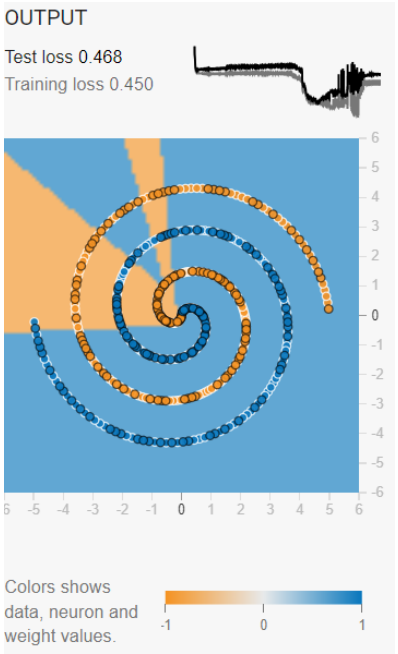
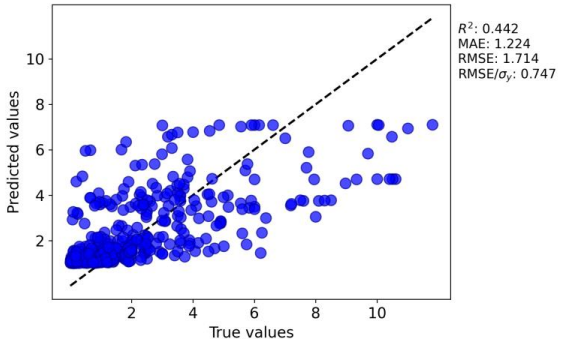
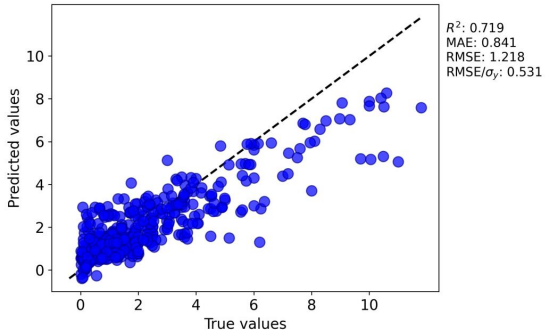
Assessment Figures



