IRCTC Stock Price Prediction

In [1]:
```
!pip install pandas-datareader
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas-datareader in c:\users\user\appdata\roaming\python\python312\site-packages (0.10.0)
Requirement already satisfied: lxml in c:\programdata\anaconda3\lib\site-packages (from pandas-datareader) (5.2.1)
Requirement already satisfied: pandas>=0.23 in c:\programdata\anaconda3\lib\site-packages (from pandas-datareader) (2.2.2)
Requirement already satisfied: requests>=2.19.0 in c:\programdata\anaconda3\lib\site-packages (from pandas-datareader) (2.32.3)
Requirement already satisfied: numpy>=1.26.0 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->pandas-datareader) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->pandas-datareader) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->pandas-datareader) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->pandas-datareader) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.19.0->pandas-datareader) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.19.0->pandas-datareader) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.19.0->pandas-datareader) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.19.0->pandas-datareader) (2024.8.30)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas>=0.23->pandas-datareader) (1.16.0)
```

In [2]:
```
pip show pandas-datareader
```

```
Name: pandas-datareaderNote: you may need to restart the kernel to use updated packages.

Version: 0.10.0
Summary: Data readers extracted from the pandas codebase,should be compatible with recent pandas versions
Home-page: https://github.com/pydata/pandas-datareader
Author: The PyData Development Team
Author-email: pydata@googlegroups.com
License: BSD License
Location: C:\Users\user\AppData\Roaming\Python\Python312\site-packages
Requires: lxml, pandas, requests
Required-by:
```

In [3]:
```
pip install pandas_ta
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas_ta in c:\users\user\appdata\roaming\python\python312\site-packages (0.3.14b0)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (from pandas_ta) (2.2.2)
Requirement already satisfied: numpy>=1.26.0 in c:\programdata\anaconda3\lib\site-packages (from pandas->pandas_ta) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3\lib\site-packages (from pandas->pandas_ta) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas->pandas_ta) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-packages (from pandas->pandas_ta) (2023.3)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas->pandas_ta) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
pip install yfinance
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: yfinance in c:\users\user\appdata\roaming\python\python312\site-packages (0.2.54)
Requirement already satisfied: pandas>=1.3.0 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (2.2.2)
Requirement already satisfied: numpy>=1.16.5 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (1.26.4)
Requirement already satisfied: requests>=2.31 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (2.32.3)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\user\appdata\roaming\python\python312\site-packages (from yfinance) (0.0.11)
Requirement already satisfied: platformdirs>=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (3.10.0)
Requirement already satisfied: pytz>=2022.5 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (2024.1)
Requirement already satisfied: frozendict>=2.3.4 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (2.4.2)
Requirement already satisfied: peewee>=3.16.2 in c:\users\user\appdata\roaming\python\python312\site-packages (from yfinance) (3.17.9)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\programdata\anaconda3\lib\site-packages (from yfinance) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in c:\programdata\anaconda3\lib\site-packages (from beautifulsoup4>=4.11.1->yfinance) (2.5)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2.9.0.post0)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2024.8.30)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [6]:
```
# import necessary libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import datetime
import pandas_ta as ta
import yfinance as yf
from sklearn.preprocessing import StandardScaler
```

In [11]:
```
# import the necessary data


start = datetime.datetime(2020, 1, 1)
end = datetime.datetime(2023, 10, 1)

try:
    irctc_data = yf.download('IRCTC.NS', start=start, end=end)
    if irctc_data is None:
        print("Error: No data returned. Check the ticker symbol or try again later.")
    else:
        print(irctc_data.head())
except Exception as e:
    print(f"An error occurred: {e}")
```

```
YF.download() has changed argument auto_adjust default to True
[*********************100%***********************]  1 of 1 completed
Price            Close        High         Low        Open   Volume
Ticker        IRCTC.NS    IRCTC.NS    IRCTC.NS    IRCTC.NS IRCTC.NS
Date
2020-01-01  181.696182  183.524024  179.858741  181.224803  8593275
2020-01-02  179.012207  181.628896  178.367657  181.628896  3287655
2020-01-03  179.204590  181.532670  178.463832  178.550425  4342850
2020-01-06  172.095276  178.367633  171.258326  178.358003  3923670
2020-01-07  174.586929  177.299826  172.585931  172.585931  3486795
```

In [12]:
```
def get_data(ticker):
    start = datetime.datetime(2020, 1, 1)
    end = datetime.datetime(2023, 10, 1)
    df = yf.download('IRCTC.NS', start=start, end=end)
    df.to_csv(ticker+'.csv')
```

In [13]:
```
get_data('IRCTC.NS')
```

```
[*********************100%***********************]  1 of 1 completed
```

In [14]:
```
pd.read_csv('IRCTC.NS.csv').shape
```

Out[14]:
```
(934, 6)
```

In [18]:
```
def data_preprocessing(ticker):
    # Load the dataset
    dataset = pd.read_csv('{}.csv'.format(ticker))

    # Convert columns to numeric
    dataset['Open'] = pd.to_numeric(dataset['Open'], errors='coerce')
    dataset['High'] = pd.to_numeric(dataset['High'], errors='coerce')
    dataset['Low'] = pd.to_numeric(dataset['Low'], errors='coerce')
    dataset['Close'] = pd.to_numeric(dataset['Close'], errors='coerce')

    # Drop rows with NaN values
    dataset = dataset.dropna()
    dataset = dataset[['Open', 'High', 'Low', 'Close']]

    # Create new features
    dataset['H-L'] = dataset['High'] - dataset['Low']
    dataset['O-C'] = dataset['Close'] - dataset['Open']

