In [1]:
```
import pandas as pd
import numpy as np
import datetime
#from mlxtend.feature_selection import SequentialFeatureSelector
from sklearn.feature_selection import mutual_info_classif
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import SGDRegressor
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import mean_squared_error
from sklearn.neighbors import KNeighborsClassifier
from skfeature.function.similarity_based import fisher_score
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import accuracy_score
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
import pyfolio as pf
from finrl.plot import backtest_plot
from sklearn.svm import SVC
```

In [2]:
```
ZTS = pd.read_csv("phase3_data/zts.csv")
ZTS = ZTS.loc[:, 'Report Date': 's_diff']
ZTS['Report Date'] = pd.to_datetime(ZTS['Report Date'])
all_date_column = ZTS['Report Date']
```

In [3]:
```
ZTS['Return'] = ZTS['Stock price'].pct_change()
ZTS['Stock price'] = ZTS['Stock price'].shift(-1)
ZTS['Y_boolean'] = ZTS['Y_boolean'].shift(-1)
ZTS['Return'] = ZTS['Return'].shift(-1)
ZTS = ZTS.dropna()
```

In [4]:
```
# function to select data given date window
def window(start, end, df, all_date_column):
    '''
    Given a start and end date, return df with data only
    from that period

    Inputs:
        start/end: start and end dates
            ex: start = '2022-01-23' ('YYYY-MM-DD')
        df: pd dataframe
        date: pd series with dates of all possible dates in data

    Returns: pd dataframe
    '''

    date_column = all_date_column[all_date_column.between(start, end, inclusive=
    df = pd.concat([date_column, df], axis = 1, join="inner")
    df = df.iloc[:, 1:]

    return df
```

In [5]:
```
def feature_selection_FS(training_dataset, testing_dataset, bar):
    """
    find the important feature using Fishers' score
    """

    X = training_dataset.iloc[:, 3:-1]
    Y_price = training_dataset['Stock price']
    Y_dummy = training_dataset['Y_boolean']
    date_train = training_dataset['Report Date']
    date_test = testing_dataset['Report Date']

    ranks = fisher_score.fisher_score(X.to_numpy(), Y_dummy)
    feat_importances = pd.Series(ranks, training_dataset.columns[3:-1])
#     feat_importances.plot(kind='barh', color ="teal")
#     plt.show()

    feat_selected = feat_importances.to_frame('importance')
    # select importance > than bar
    feat_selected = feat_selected[feat_selected['importance'] > bar]
    select_feature_lst = ['Report Date', 'Stock price', 'Y_boolean'] + list(feat

    return training_dataset[select_feature_lst], testing_dataset[select_feature_
```

In [6]:
```
def feature_selection_IG(training_dataset, testing_dataset, bar):
    """
    find the important feature using IG method
    """

    X = training_dataset.iloc[:, 3:-1]
    Y_price = training_dataset['Stock price']
    Y_dummy = training_dataset['Y_boolean']
    date_train = training_dataset['Report Date']
    date_test = testing_dataset['Report Date']
    importances = mutual_info_classif(X, Y_dummy, random_state=1241)
    feat_importances = pd.Series(importances, training_dataset.columns[3:-1])
#     feat_importances.plot(kind='barh', color ="teal")
#     plt.show()

    feat_selected = feat_importances.to_frame('importance')
    # select importance > than bar
    feat_selected = feat_selected[feat_selected['importance'] > bar]
    select_feature_lst = ['Report Date', 'Stock price', 'Y_boolean'] + list(feat

    return training_dataset[select_feature_lst], testing_dataset[select_feature_
```

In [7]:
```
def test_feature_select(training_dataset, validate_dataset):
    """
    check the performance of the current selected: use only training dataset acc
    """
    # run baseline logit
    logit = LogisticRegression(solver = 'lbfgs', random_state=1241)
    X = training_dataset.iloc[:, 3:-1]
    Y_dummy = training_dataset['Y_boolean']
```

```
    logit.fit(X, Y_dummy)
    Y_dummy_validate = validate_dataset['Y_boolean']
    X_validate = validate_dataset.iloc[:, 3:-1]
    Y_pred = logit.predict(X_validate)
    # pred = lr.predict(testing_dataset)
#     score = logit.score(testing_dataset, Y_true)

    score = accuracy_score(Y_dummy_validate, Y_pred)
    return score
```

In [8]:

```
# collect all date
date_starts = []
date = datetime.datetime(2020, 3, 2)
date_starts.append(date)

for idx in range(660):#range(722):
    date += datetime.timedelta(days=1)
    date_starts.append(date)
```

In [9]:

```
features_selected = []

for train_date_start in date_starts:
    print(train_date_start)
    train_date_end = train_date_start + datetime.timedelta(days=90)
    validate_date_start = train_date_end + datetime.timedelta(days=1)
    validate_date_end = validate_date_start + datetime.timedelta(days=30)
    test_date_start = validate_date_end + datetime.timedelta(days=1)
    test_date_end = test_date_start

    train_data = window(train_date_start, train_date_end, ZTS, all_date_column)
    validate_data = window(validate_date_start, validate_date_end, ZTS, all_date
    test_data = window(test_date_start, test_date_end, ZTS, all_date_column)

    bars_FS = range(0, 11)
    bars_IG = np.arange(0, 0.03, 0.005)#np.arange(0, 0.1005, 0.005)

    best_accuracy = 0
    best_bar = []
    best_features = []

    for bar_FS in bars_FS:
        for bar_IG in bars_IG:

            try:
                train_ZTS_selected, validate_ZTS_selected = feature_selection_FS
                train_ZTS_selected, validate_ZTS_selected = feature_selection_IG
                score = test_feature_select(train_ZTS_selected, validate_ZTS_sel
                if score > best_accuracy:
                    best_accuracy = score
                    best_bar = [bar_FS, bar_IG]
                    best_features = train_ZTS_selected.iloc[:,3:].columns
            except:
                continue
    features_selected.append(best_features)
```

```
2020-03-02 00:00:00
```

```
2020-03-03 00:00:00
2020-03-04 00:00:00
2020-03-05 00:00:00
2020-03-06 00:00:00
2020-03-07 00:00:00
2020-03-08 00:00:00
2020-03-09 00:00:00
2020-03-10 00:00:00
2020-03-11 00:00:00
2020-03-12 00:00:00
2020-03-13 00:00:00
2020-03-14 00:00:00
2020-03-15 00:00:00
2020-03-16 00:00:00
2020-03-17 00:00:00
2020-03-18 00:00:00
2020-03-19 00:00:00
2020-03-20 00:00:00
2020-03-21 00:00:00
2020-03-22 00:00:00
2020-03-23 00:00:00
2020-03-24 00:00:00
2020-03-25 00:00:00
2020-03-26 00:00:00
2020-03-27 00:00:00
2020-03-28 00:00:00
2020-03-29 00:00:00
2020-03-30 00:00:00
2020-03-31 00:00:00
2020-04-01 00:00:00
2020-04-02 00:00:00
2020-04-03 00:00:00
2020-04-04 00:00:00
2020-04-05 00:00:00
2020-04-06 00:00:00
2020-04-07 00:00:00
2020-04-08 00:00:00
2020-04-09 00:00:00
2020-04-10 00:00:00
2020-04-11 00:00:00
2020-04-12 00:00:00
2020-04-13 00:00:00
2020-04-14 00:00:00
2020-04-15 00:00:00
2020-04-16 00:00:00
2020-04-17 00:00:00
2020-04-18 00:00:00
2020-04-19 00:00:00
2020-04-20 00:00:00
2020-04-21 00:00:00
2020-04-22 00:00:00
2020-04-23 00:00:00
2020-04-24 00:00:00
2020-04-25 00:00:00
2020-04-26 00:00:00
2020-04-27 00:00:00
2020-04-28 00:00:00
2020-04-29 00:00:00
2020-04-30 00:00:00
2020-05-01 00:00:00
```

```
2020-05-02 00:00:00
2020-05-03 00:00:00
2020-05-04 00:00:00
2020-05-05 00:00:00
2020-05-06 00:00:00
2020-05-07 00:00:00
2020-05-08 00:00:00
2020-05-09 00:00:00
2020-05-10 00:00:00
2020-05-11 00:00:00
2020-05-12 00:00:00
2020-05-13 00:00:00
2020-05-14 00:00:00
2020-05-15 00:00:00
2020-05-16 00:00:00
2020-05-17 00:00:00
2020-05-18 00:00:00
2020-05-19 00:00:00
2020-05-20 00:00:00
2020-05-21 00:00:00
2020-05-22 00:00:00
2020-05-23 00:00:00
2020-05-24 00:00:00
2020-05-25 00:00:00
2020-05-26 00:00:00
2020-05-27 00:00:00
2020-05-28 00:00:00
2020-05-29 00:00:00
2020-05-30 00:00:00
2020-05-31 00:00:00
2020-06-01 00:00:00
2020-06-02 00:00:00
2020-06-03 00:00:00
2020-06-04 00:00:00
2020-06-05 00:00:00
2020-06-06 00:00:00
2020-06-07 00:00:00
2020-06-08 00:00:00
2020-06-09 00:00:00
2020-06-10 00:00:00
2020-06-11 00:00:00
2020-06-12 00:00:00
2020-06-13 00:00:00
2020-06-14 00:00:00
2020-06-15 00:00:00
2020-06-16 00:00:00
2020-06-17 00:00:00
2020-06-18 00:00:00
2020-06-19 00:00:00
2020-06-20 00:00:00
2020-06-21 00:00:00
2020-06-22 00:00:00
2020-06-23 00:00:00
2020-06-24 00:00:00
2020-06-25 00:00:00
2020-06-26 00:00:00
2020-06-27 00:00:00
2020-06-28 00:00:00
2020-06-29 00:00:00
2020-06-30 00:00:00
```

```
2020-07-01 00:00:00
2020-07-02 00:00:00
2020-07-03 00:00:00
2020-07-04 00:00:00
2020-07-05 00:00:00
2020-07-06 00:00:00
2020-07-07 00:00:00
2020-07-08 00:00:00
2020-07-09 00:00:00
2020-07-10 00:00:00
2020-07-11 00:00:00
2020-07-12 00:00:00
2020-07-13 00:00:00
2020-07-14 00:00:00
2020-07-15 00:00:00
2020-07-16 00:00:00
2020-07-17 00:00:00
2020-07-18 00:00:00
2020-07-19 00:00:00
2020-07-20 00:00:00
2020-07-21 00:00:00
2020-07-22 00:00:00
2020-07-23 00:00:00
2020-07-24 00:00:00
2020-07-25 00:00:00
2020-07-26 00:00:00
2020-07-27 00:00:00
2020-07-28 00:00:00
2020-07-29 00:00:00
2020-07-30 00:00:00
2020-07-31 00:00:00
2020-08-01 00:00:00
2020-08-02 00:00:00
2020-08-03 00:00:00
2020-08-04 00:00:00
2020-08-05 00:00:00
2020-08-06 00:00:00
2020-08-07 00:00:00
2020-08-08 00:00:00
2020-08-09 00:00:00
2020-08-10 00:00:00
2020-08-11 00:00:00
2020-08-12 00:00:00
2020-08-13 00:00:00
2020-08-14 00:00:00
2020-08-15 00:00:00
2020-08-16 00:00:00
2020-08-17 00:00:00
2020-08-18 00:00:00
2020-08-19 00:00:00
2020-08-20 00:00:00
2020-08-21 00:00:00
2020-08-22 00:00:00
2020-08-23 00:00:00
2020-08-24 00:00:00
2020-08-25 00:00:00
2020-08-26 00:00:00
2020-08-27 00:00:00
2020-08-28 00:00:00
2020-08-29 00:00:00
```

```
2020-08-30 00:00:00
2020-08-31 00:00:00
2020-09-01 00:00:00
2020-09-02 00:00:00
2020-09-03 00:00:00
2020-09-04 00:00:00
2020-09-05 00:00:00
2020-09-06 00:00:00
2020-09-07 00:00:00
2020-09-08 00:00:00
2020-09-09 00:00:00
2020-09-10 00:00:00
2020-09-11 00:00:00
2020-09-12 00:00:00
2020-09-13 00:00:00
2020-09-14 00:00:00
2020-09-15 00:00:00
2020-09-16 00:00:00
2020-09-17 00:00:00
2020-09-18 00:00:00
2020-09-19 00:00:00
2020-09-20 00:00:00
2020-09-21 00:00:00
2020-09-22 00:00:00
2020-09-23 00:00:00
2020-09-24 00:00:00
2020-09-25 00:00:00
2020-09-26 00:00:00
2020-09-27 00:00:00
2020-09-28 00:00:00
2020-09-29 00:00:00
2020-09-30 00:00:00
2020-10-01 00:00:00
2020-10-02 00:00:00
2020-10-03 00:00:00
2020-10-04 00:00:00
2020-10-05 00:00:00
2020-10-06 00:00:00
2020-10-07 00:00:00
2020-10-08 00:00:00
2020-10-09 00:00:00
2020-10-10 00:00:00
2020-10-11 00:00:00
2020-10-12 00:00:00
2020-10-13 00:00:00
2020-10-14 00:00:00
2020-10-15 00:00:00
2020-10-16 00:00:00
2020-10-17 00:00:00
2020-10-18 00:00:00
2020-10-19 00:00:00
2020-10-20 00:00:00
2020-10-21 00:00:00
2020-10-22 00:00:00
2020-10-23 00:00:00
2020-10-24 00:00:00
2020-10-25 00:00:00
2020-10-26 00:00:00
2020-10-27 00:00:00
2020-10-28 00:00:00
```

```
2020-10-29 00:00:00
2020-10-30 00:00:00
2020-10-31 00:00:00
2020-11-01 00:00:00
2020-11-02 00:00:00
2020-11-03 00:00:00
2020-11-04 00:00:00
2020-11-05 00:00:00
2020-11-06 00:00:00
2020-11-07 00:00:00
2020-11-08 00:00:00
2020-11-09 00:00:00
2020-11-10 00:00:00
2020-11-11 00:00:00
2020-11-12 00:00:00
2020-11-13 00:00:00
2020-11-14 00:00:00
2020-11-15 00:00:00
2020-11-16 00:00:00
2020-11-17 00:00:00
2020-11-18 00:00:00
2020-11-19 00:00:00
2020-11-20 00:00:00
2020-11-21 00:00:00
2020-11-22 00:00:00
2020-11-23 00:00:00
2020-11-24 00:00:00
2020-11-25 00:00:00
2020-11-26 00:00:00
2020-11-27 00:00:00
2020-11-28 00:00:00
2020-11-29 00:00:00
2020-11-30 00:00:00
2020-12-01 00:00:00
2020-12-02 00:00:00
2020-12-03 00:00:00
2020-12-04 00:00:00
2020-12-05 00:00:00
2020-12-06 00:00:00
2020-12-07 00:00:00
2020-12-08 00:00:00
2020-12-09 00:00:00
2020-12-10 00:00:00
2020-12-11 00:00:00
2020-12-12 00:00:00
2020-12-13 00:00:00
2020-12-14 00:00:00
2020-12-15 00:00:00
2020-12-16 00:00:00
2020-12-17 00:00:00
2020-12-18 00:00:00
2020-12-19 00:00:00
2020-12-20 00:00:00
2020-12-21 00:00:00
2020-12-22 00:00:00
2020-12-23 00:00:00
2020-12-24 00:00:00
2020-12-25 00:00:00
2020-12-26 00:00:00
2020-12-27 00:00:00
```

```
2020-12-28 00:00:00
2020-12-29 00:00:00
2020-12-30 00:00:00
2020-12-31 00:00:00
2021-01-01 00:00:00
2021-01-02 00:00:00
2021-01-03 00:00:00
2021-01-04 00:00:00
2021-01-05 00:00:00
2021-01-06 00:00:00
2021-01-07 00:00:00
2021-01-08 00:00:00
2021-01-09 00:00:00
2021-01-10 00:00:00
2021-01-11 00:00:00
2021-01-12 00:00:00
2021-01-13 00:00:00
2021-01-14 00:00:00
2021-01-15 00:00:00
2021-01-16 00:00:00
2021-01-17 00:00:00
2021-01-18 00:00:00
2021-01-19 00:00:00
2021-01-20 00:00:00
2021-01-21 00:00:00
2021-01-22 00:00:00
2021-01-23 00:00:00
2021-01-24 00:00:00
2021-01-25 00:00:00
2021-01-26 00:00:00
2021-01-27 00:00:00
2021-01-28 00:00:00
2021-01-29 00:00:00
2021-01-30 00:00:00
2021-01-31 00:00:00
2021-02-01 00:00:00
2021-02-02 00:00:00
2021-02-03 00:00:00
2021-02-04 00:00:00
2021-02-05 00:00:00
2021-02-06 00:00:00
2021-02-07 00:00:00
2021-02-08 00:00:00
2021-02-09 00:00:00
2021-02-10 00:00:00
2021-02-11 00:00:00
2021-02-12 00:00:00
2021-02-13 00:00:00
2021-02-14 00:00:00
2021-02-15 00:00:00
2021-02-16 00:00:00
2021-02-17 00:00:00
2021-02-18 00:00:00
2021-02-19 00:00:00
2021-02-20 00:00:00
2021-02-21 00:00:00
2021-02-22 00:00:00
2021-02-23 00:00:00
2021-02-24 00:00:00
2021-02-25 00:00:00
```

```
2021-02-26 00:00:00
2021-02-27 00:00:00
2021-02-28 00:00:00
2021-03-01 00:00:00
2021-03-02 00:00:00
2021-03-03 00:00:00
2021-03-04 00:00:00
2021-03-05 00:00:00
2021-03-06 00:00:00
2021-03-07 00:00:00
2021-03-08 00:00:00
2021-03-09 00:00:00
2021-03-10 00:00:00
2021-03-11 00:00:00
2021-03-12 00:00:00
2021-03-13 00:00:00
2021-03-14 00:00:00
2021-03-15 00:00:00
2021-03-16 00:00:00
2021-03-17 00:00:00
2021-03-18 00:00:00
2021-03-19 00:00:00
2021-03-20 00:00:00
2021-03-21 00:00:00
2021-03-22 00:00:00
2021-03-23 00:00:00
2021-03-24 00:00:00
2021-03-25 00:00:00
2021-03-26 00:00:00
2021-03-27 00:00:00
2021-03-28 00:00:00
2021-03-29 00:00:00
2021-03-30 00:00:00
2021-03-31 00:00:00
2021-04-01 00:00:00
2021-04-02 00:00:00
2021-04-03 00:00:00
2021-04-04 00:00:00
2021-04-05 00:00:00
2021-04-06 00:00:00
2021-04-07 00:00:00
2021-04-08 00:00:00
2021-04-09 00:00:00
2021-04-10 00:00:00
2021-04-11 00:00:00
2021-04-12 00:00:00
2021-04-13 00:00:00
2021-04-14 00:00:00
2021-04-15 00:00:00
2021-04-16 00:00:00
2021-04-17 00:00:00
2021-04-18 00:00:00
2021-04-19 00:00:00
2021-04-20 00:00:00
2021-04-21 00:00:00
2021-04-22 00:00:00
2021-04-23 00:00:00
2021-04-24 00:00:00
2021-04-25 00:00:00
2021-04-26 00:00:00
```

```
2021-04-27 00:00:00
2021-04-28 00:00:00
2021-04-29 00:00:00
2021-04-30 00:00:00
2021-05-01 00:00:00
2021-05-02 00:00:00
2021-05-03 00:00:00
2021-05-04 00:00:00
2021-05-05 00:00:00
2021-05-06 00:00:00
2021-05-07 00:00:00
2021-05-08 00:00:00
2021-05-09 00:00:00
2021-05-10 00:00:00
2021-05-11 00:00:00
2021-05-12 00:00:00
2021-05-13 00:00:00
2021-05-14 00:00:00
2021-05-15 00:00:00
2021-05-16 00:00:00
2021-05-17 00:00:00
2021-05-18 00:00:00
2021-05-19 00:00:00
2021-05-20 00:00:00
2021-05-21 00:00:00
2021-05-22 00:00:00
2021-05-23 00:00:00
2021-05-24 00:00:00
2021-05-25 00:00:00
2021-05-26 00:00:00
2021-05-27 00:00:00
2021-05-28 00:00:00
2021-05-29 00:00:00
2021-05-30 00:00:00
2021-05-31 00:00:00
2021-06-01 00:00:00
2021-06-02 00:00:00
2021-06-03 00:00:00
2021-06-04 00:00:00
2021-06-05 00:00:00
2021-06-06 00:00:00
2021-06-07 00:00:00
2021-06-08 00:00:00
2021-06-09 00:00:00
2021-06-10 00:00:00
2021-06-11 00:00:00
2021-06-12 00:00:00
2021-06-13 00:00:00
2021-06-14 00:00:00
2021-06-15 00:00:00
2021-06-16 00:00:00
2021-06-17 00:00:00
2021-06-18 00:00:00
2021-06-19 00:00:00
2021-06-20 00:00:00
2021-06-21 00:00:00
2021-06-22 00:00:00
2021-06-23 00:00:00
2021-06-24 00:00:00
2021-06-25 00:00:00
```

```
2021-06-26 00:00:00
2021-06-27 00:00:00
2021-06-28 00:00:00
2021-06-29 00:00:00
2021-06-30 00:00:00
2021-07-01 00:00:00
2021-07-02 00:00:00
2021-07-03 00:00:00
2021-07-04 00:00:00
2021-07-05 00:00:00
2021-07-06 00:00:00
2021-07-07 00:00:00
2021-07-08 00:00:00
2021-07-09 00:00:00
2021-07-10 00:00:00
2021-07-11 00:00:00
2021-07-12 00:00:00
2021-07-13 00:00:00
2021-07-14 00:00:00
2021-07-15 00:00:00
2021-07-16 00:00:00
2021-07-17 00:00:00
2021-07-18 00:00:00
2021-07-19 00:00:00
2021-07-20 00:00:00
2021-07-21 00:00:00
2021-07-22 00:00:00
2021-07-23 00:00:00
2021-07-24 00:00:00
2021-07-25 00:00:00
2021-07-26 00:00:00
2021-07-27 00:00:00
2021-07-28 00:00:00
2021-07-29 00:00:00
2021-07-30 00:00:00
2021-07-31 00:00:00
2021-08-01 00:00:00
2021-08-02 00:00:00
2021-08-03 00:00:00
2021-08-04 00:00:00
2021-08-05 00:00:00
2021-08-06 00:00:00
2021-08-07 00:00:00
2021-08-08 00:00:00
2021-08-09 00:00:00
2021-08-10 00:00:00
2021-08-11 00:00:00
2021-08-12 00:00:00
2021-08-13 00:00:00
2021-08-14 00:00:00
2021-08-15 00:00:00
2021-08-16 00:00:00
2021-08-17 00:00:00
2021-08-18 00:00:00
2021-08-19 00:00:00
2021-08-20 00:00:00
2021-08-21 00:00:00
2021-08-22 00:00:00
2021-08-23 00:00:00
2021-08-24 00:00:00
```

```
2021-08-25 00:00:00
2021-08-26 00:00:00
2021-08-27 00:00:00
2021-08-28 00:00:00
2021-08-29 00:00:00
2021-08-30 00:00:00
2021-08-31 00:00:00
2021-09-01 00:00:00
2021-09-02 00:00:00
2021-09-03 00:00:00
2021-09-04 00:00:00
2021-09-05 00:00:00
2021-09-06 00:00:00
2021-09-07 00:00:00
2021-09-08 00:00:00
2021-09-09 00:00:00
2021-09-10 00:00:00
2021-09-11 00:00:00
2021-09-12 00:00:00
2021-09-13 00:00:00
2021-09-14 00:00:00
2021-09-15 00:00:00
2021-09-16 00:00:00
2021-09-17 00:00:00
2021-09-18 00:00:00
2021-09-19 00:00:00
2021-09-20 00:00:00
2021-09-21 00:00:00
2021-09-22 00:00:00
2021-09-23 00:00:00
2021-09-24 00:00:00
2021-09-25 00:00:00
2021-09-26 00:00:00
2021-09-27 00:00:00
2021-09-28 00:00:00
2021-09-29 00:00:00
2021-09-30 00:00:00
2021-10-01 00:00:00
2021-10-02 00:00:00
2021-10-03 00:00:00
2021-10-04 00:00:00
2021-10-05 00:00:00
2021-10-06 00:00:00
2021-10-07 00:00:00
2021-10-08 00:00:00
2021-10-09 00:00:00
2021-10-10 00:00:00
2021-10-11 00:00:00
2021-10-12 00:00:00
2021-10-13 00:00:00
2021-10-14 00:00:00
2021-10-15 00:00:00
2021-10-16 00:00:00
2021-10-17 00:00:00
2021-10-18 00:00:00
2021-10-19 00:00:00
2021-10-20 00:00:00
2021-10-21 00:00:00
2021-10-22 00:00:00
2021-10-23 00:00:00
```

```
2021-10-24 00:00:00
2021-10-25 00:00:00
2021-10-26 00:00:00
2021-10-27 00:00:00
2021-10-28 00:00:00
2021-10-29 00:00:00
2021-10-30 00:00:00
2021-10-31 00:00:00
2021-11-01 00:00:00
2021-11-02 00:00:00
2021-11-03 00:00:00
2021-11-04 00:00:00
2021-11-05 00:00:00
2021-11-06 00:00:00
2021-11-07 00:00:00
2021-11-08 00:00:00
2021-11-09 00:00:00
2021-11-10 00:00:00
2021-11-11 00:00:00
2021-11-12 00:00:00
2021-11-13 00:00:00
2021-11-14 00:00:00
2021-11-15 00:00:00
2021-11-16 00:00:00
2021-11-17 00:00:00
2021-11-18 00:00:00
2021-11-19 00:00:00
2021-11-20 00:00:00
2021-11-21 00:00:00
2021-11-22 00:00:00
2021-11-23 00:00:00
2021-11-24 00:00:00
2021-11-25 00:00:00
2021-11-26 00:00:00
2021-11-27 00:00:00
2021-11-28 00:00:00
2021-11-29 00:00:00
2021-11-30 00:00:00
2021-12-01 00:00:00
2021-12-02 00:00:00
2021-12-03 00:00:00
2021-12-04 00:00:00
2021-12-05 00:00:00
2021-12-06 00:00:00
2021-12-07 00:00:00
2021-12-08 00:00:00
2021-12-09 00:00:00
2021-12-10 00:00:00
2021-12-11 00:00:00
2021-12-12 00:00:00
2021-12-13 00:00:00
2021-12-14 00:00:00
2021-12-15 00:00:00
2021-12-16 00:00:00
2021-12-17 00:00:00
2021-12-18 00:00:00
2021-12-19 00:00:00
2021-12-20 00:00:00
2021-12-21 00:00:00
2021-12-22 00:00:00
```

