Regression Mad Science

March 25, 2018

In [1]:
```
import os
import warnings
import copy
import time

import numpy as np
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
from plotly.graph_objs import *
import plotly.figure_factory as ff

from sklearn.model_selection import train_test_split, KFold
from sklearn.metrics import explained_variance_score, r2_score
from sklearn.linear_model import LinearRegression, Lasso, lasso_path, lars_path, LassoLa
from sklearn.neural_network import MLPRegressor

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3' #Hide messy TensorFlow warnings
warnings.filterwarnings("ignore") #Hide messy Numpy warnings

import keras
from keras.layers.core import Dense, Activation, Dropout
from keras.layers import Input
from keras.models import Model

from keras.layers.recurrent import LSTM
from keras.regularizers import l1
from keras.models import Sequential
from keras.models import load_model

init_notebook_mode(connected=True)
```

```
Using TensorFlow backend.
```

In [2]:
```
# create a data set, sin wave plus random noise
nobs = 1000
x = np.linspace(0, 3*np.pi, num=nobs)
y = -np.cos(x) + x/(3*np.pi) + np.random.normal(0, 0.25, nobs)
```

In [3]:
```
# chart it

def mychart(*args):

    # pass some 2d n x 1 arrays, x, y, z

    # 1st array is independent vars
    # reshape to 1 dimensional array
    x = args[0].reshape(-1)

    # following are dependent vars plotted on y axis
    data = []
    for i in range(1, len(args)):
        data.append(Scatter(x=x,
                            y=args[i].reshape(-1),
                            mode = 'markers'))

    layout = Layout(
        yaxis=dict(
            autorange=True))

    fig = Figure(data=data, layout=layout)

    return iplot(fig, image='png') # png to save notebook w/static image

mychart(x,y)
```

```
<IPython.core.display.HTML object>
```

In [4]:
```
# fit with sklearn MLPRegressor

layer1_sizes=[1,2,3,4]
layer2_sizes=[1,2,3,4]
import itertools
from plotly import tools

def run_grid(build_model_fn, layer1_sizes, layer2_sizes, x, y):
    nrows = len(layer1_sizes)
    ncols = len(layer2_sizes)

    hyperparameter_list = list(itertools.product(layer1_sizes, layer2_sizes))
    subplot_titles = ["%d units, %d units" %
                      (layer1_size, layer2_size) for (layer1_size, layer2_size) in hyper

    fig = tools.make_subplots(rows=nrows,
                              cols=ncols,
                              subplot_titles=subplot_titles)
```

```
        for count, (layer1_size, layer2_size) in enumerate(hyperparameter_list):
            print("Layer 1 units: %d, Layer 2 units %d:" % (layer1_size, layer2_size))

            print("Running experiment %d of %d : %d %d" % (count+1, len(hyperparameter_list)
            model = build_model_fn(hidden_layer_sizes=(layer1_size, layer2_size),
                                   max_iter=10000, tol=1e-8,
                                   solver='lbfgs')
            x = x.reshape(-1,1)
            model.fit(x,y)
            z = model.predict(x)
            trace = Scatter(
                x = x.reshape(-1),
                y = z.reshape(-1),
                name = 'fit',
                mode = 'markers',
                marker = dict(size = 2)
            )
            fig.append_trace(trace, count // nrows + 1, count % ncols +1)
        return(iplot(fig))

    run_grid(MLPRegressor, layer1_sizes, layer2_sizes, x, y)
```

```
This is the format of your plot grid:
[ (1,1) x1,y1 ]    [ (1,2) x2,y2 ]    [ (1,3) x3,y3 ]    [ (1,4) x4,y4 ]
[ (2,1) x5,y5 ]    [ (2,2) x6,y6 ]    [ (2,3) x7,y7 ]    [ (2,4) x8,y8 ]
[ (3,1) x9,y9 ]    [ (3,2) x10,y10 ]  [ (3,3) x11,y11 ]  [ (3,4) x12,y12 ]
[ (4,1) x13,y13 ]  [ (4,2) x14,y14 ]  [ (4,3) x15,y15 ]  [ (4,4) x16,y16 ]

Layer 1 units: 1, Layer 2 units 1:
Running experiment 1 of 16 : 1 1
Layer 1 units: 1, Layer 2 units 2:
Running experiment 2 of 16 : 1 2
Layer 1 units: 1, Layer 2 units 3:
Running experiment 3 of 16 : 1 3
Layer 1 units: 1, Layer 2 units 4:
Running experiment 4 of 16 : 1 4
Layer 1 units: 2, Layer 2 units 1:
Running experiment 5 of 16 : 2 1
Layer 1 units: 2, Layer 2 units 2:
Running experiment 6 of 16 : 2 2
Layer 1 units: 2, Layer 2 units 3:
Running experiment 7 of 16 : 2 3
Layer 1 units: 2, Layer 2 units 4:
Running experiment 8 of 16 : 2 4
Layer 1 units: 3, Layer 2 units 1:
Running experiment 9 of 16 : 3 1
Layer 1 units: 3, Layer 2 units 2:
```

```
Running experiment 10 of 16 : 3 2
Layer 1 units: 3, Layer 2 units 3:
Running experiment 11 of 16 : 3 3
Layer 1 units: 3, Layer 2 units 4:
Running experiment 12 of 16 : 3 4
Layer 1 units: 4, Layer 2 units 1:
Running experiment 13 of 16 : 4 1
Layer 1 units: 4, Layer 2 units 2:
Running experiment 14 of 16 : 4 2
Layer 1 units: 4, Layer 2 units 3:
Running experiment 15 of 16 : 4 3
Layer 1 units: 4, Layer 2 units 4:
Running experiment 16 of 16 : 4 4
```