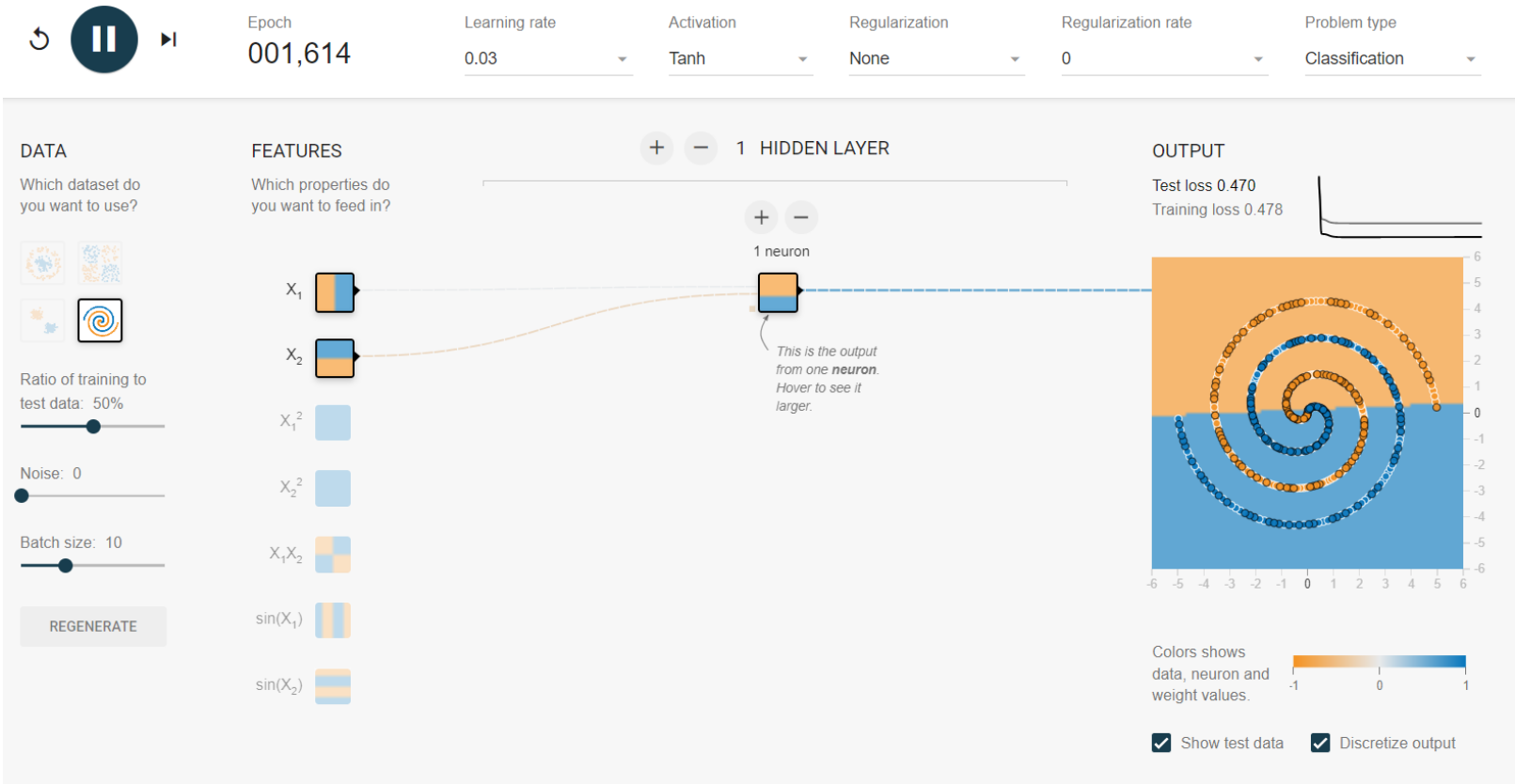




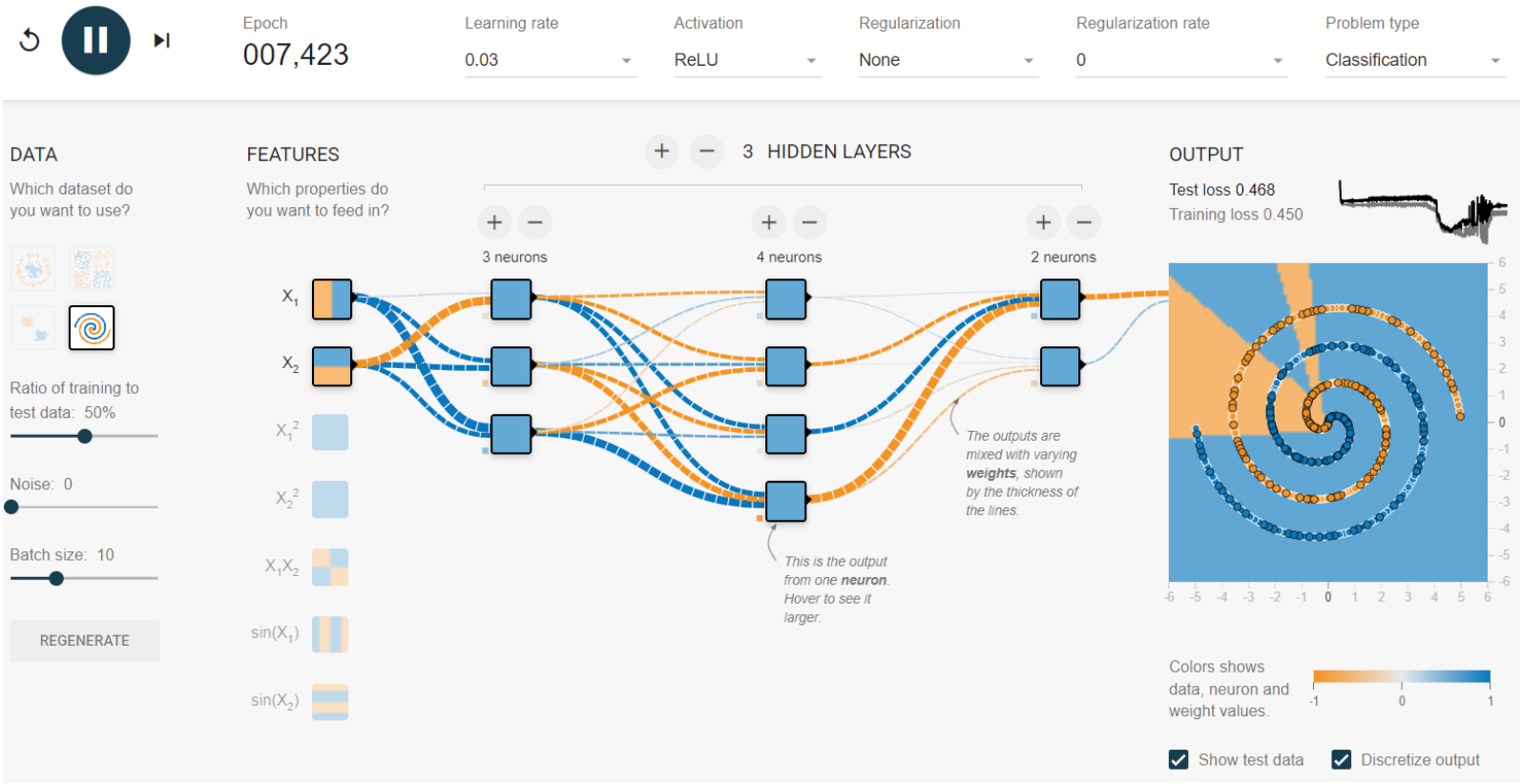
ML4ER - Assignment 6 Activities

Muhammad Zain Azeem,
Informatics Skunkworks (**non-credits),** Week 3
04/08/2024

Progress


Epoch
001,614
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
1 HIDDEN LAYER
1 neuron
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.470
Training loss 0.478
Colors shows data, neuron and weight values.
Show test data
Discretize output

Progress


Epoch
007,423
Learning rate
0.03
Activation
ReLU
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
3 HIDDEN LAYERS
3 neurons
4 neurons
2 neurons
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.468
Training loss 0.450
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output

Progress

```
# Activity 6 - Part 1

default_model = SklearnModel(model='MLPRegressor')
models = [default_model]
selector = [NoSelect()]
metrics = ['r2_score', 'mean_absolute_error', 'root_mean_squared_error', 'rmse_over_stdev']

### here's the key grid search settings we're going to edit
# hyperparams = 'param1; param2; param3'
# param_vals = '1 5 3 log float; 2 10 5 log float; activation1 activation2 activation3 activation4 activation5'
hyperparams = 'alpha ; learning_rate_init ; activation'
