Strategy Code

##### Importing the Libraries###################################################

from sklearn.svm import SVC

from sklearn.metrics import scorer

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from pandas\_datareader import data as web

################### Downlaoding S&P data ######################################

Df= web.DataReader('SPY', data\_source='google')

Df=Df[['Open','High','Low','Close']]

##########Creating input Parameters#############################################

Df['high']=Df['High'].shift(1)

Df['low']=Df['Low'].shift(1)

Df['close']=Df['Close'].shift(1)

Df['Signal']=1

Df['Signal'][Df['Open'].shift(-1)<Df['Open']]=-1

##############Show the Current DataFrame######################################

print(Df.head())

Df= Df.dropna()

#############################################################################

##############Creation of X and y datasets######################################

X=Df[['Open','high','low','close']]

y=Df['Signal']

######################Test and Train Split######################################

t=.8

split = int(t\*len(Df))

############### Data Pre-Processing Completed###################################

#############################################################################

################# Regression##################################################

reg = SVC(C=1, cache\_size=200, class\_weight=None, coef0=0,

decision\_function\_shape=None, degree=3, gamma='auto', kernel='rbf',

max\_iter=1000, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False)

reg.fit(X[:split],y[:split])

y\_predict =reg.predict(X[split:])

################Prediction####################################################

Df = Df.assign(P\_Trend =pd.Series(np.zeros(len(X))).values)

Df['P\_Trend'][split:]=y\_predict

accuracy = scorer.accuracy\_score(Df['Signal'][split:],Df['P\_Trend'][split:])

############### Strategy Implementation#########################################

Df = Df.assign(Ret =pd.Series(np.zeros(len(X))).values)

Df['Ret']=np.log(Df['Open'].shift(-1)/Df['Open'])

Df = Df.assign(Ret1 =pd.Series(np.zeros(len(X))).values)

Df['Ret1']=Df['P\_Trend']\*Df['Ret']

Df = Df.assign(Cu\_Ret1 =pd.Series(np.zeros(len(X))).values)

Df['Cu\_Ret1']=np.cumsum(Df['Ret1'][split:])

Df = Df.assign(Cu\_Ret =pd.Series(np.zeros(len(X))).values)

Df['Cu\_Ret']=np.cumsum(Df['Ret'][split:])

Std =pd.expanding\_std(Df['Cu\_Ret1'])

Sharpe = (Df['Cu\_Ret1']-Df['Cu\_Ret'])/Std

Sharpe=Sharpe[split:].mean()

print('\n\nAccuracy:',accuracy)

plt.plot(Df['Cu\_Ret1'],color='r',label='Strategy Returns')

plt.plot(Df['Cu\_Ret'],color='g',label='Market Returns')

plt.figtext(0.14,0.7,s='Sharpe ratio: %.2f'%Sharpe)

plt.legend(loc='best')

################################The End ######################################