In [10]:
```
list(features_selected[0])
```

Out[10]:
```
['Debt Ratio',
 'R&D / Revenue',
 'Return on Research Capital',
 'tweet_subjectivity',
 'RSI_ratio',
 'MACD']
```

In [11]:
```
features_selected[0]
```

Out[11]:
```
Index(['Debt Ratio', 'R&D / Revenue', 'Return on Research Capital',
       'tweet_subjectivity', 'RSI_ratio', 'MACD'],
      dtype='object')
```

In [12]:
```
import pickle
```

In [13]:
```
with open('features', 'wb') as f:
    pickle.dump(features_selected, f)
```

In [14]:
```
with open('features', 'rb') as f:
    features_selected = pickle.load(f)
```

In [15]:
```
models = {'random forest': RandomForestClassifier(random_state=0,
                                n_jobs=-1,
                                n_estimators=10,
                                class_weight='balanced'),
          'logistic regression': LogisticRegression(max_iter=1000,
                                                    penalty='l2',
                                                    tol=0.001,
                                                    solver='newton-cg'),
          'SVC': SVC(kernel='poly'),
          'KNN': KNeighborsClassifier(n_neighbors=5),
          'gradient boosting':GradientBoostingClassifier(max_depth= 10, random_s
         }
```

In [16]:
```
results = {}
for model_name, model in models.items():
    if model_name not in results:
        dates = []
        predictions = []
        truth = []
        returns = []
        for train_date_start, features in zip(date_starts, features_selected):
            train_date_end = train_date_start + datetime.timedelta(days=90)
            validate_date_start = train_date_end + datetime.timedelta(days=1)
            validate_date_end = validate_date_start + datetime.timedelta(days=30
            test_date_start = validate_date_end + datetime.timedelta(days=1)
            test_date_end = test_date_start

            train_data = window(train_date_start, train_date_end, ZTS, all_date_
            validate_data = window(validate_date_start, validate_date_end, ZTS,
```

```
            test_data = window(test_date_start, test_date_end, ZTS, all_date_col

            train = pd.concat([train_data, validate_data])

            model.fit(train[features], train['Y_boolean'])
            predictions.append(model.predict(test_data[features])[0])
            dates.append(test_data['Report Date'].values[0])
            truth.append(test_data['Y_boolean'].values[0])
            returns.append(test_data['Return'].values[0])
        results[model_name] = pd.DataFrame({'date':dates,
                                            'prediction': predictions,
                                            'truth': truth,
                                            'returns': returns})
        results[model_name]['portfolio returns'] = results[model_name]['returns'
        results[model_name]['portfolio cumulative returns'] = np.cumprod(1 + res
```

In [17]:

```
for model_name in results.keys():
    acc = sum(results[model_name]['prediction'] == results[model_name]['truth'])
    print(f'{model_name} score:{round(acc,3)}')
```

```
random forest score:0.383
logistic regression score:0.363
SVC score:0.366
KNN score:0.369
gradient boosting score:0.36
```

In [18]:

```
def backtest(df):
    df_account_value = df[['date', 'portfolio cumulative returns']]
    df_account_value.columns = ['date', 'account_value']
    backtest_plot(df_account_value,
              baseline_ticker='SPY',
              baseline_start = df_account_value.date.iloc[0],
              baseline_end = df_account_value.date.iloc[-1])
```

In [19]:

```
backtest(results['gradient boosting'])
```

```
[**********************100%***********************]  1 of 1 completed
Shape of DataFrame:  (456, 8)
```

Start date	2020-07-02
End date	2022-04-23
Total months	31

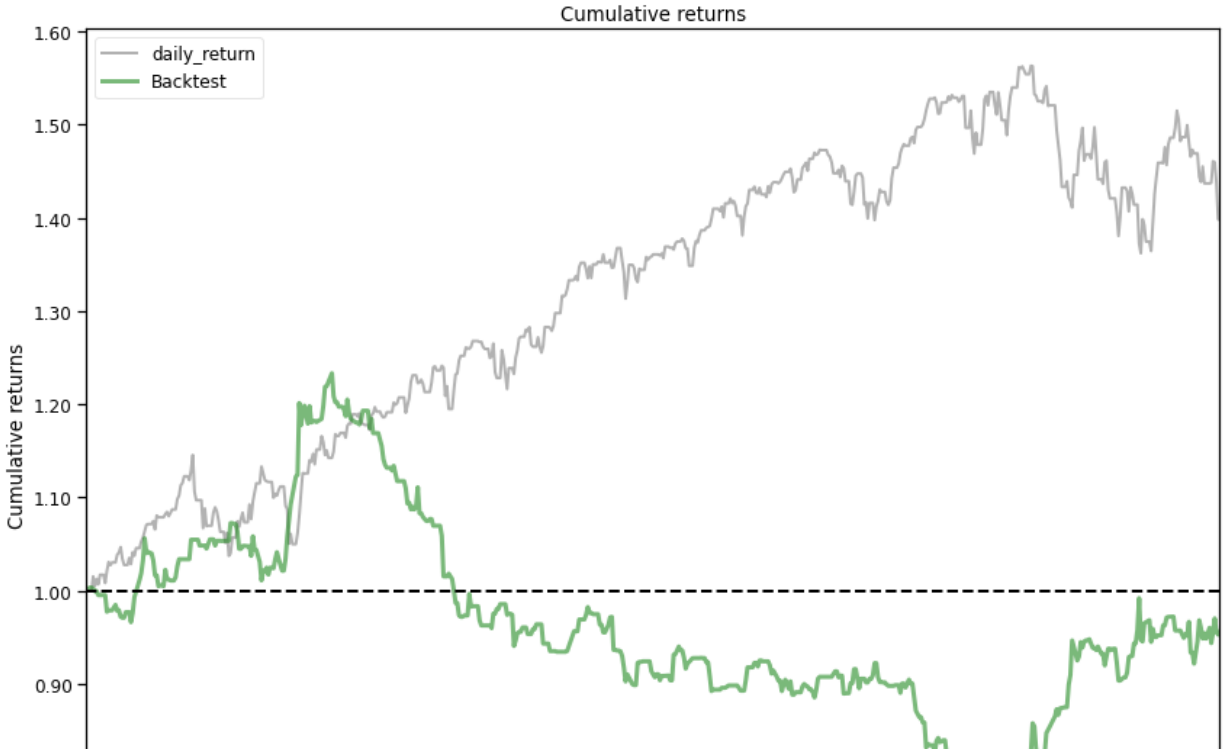
	Backtest
Annual return	-1.707%
Cumulative returns	-4.415%
Annual volatility	14.661%
Sharpe ratio	-0.04
Calmar ratio	-0.05
Stability	0.47
Max drawdown	-37.385%

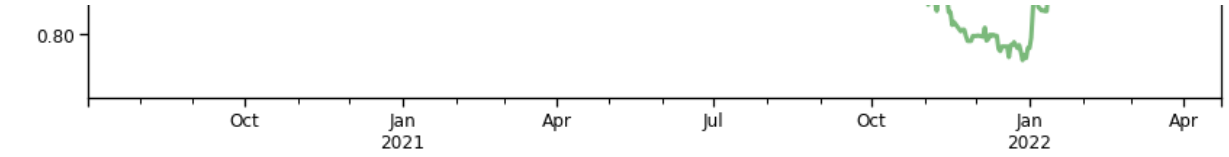
Start date	2020-07-02
End date	2022-04-23
Total months	31

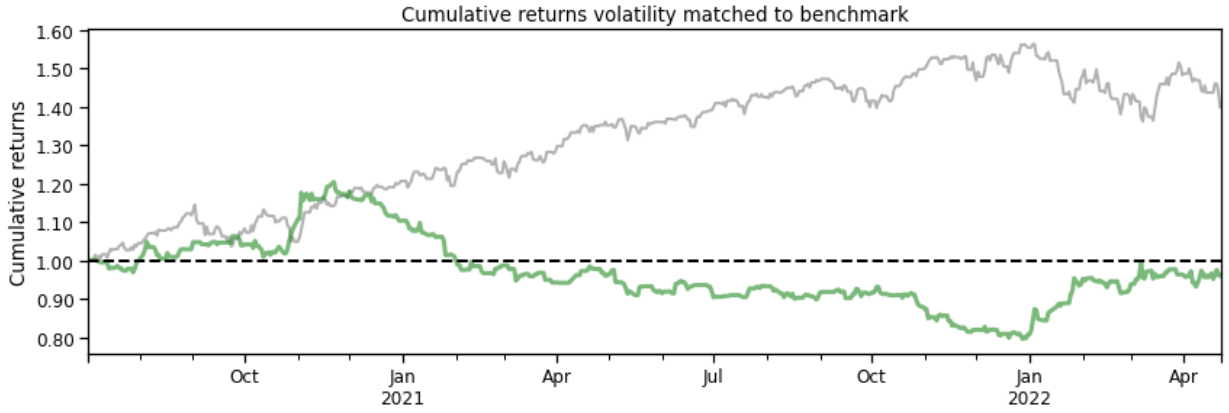
	Backtest
Omega ratio	0.99
Sortino ratio	-0.06
Skew	NaN
Kurtosis	NaN
Tail ratio	0.99
Daily value at risk	-1.85%
Alpha	-0.00
Beta	-0.02

