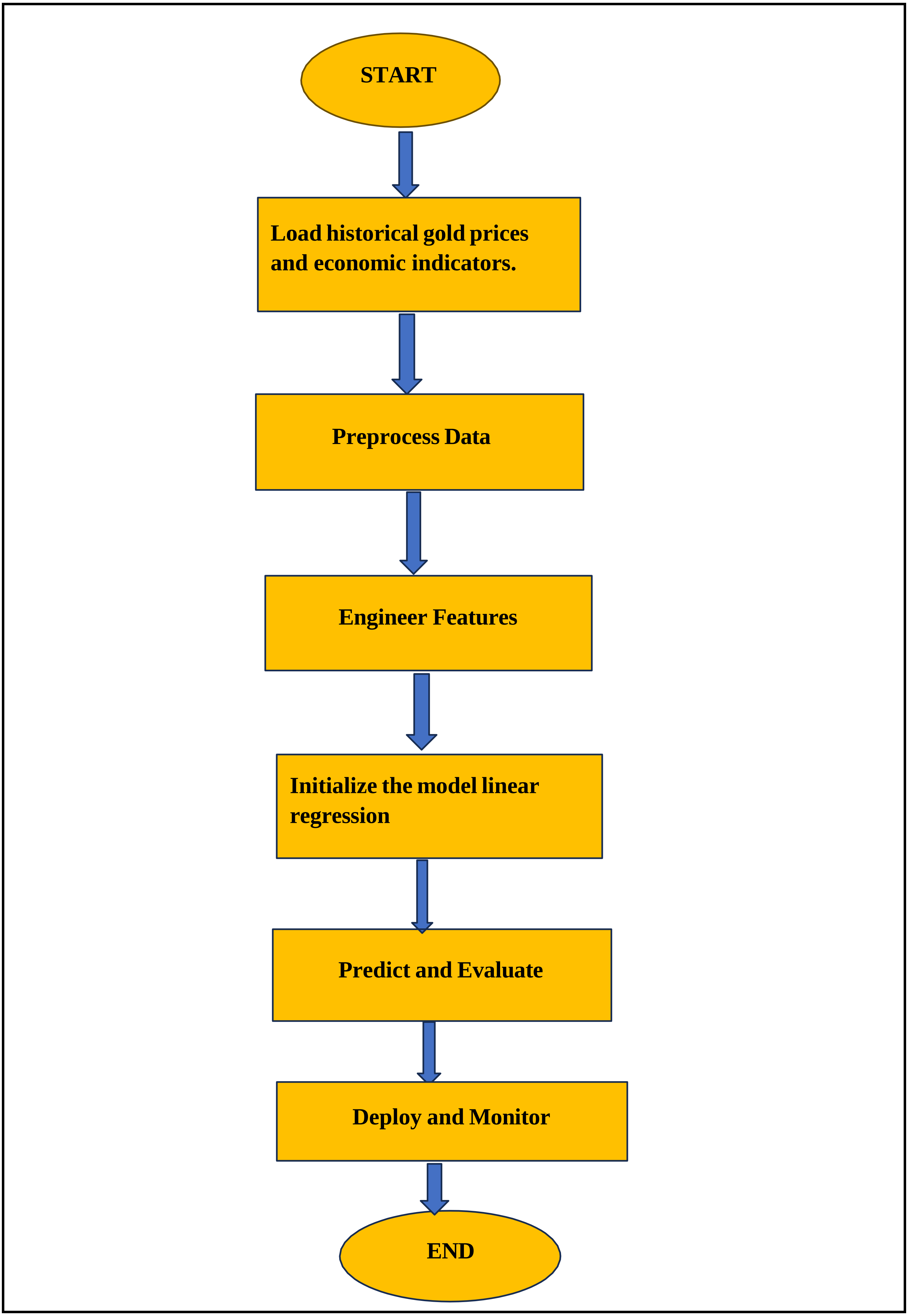
- Predict gold prices on the test set using the trained model.
- Evaluate the model's performance using metrics.

**Step 7**: **Deploy Model**

- Deploy the model for real-world use.

**Step 8: End**

## Flow Chart:




717822P306-T.Bhuvaneshwari

| Year | Month | Gold_Price_INR | Inflation_Rate | Unemployment_Rate | Interest_Rate | GDP_Growth |
|---|---|---|---|---|---|---|
| 2023 | January | 149,400 | 3.1 | 2.4 | 4.2 | 2.5 |
| 2023 | February | 151,060 | 3.3 | 2.5 | 4.1 | 2.6 |
| 2023 | March | 151,690 | 3.2 | 2.6 | 4.0 | 2.4 |
| 2023 | April | 153,550 | 3.4 | 2.5 | 4.1 | 2.7 |
| 2023 | May | 155,410 | 3.6 | 2.6 | 3.9 | 2.8 |
| 2023 | June | 156,440 | 3.7 | 2.7 | 4.0 | 2.9 |
| 2023 | July | 157,700 | 3.8 | 2.8 | 4.2 | 3.0 |
| 2023 | August | 159,360 | 3.9 | 2.9 | 4.1 | 3.1 |
| 2023 | September | 161,420 | 4.0 | 3.0 | 4.0 | 3.2 |
| 2023 | October | 161,850 | 4.1 | 3.1 | 4.2 | 3.3 |
| 2023 | November | 163,410 | 4.2 | 3.2 | 4.1 | 3.4 |
| 2023 | December | 165,070 | 4.3 | 3.3 | 4.0 | 3.5 |

