


```

In [1]: import sqlalchemy
        from sqlalchemy import create_engine
        import pymysql
        import os
        import pandas as pd
        import matplotlib.pyplot as plt
        from finta import TA

        from datetime import datetime

        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import ray
        import ray.rllib.agents.ppo as ppo
        import tensortrade.env.default as default
        from gym.spaces import Discrete
        from ray import tune
        from ray.tune.registry import register_env
        from sympfit import parameters, variables, sin, cos, Fit
        from tensortrade.env.default.actions import TensorTradeActionScheme
        from tensortrade.env.default.rewards import TensorTradeRewardScheme
        from tensortrade.env.generic import Renderer
        from tensortrade.feed.core import DataFeed, Stream
        from tensortrade.oms.exchanges import Exchange
        from tensortrade.oms.exchanges import ExchangeOptions
        from tensortrade.oms.instruments import Instrument
        from tensortrade.oms.orders import proportion_order
        from tensortrade.oms.services.execution.simulated import execute_order
        from tensortrade.oms.wallets import Wallet, Portfolio

        from tensortrade.env.default.actions import BSH
        from tensortrade.env.default.rewards import PBR

        import tensortrade.stochastic as sp

        import tensortrade.env.default.rewards as rewards
        from tensortrade.env.default.actions import ManagedRiskOrders
        from tensortrade.env.default.rewards import SimpleProfit

```

WARNING:tensorflow:From /root/anaconda3/lib/python3.8/site-packages/tensorflow/python/compat/v2_compat.py:9

6: `disable_resource_variables` (from `tensorflow.python.ops.variable_scope`) is deprecated and will be removed in a future version.

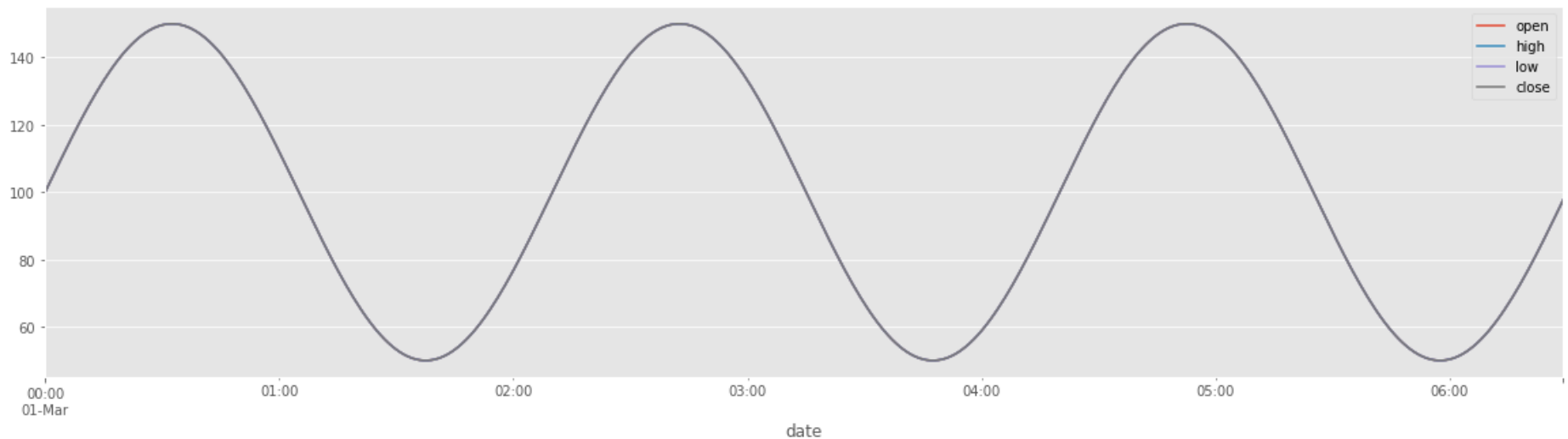
Instructions for updating:

non-resource variables are not supported in the long term

```
In [2]: def get_data():
        x = np.arange(0, 2*np.pi, 2*np.pi / 390)
        y = 50*np.sin(3*x) + 100
        data = pd.DataFrame(data=y, columns=["open"])
        data["date"] = pd.date_range(start='03/01/2021', freq='MIN', periods=390)
        data["high"] = data["open"]
        data["low"] = data["open"]
        data["close"] = data["open"]
        data["volume"] = 10
        data = data[["date", "open", "high", "low", "close", "volume"]]
        return data
```

```
In [3]: data = get_data()
        data.plot(x="date", y=["open", "high", "low", "close"], figsize=(20,5))
```

Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x7fa54d16dc70>



```

In [4]: USD = Instrument("USD", 2, "U.S. Dollar")
        TSLA = Instrument("TSLA", 8, "Tesla, Inc.")

def create_env(config):

    # Sine wave
    data = get_data()

    p = Stream.source(list(data["close"]), dtype="float").rename("USD-TSLA")

    simulated_exchange = Exchange(
        "simulated_exchange",
        service=execute_order,
        options=ExchangeOptions(commission=0.03)
    )( p )

    cash = Wallet(simulated_exchange, 10000 * USD)
    asset = Wallet(simulated_exchange, 0 * TSLA)

    portfolio = Portfolio(USD, [
        cash,
        asset
    ])

    feed = DataFeed([
        Stream.source(list(data["open"]), dtype="float").rename("open"),
        Stream.source(list(data["high"]), dtype="float").rename("high"),
        Stream.source(list(data["low"]), dtype="float").rename("low"),
        Stream.source(list(data["close"]), dtype="float").rename("close"),
        Stream.source(list(data["volume"]), dtype="float").rename("volume"),
        p.ewm(span=10).mean().rename("fast"),
        p.ewm(span=50).mean().rename("medium"),
        p.ewm(span=100).mean().rename("slow"),
        p.log().diff().fillna(0).rename("lr")
    ])

    # Position-based returns reward scheme
    reward_scheme = PBR(price=p)

    # Buy, sell, or hold action scheme
    action_scheme = BSH(

```

```

        cash=cash,
        asset=asset
    ).attach(reward_scheme)

rendered_feed = DataFeed([
    Stream.source(list(data["date"])), dtype="datetime64").rename("date"),
    Stream.source(list(data["open"])), dtype="float").rename("open"),
    Stream.source(list(data["high"])), dtype="float").rename("high"),
    Stream.source(list(data["low"])), dtype="float").rename("low"),
    Stream.source(list(data["close"])), dtype="float").rename("close"),
    Stream.source(list(data["volume"])), dtype="float").rename("volume"),
])

environment = default.create(
    feed=feed,
    portfolio=portfolio,
    action_scheme=action_scheme,
    reward_scheme=reward_scheme,
    rendered_feed=rendered_feed,
    renderer=default.renderers.PlotlyTradingChart(display=True, auto_open_html=False, save_format="png"),
    window_size=25,
    max_allowed_loss=0.6
)
return environment

register_env("TradingEnv", create_env)

```

```
In [5]: analysis = tune.run(
    "PPO",
    stop={
        "info/num_steps_trained": 200000,
    },
    config={
        "env": "TradingEnv",
        "env_config": {
            "window_size": 25
        },
        "framework": "torch",
        "ignore_worker_failures": True,
        "num_workers": os.cpu_count() - 1,
        "num_gpus": 1,
        "clip_rewards": True,
        "lr": 8e-6,
        "gamma": 0,
        "observation_filter": "MeanStdFilter",
        "lambda": 0.72,
        "vf_loss_coeff": 0.5,
        "entropy_coeff": 0.01
    },
    checkpoint_at_end=True
)
```

```
learn_throughput: 561.696
learn_time_ms: 7477.356
sample_throughput: 1451.287
sample_time_ms: 2893.982
update_time_ms: 2.177
timestamp: 1616765089
timesteps_since_restore: 0
timesteps_total: 4200
training_iteration: 1
trial_id: 98fd3_00000
```

== Status ==

Memory usage on this node: 14.2/62.7 GiB

Using FIFO scheduling algorithm.

Resources requested: 4/4 CPUs, 1/1 GPUs, 0.0/31.25 GiB heap, 0.0/10.74 GiB objects (0/1.0 accelerator_type:X)

Result logdir: /root/ray_results/PPO

Number of trials: 1/1 (1 RUNNING)

```

In [6]: checkpoints = analysis.get_trial_checkpoints_paths(
        trial=analysis.get_best_trial(metric="episode_reward_mean", mode="max"),
        metric="episode_reward_mean"
    )
    checkpoint_path = checkpoints[0][0]

    # Restore agent
    agent = ppo.PPOTrainer(
        env="TradingEnv",
        config={
            "env_config": {
                "window_size": 25
            },
            "framework": "torch",
            "ignore_worker_failures": True,
            "num_workers": 1,
            "num_gpus": 0,
            "clip_rewards": True,
            "lr": 8e-6,
            "gamma": 0,
            "observation_filter": "MeanStdFilter",
            "lambda": 0.72,
            "vf_loss_coeff": 0.5,
            "entropy_coeff": 0.01
        }
    )
    agent.restore(checkpoint_path)

```

```

2021-03-26 13:32:50,406 INFO trainer.py:641 -- Current log_level is WARN. For more information, set 'log_level': 'INFO' / 'DEBUG' or use the -v and -vv flags.
(pid=1052160) WARNING:tensorflow:From /root/anaconda3/lib/python3.8/site-packages/tensorflow/python/compat/v2_compat.py:96: disable_resource_variables (from tensorflow.python.ops.variable_scope) is deprecated and will be removed in a future version.
(pid=1052160) Instructions for updating:
(pid=1052160) non-resource variables are not supported in the long term
2021-03-26 13:32:55,776 INFO trainable.py:371 -- Restored on 192.168.0.18 from checkpoint: /root/ray_results/PPO/PPO_TradingEnv_98fd3_00000_0_2021-03-26_13-24-31/checkpoint_48/checkpoint-48
2021-03-26 13:32:55,784 INFO trainable.py:379 -- Current state after restoring: {'_iteration': 48, '_timesteps_total': None, '_time_total': 489.95394706726074, '_episodes_total': 615}

```



```

In [7]: env = create_env({
        "window_size": 25
    })

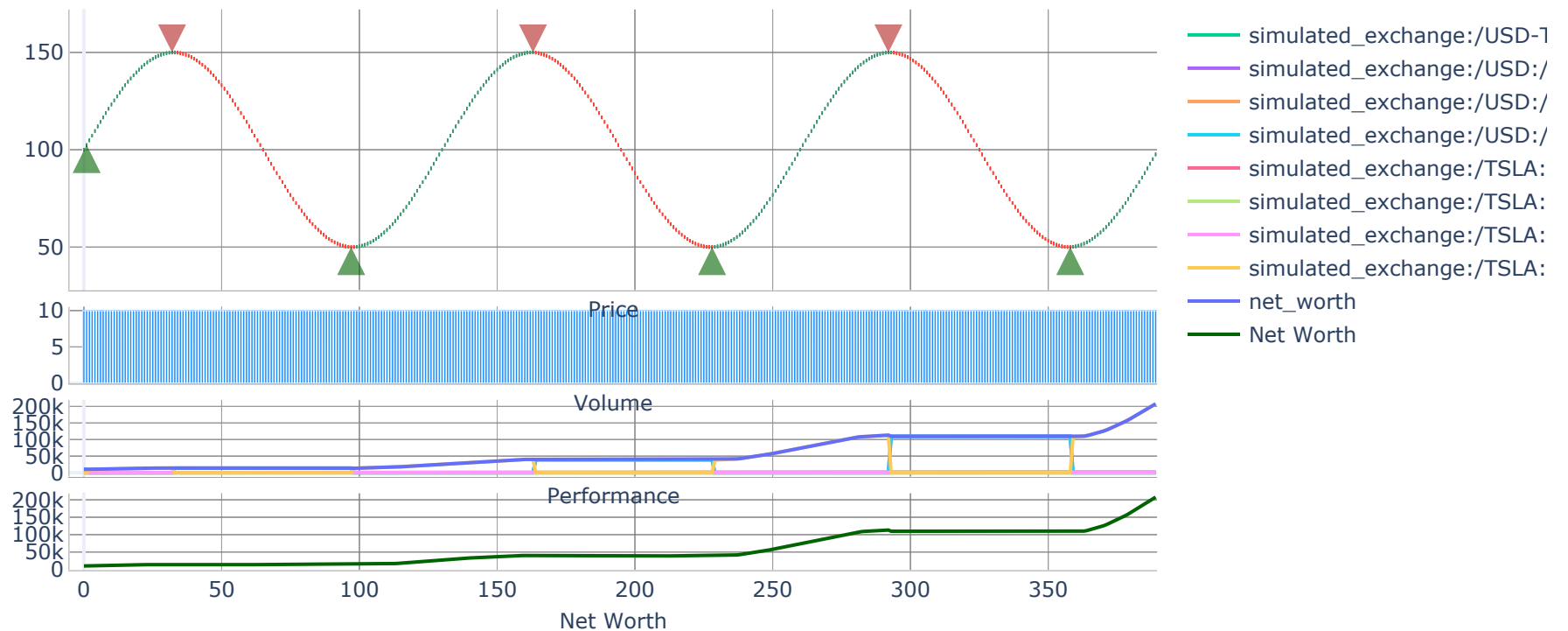
episode_reward = 0
done = False
obs = env.reset()

while not done:
    action = agent.compute_action(obs)
    obs, reward, done, info = env.step(action)
    episode_reward += reward

env.render()

```

[2021-03-26 13:32:56 PM] Step: 390/



[2021-03-26 13:32:56 PM] Step: 390/