param_vals = '-8 2 5 log float ; -5 1 5 log float ; identity logistic tanh relu str'
###

# Grid Addition
grid1 = GridSearch(param_names=hyperparams,param_values=param_vals,scoring='root_mean_squared_error')
grids = [grid1]

# no split!
splitter = NoSplit() # note: even though we're using nosplit here by manually specifying the "leaveout_inds" opt

# Evaluate the model using cross-validation and test set
splitter.evaluate(X=X,
                  y=y,
                  models=models,
                  preprocessor=None,
                  selectors=selector,
                  metrics=metrics,
                  savepath=savepath,
                  X_extra=X_extra,
                  leaveout_inds=X_testdata,
                  hyperopts = grids,
                  recalibrate_errors = True,
                  verbosity=3)
```

- Determining the best hyperparameter for the default model

Results:

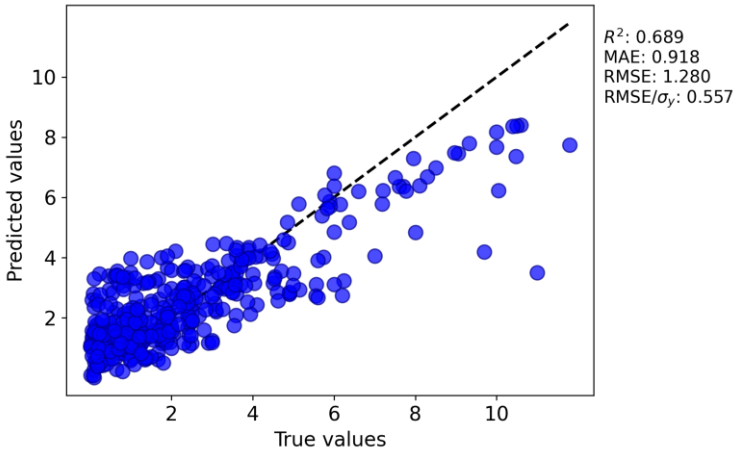
A	B	C	D
	activation	**alpha**	**learning_rate_init**
Best Parameters	logistic	0.316227766	0.316227766