**Python Code:**

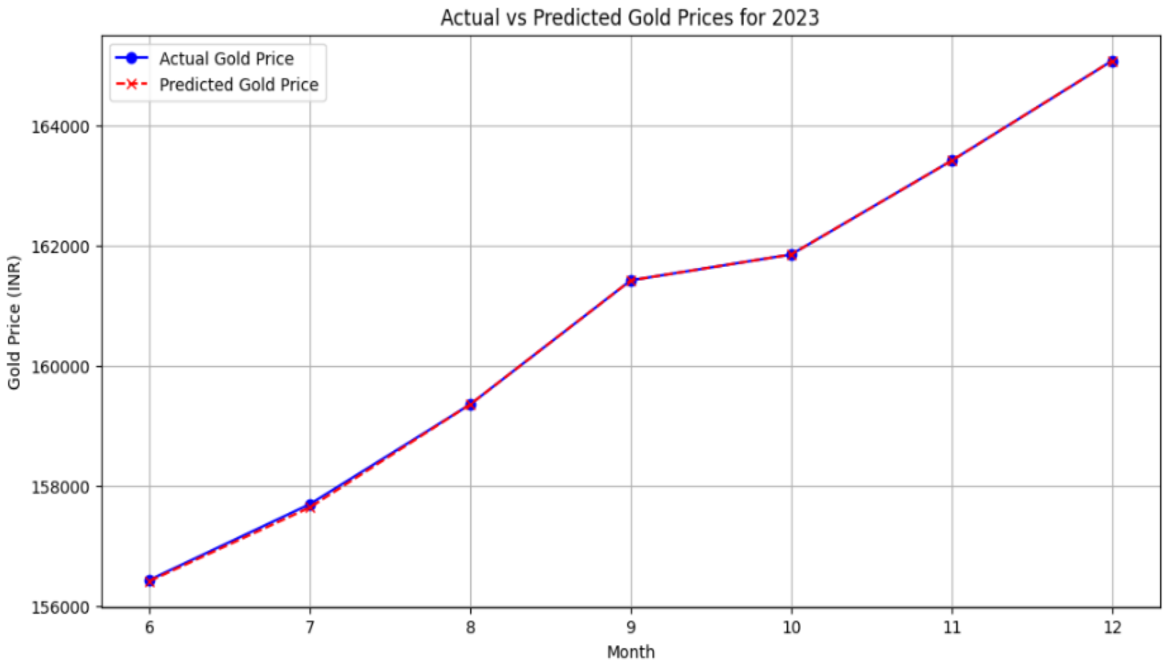
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
data = {
    'Year': [2023]*12,
    'Month': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
    'Gold_Price_INR': [149400, 151060, 151690, 153550, 155410, 156440, 157700, 159360, 161420, 161850, 163410,
165070],
    'Inflation_Rate': [3.1, 3.3, 3.2, 3.4, 3.6, 3.7, 3.8, 3.9, 4.0, 4.1, 4.2, 4.3],
    'Unemployment_Rate': [2.4, 2.5, 2.6, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.3],
    'Interest_Rate': [4.2, 4.1, 4.0, 4.1, 3.9, 4.0, 4.2, 4.1, 4.0, 4.2, 4.1, 4.0],
    'GDP_Growth': [2.5, 2.6, 2.4, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.3, 3.4, 3.5]
}

df = pd.DataFrame(data)

df['Gold_Price_Lag_1'] = df['Gold_Price_INR'].shift(1)
df['Gold_Price_Lag_2'] = df['Gold_Price_INR'].shift(2)
df['MA_3'] = df['Gold_Price_INR'].rolling(window=3).mean()
df['MA_6'] = df['Gold_Price_INR'].rolling(window=6).mean()
df['Volatility_3'] = df['Gold_Price_INR'].rolling(window=3).std()
df.dropna(inplace=True)
features = ['Gold_Price_Lag_1', 'Gold_Price_Lag_2', 'MA_3', 'MA_6', 'Volatility_3', 'Inflation_Rate',
'Unemployment_Rate', 'Interest_Rate', 'GDP_Growth']
target = 'Gold_Price_INR'

X = df[features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
```

```
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')


df['Predicted_Gold_Price_INR'] = model.predict(X)


print("\nActual vs Predicted Gold Prices:")
for index, row in df.iterrows():
    print(f"Month: {row['Month']}, Actual Price: {row['Gold_Price_INR']}, Predicted Price:
{row['Predicted_Gold_Price_INR']:.2f}")
```

## Output:

```
Mean Squared Error: 1668.6907908474564
R^2 Score: 0.9957956896174164

Actual vs Predicted Gold Prices:
Month: 6.0, Actual Price: 156440.0, Predicted Price: 156421.02
Month: 7.0, Actual Price: 157700.0, Predicted Price: 157645.44
Month: 8.0, Actual Price: 159360.0, Predicted Price: 159360.00
Month: 9.0, Actual Price: 161420.0, Predicted Price: 161420.00
Month: 10.0, Actual Price: 161850.0, Predicted Price: 161850.00
Month: 11.0, Actual Price: 163410.0, Predicted Price: 163410.00
Month: 12.0, Actual Price: 165070.0, Predicted Price: 165070.00
```

**Graph :**


Actual vs Predicted Gold Prices for 2023
Actual Gold Price
Predicted Gold Price
164000
162000
160000
158000
156000
Gold Price (INR)
6
7
8
9
10
11
12
Month