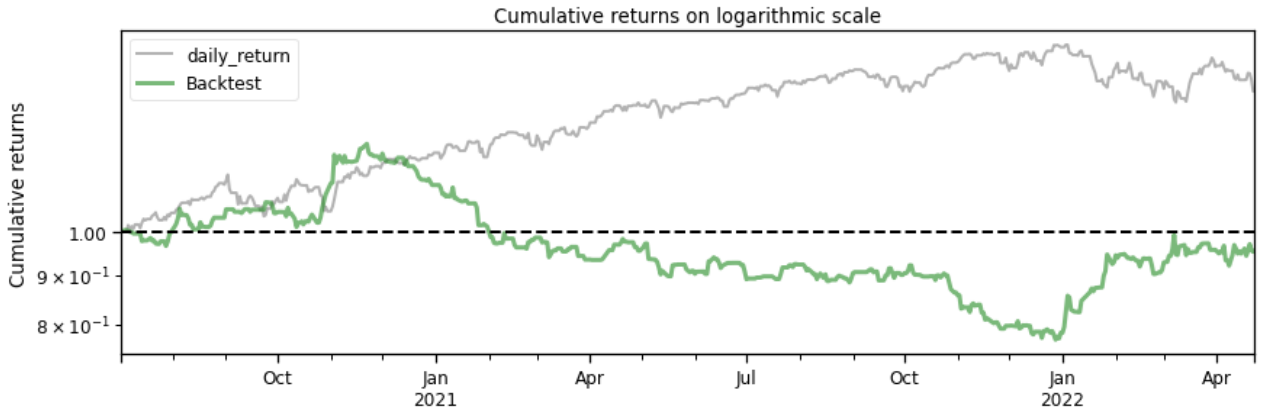
Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0	37.38	2020-11-22	2021-12-28	NaT	NaN
1	5.74	2020-09-25	2020-10-12	2020-10-28	24
2	4.86	2020-08-05	2020-08-13	2020-09-24	37
3	3.70	2020-07-05	2020-07-28	2020-08-01	20
4	1.98	2020-11-03	2020-11-04	2020-11-18	12

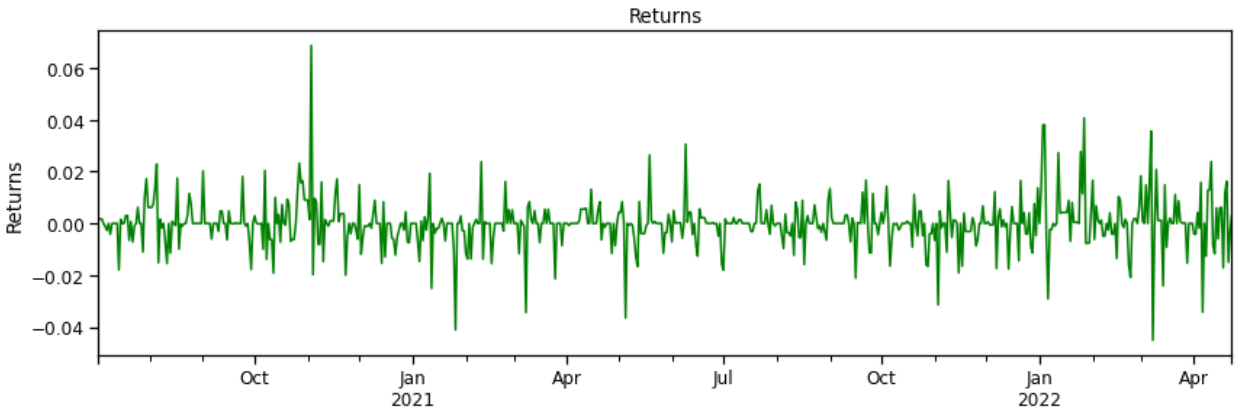
Stress Events	mean	min	max
New Normal	-0.00%	-4.52%	6.87%

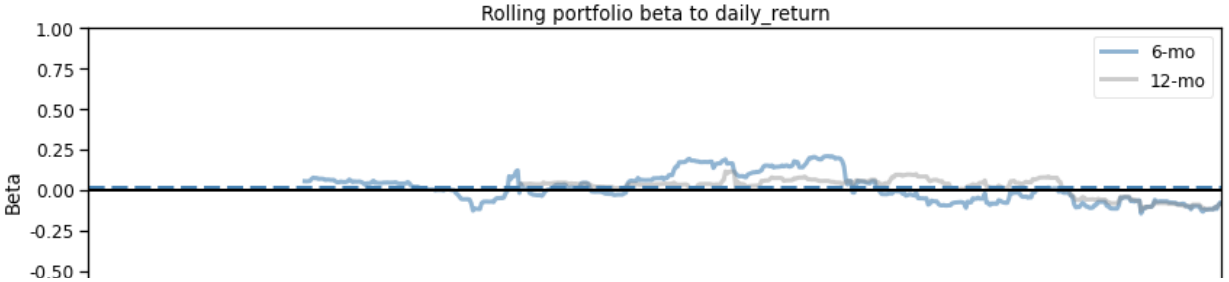


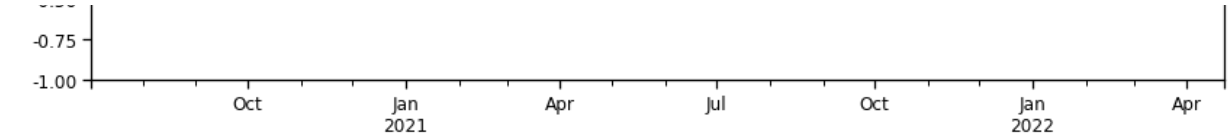

0.80
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

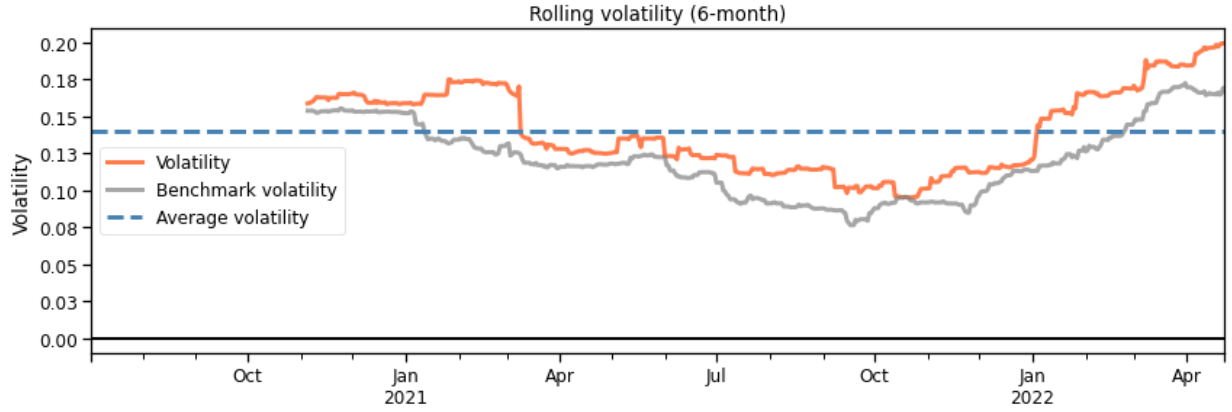

Cumulative returns volatility matched to benchmark
Cumulative returns
1.60
1.50
1.40
1.30
1.20
1.10
1.00
0.90
0.80
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

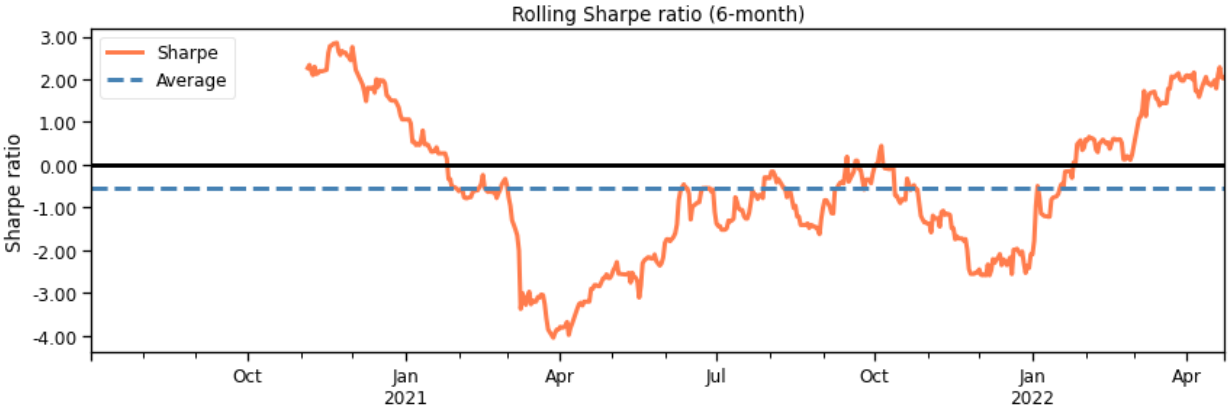

Cumulative returns on logarithmic scale
daily_return
Backtest
Cumulative returns
1.00
9×10^{-1}
8×10^{-1}
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

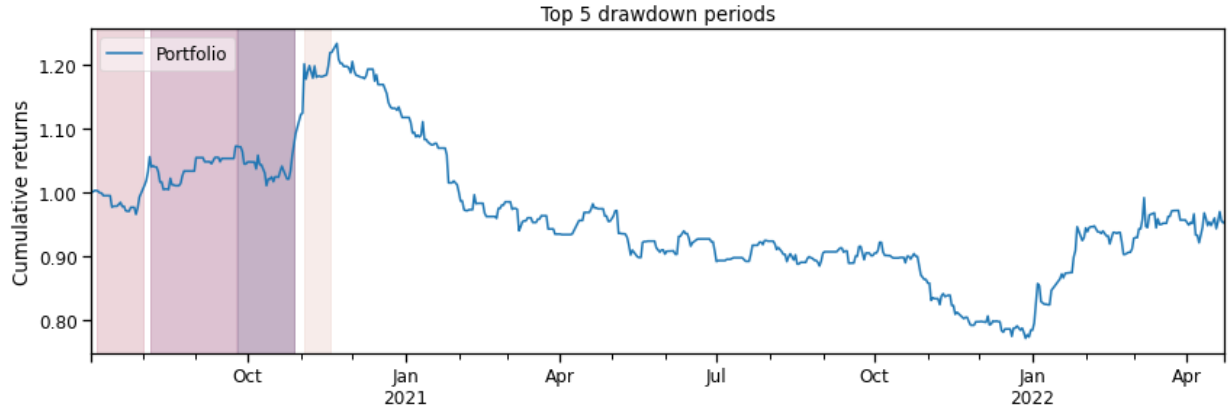

Returns
Returns
0.06
0.04
0.02
0.00
−0.02
−0.04
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

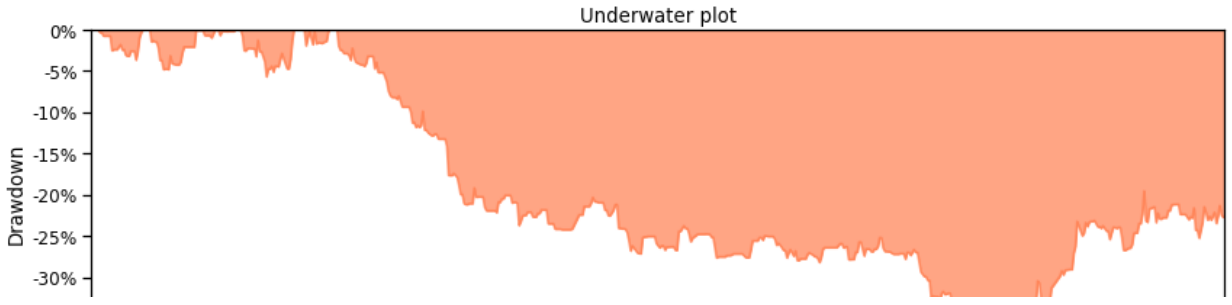

Rolling portfolio beta to daily_return
Beta
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
6-mo
12-mo

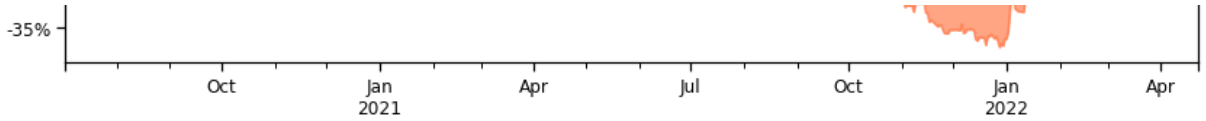

-0.75
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

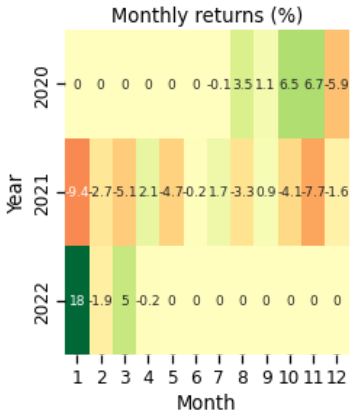

Rolling volatility (6-month)
Volatility
0.20
0.18
0.15
0.13
0.10
0.08
0.05
0.03
0.00
Volatility
Benchmark volatility
Average volatility
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

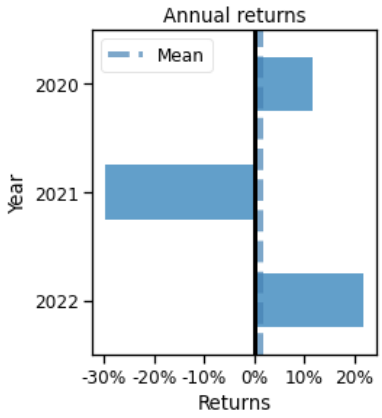

Rolling Sharpe ratio (6-month)
Sharpe ratio
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
Sharpe
Average
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

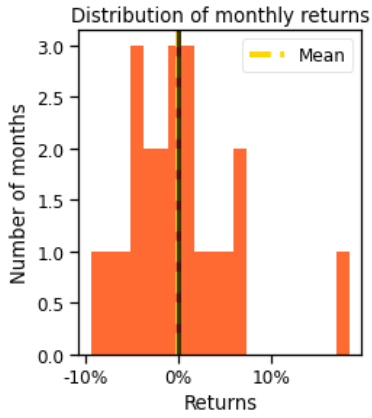

Top 5 drawdown periods
Cumulative returns
1.20
1.10
1.00
0.90
0.80
Portfolio
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

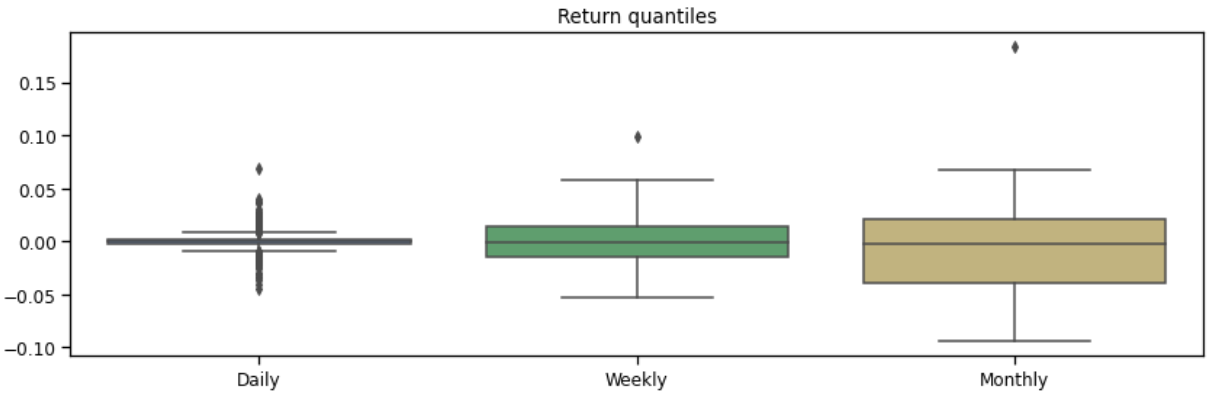

Underwater plot
Drawdown
0%
-5%
-10%
-15%
-20%
-25%
-30%

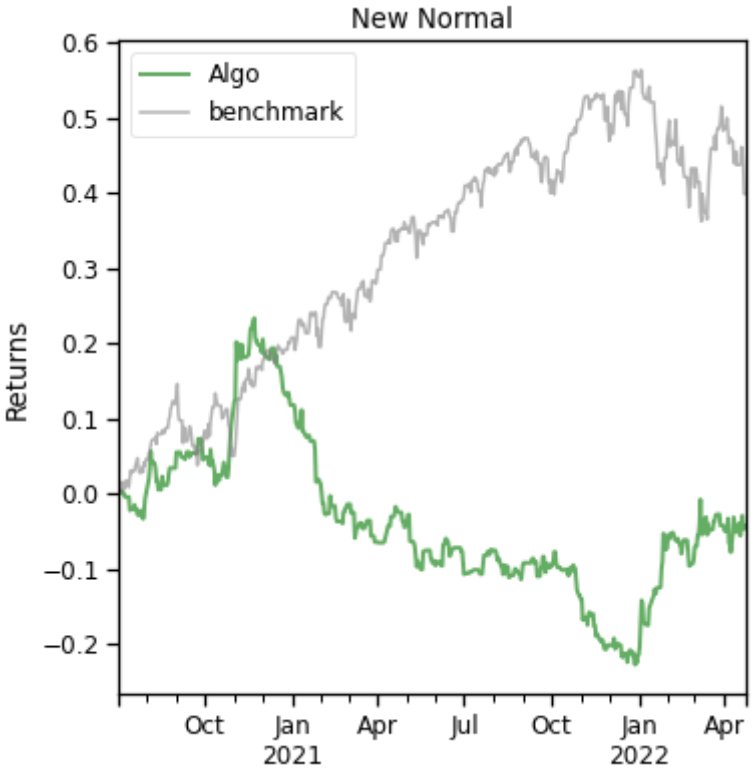












In [20]:
```
backtest(results['random forest'])
```

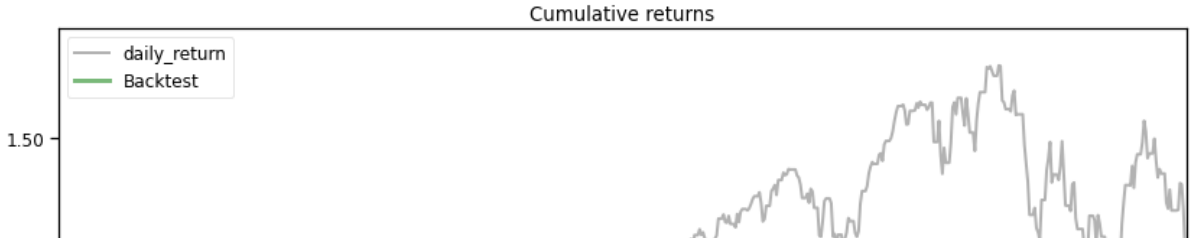
```
[*********************100%***********************]  1 of 1 completed
Shape of DataFrame:  (456, 8)
```

