EX : 3 Linear Regression using Time Series Data

AIM:

To implement the linear regression using the time series data

**CODE:**

import numpy as np

import pandas as pd

from pathlib import Path

%matplotlib inline

# Currency pair exchange rates for CAD/JPY

cad\_jpy\_df = pd.read\_csv(

Path("cad\_jpy.csv"), index\_col="Date", infer\_datetime\_format=True, parse\_dates=True

)

cad\_jpy\_df.head()

# Trim the dataset to begin on January 1st, 1990

cad\_jpy\_df = cad\_jpy\_df.loc["1990-01-01":, :]

cad\_jpy\_df.head()

# Create a series using "Price" percentage returns, drop any nan"s, and check the results:

# (Make sure to multiply the pct\_change() results by 100)

# In this case, you may have to replace inf, -inf values with np.nan"s

cad\_jpy\_df['Return'] = (cad\_jpy\_df[["Price"]].pct\_change() \* 100)

returns = cad\_jpy\_df.replace(-np.inf, np.nan).dropna()

returns.tail()

# Create a lagged return using the shift function

cad\_jpy\_df['Lagged\_Return'] = cad\_jpy\_df["Return"].shift()

cad\_jpy\_df = cad\_jpy\_df.dropna()

cad\_jpy\_df.tail()

# Create a train/test split for the data using 2018-2019 for testing and the rest for training

train = cad\_jpy\_df[:'2018']

test = cad\_jpy\_df['2018':]

# Create a Linear Regression model and fit it to the training data

from sklearn.linear\_model import LinearRegression

# Fit a SKLearn linear regression using just the training set (X\_train, Y\_train):

model = LinearRegression()

model.fit(x\_train, y\_train)

# Make a prediction of "y" values using just the test dataset

predictions = model.predict(x\_test)

# Assemble actual y data (Y\_test) with predicted y data (from just above) into two columns in a dataframe:

results = y\_test.to\_frame()

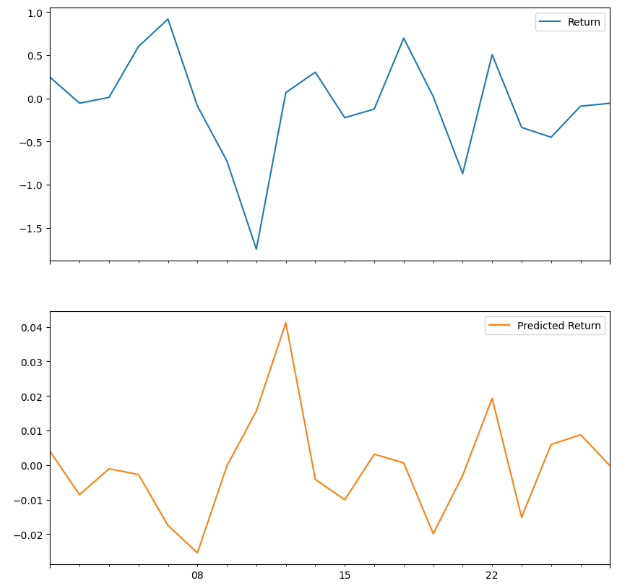
results["Predicted Return"] = predictions

results.head(2)

# Plot the first 20 predictions vs the true values

results[:20].plot(subplots=True, figsize = (10, 10))

output:



**Result:**

**Th**us the implementation of linear regression using time series data has bee done