In [5]:
```
# sklearn MLP regression didn't work well at all
# let's build our own keras model by wrappping it in sklearn interface

def build_ols_model(input_size = 1, hidden_layer_sizes=[4]):

    main_input = Input(shape=(input_size,), dtype='float32', name='main_input')

    lastlayer=main_input
    for layer_size in hidden_layer_sizes:
        lastlayer = Dense(layer_size,
                          kernel_initializer=keras.initializers.glorot_normal(seed=None)
                          bias_initializer=keras.initializers.glorot_normal(seed=None),
                          activation='relu')(lastlayer)

    output = Dense(1, activation='linear')(lastlayer)

    model = Model(inputs=[main_input], outputs=[output])

    model.compile(loss="mean_squared_error",
                  optimizer=keras.optimizers.Adam()
                 )
    print(model.summary())

    return model
```

In [6]:
```
EPOCHS=501
BATCH_SIZE=32
def run_experiment (model, x, y):

    models = []
```

```
        losses = []

        for epoch in range(EPOCHS):
            fit = model.fit(
                x,
                y,
                batch_size=BATCH_SIZE,
                epochs=1,
                verbose=0
            )

            train_loss = fit.history['loss'][-1]

            losses.append(train_loss)
            models.append(copy.copy(model))

            bestloss_index = np.argmin(losses)
            bestloss_value = losses[bestloss_index]
            if epoch % 10 == 0:
                print("%s Epoch %d of %d Loss %.6f Best Loss %.6f" % (time.strftime("%H:%M:%
                                                                       epoch,
                                                                       EPOCHS, train_loss,
                                                                       bestloss_value))
            # stop if loss rises by 20% from best
            if train_loss / bestloss_value > 1.2:
                print("%s Stopping..." % (time.strftime("%H:%M:%S")))
                break

        print ("%s Best training loss epoch %d, value %f" % (time.strftime("%H:%M:%S"), best
        model = models[bestloss_index]

        train_score = model.evaluate(x, y)
        print(train_score)
        print("%s Train MSE: %.6f" % (time.strftime("%H:%M:%S"), train_score))
        print("%s Train R-squared: %.6f" % (time.strftime("%H:%M:%S"), 1-train_score/y.var()

        return mychart(x, y, model.predict(x))
```

```
In [7]: model = build_ols_model(hidden_layer_sizes=[16])
        run_experiment(model, x, y)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 16)                32
_________________________________________________________________
```

```
dense_2 (Dense)              (None, 1)                 17
=================================================================
Total params: 49
Trainable params: 49
Non-trainable params: 0
_________________________________________________________________
None
11:44:23 Epoch 0 of 501 Loss 15.426664 Best Loss 15.426664
11:44:25 Epoch 10 of 501 Loss 0.509588 Best Loss 0.509588
11:44:26 Epoch 20 of 501 Loss 0.443615 Best Loss 0.443615