Progress

- Implementing best parameters

```
# Activity 6 - Part 2

default_model = SklearnModel(model='MLPRegressor',activation='logistic',alpha=0.316227766,learning_rate_init=0.316227766)
models = [default_model]
selector = [NoSelect()]
metrics = ['r2_score', 'mean_absolute_error', 'root_mean_squared_error', 'rmse_over_stdev']

### here's the key grid search settings we're going to edit
# hyperparams = 'param1; param2; param3'
# param_vals = '1 5 3 log float; 2 10 5 log float; activation1 activation2 activation3 activation4 activation5'
hyperparams = 'alpha ; learning_rate_init ; activation'
param_vals = '-8 2 5 log float ; -5 1 5 log float ; identity logistic tanh relu str'
###

# Grid Addition
grid1 = GridSearch(param_names=hyperparams,param_values=param_vals,scoring='root_mean_squared_error')
grids = [grid1]

# no split!
splitter = NoSplit() # note: even though we're using nosplit here by manually specifying the "leaveout_inds" option below

# Evaluate the model using cross-validation and test set
splitter.evaluate(X=X,
```

Results:

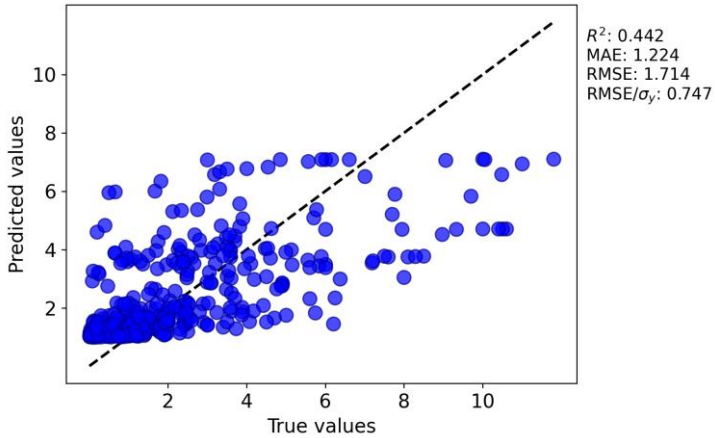


Progress

- Default NN Mode with best parameters

```
# Activity 6 - Part 3


default_model = SklearnModel(model='MLPRegressor',activation='logistic',alpha=0.316227766,learning_rate_init=0.316227766) # Best optimized parameters
models = [default_model]
selector = [NoSelect()]
metrics = ['r2_score', 'mean_absolute_error', 'root_mean_squared_error', 'rmse_over_stdev']

### Neural network
hyperparams = 'hidden_layer_sizes'
param_vals = '(100,) (50,10) (40,) (30,) (20,) (10,) (20,20) (20,20,20) tup'


grid1 = GridSearch(param_names=hyperparams,param_values=param_vals,scoring='root_mean_squared_error')
grids = [grid1]
splitter = NoSplit()
splitter.evaluate(X=X,
                  y=y,
                  models=models,
                  preprocessor=None,
                  selectors=selector,
                  metrics=metrics,
                  savepath=savepath,
                  X_extra=X_extra,
                  leaveout_inds=X_testdata,
                  hyperopts = grids,
                  recalibrate_errors = True,
                  verbosity=3)
```

Results:

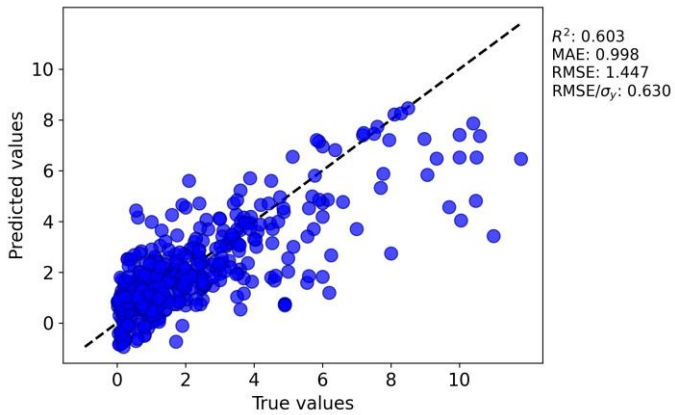


Progress

- **Optimization:** setting all single layers to 50

```
# Activity 6 - Part 4


default_model = SklearnModel(model='MLPRegressor',activation='logistic',alpha=0.316227766,learning_rate_init=0.316227766) # Best optimized parameters
models = [default_model]
selector = [NoSelect()]
metrics = ['r2_score', 'mean_absolute_error', 'root_mean_squared_error', 'rmse_over_stdev']

### Reduce the number of neurons in the single layer to 50. How does this affect the results? Does the simpler model cause a drop in performance?
# Neural network
hyperparams = 'hidden_layer_sizes'
param_vals = '(50,) (50,10) (50,) (50,) (50,) (20,20) (20,20,20) tup'


grid1 = GridSearch(param_names=hyperparams,param_values=param_vals,scoring='root_mean_squared_error')
grids = [grid1]
splitter = NoSplit()
splitter.evaluate(X=X,
                  y=y,
                  models=models,
                  preprocessor=None,
                  selectors=selector,
                  metrics=metrics,
                  savepath=savepath,
                  X_extra=X_extra,
                  leaveout_inds=X_testdata,
                  hyperopts = grids,
                  recalibrate_errors = True,
                  verbosity=3)
```