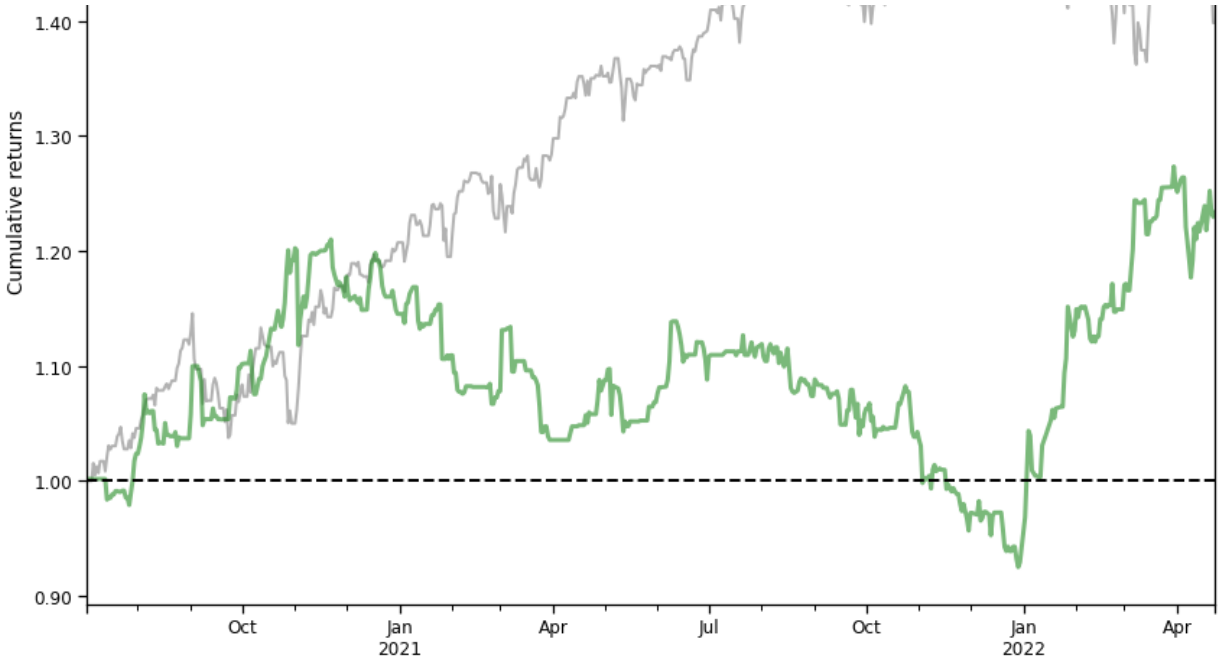
Start date	2020-07-02
End date	2022-04-23
Total months	31

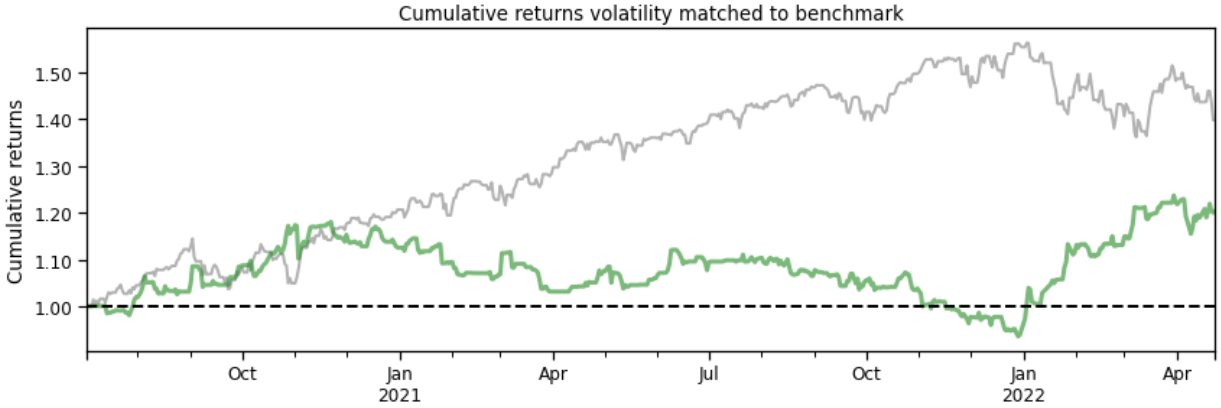
	Backtest
Annual return	8.331%
Cumulative returns	23.354%
Annual volatility	14.951%
Sharpe ratio	0.61
Calmar ratio	0.35
Stability	0.01
Max drawdown	-23.556%
Omega ratio	1.15
Sortino ratio	0.87
Skew	NaN
Kurtosis	NaN
Tail ratio	0.99
Daily value at risk	-1.847%
Alpha	0.10
Beta	-0.02

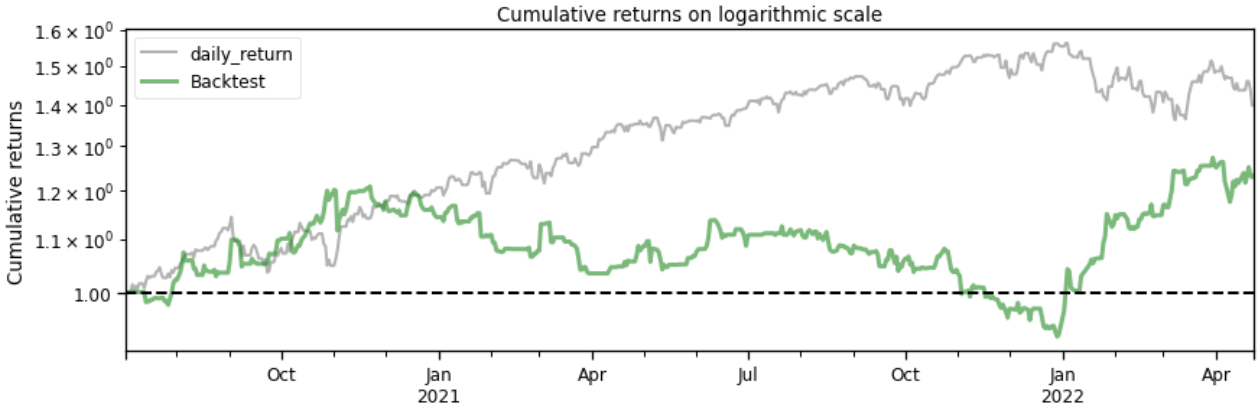
Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0	23.56	2020-11-22	2021-12-29	2022-03-07	336
1	7.59	2022-03-30	2022-04-09	NaT	NaN
2	7.01	2020-11-01	2020-11-03	2020-11-20	15
3	4.68	2020-09-05	2020-09-08	2020-10-01	19
4	4.19	2020-08-05	2020-08-24	2020-09-02	21

Stress Events	mean	min	max
New Normal	0.04%	-6.87%	4.92%

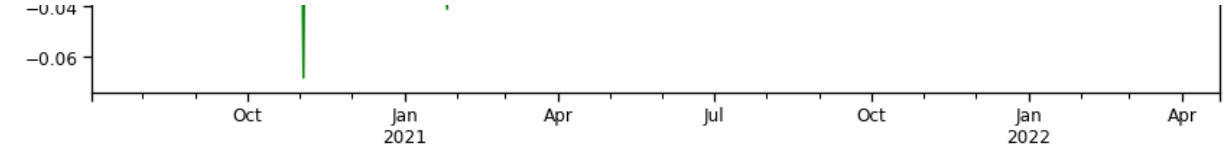


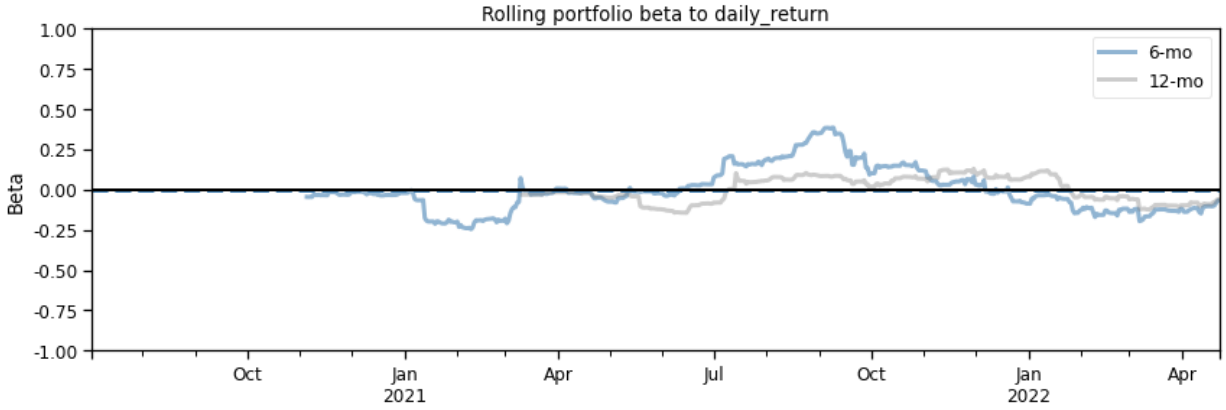

Cumulative returns
1.40
1.30
1.20
1.10
1.00
0.90
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

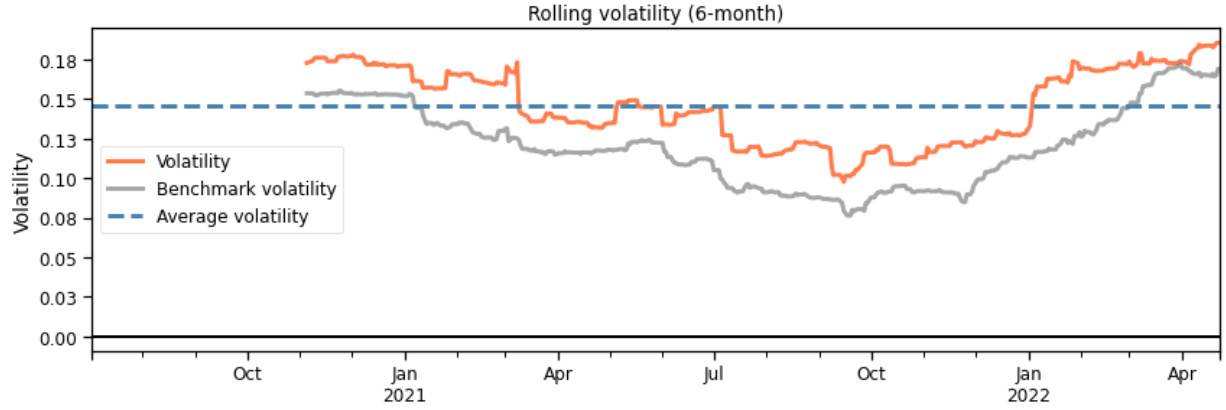

Cumulative returns volatility matched to benchmark
Cumulative returns
1.50
1.40
1.30
1.20
1.10
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

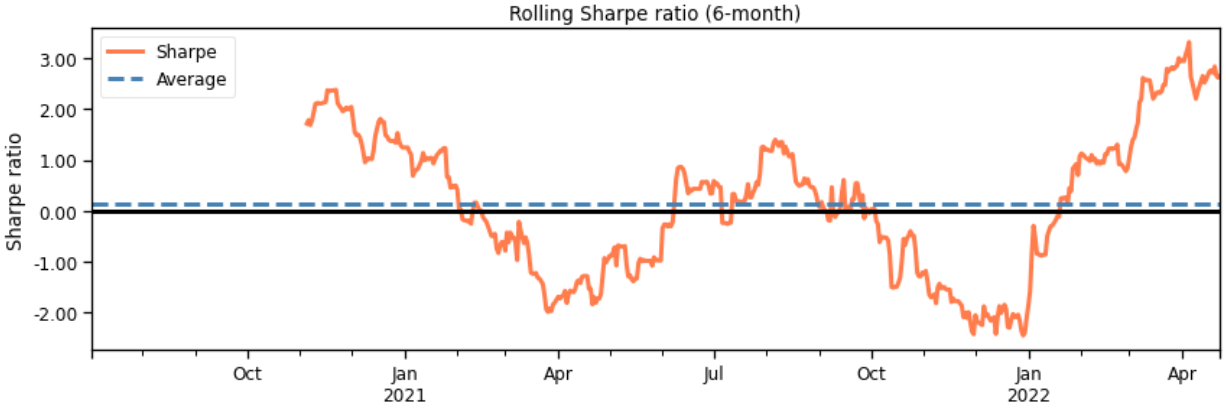

Cumulative returns on logarithmic scale
daily_return
Backtest
Cumulative returns
1.6 × 10^0
1.5 × 10^0
1.4 × 10^0
1.3 × 10^0
1.2 × 10^0
1.1 × 10^0
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

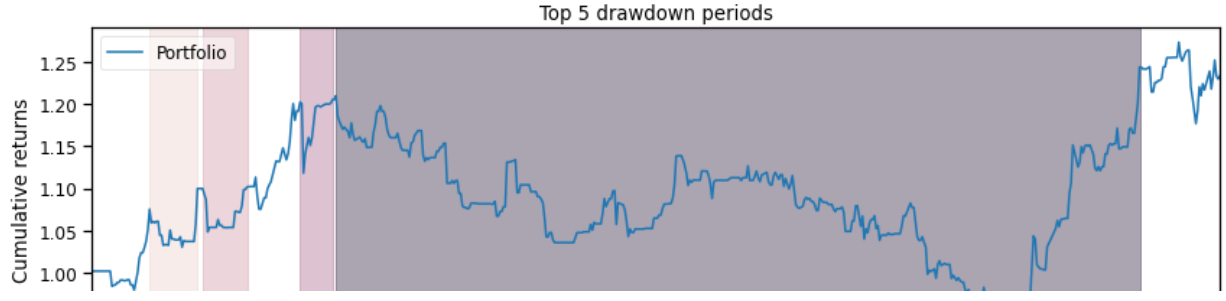

Returns
Returns
0.04
0.02
0.00
−0.02

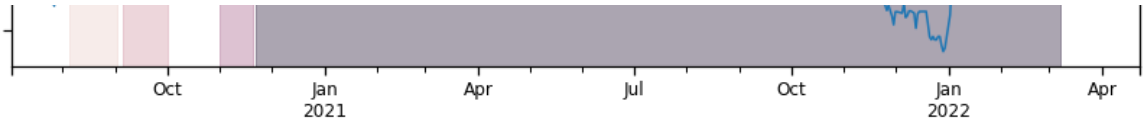

−0.04
−0.06
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

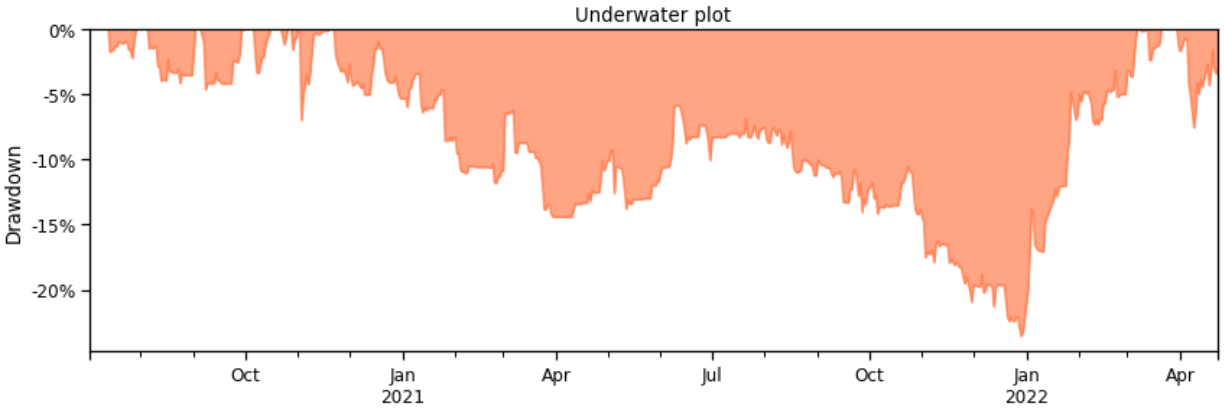

Rolling portfolio beta to daily_return
Beta
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00
6-mo
12-mo
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

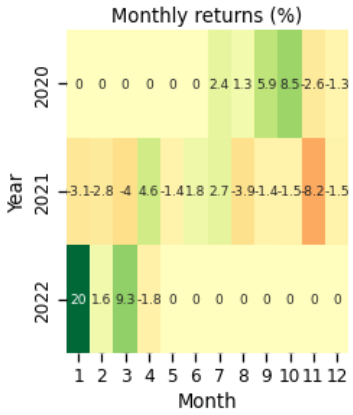

Rolling volatility (6-month)
Volatility
0.18
0.15
0.13
0.10
0.08
0.05
0.03
0.00
Volatility
Benchmark volatility
Average volatility
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

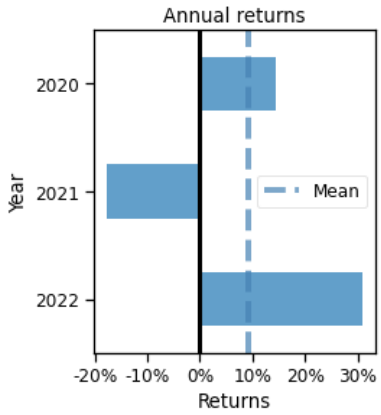

Rolling Sharpe ratio (6-month)
Sharpe ratio
3.00
2.00
1.00
0.00
-1.00
-2.00
Sharpe
Average
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

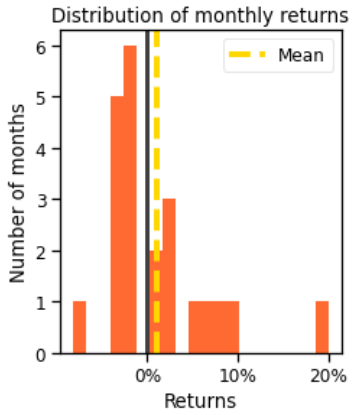

Top 5 drawdown periods
Cumulative returns
1.25
1.20
1.15
1.10
1.05
1.00
Portfolio

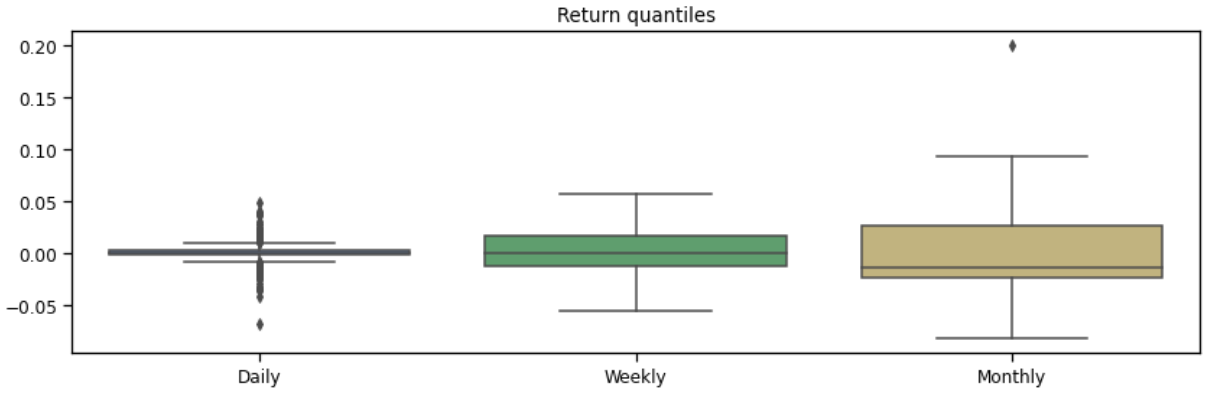

0.95
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

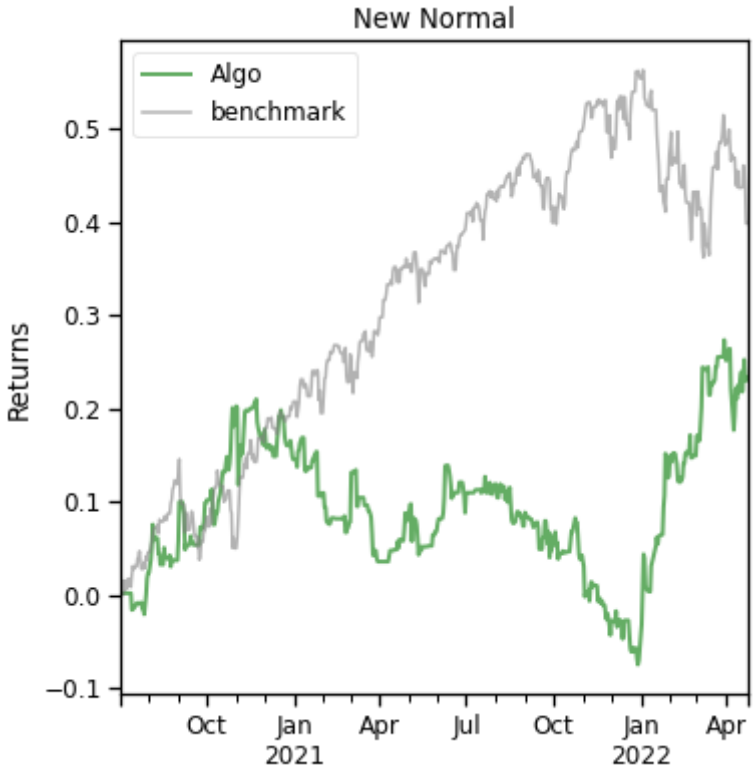

Underwater plot
Drawdown
0%
-5%
-10%
-15%
-20%
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr


Monthly returns (%)
Year
2020
2021
2022
0 0 0 0 0 0 2.4 1.3 5.9 8.5 -2.6 -1.3
-3.1 -2.8 -4 4.6 -1.4 1.8 2.7 -3.9 -1.4 -1.5 8.2 -1.5
20 1.6 9.3 -1.8 0 0 0 0 0 0 0 0
1 2 3 4 5 6 7 8 9 10 11 12
Month


Annual returns
Year
2020
2021
2022
Mean
-20% -10% 0% 10% 20% 30%
Returns


Distribution of monthly returns
Mean
Number of months
0
1
2
3
4
5
6
0%
10%
20%
Returns


Return quantiles
0.20
0.15
0.10
0.05
0.00
−0.05
Daily
Weekly
Monthly



In [21]:
```
backtest(results['logistic regression'])
```

```
[**********100%**********]  1 of 1 completed
Shape of DataFrame:  (456, 8)
```

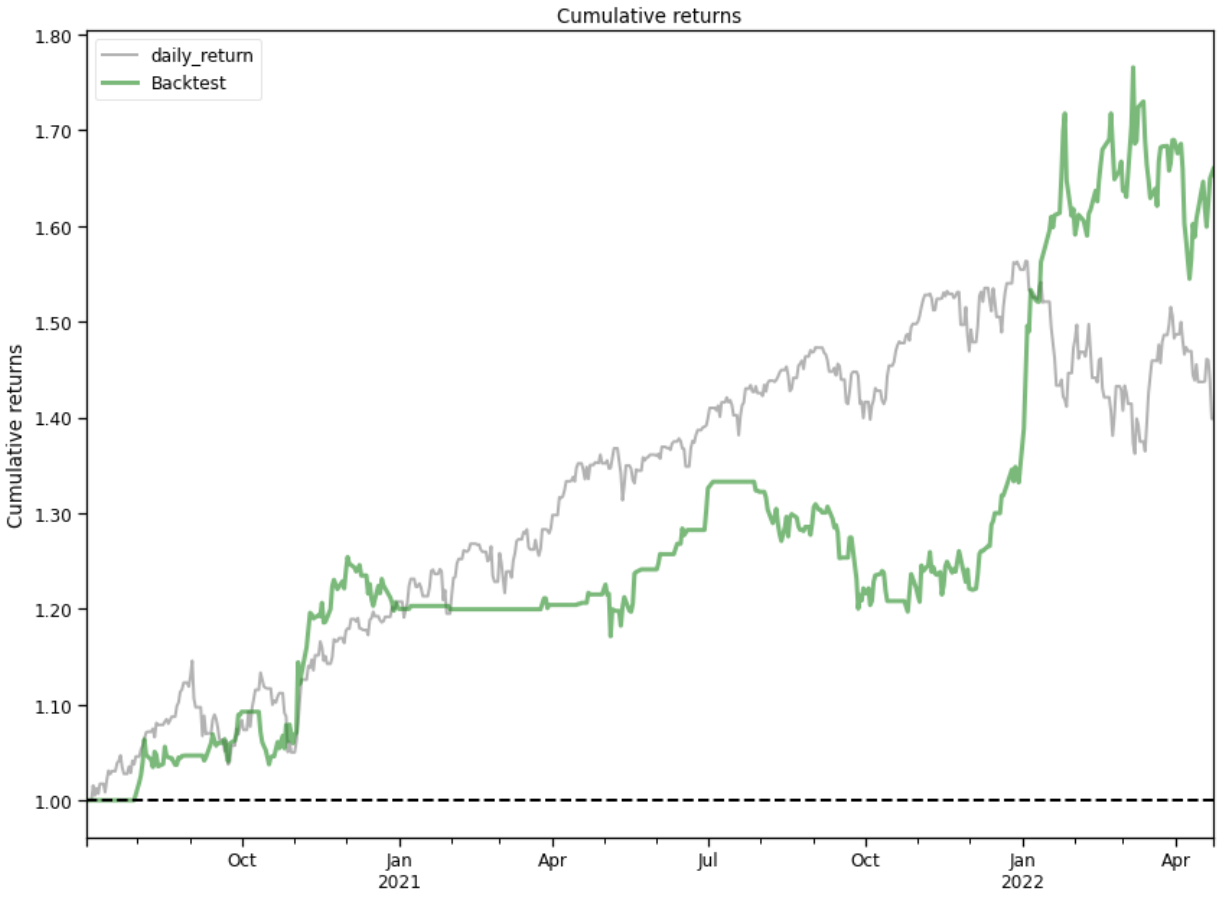
Start date	2020-07-02
End date	2022-04-23
Total months	31

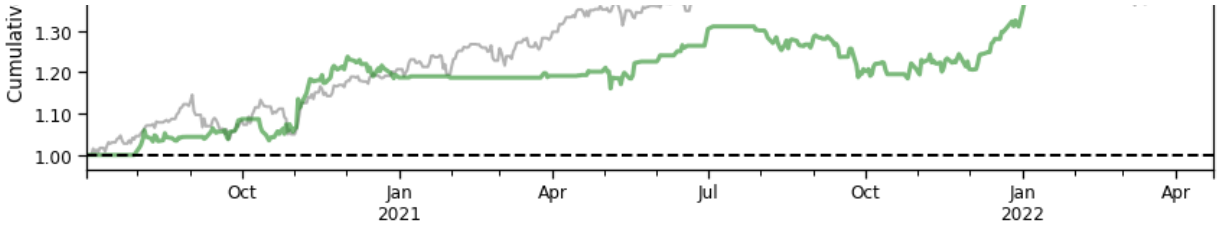
	Backtest
Annual return	21.302%
Cumulative returns	65.953%
Annual volatility	13.797%
Sharpe ratio	1.47
Calmar ratio	1.70
Stability	0.77
Max drawdown	-12.508%
Omega ratio	1.41
Sortino ratio	2.38
Skew	NaN
Kurtosis	NaN
Tail ratio	1.28
Daily value at risk	-1.658%
Alpha	0.22

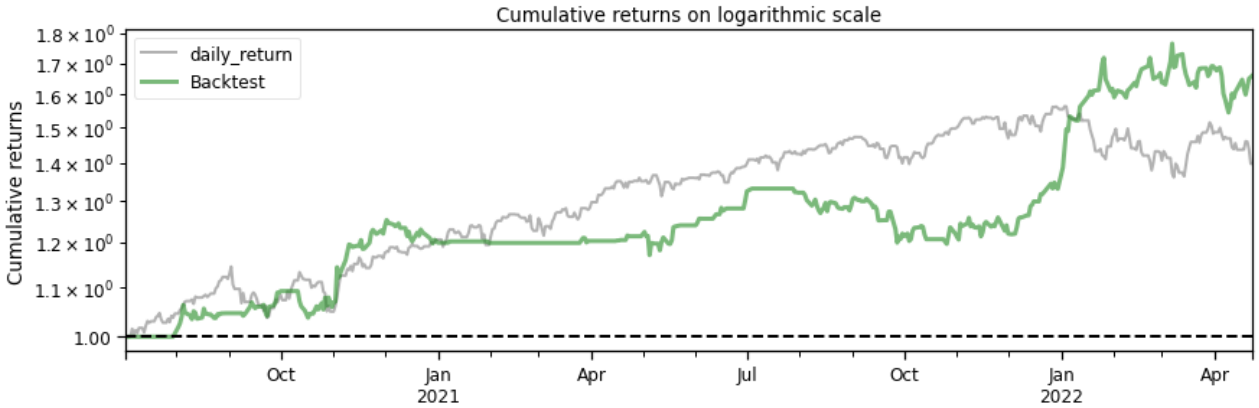
Start date	2020-07-02
End date	2022-04-23
Total months	31
	Backtest
Beta	0.02

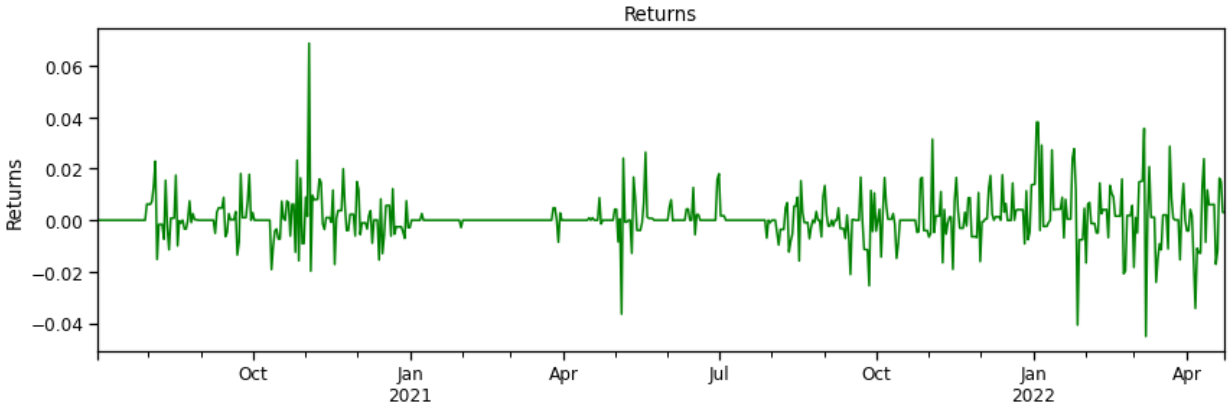
	Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0		12.51	2022-03-07	2022-04-09	NaT	NaN
1		10.19	2021-07-28	2021-10-26	2021-12-24	108
2		7.44	2022-01-26	2022-02-08	2022-03-07	29
3		6.60	2020-12-02	2021-05-05	2021-06-03	132
4		5.03	2020-10-11	2020-10-17	2020-11-03	17

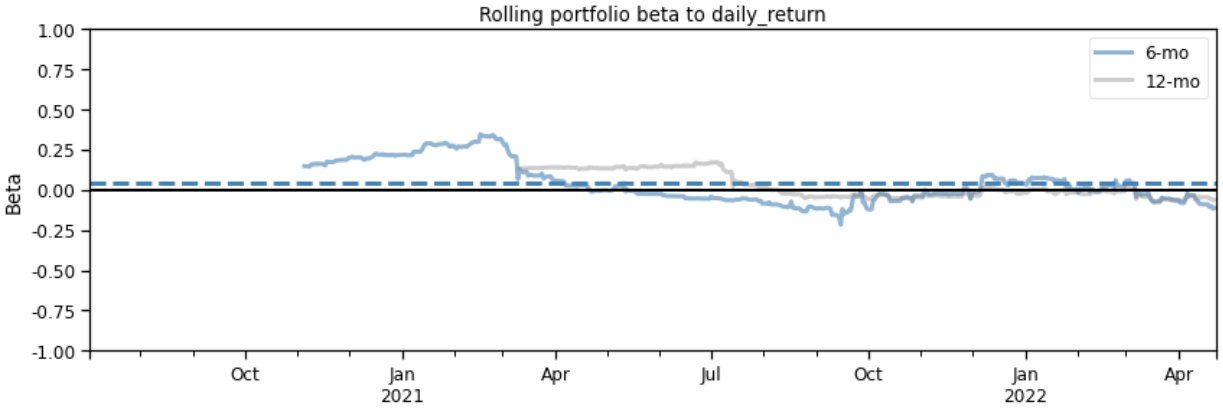
Stress Events	mean	min	max
New Normal	0.08%	-4.52%	6.87%

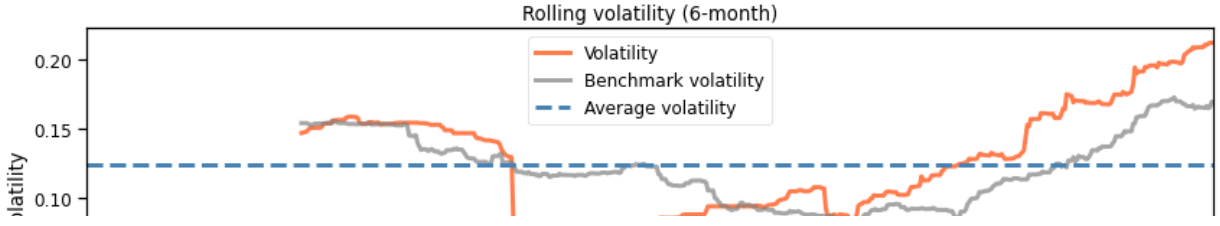


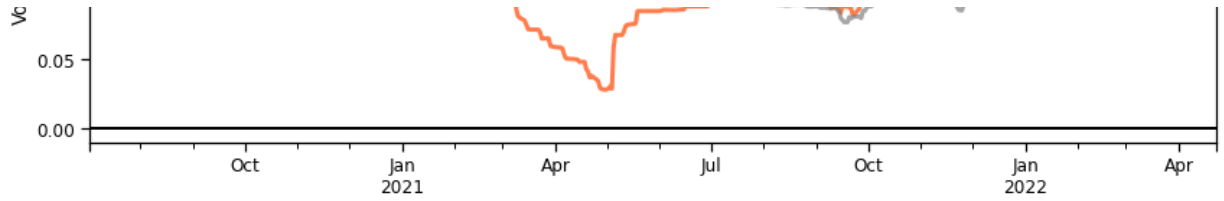

Cumulativ
1.30
1.20
1.10
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

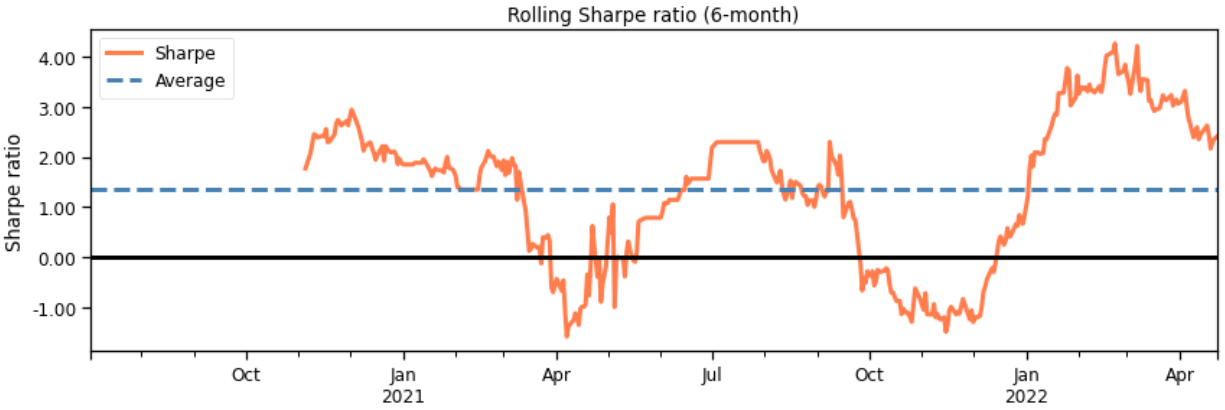

Cumulative returns on logarithmic scale
daily_return
Backtest
Cumulative returns
1.8 × 10^0
1.7 × 10^0
1.6 × 10^0
1.5 × 10^0
1.4 × 10^0
1.3 × 10^0
1.2 × 10^0
1.1 × 10^0
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