11:44:28 Epoch 30 of 501 Loss 0.441543 Best Loss 0.441543
11:44:30 Epoch 40 of 501 Loss 0.440950 Best Loss 0.440836
11:44:31 Epoch 50 of 501 Loss 0.440622 Best Loss 0.440603
11:44:33 Epoch 60 of 501 Loss 0.440766 Best Loss 0.440427
11:44:35 Epoch 70 of 501 Loss 0.440953 Best Loss 0.440094
11:44:36 Epoch 80 of 501 Loss 0.440382 Best Loss 0.440094
11:44:38 Epoch 90 of 501 Loss 0.440569 Best Loss 0.439991
11:44:40 Epoch 100 of 501 Loss 0.440304 Best Loss 0.439974
11:44:41 Epoch 110 of 501 Loss 0.441213 Best Loss 0.439974
11:44:43 Epoch 120 of 501 Loss 0.439972 Best Loss 0.439972
11:44:45 Epoch 130 of 501 Loss 0.440429 Best Loss 0.439972
11:44:47 Epoch 140 of 501 Loss 0.440240 Best Loss 0.439607
11:44:48 Epoch 150 of 501 Loss 0.441049 Best Loss 0.439607
11:44:50 Epoch 160 of 501 Loss 0.441038 Best Loss 0.439607
11:44:52 Epoch 170 of 501 Loss 0.442192 Best Loss 0.439607
11:44:53 Epoch 180 of 501 Loss 0.441794 Best Loss 0.439607
11:44:55 Epoch 190 of 501 Loss 0.442280 Best Loss 0.439607
11:44:57 Epoch 200 of 501 Loss 0.441512 Best Loss 0.439607
11:44:58 Epoch 210 of 501 Loss 0.441798 Best Loss 0.439607
11:45:00 Epoch 220 of 501 Loss 0.441498 Best Loss 0.439607
11:45:02 Epoch 230 of 501 Loss 0.441671 Best Loss 0.439607
11:45:03 Epoch 240 of 501 Loss 0.440586 Best Loss 0.439607
11:45:05 Epoch 250 of 501 Loss 0.440256 Best Loss 0.439607
11:45:07 Epoch 260 of 501 Loss 0.440907 Best Loss 0.439607
11:45:08 Epoch 270 of 501 Loss 0.442421 Best Loss 0.439607
11:45:10 Epoch 280 of 501 Loss 0.440259 Best Loss 0.439607
11:45:12 Epoch 290 of 501 Loss 0.441640 Best Loss 0.439607
11:45:13 Epoch 300 of 501 Loss 0.440246 Best Loss 0.439607
11:45:15 Epoch 310 of 501 Loss 0.440654 Best Loss 0.439607
11:45:17 Epoch 320 of 501 Loss 0.442664 Best Loss 0.439607
11:45:19 Epoch 330 of 501 Loss 0.441344 Best Loss 0.439607
11:45:20 Epoch 340 of 501 Loss 0.441623 Best Loss 0.439607
11:45:22 Epoch 350 of 501 Loss 0.440543 Best Loss 0.439607
11:45:24 Epoch 360 of 501 Loss 0.442160 Best Loss 0.439607
11:45:25 Epoch 370 of 501 Loss 0.440666 Best Loss 0.439607
11:45:27 Epoch 380 of 501 Loss 0.442044 Best Loss 0.439607
11:45:29 Epoch 390 of 501 Loss 0.440488 Best Loss 0.439607
11:45:30 Epoch 400 of 501 Loss 0.440266 Best Loss 0.439607
```

```
11:45:32 Epoch 410 of 501 Loss 0.445000 Best Loss 0.439607
11:45:34 Epoch 420 of 501 Loss 0.441947 Best Loss 0.439607
11:45:35 Epoch 430 of 501 Loss 0.440754 Best Loss 0.439607
11:45:37 Epoch 440 of 501 Loss 0.440188 Best Loss 0.439607
11:45:39 Epoch 450 of 501 Loss 0.442832 Best Loss 0.439607
11:45:41 Epoch 460 of 501 Loss 0.440830 Best Loss 0.439607
11:45:42 Epoch 470 of 501 Loss 0.440467 Best Loss 0.439607
11:45:44 Epoch 480 of 501 Loss 0.440286 Best Loss 0.439607
11:45:46 Epoch 490 of 501 Loss 0.440185 Best Loss 0.439607
11:45:47 Epoch 500 of 501 Loss 0.433730 Best Loss 0.433730
11:45:47 Best training loss epoch 500, value 0.433730
1000/1000 [==============================] - 0s 73us/step
0.4310117630958557
11:45:47 Train MSE: 0.431012
11:45:47 Train R-squared: 0.371699
```

```
<IPython.core.display.HTML object>
```

In [8]:
```python
# wrap our keras model in sklearn wrapper so we can run grid like above MLPRegressor
from keras.wrappers.scikit_learn import KerasRegressor

# closure for sklearn wrapper
def make_build(layer_sizes):
    def myclosure():
        return build_ols_model(input_size=1, hidden_layer_sizes=layer_sizes)
    return myclosure

def make_keras_model(**kwargs):
    # make a function that takes hidden layer sizes kwarg and returns estimator of corre
    build_fn = make_build(kwargs['hidden_layer_sizes'])

    keras_estimator = KerasRegressor(build_fn=build_fn,
                                     nb_epoch=5000,
                                     batch_size=32,
                                     verbose=1)
    return keras_estimator
```

In [9]:
```python
run_grid(make_keras_model, layer1_sizes, layer2_sizes, x, y)
# this also doesn't perform well
# takeaway: feedforward NN sucks for regression
```

```
This is the format of your plot grid:
[ (1,1) x1,y1 ]    [ (1,2) x2,y2 ]    [ (1,3) x3,y3 ]    [ (1,4) x4,y4 ]
[ (2,1) x5,y5 ]    [ (2,2) x6,y6 ]    [ (2,3) x7,y7 ]    [ (2,4) x8,y8 ]
[ (3,1) x9,y9 ]    [ (3,2) x10,y10 ]  [ (3,3) x11,y11 ]  [ (3,4) x12,y12 ]
```

```
[ (4,1) x13,y13 ]  [ (4,2) x14,y14 ]  [ (4,3) x15,y15 ]  [ (4,4) x16,y16 ]

Layer 1 units: 1, Layer 2 units 1:
Running experiment 1 of 16 : 1 1
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 2
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 2
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 2
=================================================================
Total params: 6
Trainable params: 6
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 291us/step - loss: 0.9116
1000/1000 [==============================] - 0s 55us/step
Layer 1 units: 1, Layer 2 units 2:
Running experiment 2 of 16 : 1 2
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 2
_________________________________________________________________
dense_7 (Dense)              (None, 2)                 4
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 3
=================================================================
Total params: 9
Trainable params: 9
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 283us/step - loss: 12.2443
1000/1000 [==============================] - 0s 79us/step
Layer 1 units: 1, Layer 2 units 3:
Running experiment 3 of 16 : 1 3
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_9 (Dense)              (None, 1)                 2
_________________________________________________________________
dense_10 (Dense)             (None, 3)                 6
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 4
=================================================================
Total params: 12
Trainable params: 12
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 297us/step - loss: 1.5040
1000/1000 [==============================] - 0s 73us/step
Layer 1 units: 1, Layer 2 units 4:
Running experiment 4 of 16 : 1 4
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_12 (Dense)             (None, 1)                 2
_________________________________________________________________
dense_13 (Dense)             (None, 4)                 8
_________________________________________________________________
dense_14 (Dense)             (None, 1)                 5
=================================================================
Total params: 15
Trainable params: 15
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 330us/step - loss: 0.8987
1000/1000 [==============================] - 0s 95us/step
Layer 1 units: 2, Layer 2 units 1:
Running experiment 5 of 16 : 2 1
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_15 (Dense)             (None, 2)                 4
_________________________________________________________________
dense_16 (Dense)             (None, 1)                 3
```

```
_________________________________________________________________
dense_17 (Dense)             (None, 1)                 2
=================================================================
Total params: 9
Trainable params: 9
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 315us/step - loss: 0.8781
1000/1000 [==============================] - 0s 86us/step
Layer 1 units: 2, Layer 2 units 2:
Running experiment 6 of 16 : 2 2
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_18 (Dense)             (None, 2)                 4
_________________________________________________________________
dense_19 (Dense)             (None, 2)                 6
_________________________________________________________________
dense_20 (Dense)             (None, 1)                 3
=================================================================
Total params: 13
Trainable params: 13
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 341us/step - loss: 0.9106
1000/1000 [==============================] - 0s 95us/step
Layer 1 units: 2, Layer 2 units 3:
Running experiment 7 of 16 : 2 3
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_21 (Dense)             (None, 2)                 4
_________________________________________________________________
dense_22 (Dense)             (None, 3)                 9
_________________________________________________________________
dense_23 (Dense)             (None, 1)                 4
=================================================================
Total params: 17
Trainable params: 17
Non-trainable params: 0
```

```
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 338us/step - loss: 1.2137
1000/1000 [==============================] - 0s 114us/step
Layer 1 units: 2, Layer 2 units 4:
Running experiment 8 of 16 : 2 4
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_24 (Dense)             (None, 2)                 4
_________________________________________________________________
dense_25 (Dense)             (None, 4)                 12
_________________________________________________________________
dense_26 (Dense)             (None, 1)                 5
=================================================================
Total params: 21
Trainable params: 21
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 375us/step - loss: 0.7322
1000/1000 [==============================] - 0s 106us/step
Layer 1 units: 3, Layer 2 units 1:
Running experiment 9 of 16 : 3 1
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_27 (Dense)             (None, 3)                 6
_________________________________________________________________
dense_28 (Dense)             (None, 1)                 4
_________________________________________________________________
dense_29 (Dense)             (None, 1)                 2
=================================================================
Total params: 12
Trainable params: 12
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 415us/step - loss: 0.5695
1000/1000 [==============================] - 0s 132us/step
Layer 1 units: 3, Layer 2 units 2:
```

```
Running experiment 10 of 16 : 3 2
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_30 (Dense)             (None, 3)                 6
_________________________________________________________________
dense_31 (Dense)             (None, 2)                 8
_________________________________________________________________
dense_32 (Dense)             (None, 1)                 3
=================================================================
Total params: 17
Trainable params: 17
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 392us/step - loss: 0.7649
1000/1000 [==============================] - 0s 125us/step
Layer 1 units: 3, Layer 2 units 3:
Running experiment 11 of 16 : 3 3
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_33 (Dense)             (None, 3)                 6
_________________________________________________________________
dense_34 (Dense)             (None, 3)                 12
_________________________________________________________________
dense_35 (Dense)             (None, 1)                 4
=================================================================
Total params: 22
Trainable params: 22
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 421us/step - loss: 22.1917
1000/1000 [==============================] - 0s 116us/step
Layer 1 units: 3, Layer 2 units 4:
Running experiment 12 of 16 : 3 4
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
```

```
dense_36 (Dense)             (None, 3)                 6
_________________________________________________________________
dense_37 (Dense)             (None, 4)                 16
_________________________________________________________________
dense_38 (Dense)             (None, 1)                 5
=================================================================
Total params: 27
Trainable params: 27
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 411us/step - loss: 5.4467
1000/1000 [==============================] - 0s 156us/step
Layer 1 units: 4, Layer 2 units 1:
Running experiment 13 of 16 : 4 1
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_39 (Dense)             (None, 4)                 8
_________________________________________________________________
dense_40 (Dense)             (None, 1)                 5
_________________________________________________________________
dense_41 (Dense)             (None, 1)                 2
=================================================================
Total params: 15
Trainable params: 15
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 482us/step - loss: 0.9039
1000/1000 [==============================] - 0s 140us/step
Layer 1 units: 4, Layer 2 units 2:
Running experiment 14 of 16 : 4 2
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_42 (Dense)             (None, 4)                 8
_________________________________________________________________
dense_43 (Dense)             (None, 2)                 10
_________________________________________________________________
dense_44 (Dense)             (None, 1)                 3
=================================================================
```

```
Total params: 21
Trainable params: 21
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 0s 475us/step - loss: 0.9101
1000/1000 [==============================] - 0s 143us/step
Layer 1 units: 4, Layer 2 units 3:
Running experiment 15 of 16 : 4 3
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_45 (Dense)             (None, 4)                 8
_________________________________________________________________
dense_46 (Dense)             (None, 3)                 15
_________________________________________________________________
dense_47 (Dense)             (None, 1)                 4
=================================================================
Total params: 27
Trainable params: 27
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
1000/1000 [==============================] - 1s 546us/step - loss: 0.7803
1000/1000 [==============================] - 0s 192us/step
Layer 1 units: 4, Layer 2 units 4:
Running experiment 16 of 16 : 4 4
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
main_input (InputLayer)      (None, 1)                 0
_________________________________________________________________
dense_48 (Dense)             (None, 4)                 8
_________________________________________________________________
dense_49 (Dense)             (None, 4)                 20
_________________________________________________________________
dense_50 (Dense)             (None, 1)                 5
=================================================================
Total params: 33
Trainable params: 33
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/1
```

```
1000/1000 [==============================] - 0s 500us/step - loss: 0.5420
1000/1000 [==============================] - 0s 192us/step
```

```
In [10]: # try RandomForestRegressor
         from sklearn.ensemble import RandomForestRegressor
         from sklearn.model_selection import RandomizedSearchCV

         rf = RandomForestRegressor(random_state = 42)
         from pprint import pprint
         # Look at parameters used by our current forest
         print('Parameters currently in use:\n')
         pprint(rf.get_params())

         # First 2 are most important
         # Number of trees in random forest
         n_estimators = [int(a) for a in np.linspace(start = 200, stop = 2000, num = 10)]
         # Number of features to consider at every split
         max_features = ['auto', 'sqrt']
         # Maximum number of levels in tree
         max_depth = [int(a) for a in np.linspace(10, 110, num = 11)]
         max_depth.append(None)
         # Minimum number of samples required to split a node
         min_samples_split = [2, 5, 10]
         # Minimum number of samples required at each leaf node
         min_samples_leaf = [1, 2, 4]
         # Method of selecting samples for training each tree
         bootstrap = [True, False]

         # Create the random grid
         print("Search grid:")
         random_grid = {'n_estimators': n_estimators,
                        'max_features': max_features,
                        'max_depth': max_depth,
                        'min_samples_split': min_samples_split,
                        'min_samples_leaf': min_samples_leaf,
                        'bootstrap': bootstrap}
         pprint(random_grid)
```

```
Parameters currently in use:

{'bootstrap': True,
 'criterion': 'mse',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'min_impurity_decrease': 0.0,
 'min_impurity_split': None,
```

```
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 10,
 'n_jobs': 1,
 'oob_score': False,
 'random_state': 42,
 'verbose': 0,
 'warm_start': False}
Search grid:
{'bootstrap': [True, False],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]}
```

In [11]:
```
x.shape
```

Out[11]:
```
(1000,)
```

In [12]:
```
x=x.reshape(x.shape[0],1)
y=y.reshape(y.shape[0],1)
```

In [13]:
```
rf = RandomForestRegressor()
rf.fit(x, y)
z = rf.predict(x)
mychart(x, y, z)
```

```
<IPython.core.display.HTML object>
```

In [14]:
```
# Use the random grid to search for best hyperparameters
# First create the base model to tune
rf = RandomForestRegressor()
# Random search of parameters, using 3 fold cross validation,
# search across 100 different combinations, and use all available cores
rf_random = RandomizedSearchCV(estimator = rf,
                               param_distributions = random_grid,
                               n_iter = 100,
                               cv = 3, verbose=2,
                               random_state=42,
                               n_jobs = 1)
# Fit the random search model
rf_random.fit(x, y)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=
```

```
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.7s remaining:    0.0s


[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=auto, min_samples_split
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
```

```
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
```

```
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1200, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=200, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spl
```

```
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=1200, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1800, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1400, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=1400, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=400, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
```

```
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=auto, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1000, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=800, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=400, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
```

```
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=1600, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=1000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=600, max_features=sqrt, min_samples_split
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=2000, max_features=sqrt, min_samples_spl
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=1, n_estimators=800, max_features=sqrt, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split=
[CV]  bootstrap=True, min_samples_leaf=4, n_estimators=600, max_features=auto, min_samples_split
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=600, max_features=sqrt, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_spli
```

```
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=2, n_estimators=400, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spli
[CV]  bootstrap=False, min_samples_leaf=1, n_estimators=1000, max_features=auto, min_samples_spl
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_split
[CV]  bootstrap=False, min_samples_leaf=4, n_estimators=200, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=sqrt, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli
[CV] bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_split
[CV]  bootstrap=True, min_samples_leaf=2, n_estimators=2000, max_features=auto, min_samples_spli


[Parallel(n_jobs=1)]: Done 300 out of 300 | elapsed:  8.3min finished
```

```
Out[14]: RandomizedSearchCV(cv=3, error_score='raise',
                   estimator=RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=No
                    max_features='auto', max_leaf_nodes=None,
                    min_impurity_decrease=0.0, min_impurity_split=None,
                    min_samples_leaf=1, min_samples_split=2,
                    min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                    oob_score=False, random_state=None, verbose=0, warm_start=False),
                   fit_params=None, iid=True, n_iter=100, n_jobs=1,
                   param_distributions={'bootstrap': [True, False], 'min_samples_leaf': [1, 2, 4
                   pre_dispatch='2*n_jobs', random_state=42, refit=True,
                   return_train_score='warn', scoring=None, verbose=2)
```

```
In [15]: rf_random.best_params_
         {'bootstrap': True,
          'max_depth': 70,
```

```
 'max_features': 'auto',
 'min_samples_leaf': 4,
 'min_samples_split': 10,
 'n_estimators': 400}
```

Out[15]:
```
{'bootstrap': True,
 'max_depth': 70,
 'max_features': 'auto',
 'min_samples_leaf': 4,
 'min_samples_split': 10,
 'n_estimators': 400}
```

In [16]:
```
rf = RandomForestRegressor(n_estimators=400,
                           max_features='auto',
                           max_depth=None,
                           min_samples_leaf=4,
                           min_samples_split=10,
                          )
rf.fit(x, y)
z = rf.predict(x)
mychart(x, y, z)
```

```
<IPython.core.display.HTML object>
```

In [17]:
```
rf.predict(np.random.uniform(low=0, high=8, size=10).reshape(10,1))
```

Out[17]:
```
array([-0.21986661,  1.40764951, -0.33172553,  1.40203883, -0.21986661,
       -0.31580421,  1.41862084,  0.06309863, -0.98201201,  0.57664036])
```

In [19]:
```
from scipy.interpolate import UnivariateSpline
spl = UnivariateSpline(x, y)
z = spl(x)
mychart(x, y, z)
```

```
<IPython.core.display.HTML object>
```

In [21]:
```
spl.set_smoothing_factor(94.0)
print(spl.get_knots())
z = spl(x)
mychart(x, y, z)
```

```
[0.         2.35855304 4.71710609 7.07565913 9.42477796]
```

```
<IPython.core.display.HTML object>
```

```
In [22]: from sklearn.metrics import mean_squared_error

         def spline_cv_test(smoothing_factor):
             kf = KFold(n_splits=5)
             errors = []
             for train_index, test_index in kf.split(x):
                 x_train, x_test = x[train_index], x[test_index]
                 y_train, y_test = y[train_index], y[test_index]
                 spline = UnivariateSpline(x_train, y_train)
                 spline.set_smoothing_factor(smoothing_factor)
                 y_pred_test = spline(x_test)
                 errors.append(mean_squared_error(y_test, y_pred_test))
                 return np.mean(np.array(errors))

         print(spline_cv_test(49.856))
         for sf in np.linspace(10, 110, num=101):
             print(sf, spline_cv_test(sf))
```

```
6.323464248167933
(10.0, 29139241297.354435)
(11.0, 26888054192.5276)
(12.0, 12581312461.24114)
(13.0, 24138738188.133846)
(14.0, 21455551677.259903)
(15.0, 21259883710.04871)
(16.0, 14779297197.220016)
(17.0, 40916686239.17037)
(18.0, 54077792589.58279)
(19.0, 61307837703.81329)
(20.0, 56340536992.4181)
(21.0, 53787054058.36845)
(22.0, 72776037115.01947)
(23.0, 71210739948.9296)
(24.0, 82548803412.35509)
(25.0, 105362949489.5807)
(26.0, 109784451763.66956)
(27.0, 93579774593.80069)
(28.0, 63364078853.069275)
(29.0, 34356665411.262726)
(30.0, 85541570240.12613)
(31.0, 71614182736.32814)
(32.0, 55795395272.728874)
(33.0, 44100931744.258064)
(34.0, 1041856210.285058)
(35.0, 1033213576.8894588)
(36.0, 1025434996.7409288)
(37.0, 868639279.2679269)
(38.0, 810230666.0885332)
```

```
(39.0, 122286009.38749148)
(40.0, 1368250.6692457518)
(41.0, 21474652.06020119)
(42.0, 18006378.273677878)
(43.0, 81313.15991493732)
(44.0, 125723.38298874408)
(45.0, 56980.94063085092)
(46.0, 27695.0094338631)
(47.0, 7183.534917414842)
(48.0, 2024.4051849757248)
(49.0, 4.2379255538503395)
(50.0, 6.712709188171314)
(51.0, 11.564035750842494)
(52.0, 10.864186968293035)
(53.0, 9.88416575770284)
(54.0, 15.154027426160027)
(55.0, 11.450273894651065)
(56.0, 10.038683749329609)
(57.0, 9.01910826856821)
(58.0, 8.180655612045198)
(59.0, 7.523481461059173)
(60.0, 6.9136535317497065)
(61.0, 6.390040756997094)
(62.0, 5.94683450277738)
(63.0, 5.505555680276249)
(64.0, 5.131031105379006)
(65.0, 4.770410769155931)
(66.0, 4.446097340385663)
(67.0, 4.144773332596627)
(68.0, 3.863548360875149)
(69.0, 3.6003935614382723)
(70.0, 3.3762453486124935)
(71.0, 3.152456192150913)
(72.0, 2.9431753954994377)
(73.0, 2.747202012290892)
(74.0, 2.563488830601798)
(75.0, 2.3911169397967185)
(76.0, 2.2292757019803697)
(77.0, 2.0772465961240063)
(78.0, 1.934389939365106)
(79.0, 1.8001338246757017)
(80.0, 1.673964819097554)
(81.0, 1.555420093472375)
(82.0, 1.4440807356398508)
(83.0, 1.3395660535933496)
(84.0, 1.2415287138038553)
(85.0, 1.1488471763606853)
(86.0, 1.063015457607016)
```

```
(87.0, 0.9894041064435083)
(88.0, 0.911425768825741)
(89.0, 0.8398305073897098)
(90.0, 0.7738907373198226)
(91.0, 0.7130830429603775)
(92.0, 0.6570233353371675)
(93.0, 0.6054205856692536)
(94.0, 0.5580449275601991)
(95.0, 0.5109713515666161)
(96.0, 0.4731620543556438)
(97.0, 0.43839618925746143)
(98.0, 0.40679955697508663)
(99.0, 0.378420069080736)
(100.0, 0.35325111921661656)
(101.0, 0.33125005232235827)
(102.0, 0.3123520910069939)
(103.0, 0.2977487131215401)
(104.0, 0.2841228023008037)
(105.0, 0.2737114166133326)
(106.0, 0.26627154307819695)
(107.0, 0.2616261208929345)
(108.0, 0.259644921072949)
(109.0, 0.26038765887874205)
(110.0, 0.26344799910192446)
```

In [23]:
```
from xgboost import XGBRegressor
from sklearn.model_selection import RandomizedSearchCV

import scipy.stats as st

xgbreg = XGBRegressor(nthreads=1)

one_to_left = st.beta(10, 1)
from_zero_positive = st.expon(0, 50)

param_grid = {
        'silent': [False],
        'max_depth': [6, 10, 15, 20],
        'learning_rate': [0.001, 0.01, 0.1, 0.2, 0,3],
        'subsample': [0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
        'colsample_bytree': [0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
        'colsample_bylevel': [0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
        'min_child_weight': [0.5, 1.0, 3.0, 5.0, 7.0, 10.0],
        'gamma': [0, 0.25, 0.5, 1.0],
        'reg_lambda': [0.1, 1.0, 5.0, 10.0, 50.0, 100.0],
        'n_estimators': [100]}
```

```
        params = {
            "n_estimators": [10, 20, 40, 80, 160, 320, 640],
            "max_depth": [4, 8, 16, 32, 64, 128],
            "learning_rate": [0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50],
            "colsample_bytree": [1.0, 0.99, 0.95, 0.90, 0.85, 0.80, 0.60],
            "subsample": [1.0, 0.99, 0.95, 0.90, 0.85, 0.80, 0.60],
            "gamma": [0, 1.0, 1.5, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0],
            'reg_alpha': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100],
            "min_child_weight": [10, 20, 30, 40, 50, 60, 70, 80, 90, 100],
        }

        gs = RandomizedSearchCV(xgbreg, params, n_jobs=1)
        gs.fit(x, y)
        gs.best_params_
```

```
Out[23]: {'colsample_bytree': 1.0,
          'gamma': 1.0,
          'learning_rate': 0.2,
          'max_depth': 64,
          'min_child_weight': 30,
          'n_estimators': 10,
          'reg_alpha': 70,
          'subsample': 0.6}
```

```
In [24]: xgbreg = XGBRegressor(nthreads=1,
                               colsample_bytree=1.0,
                               gamma=9.0,
                               learning_rate=0.5,
                               max_depth=128,
                               min_child_weight=60,
                               n_estimators=10,
                               reg_alpha=100,
                               subsample=0.85)

         xgbreg.fit(x, y)
         z = xgbreg.predict(x)
         mychart(x, y, z)
```

```
<IPython.core.display.HTML object>
```