Results:

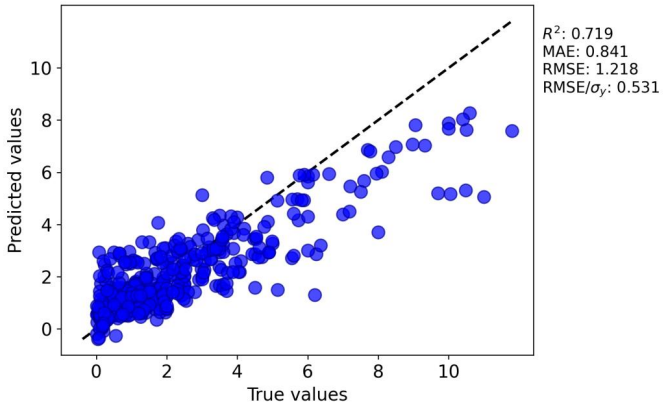


Progress

- **Optimization:** increased both double and triple layers

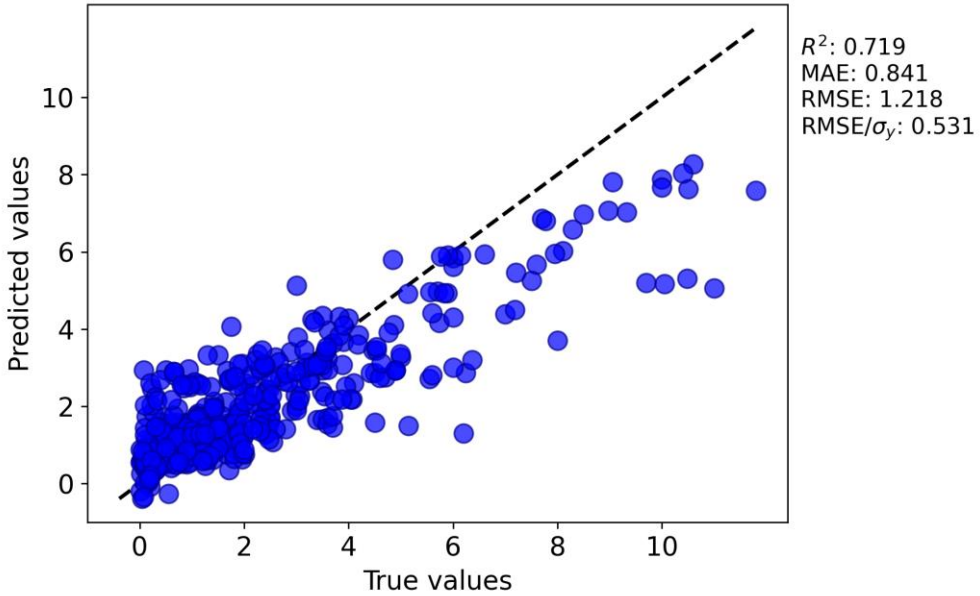
```
# Activity 6 - Part 5


default_model = SklearnModel(model='MLPRegressor',activation='logistic',alpha=0.316227766,learning_rate_init=0.316227766) # Best optimized parameters
models = [default_model]
selector = [NoSelect()]
metrics = ['r2_score', 'mean_absolute_error', 'root_mean_squared_error', 'rmse_over_stdev']

### Let's try multiple layers. Using the previous result of 20 neurons lets increase the number of layers to 2 and then 3 and see how the performance i
# Neural network
hyperparams = 'hidden_layer_sizes'
param_vals = '(50,) (50,100) (50,150) (50,200) (50,250) (50,300) (50,350) (50,100,150) tup'


grid1 = GridSearch(param_names=hyperparams,param_values=param_vals,scoring='root_mean_squared_error')
grids = [grid1]
splitter = NoSplit()
splitter.evaluate(X=X,
                  y=y,
                  models=models,
                  preprocessor=None,
                  selectors=selector,
                  metrics=metrics,
                  savepath=savepath,
                  X_extra=X_extra,
                  leaveout_inds=X_testdata,
                  hyperopts = grids,
                  recalibrate_errors = True,
                  verbosity=3)
```

Results:



Progress

- Final optimized model results:



Conclusion: increased from R^2: 0.442 to R^2 : 0.719