    # Moving Averages
    dataset['ma_5'] = dataset['Close'].rolling(window=5).mean()
    dataset['ma_10'] = dataset['Close'].rolling(window=10).mean()

    # Exponential Moving Average
    dataset['EWMA_12'] = dataset['Close'].ewm(span=12).mean()

    # Standard Deviation
    dataset['std_5'] = dataset['Close'].rolling(window=5).std()
    dataset['std_10'] = dataset['Close'].rolling(window=10).std()

    # Technical Indicators using pandas_ta
    dataset['RSI'] = ta.rsi(dataset['Close'], length=14)  # RSI
    dataset['Williams %R'] = ta.willr(dataset['High'], dataset['Low'], dataset['Close'], length=7)  # Williams %R
    dataset['SAR'] = ta.psar(dataset['High'], dataset['Low'], acceleration=0.02, maximum=0.2)['PSARl_0.02_0.2'] #SAR
    dataset['ADX'] = ta.adx(dataset['High'], dataset['Low'], dataset['Close'], length=10)['ADX_10']  # ADX

    # Target variable (Price Rise)
    dataset['Price_Rise'] = np.where(dataset['Close'].shift(-1) > dataset['Close'], 1, 0)

    # Drop rows with NaN values (created by rolling/indicator calculations)
    dataset = dataset.dropna()

    # Features (X) and Target (y)
    X = dataset.iloc[:, 4:-1]  # All columns except the first 4 and the last one
    y = dataset.iloc[:, -1]    # Last column (Price_Rise)

    # Split the data into training and testing sets
    split = int(len(dataset) * 0.8)
    X_train, X_test, y_train, y_test = X[:split], X[split:], y[:split], y[split:]

    # Standardize the features
    sc = StandardScaler()
    X_train = sc.fit_transform(X_train)
    X_test = sc.transform(X_test)

    return X_train, X_test, y_train, y_test

# Example usage
data_preprocessing('IRCTC.NS')
```

Out[18]:
```
(array([[ 0.6055487 ,  1.5279733 , -1.35728293, ...,  1.21492839,
         -1.59786048,  1.82932593],
        [ 0.13029358,  0.61232015, -1.28691397, ...,  1.15269065,
         -1.58704863,  2.07142312],
        [-0.03583366, -0.40829194, -1.22103111, ...,  0.97711405,
         -1.56155384,  2.28072143],
        ...,
        [-0.71567253, -0.1755186 ,  0.45745608, ..., -0.53148713,
          0.59773925, -0.30742654],
        [-0.52391601, -0.04503378,  0.4593196 , ..., -0.44527079,
          0.60146529, -0.48603008],
        [-0.02322545, -0.88663675,  0.44786562, ..., -2.19790811,
          0.60511681, -0.48614388]]),
 array([[ 0.92489359, -0.89642435,  0.38750508, ..., -0.4439243 ,
          0.51698353, -0.39968751],
        [-0.37744014,  0.09196948,  0.38991399, ..., -0.28021628,
          0.52212611, -0.56624044],
        [-0.18302097, -0.28968533,  0.40409504, ..., -0.48960748,
          0.52716584, -0.66857392],
        ...,
        [-0.50377933, -0.09433213,  0.76064484, ...,  0.71626846,
          0.89765578,  1.68502276],
        [ 1.73107526,  0.97163642,  0.7947516 , ...,  0.00576111,
          0.92848169,  1.86783895],
        [-0.06594027, -1.07804609,  0.80772499, ..., -0.43339643,
          0.99250305,  2.03237353]]),
 26     1
 27     0
 28     1
 29     0
 30     0
       ..
 762    1
 763    1
 764    1
 765    0
 766    0
 Name: Price_Rise, Length: 376, dtype: int32,
 770    1
 771    0
 772    1
 773    1
 774    1
       ..
 916    1
 917    0
 918    1
 919    0
 920    0
 Name: Price_Rise, Length: 95, dtype: int32)
```

In [19]:
```
from sklearn import svm
from collections import Counter

def svm_linear(ticker):
    X_train, X_test, y_train, y_test = data_preprocessing(ticker)
    clf = svm.SVC(kernel = 'linear')
    clf.fit(X_train, y_train)
    confidence = clf.score(X_test, y_test)
    print('accuracy:',confidence)
    predictions = clf.predict(X_test)
    print('predicted class counts:',Counter(predictions))

svm_linear('IRCTC.NS')
```

```
accuracy: 0.5157894736842106
predicted class counts: Counter({1: 81, 0: 14})
```

In [21]:
```
def svm_poly(ticker):
    X_train, X_test, y_train, y_test = data_preprocessing(ticker)
    clf = svm.SVC(kernel = 'poly')
    clf.fit(X_train, y_train)
    confidence = clf.score(X_test, y_test)
    print('accuracy:',confidence)
    predictions = clf.predict(X_test)
    print('predicted class counts:',Counter(predictions))

svm_poly('IRCTC.NS')
```

```
accuracy: 0.4842105263157895
predicted class counts: Counter({0: 89, 1: 6})
```

In [22]:
```
def svm_rbf(ticker):
    X_train, X_test, y_train, y_test = data_preprocessing(ticker)
    clf = svm.SVC(kernel = 'rbf')
    clf.fit(X_train, y_train)
    confidence = clf.score(X_test, y_test)
    print('accuracy:',confidence)
    predictions = clf.predict(X_test)
    print('predicted class counts:',Counter(predictions))

svm_rbf('IRCTC.NS')
```

```
accuracy: 0.5157894736842106
predicted class counts: Counter({1: 87, 0: 8})
```

In [ ]: