Installing packages

```
pip install git+https://github.com/Techtonique/nnetsauce.git

pip install matplotlib==3.1.3
```

Obtaining predictions from nnetsauce's MTS

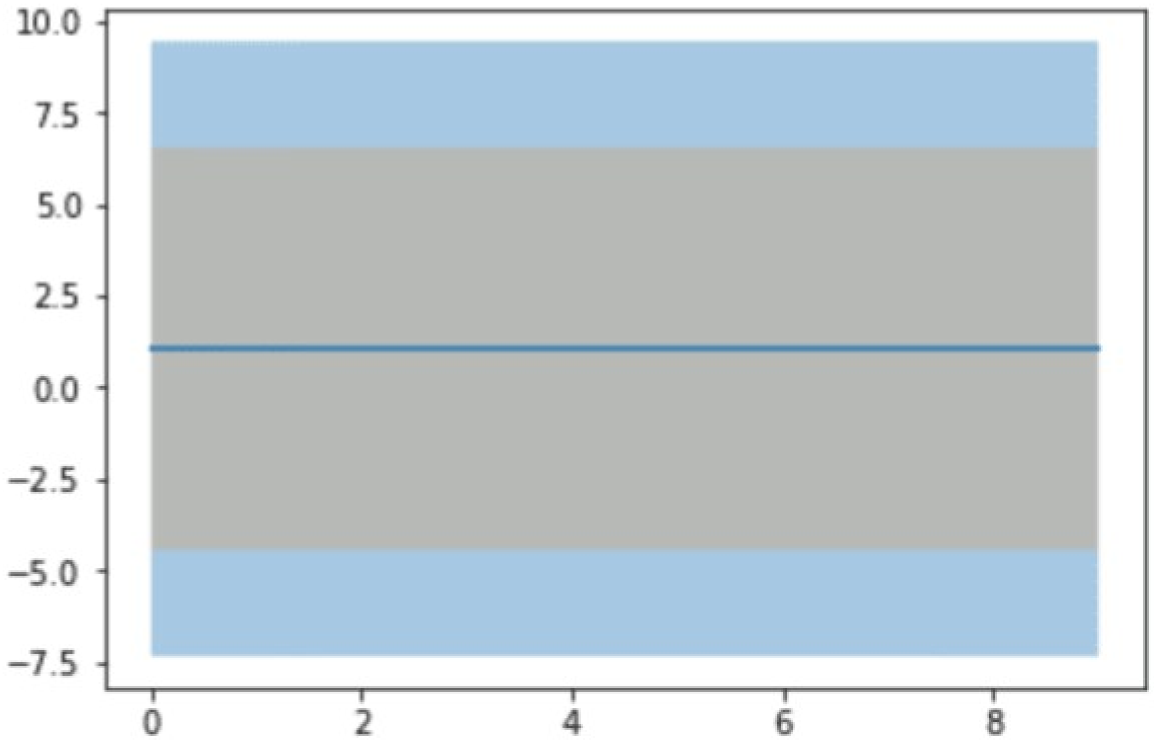
```
import nnetsauce as ns
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets, metrics
from sklearn import linear_model


# a simple example with 3 made-up multivariate time series
np.random.seed(123)
M = np.random.rand(10, 3)
M[:,0] = 10*M[:,0]
M[:,2] = 25*M[:,2]
M[:,1] = M[:,2]/23 + 0.5

print("Initial series")
print(M)
print("\n")

# using a Bayesian model along with nnetsauce's MTS
regr = linear_model.BayesianRidge()
obj_MTS = ns.MTS(regr, lags = 2, n_hidden_features=5)
obj_MTS.fit(M)

print("mean forecast -----")
preds = obj_MTS.predict(h=10)
print(preds)
print("\n")

# confidence level = 80%
# makes the assumption of Gaussian uncertainty and works
# only if the shared supervised learning model has an argument
`return_std`
# in method `predict`
print("predict with credible intervals and confidence level = 80%
-----")
preds_80 = obj_MTS.predict(h=10, return_std=True, level=80)
print(preds_80)
print("\n")

# confidence level = 95%
# makes the assumption of Gaussian uncertainty and works
# only if the shared supervised learning model has an argument
```

```
`return_std`
# in method `predict`
print("predict with credible intervals and confidence level = 95%
-----")
preds_95 = obj_MTS.predict(h=10, return_std=True, level=95)
print(preds_95)
print("\n")
```

```
Initial series
[[ 6.96469186  0.74657767  5.67128634]
 [ 5.51314769  0.95989833 10.5776615 ]
 [ 9.80764198  1.02275207 12.02329754]
 [ 3.92117518  1.29244533 18.22624268]
 [ 4.38572245  0.9326568   9.95110638]
 [ 7.37995406  0.69070843  4.3862939 ]
 [ 5.31551374  1.18956626 15.86002396]
 [ 8.49431794  1.16415599 15.27558777]
 [ 7.22443383  0.89324854  9.04471639]
 [ 2.28263231  1.18584361 15.7744031 ]]


mean forecast -----
[[ 6.09958391  1.0445821  12.56586894]
 [ 6.09858969  1.04358789 12.56487472]
 [ 6.10102293  1.04602113 12.56730796]
 [ 6.10101234  1.04601054 12.56729738]
 [ 6.10097759  1.04597578 12.56726262]
 [ 6.10097809  1.04597629 12.56726312]
 [ 6.10097859  1.04597678 12.56726362]
 [ 6.10097857  1.04597677 12.5672636 ]
 [ 6.10097857  1.04597676 12.5672636 ]
 [ 6.10097857  1.04597676 12.5672636 ]]


predict with credible intervals and confidence level = 80% -----
{'mean': array([[ 6.09958391,  1.0445821 , 12.56586894],
       [ 6.09858969,  1.04358789, 12.56487472],
       [ 6.10102293,  1.04602113, 12.56730796],
       [ 6.10101234,  1.04601054, 12.56729738],
       [ 6.10097759,  1.04597578, 12.56726262],
       [ 6.10097809,  1.04597629, 12.56726312],
       [ 6.10097859,  1.04597678, 12.56726362],
       [ 6.10097857,  1.04597677, 12.5672636 ],
       [ 6.10097857,  1.04597676, 12.5672636 ],
       [ 6.10097857,  1.04597676, 12.5672636 ]]), 'std':
array([[4.28508739, 4.28508739, 4.28508739],
       [4.28508739, 4.28508739, 4.28508739],
       [4.28508739, 4.28508739, 4.28508739],
       [4.2850874 , 4.2850874 , 4.2850874 ],
       [4.28508736, 4.28508736, 4.28508736],
       [4.2850873 , 4.2850873 , 4.2850873 ],
       [4.28509003, 4.28509003, 4.28509003],
       [4.28509202, 4.28509202, 4.28509202],
       [4.28630194, 4.28630194, 4.28630194],
       [4.28696269, 4.28696269, 4.28696269]]), 'lower': array([[
0.60802345, -4.44697835,  7.07430848],
       [ 0.60702924, -4.44797257,  7.07331427],
       [ 0.60946248, -4.44553933,  7.07574751],
       [ 0.60945189, -4.44554992,  7.07573692],
       [ 0.60941718, -4.44558463,  7.07570221],
       [ 0.60941775, -4.44558405,  7.07570278],
       [ 0.60941475, -4.44558705,  7.07569978],
       [ 0.60941219, -4.44558962,  7.07569722],
       [ 0.6078616 , -4.4471402 ,  7.07414663],
       [ 0.60701482, -4.44798698,  7.07329985]]), 'upper':
array([[11.59114437,  6.53614256, 18.0574294 ],
       [11.59015015,  6.53514835, 18.05643518],
       [11.59258339,  6.53758159, 18.05886842],
       [11.5925728 ,  6.537571  , 18.05885783],
       [11.592538  ,  6.53753619, 18.05882303],
       [11.59253843,  6.53753663, 18.05882347],
       [11.59254242,  6.53754061, 18.05882745],
       [11.59254496,  6.53754315, 18.05882999],
       [11.59409553,  6.53909373, 18.06038056],
       [11.59494231,  6.53994051, 18.06122734]])}


predict with credible intervals and confidence level = 95% -----
{'mean': array([[ 6.09958391,  1.0445821 , 12.56586894],
       [ 6.09858969,  1.04358789, 12.56487472],
       [ 6.10102293,  1.04602113, 12.56730796],
       [ 6.10101234,  1.04601054, 12.56729738],
       [ 6.10097759,  1.04597578, 12.56726262],
       [ 6.10097809,  1.04597629, 12.56726312],
       [ 6.10097859,  1.04597678, 12.56726362],
       [ 6.10097857,  1.04597677, 12.5672636 ],
       [ 6.10097857,  1.04597676, 12.5672636 ],
       [ 6.10097857,  1.04597676, 12.5672636 ]]), 'std':
array([[4.28508739, 4.28508739, 4.28508739],
       [4.28508739, 4.28508739, 4.28508739],
       [4.28508739, 4.28508739, 4.28508739],
       [4.2850874 , 4.2850874 , 4.2850874 ],
       [4.28508736, 4.28508736, 4.28508736],
       [4.2850873 , 4.2850873 , 4.2850873 ],
       [4.28509003, 4.28509003, 4.28509003],
       [4.28509202, 4.28509202, 4.28509202],
       [4.28630194, 4.28630194, 4.28630194],
       [4.28696269, 4.28696269, 4.28696269]]), 'lower':
array([[-2.29903305, -7.35403486,  4.16725198],
       [-2.30002727, -7.35502907,  4.16625776],
       [-2.29759403, -7.35259583,  4.168691  ],
       [-2.29760462, -7.35260643,  4.16868041],
       [-2.2976393 , -7.35264111,  4.16864573],
       [-2.29763869, -7.3526405 ,  4.16864634],
       [-2.29764354, -7.35264535,  4.16864149],
       [-2.29764745, -7.35264926,  4.16863758],
       [-2.30001887, -7.35502067,  4.16626616],
       [-2.30131391, -7.35631571,  4.16497112]]), 'upper':
array([[14.49820087,  9.44319907, 20.9644859 ],
       [14.49720666,  9.44220485, 20.96349169],
       [14.4996399 ,  9.44463809, 20.96592493],
       [14.49962931,  9.4446275 , 20.96591434],
       [14.49959448,  9.44459267, 20.96587951],
       [14.49959488,  9.44459307, 20.96587991],
       [14.49960071,  9.4445989 , 20.96588574],
       [14.4996046 ,  9.44460279, 20.96588963],
       [14.501976  ,  9.44697419, 20.96826103],
       [14.50327104,  9.44826923, 20.96955607]])}
```

Visualizing credible (because Bayesian, “credible”) intervals

```
j0 = 1 # choice of time series index

x = np.linspace(0, 9, 10)
y_est = preds_95['mean'][:,j0]
y_est_upper = preds_95['upper'][:,j0]
y_est_lower = preds_95['lower'][:,j0]
y_est_upper2 = preds_80['upper'][:,j0]
y_est_lower2 = preds_80['lower'][:,j0]


fig, ax = plt.subplots()
ax.plot(x, y_est, '-')
ax.fill_between(x, y_est_lower, y_est_upper, alpha=0.4)
ax.fill_between(x, y_est_lower2, y_est_upper2, alpha=0.2)
```


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
0
2
4
6
8