```
!pip install 'cipher-bt[mplfinance]'
```

```python
import pandas_ta as ta
import numpy as np

from cipher import Cipher, Session, Strategy, quote


class VuManChuCipherBStrategy(Strategy):
    def __init__(
        self,
        wave_channel_length=9,
        wave_average_length=12,
        wave_ma_length=3,
        wave_k=0.015,
        mfi_length=60,
        atr_length=24,
        stop_loss_k=2,
        take_profit_k=4,
        slow_ema_length=200,
        fast_ema_length=50,
    ):
        self.wave_channel_length = wave_channel_length
        self.wave_average_length = wave_average_length
        self.wave_ma_length = wave_ma_length
        self.wave_k = wave_k
        self.mfi_length = mfi_length
        self.atr_length = atr_length
        self.stop_loss_k = stop_loss_k
        self.take_profit_k = take_profit_k
        self.slow_ema_length = slow_ema_length
        self.fast_ema_length = fast_ema_length

    def compose(self):
        df = self.datas.df

        hlc3 = (df["high"] + df["low"] + df["close"]) / 3
        esa = ta.ema(hlc3, length=self.wave_channel_length)
        de = ta.ema(abs(hlc3 - esa), length=self.wave_channel_length)
        ci = (hlc3 - esa) / (de * self.wave_k)

        df["wt1"] = ta.ema(ci, length=self.wave_average_length)
        df["wt2"] = ta.sma(df["wt1"], length=self.wave_ma_length)
        df["mfi"] = df.ta.mfi(length=self.mfi_length) - 50
        df["atr"] = df.ta.atr(length=self.atr_length)
        df["fast_ema"] = df.ta.ema(length=self.fast_ema_length)
        df["slow_ema"] = df.ta.ema(length=self.slow_ema_length)

        difference = df["wt2"] - df["wt1"]
        cross = np.sign(difference.shift(1)) != np.sign(difference)

        df["long_entry"] = (
            (df["close"] > df["slow_ema"])
            & (df["low"] < df["fast_ema"])
            & (df["mfi"] > 0)
            & (df["wt1"] < 0)
            & cross
            & (difference < 0)
        )
        df["short_entry"] = (
            (df["close"] < df["slow_ema"])
            & (df["low"] > df["fast_ema"])
            & (df["mfi"] < 0)
            & (df["wt1"] > 0)
            & cross
            & (difference > 0)
        )

        df["entry"] = df["long_entry"] | df["short_entry"]

        return df

    def on_entry(self, row: dict, session: Session):
        if self.wallet.base != 0:
            # keep only one position open
            return

        if row["long_entry"]:
            session.position += quote(100)
            session.stop_loss = row["close"] - row["atr"] * self.stop_loss_k
```

```
            session.take_profit = row["close"] + row["atr"] * self.take_profit_k
        else:
            session.position -= quote(100)
            session.stop_loss = row["close"] + row["atr"] * self.stop_loss_k
            session.take_profit = row["close"] - row["atr"] * self.take_profit_k

cipher = Cipher()
cipher.add_source("gateio_spot_ohlc", symbol="DOGE_USDT", interval="1h")
cipher.set_strategy(VuManChuCipherBStrategy())
cipher.run(start_ts="2020-01-01", stop_ts="2020-04-01")
```

```
cipher.set_commission("0.0025")
cipher.stats
```

```
----------------  ------------------------------  -----
start             2020-01-09 07:00
stop              2020-03-31 23:00
period            82d 16h
trades            13
longs             9                               69.2%
shorts            4                               30.8%
period median     1d 10h
period max        6d 2h
success           6                               46.2%
success median    3.354757999781781716394474039
success max       7.285257787325456498388829202
success row       2
failure           7                               53.8%
failure median    3.224282397636133389615871688
failure max       7.162470167064439140811455844
failure row       3
spf               0.8571428571428571
pnl               -3.814852202301978261666303180
commission        6.470194899196096791199981580
balance min       -6.977559221631054
balance max       10.815903701787263
balance drawdown  17.793462923418318
romad             -0.21439627680799436
----------------  ------------------------------  -----
```

```
cipher.plot(
    rows=[
        ["ohlc", "slow_ema", "fast_ema", "sessions"],
        ["wt1", "wt2"],
    ]
)
```

```
WARNING:cipher.plotters.base:Only a part of the dataframe is shown on the plot, use start/limit plot arguments to paginat
```


```
cipher.plot(
    rows=[
```

```
        ["ohlc"],
        ["position", "balance"],
    ]
)
```

```
WARNING:cipher.plotters.base:Only a part of the dataframe is shown on the plot, use start/limit plot arguments to paginat
```

Colab paid products - Cancel contracts here

✓ 1s completed at 6:12 PM