pip install pandas numpy matplotlib

import pandas as pd

import numpy as npm

import matplotlib.pyplot as plt

# Sample DataFrame with OHLC data

# Replace these lists with your actual OHLC data

data = pd.DataFrame({

'high': [/\* list of high prices \*/],

'low': [/\* list of low prices \*/],

'close': [/\* list of close prices \*/]

})

# Parameters

periods = 30

ema\_period = 34

# Calculate the HLC3

data['hlc3'] = (data['high'] + data['low'] + data['close']) / 3

# Calculate the EMA

data['ema34'] = data['close'].ewm(span=ema\_period, adjust=False).mean()

# Calculate highestHigh and lowestLow over 'periods' periods

data['highestHigh'] = data['high'].rolling(window=periods).max()

data['lowestLow'] = data['low'].rolling(window=periods).min()

# Calculate span

data['span'] = 25 / (data['highestHigh'] - data['lowestLow']) \* data['lowestLow']

# Calculate x1, x2, y1, y2

data['x1\_ema34'] = 0

data['x2\_ema34'] = 1

data['y1\_ema34'] = 0

data['y2\_ema34'] = (data['ema34'].shift(1) - data['ema34']) / data['hlc3'] \* data['span']

# Calculate c\_ema34

data['c\_ema34'] = np.sqrt((data['x2\_ema34'] - data['x1\_ema34'])\*\*2 + (data['y2\_ema34'] - data['y1\_ema34'])\*\*2)

# Calculate emaAngle

pi = np.pi

data['emaAngle\_1'] = 180 \* np.arccos((data['x2\_ema34'] - data['x1\_ema34']) / data['c\_ema34']) / pi

data['emaAngle'] = np.where(data['y2\_ema34'] > 0, -data['emaAngle\_1'], data['emaAngle\_1'])

# Determine chopZoneColor based on emaAngle

if angle >= 5:

BUY

elif -5 <= angle < -1 \* 5:

SELL

pip install pandas numpy matplotlib

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Sample DataFrame with OHLC data

# Replace these lists with your actual OHLC data

data = pd.DataFrame({

'high': [/\* list of high prices \*/],

'low': [/\* list of low prices \*/],

'close': [/\* list of close prices \*/]

})

# Parameters

period\_ = 14

# Calculate True Range (TR) for ATR calculation

def true\_range(high, low, close\_prev):

return np.maximum(np.maximum(high - low, np.abs(high - close\_prev)), np.abs(low - close\_prev))

# Calculate ATR

def average\_true\_range(high, low, close, period):

tr = true\_range(high, low, close.shift(1))

atr = tr.rolling(window=period, min\_periods=1).mean()

return atr

# Calculate Vortex Indicator

def vortex\_indicator(data, period):

data['close\_prev'] = data['close'].shift(1)

# True Range and ATR calculation

data['tr'] = true\_range(data['high'], data['low'], data['close\_prev'])

data['atr'] = average\_true\_range(data['high'], data['low'], data['close'], period)

# VMP and VMM calculation

data['vmp'] = np.abs(data['high'] - data['low'].shift(1))

data['vmm'] = np.abs(data['low'] - data['high'].shift(1))

# Rolling sums for VMP and VMM

data['vmp\_sum'] = data['vmp'].rolling(window=period).sum()

data['vmm\_sum'] = data['vmm'].rolling(window=period).sum()

data['atr\_sum'] = data['atr'].rolling(window=period).sum()

# VI+ and VI- calculation

data['vip'] = data['vmp\_sum'] / data['atr\_sum']

data['vim'] = data['vmm\_sum'] / data['atr\_sum']

return data

# Compute Vortex Indicator

data = vortex\_indicator(data, period\_)

Vortex Indicator Pine Script

//**@version=**5

indicator(title = "Vortex Indicator", shorttitle="VI", format=format.price, precision=4, timeframe="", timeframe\_gaps=true)

period\_ = input.int(14, title="Length", minval=2)

VMP = math.sum( math.abs( high - low[1]), period\_ )

VMM = math.sum( math.abs( low - high[1]), period\_ )

STR = math.sum( ta.atr(1), period\_ )

VIP = VMP / STR

VIM = VMM / STR

plot(VIP, title="VI +", color=#2962FF)

plot(VIM, title="VI -", color=#E91E63)

=====================================================

Chop Zone: Pine Script

//**@version=**5

indicator(title = "Chop Zone", format=format.price, precision=0, timeframe="", timeframe\_gaps=true)

colorTurquoise = #26C6DA

colorDarkGreen = #43A047

colorPaleGreen = #A5D6A7

colorLime = #009688

colorDarkRed = #D50000

colorRed = #E91E63

colorOrange = #FF6D00

colorLightOrange = #FFB74D

colorYellow = #FDD835

source = close

avg = hlc3

pi = math.atan(1) \* 4

periods = 30

highestHigh = ta.highest(periods)

lowestLow = ta.lowest(periods)

span = 25 / (highestHigh - lowestLow) \* lowestLow

ema34 = ta.ema(source, 34)

x1\_ema34 = 0

x2\_ema34 = 1

y1\_ema34 = 0

y2\_ema34 = (ema34[1] - ema34) / avg \* span

c\_ema34 = math.sqrt((x2\_ema34 - x1\_ema34)\*(x2\_ema34 - x1\_ema34) + (y2\_ema34 - y1\_ema34)\*(y2\_ema34 - y1\_ema34))

emaAngle\_1 = math.round(180 \* math.acos((x2\_ema34 - x1\_ema34)/c\_ema34) / pi)

emaAngle = y2\_ema34 > 0? - emaAngle\_1: emaAngle\_1

chopZoneColor = emaAngle >= 5 ? colorTurquoise : emaAngle < 5 and emaAngle >= 3.57 ? colorDarkGreen : emaAngle < 3.57 and emaAngle >= 2.14 ? colorPaleGreen : emaAngle < 2.14 and emaAngle >= .71 ? colorLime : emaAngle <= -1 \* 5 ? colorDarkRed : emaAngle > -1 \* 5 and emaAngle <= -1 \* 3.57 ? colorRed : emaAngle > -1 \* 3.57 and emaAngle <= -1 \* 2.14 ? colorOrange : emaAngle > -1 \* 2.14 and emaAngle <= -1 \* .71 ? colorLightOrange : colorYellow

plot(1, color=chopZoneColor, style=plot.style\_columns)