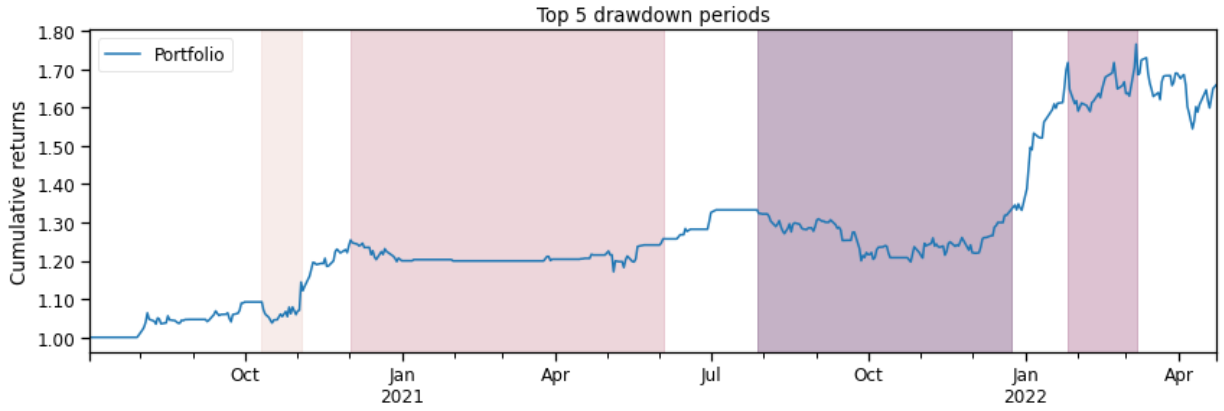

Returns
Returns
0.06
0.04
0.02
0.00
−0.02
−0.04
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

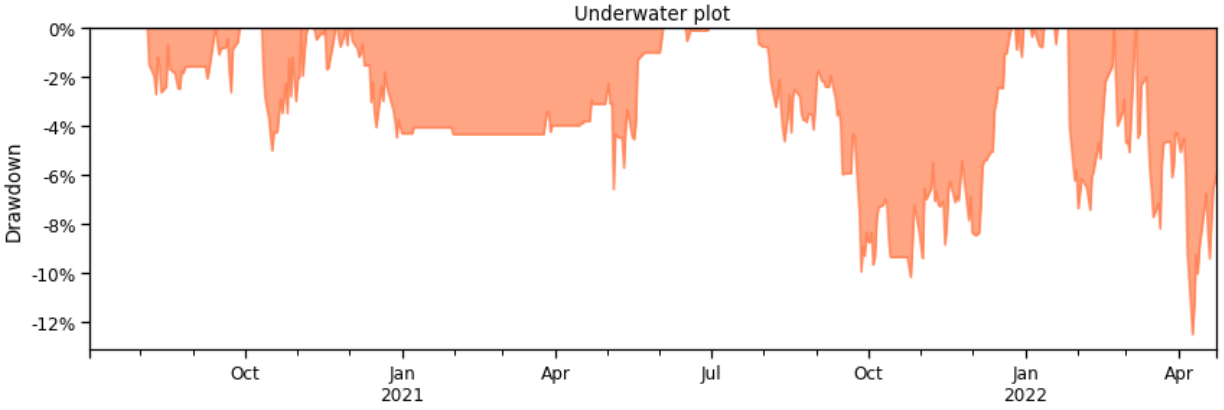

Rolling portfolio beta to daily_return
6-mo
12-mo
Beta
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

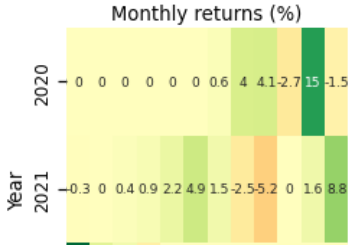

Rolling volatility (6-month)
Volatility
Benchmark volatility
Average volatility
0.20
0.15
0.10

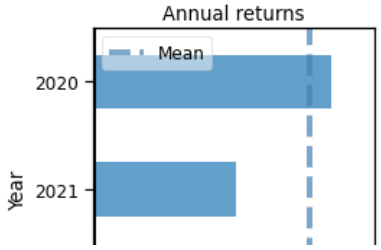

Vc
0.05
0.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

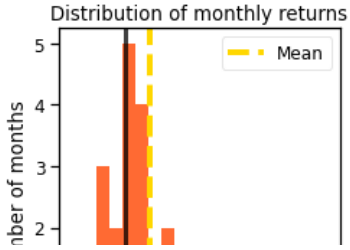

Rolling Sharpe ratio (6-month)
Sharpe
Average
Sharpe ratio
4.00
3.00
2.00
1.00
0.00
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

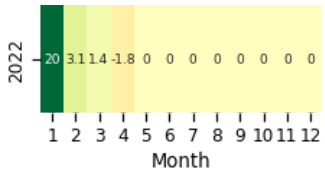

Top 5 drawdown periods
Portfolio
Cumulative returns
1.80
1.70
1.60
1.50
1.40
1.30
1.20
1.10
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

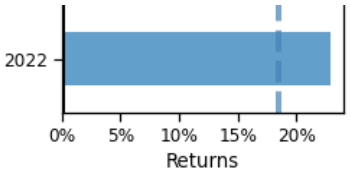

Underwater plot
Drawdown
0%
-2%
-4%
-6%
-8%
-10%
-12%
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

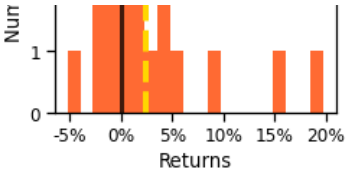

Monthly returns (%)
Year
2020
2021
0 0 0 0 0 0 0.6 4 4.1 -2.7 15 -1.5
-0.3 0 0.4 0.9 2.2 4.9 1.5 -2.5 -5.2 0 1.6 8.8

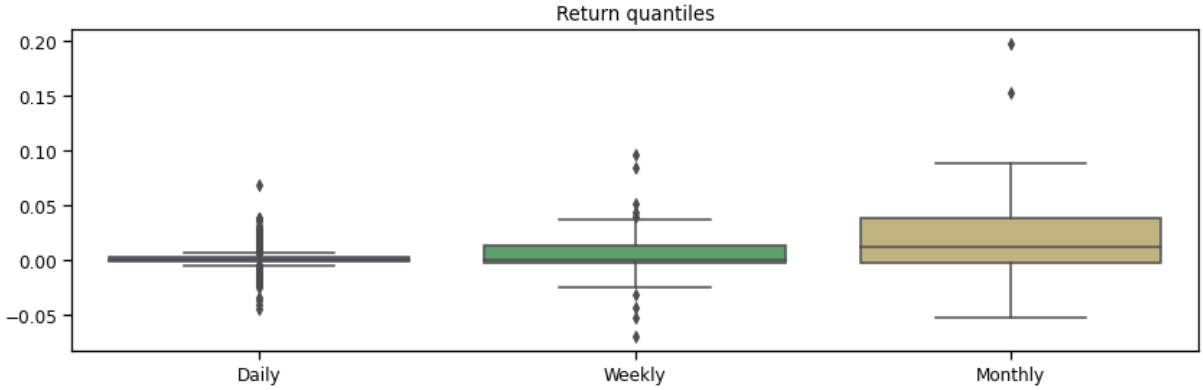

Annual returns
Mean
Year
2020
2021

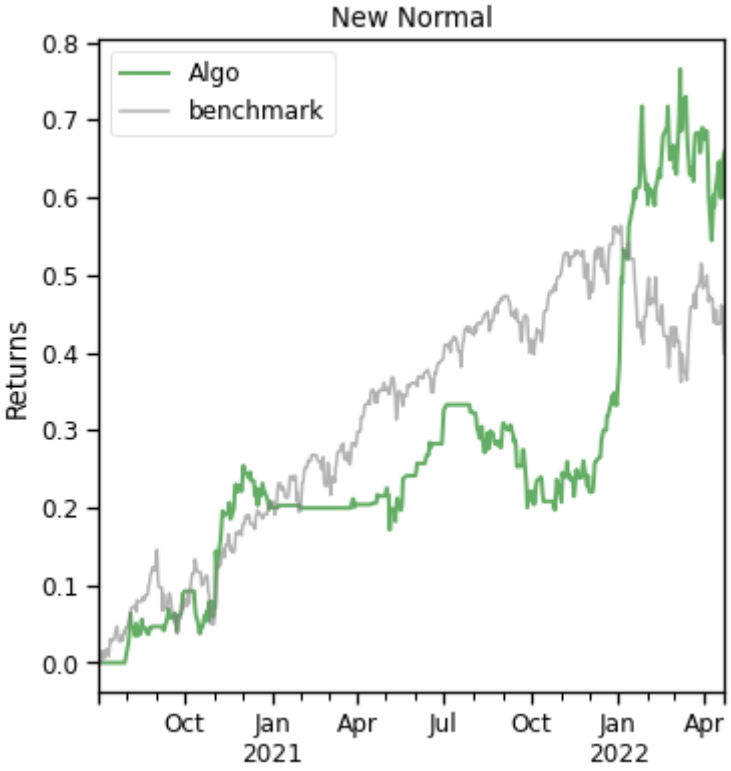

Distribution of monthly returns
Mean
nber of months
5
4
3
2











In [22]:
```
backtest(results['SVC'])
```

```
[**********************100%**********************]  1 of 1 completed
Shape of DataFrame:  (456, 8)
```

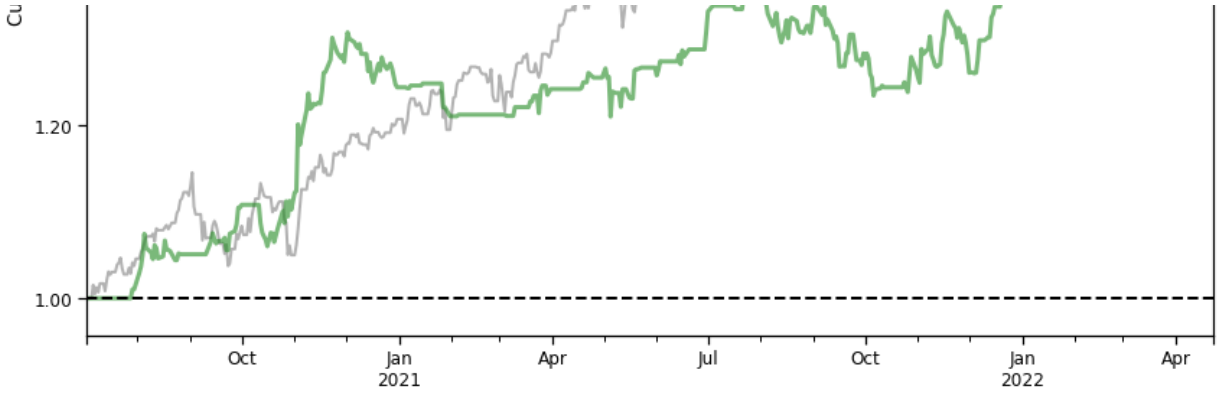
Start date	2020-07-02
End date	2022-04-23
Total months	31
	Backtest
Annual return	23.696%
Cumulative returns	74.683%

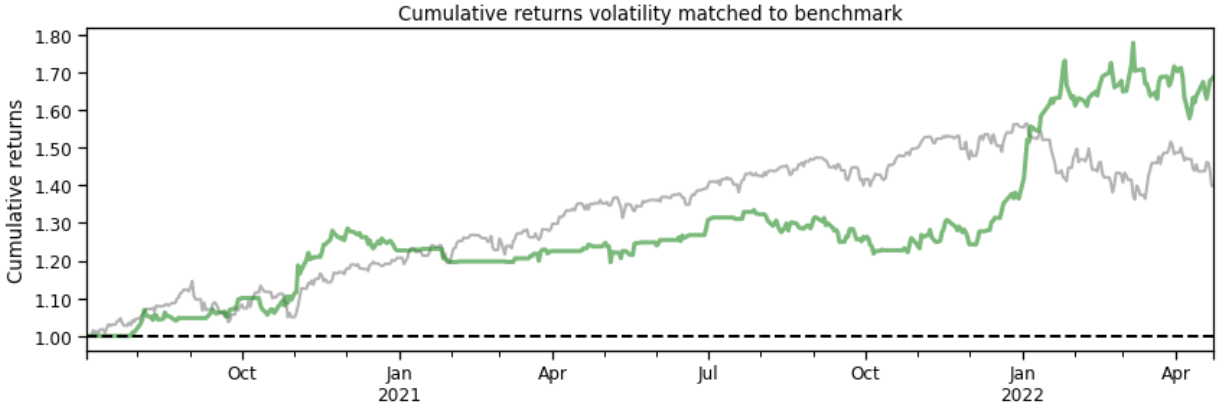
Start date	2020-07-02
End date	2022-04-23
Total months	31

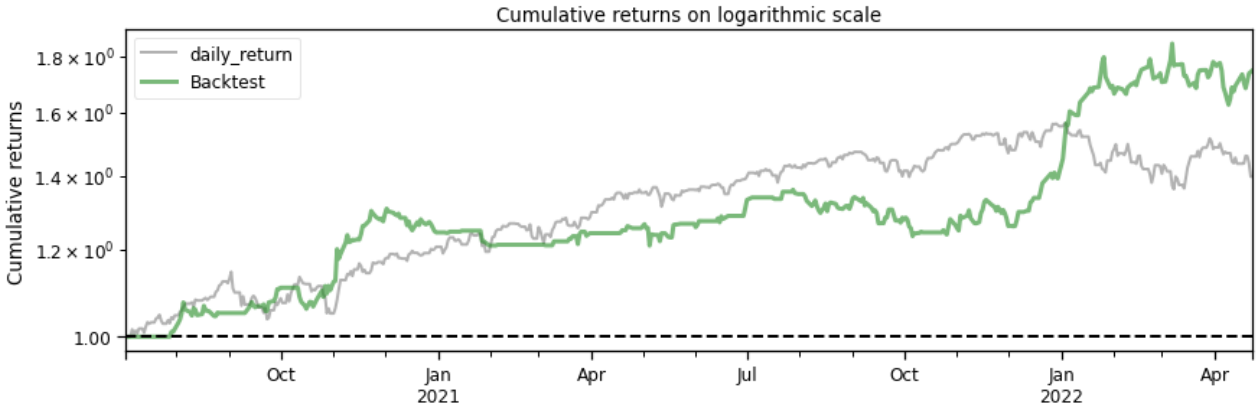
	Backtest
Annual volatility	13.89%
Sharpe ratio	1.60
Calmar ratio	1.97
Stability	0.78
Max drawdown	-12.032%
Omega ratio	1.44
Sortino ratio	2.63
Skew	NaN
Kurtosis	NaN
Tail ratio	1.30
Daily value at risk	-1.662%
Alpha	0.24
Beta	0.04

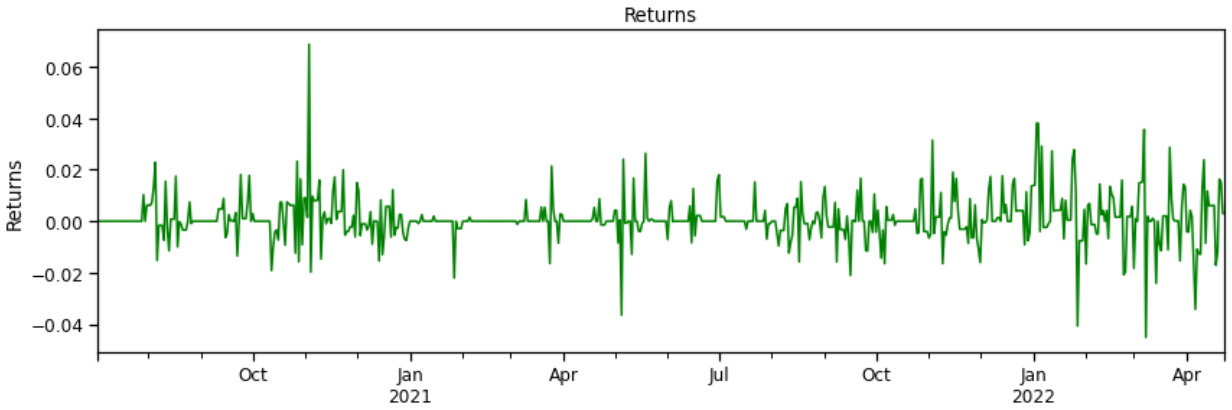
Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0	12.03	2022-03-07	2022-04-09	NaT	NaN
1	9.26	2021-07-28	2021-10-06	2021-12-21	105
2	7.46	2020-12-02	2021-05-05	2021-06-30	151
3	7.44	2022-01-26	2022-02-08	2022-03-07	29
4	4.33	2020-10-11	2020-10-16	2020-10-27	12

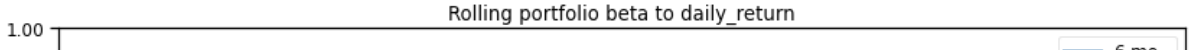
Stress Events	mean	min	max
New Normal	0.09%	-4.52%	6.87%

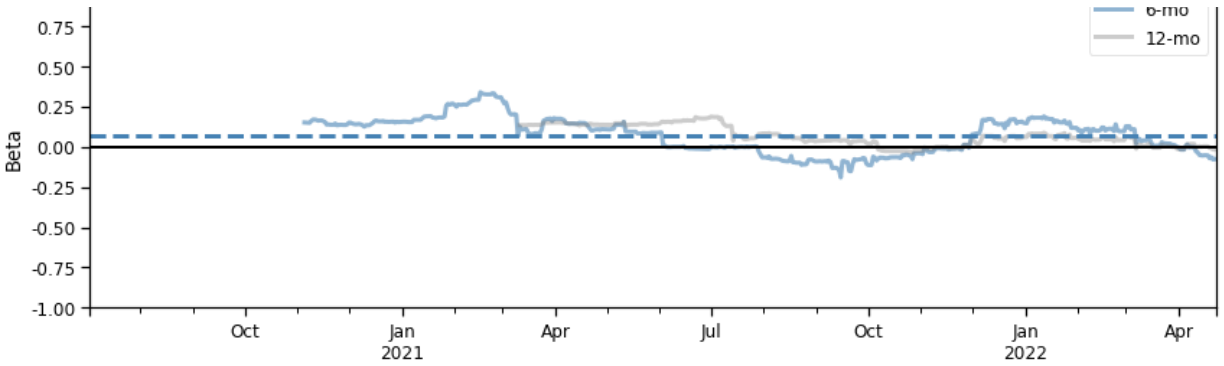

1.20
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

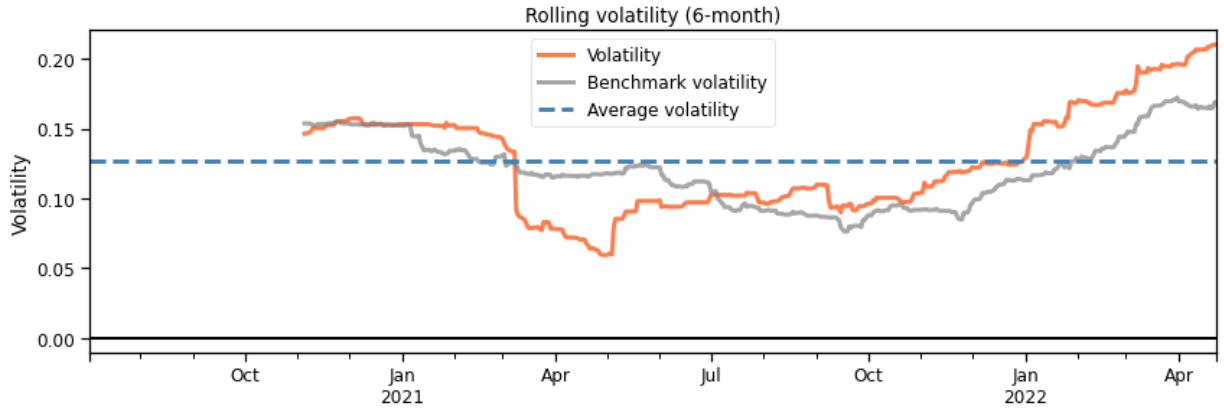

Cumulative returns volatility matched to benchmark
Cumulative returns
1.80
1.70
1.60
1.50
1.40
1.30
1.20
1.10
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

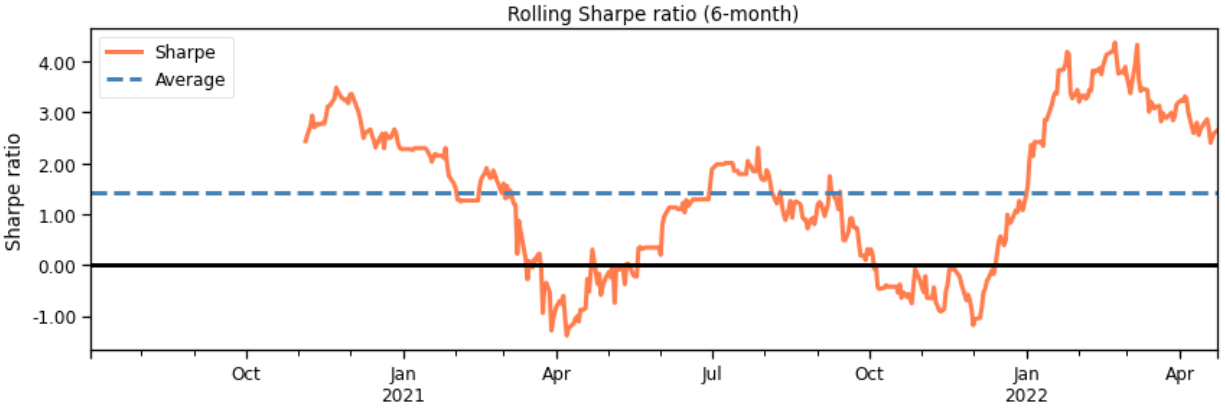

Cumulative returns on logarithmic scale
daily_return
Backtest
Cumulative returns
1.8 × 10^0
1.6 × 10^0
1.4 × 10^0
1.2 × 10^0
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

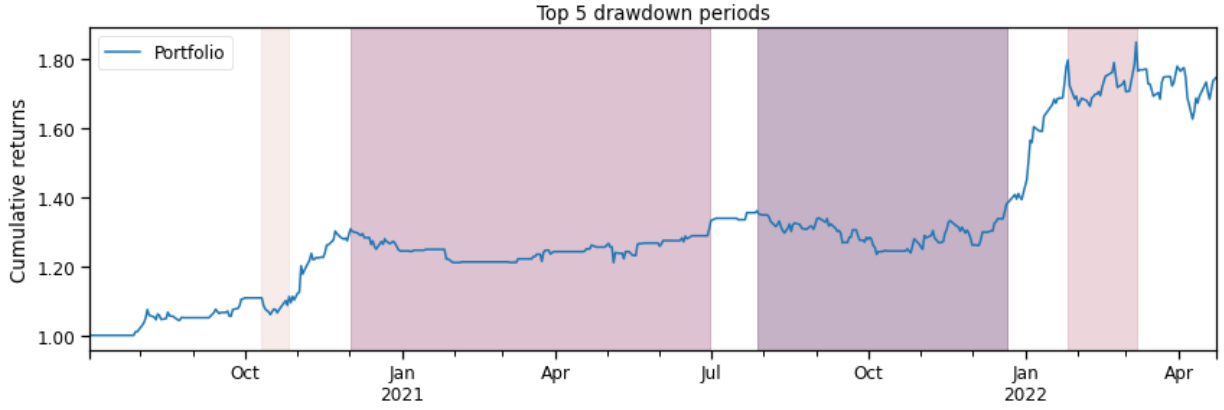

Returns
Returns
0.06
0.04
0.02
0.00
−0.02
−0.04
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

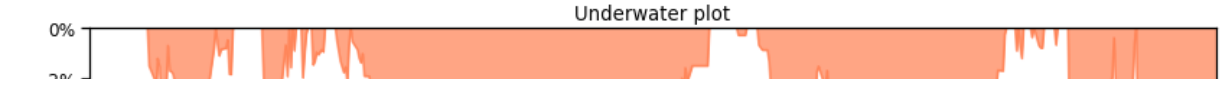

Rolling portfolio beta to daily_return
1.00

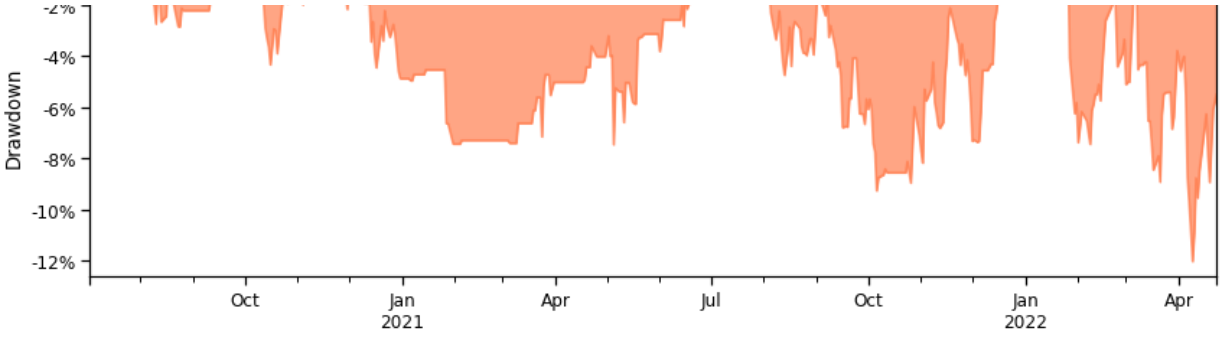

6-mo
12-mo
Beta
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

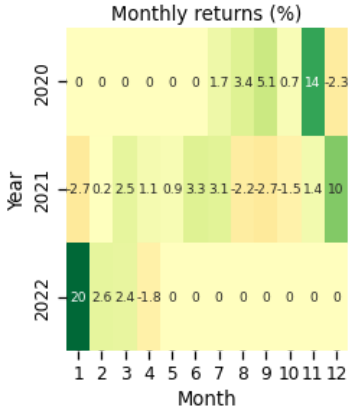

Rolling volatility (6-month)
Volatility
Benchmark volatility
Average volatility
Volatility
0.20
0.15
0.10
0.05
0.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

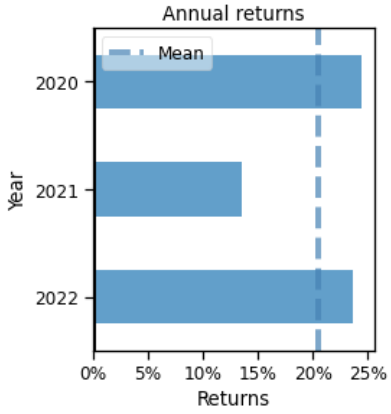

Rolling Sharpe ratio (6-month)
Sharpe
Average
Sharpe ratio
4.00
3.00
2.00
1.00
0.00
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

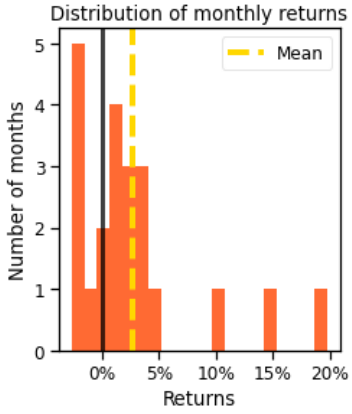

Top 5 drawdown periods
Portfolio
Cumulative returns
1.80
1.60
1.40
1.20
1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

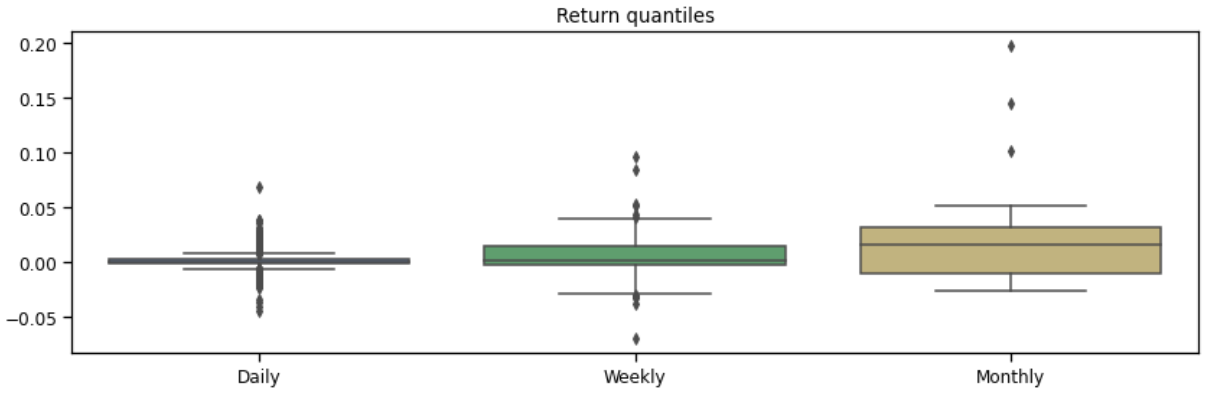

Underwater plot
0%

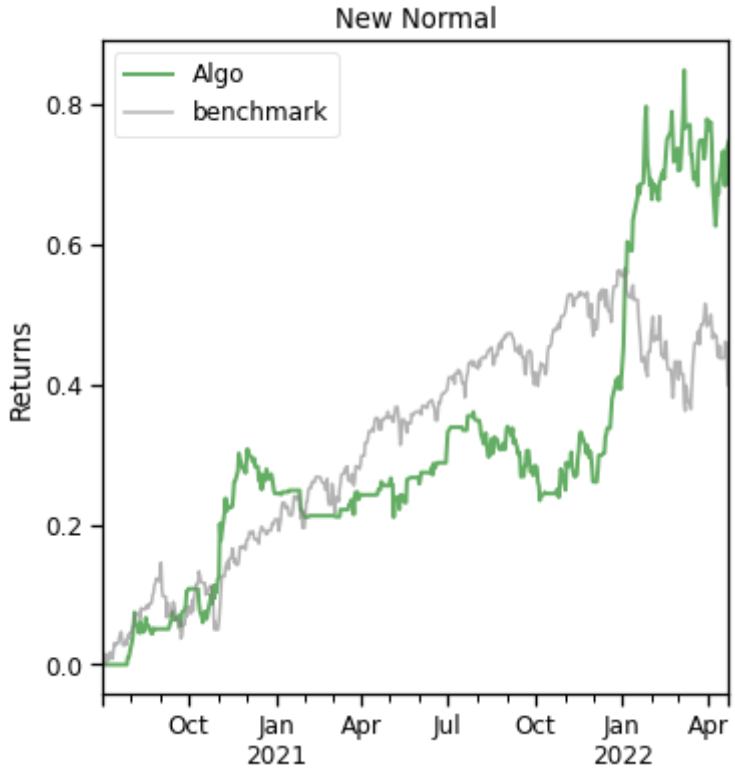

Drawdown
-2%
-4%
-6%
-8%
-10%
-12%
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr


Monthly returns (%)
Year
2020
2021
2022
0 0 0 0 0 0 1.7 3.4 5.1 0.7 14 -2.3
-2.7 0.2 2.5 1.1 0.9 3.3 3.1 -2.2 -2.7 -1.5 1.4 10
20 2.6 2.4 -1.8 0 0 0 0 0 0 0 0
1 2 3 4 5 6 7 8 9 10 11 12
Month


Annual returns
Mean
Year
2020
2021
2022
0% 5% 10% 15% 20% 25%
Returns


Distribution of monthly returns
Mean
Number of months
0
1
2
3
4
5
0% 5% 10% 15% 20%
Returns


Return quantiles
0.20
0.15
0.10
0.05
0.00
−0.05
Daily
Weekly
Monthly



In [23]:
```
backtest(results['KNN'])
```

```
[**********100%**********]  1 of 1 completed
Shape of DataFrame:  (456, 8)
```

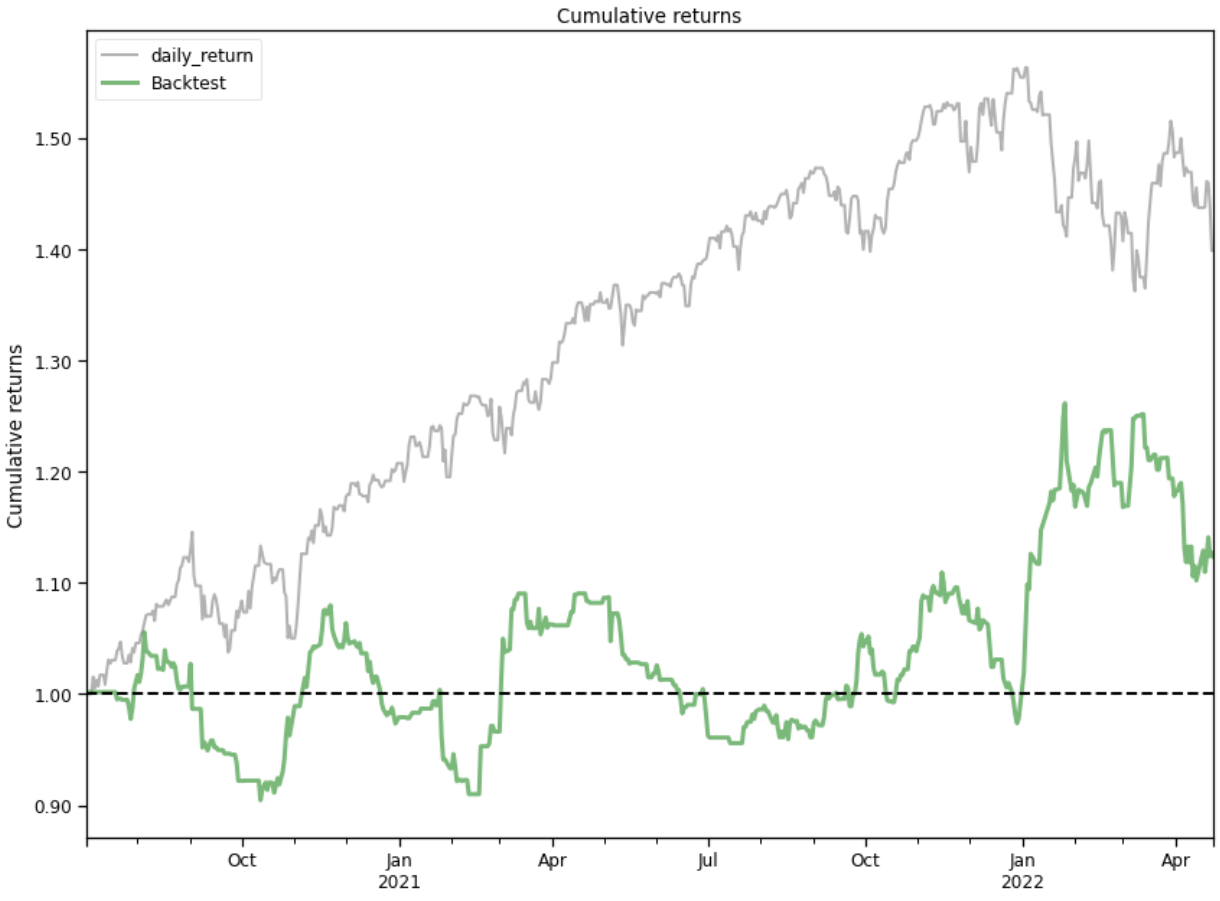
Start date	2020-07-02
End date	2022-04-23
Total months	31

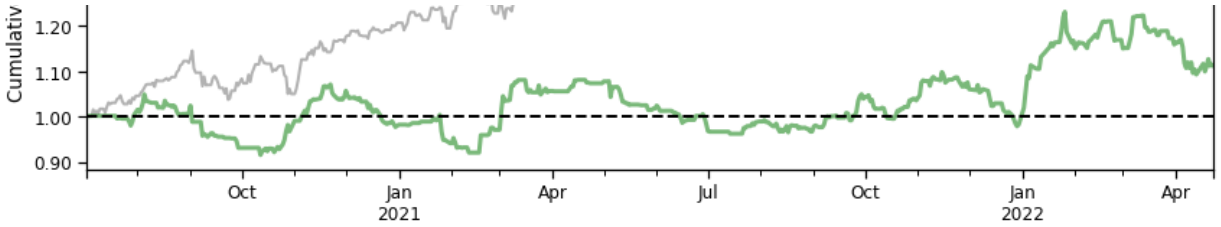
	Backtest
Annual return	4.547%
Cumulative returns	12.371%
Annual volatility	14.628%
Sharpe ratio	0.38
Calmar ratio	0.29
Stability	0.40
Max drawdown	-15.729%
Omega ratio	1.09
Sortino ratio	0.56
Skew	NaN
Kurtosis	NaN
Tail ratio	0.97
Daily value at risk	-1.821%
Alpha	0.06

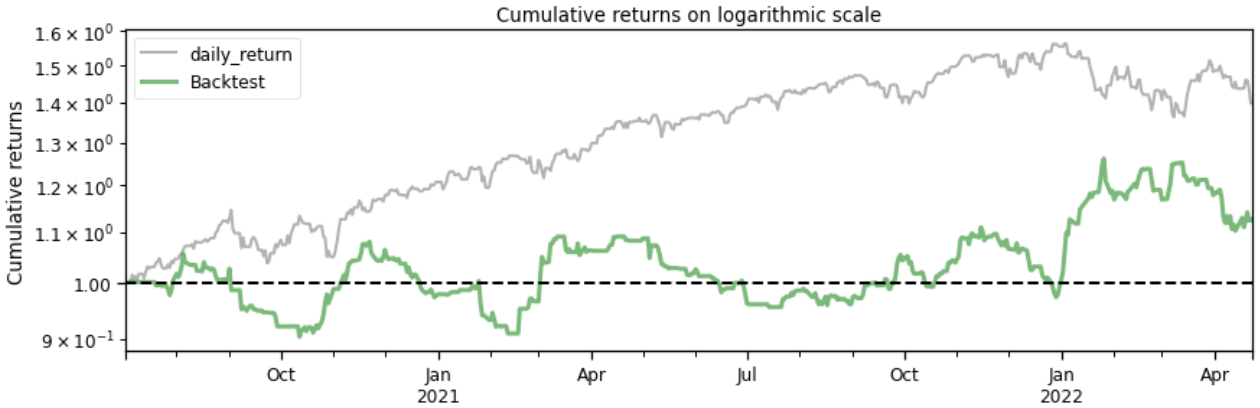
Start date	2020-07-02
End date	2022-04-23
Total months	31
	Backtest
Beta	-0.03

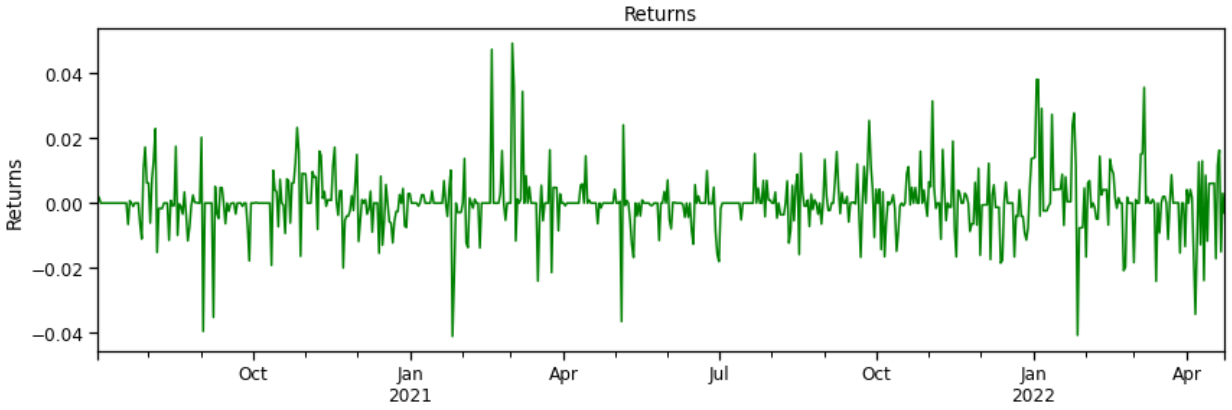
Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0	15.73	2020-11-22	2021-02-11	2021-03-10	78
1	14.31	2020-08-05	2020-10-12	2020-11-17	75
2	12.63	2022-01-26	2022-04-13	NaT	NaN
3	12.34	2021-04-20	2021-07-14	2021-11-09	146
4	12.22	2021-11-15	2021-12-29	2022-01-06	39

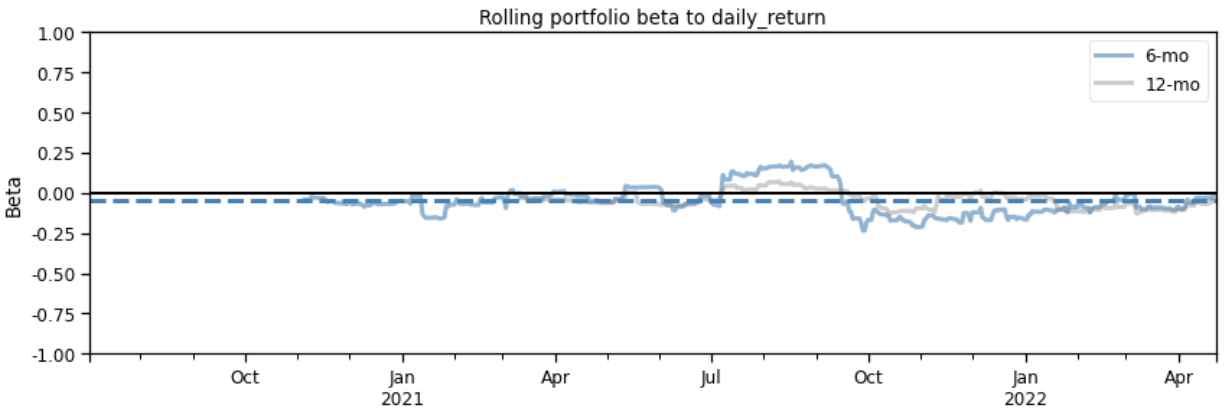
Stress Events	mean	min	max
New Normal	0.02%	-4.11%	4.92%

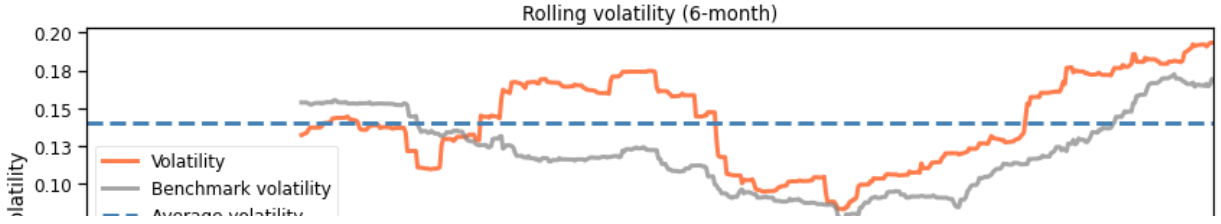


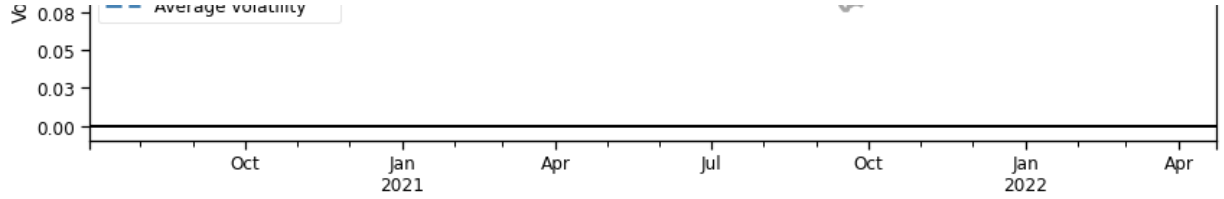

Cumulativ
1.20
1.10
1.00
0.90
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

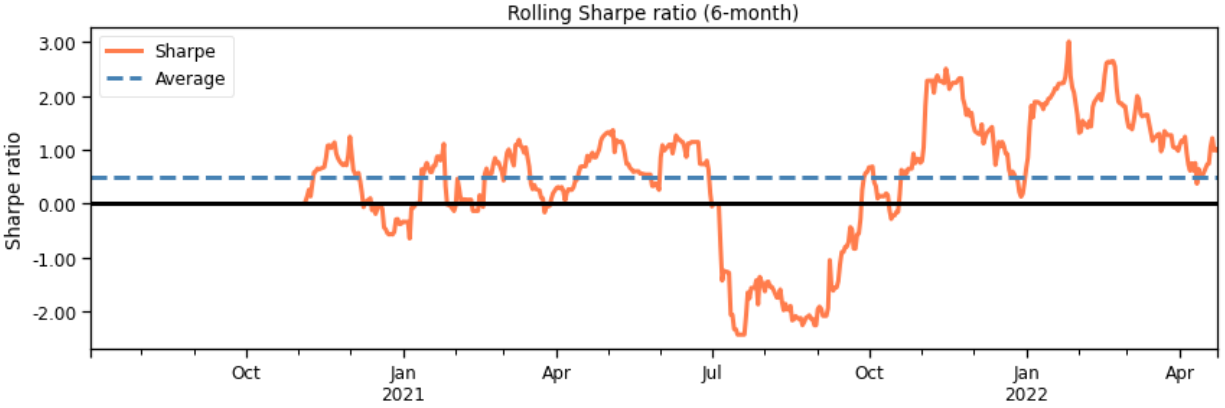

Cumulative returns on logarithmic scale
daily_return
Backtest
Cumulative returns
1.6 × 10^0
1.5 × 10^0
1.4 × 10^0
1.3 × 10^0
1.2 × 10^0
1.1 × 10^0
1.00
9 × 10^−1
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

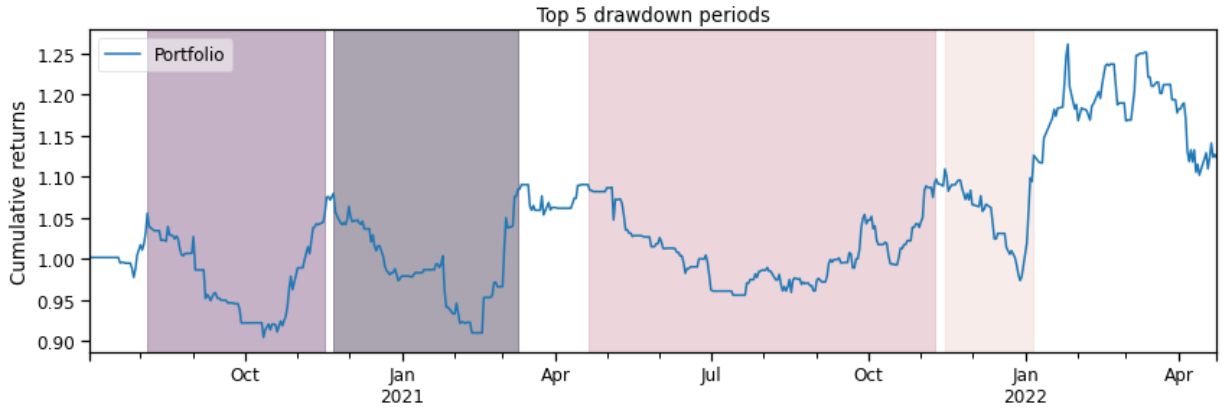

Returns
Returns
0.04
0.02
0.00
−0.02
−0.04
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

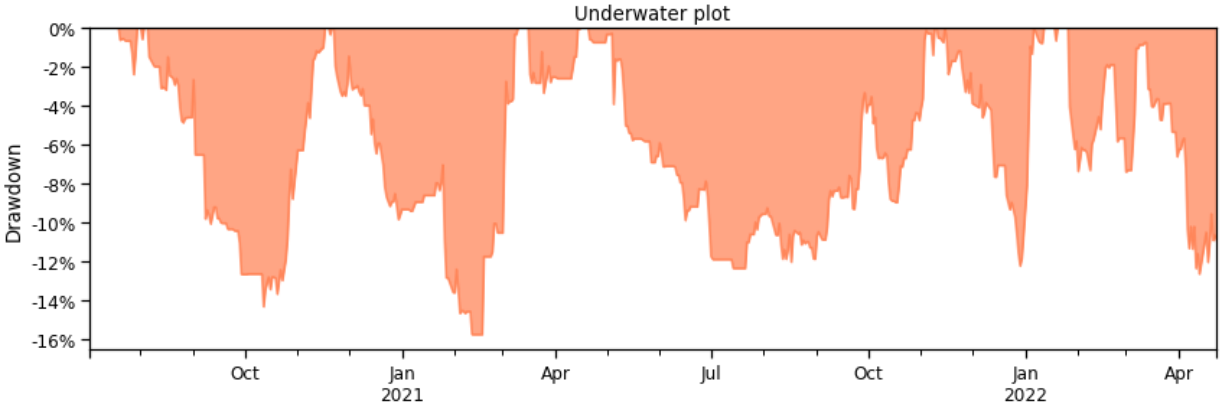

Rolling portfolio beta to daily_return
6-mo
12-mo
Beta
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

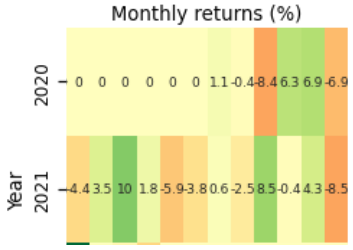

Rolling volatility (6-month)
Volatility
Benchmark volatility
0.20
0.18
0.15
0.13
0.10

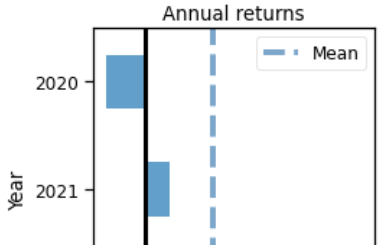

Vc
Average volatility
0.08
0.05
0.03
0.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

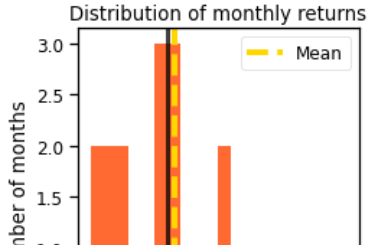

Rolling Sharpe ratio (6-month)
Sharpe
Average
Sharpe ratio
3.00
2.00
1.00
0.00
-1.00
-2.00
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

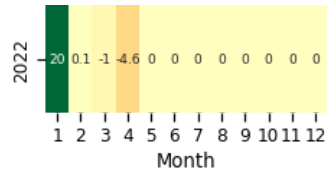

Top 5 drawdown periods
Portfolio
Cumulative returns
1.25
1.20
1.15
1.10
1.05
1.00
0.95
0.90
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

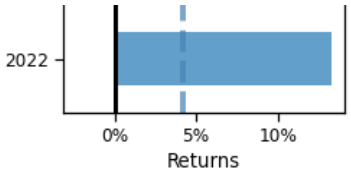

Underwater plot
Drawdown
0%
-2%
-4%
-6%
-8%
-10%
-12%
-14%
-16%
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

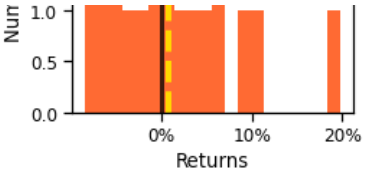

Monthly returns (%)
Year
2020
2021
0 0 0 0 0 0 1.1 -0.4 -8.4 6.3 6.9 -6.9
-4.4 3.5 10 1.8 -5.9 -3.8 0.6 -2.5 8.5 -0.4 4.3 -8.5

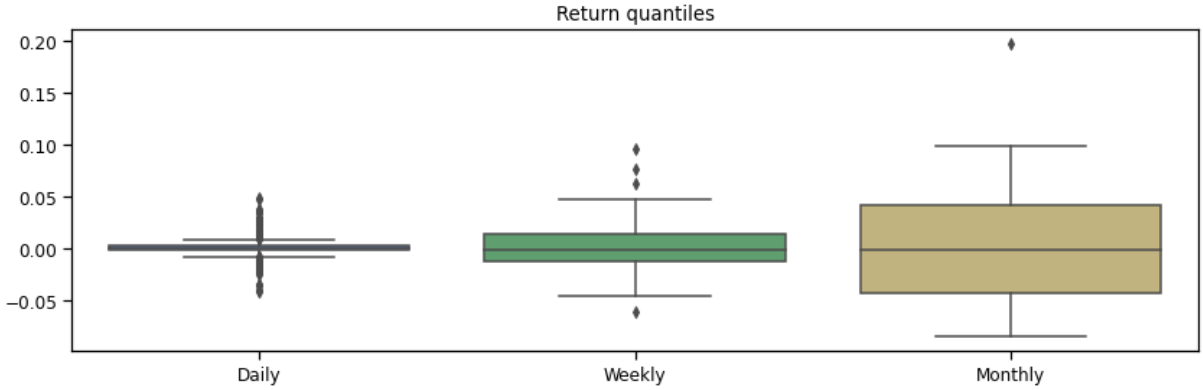

Annual returns
Mean
Year
2020
2021

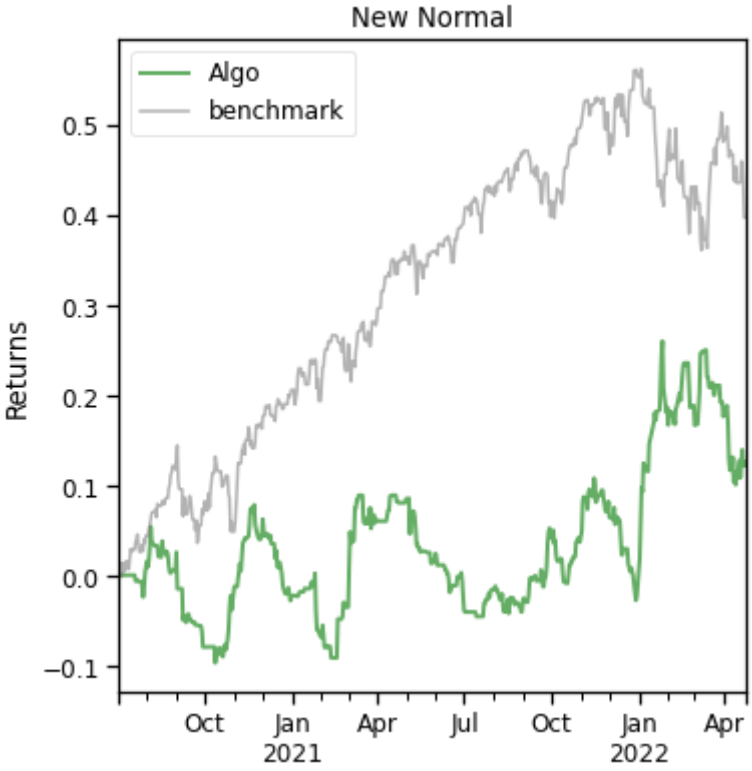

Distribution of monthly returns
Mean
3.0
2.5
2.0
1.5


2022
20 0.1 -1 -4.6 0 0 0 0 0 0 0 0
1 2 3 4 5 6 7 8 9 10 11 12
Month


2022
0%
5%
10%
Returns


1.0
0.5
0.0
0%
10%
20%
Returns


Return quantiles
0.20
0.15
0.10
0.05
0.00
−0.05
Daily
Weekly
Monthly


New Normal
Algo
benchmark
Returns
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
Oct
Jan
2021
Apr
Jul
Oct
Jan
2022
Apr

In [ ]:

In [ ]: