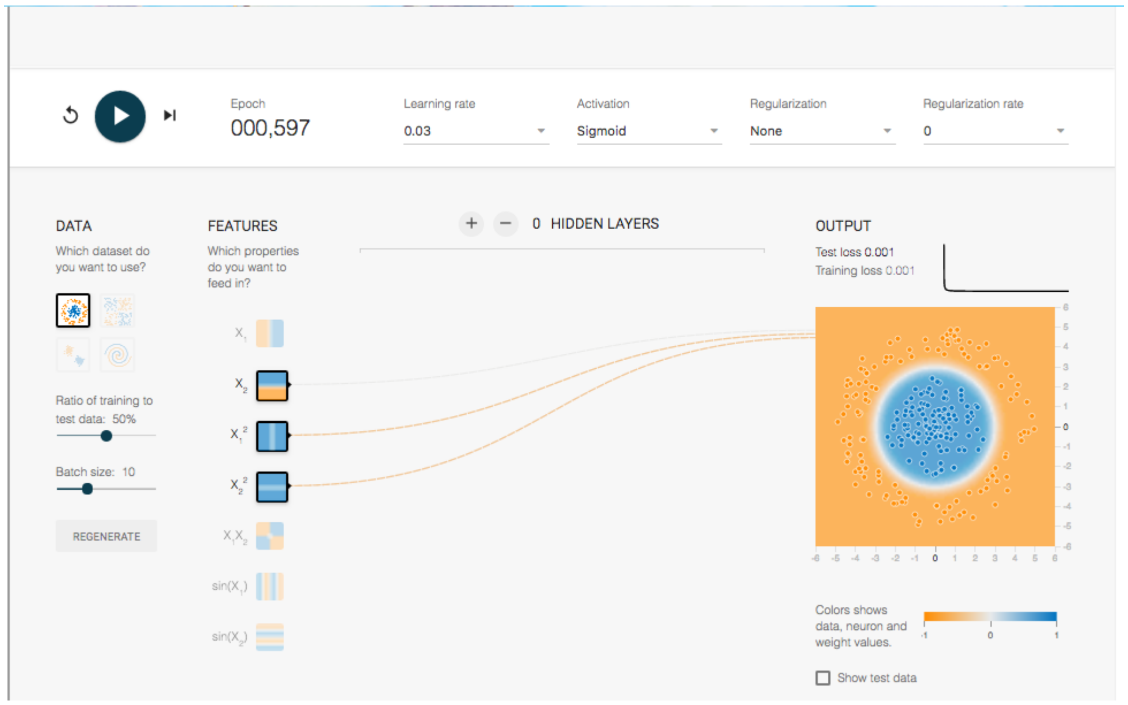
Brandon Peck
CS4501 - Machine Learning
Homework 3

1.1

	Features	Iterations	Test Loss
Circle	X_2, X_1^2, X_2^2	573	0.001
Exclusive OR	X_1X_2	421	0.001
Gaussian	X_1, X_2	326	0.001
Spiral	$\sin(X_1)$, $\sin(X_2)$	757	0.530

Circle:



Epoch 000,597
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?

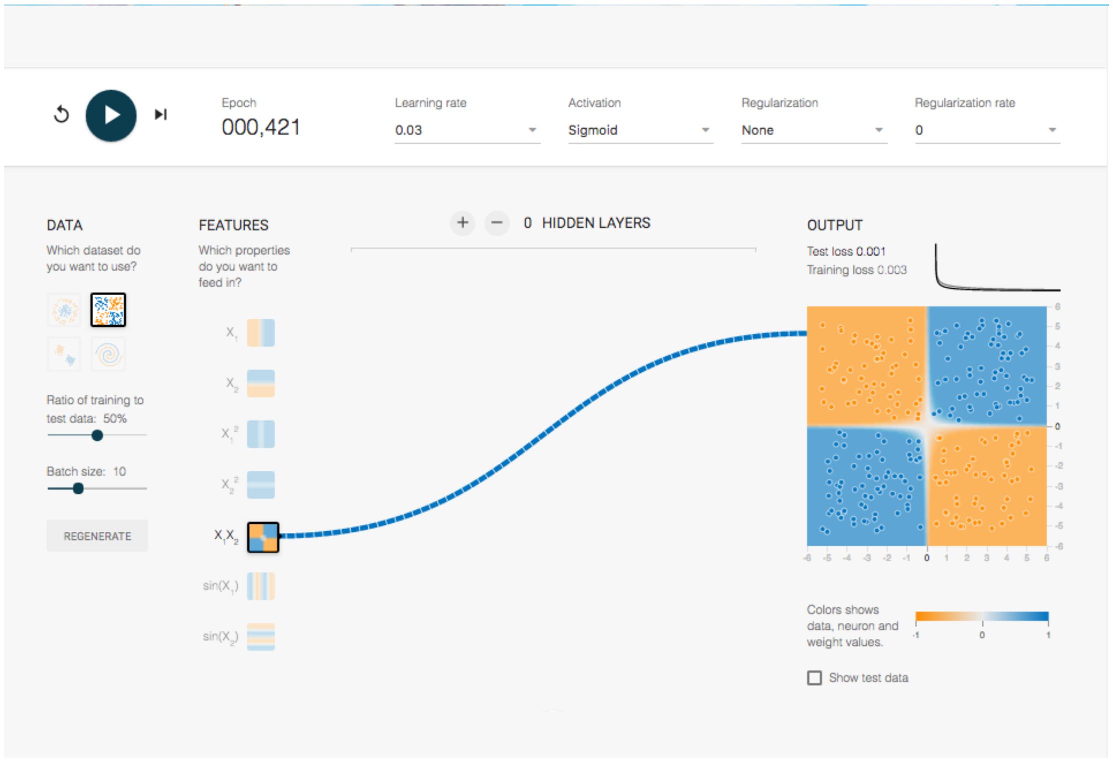
0 HIDDEN LAYERS

OUTPUT
Test loss 0.001
Training loss 0.001

Colors shows data, neuron and weight values.
Show test data

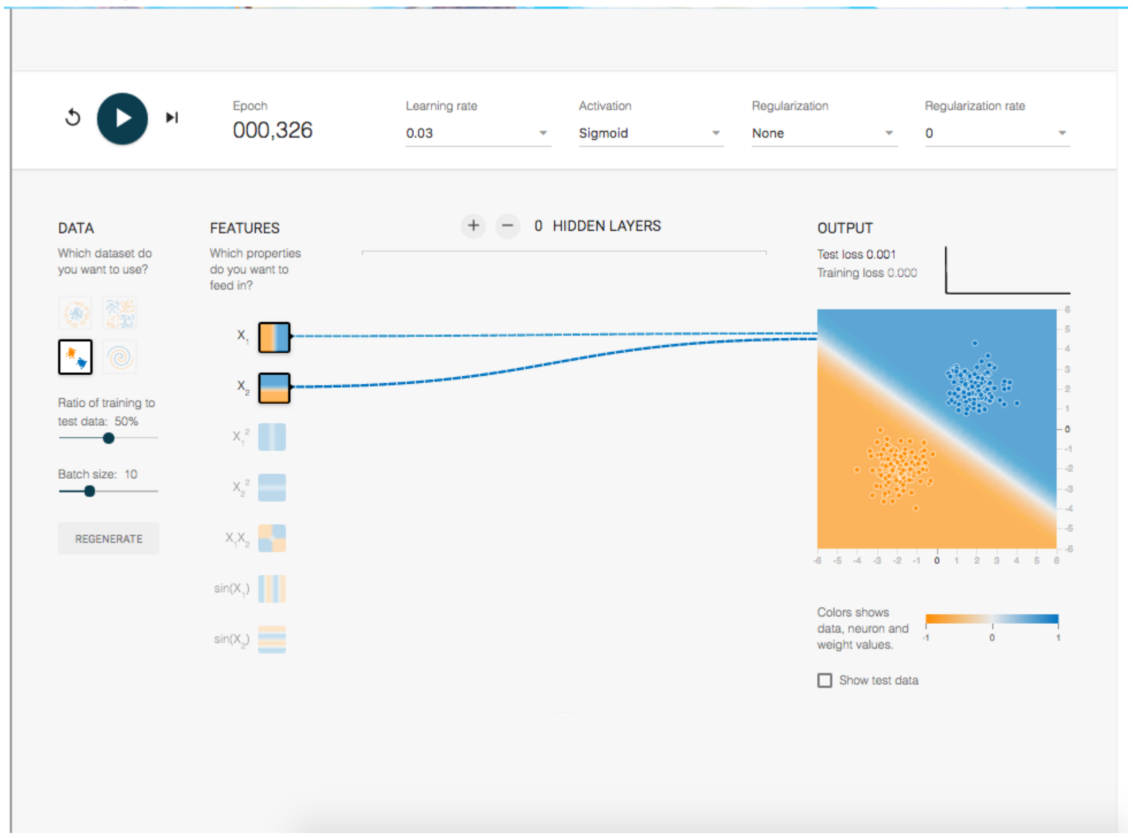
Intuition:
From this configuration we see that X_1^2 and X_2^2 are the perceptrons primarily used to create the model. This because the model creates a circle around the data points. We know that the equation of a circle is $(X_1 - h)^2 + (X_2 – k)^2 = r^2$, so it is up to the model to determine the coefficients and radius.

Exclusive OR:


Epoch 000,421
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0
0 HIDDEN LAYERS
Ratio of training to test data: 50%
Batch size: 10
Test loss 0.001
Training loss 0.003
DATA
Which dataset do you want to use?
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
REGENERATE
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1
Show test data

Intuition:
From this configuration we see that the X_1X_2 perceptron creates the most accurate model around exclusive or. Setting X_1X_2 to a constant, we can divide by X_2 and easily see how the function models the 1/X equation, so $X_1 = c/X_2$ and creates an accurate separation of data points across exclusive or.

Gaussian:


Epoch 000,326

Learning rate 0.03

Activation Sigmoid

Regularization None

Regularization rate 0

DATA
Which dataset do you want to use?

Ratio of training to test data: 50%

Batch size: 10

FEATURES
Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT
Test loss 0.001
Training loss 0.000

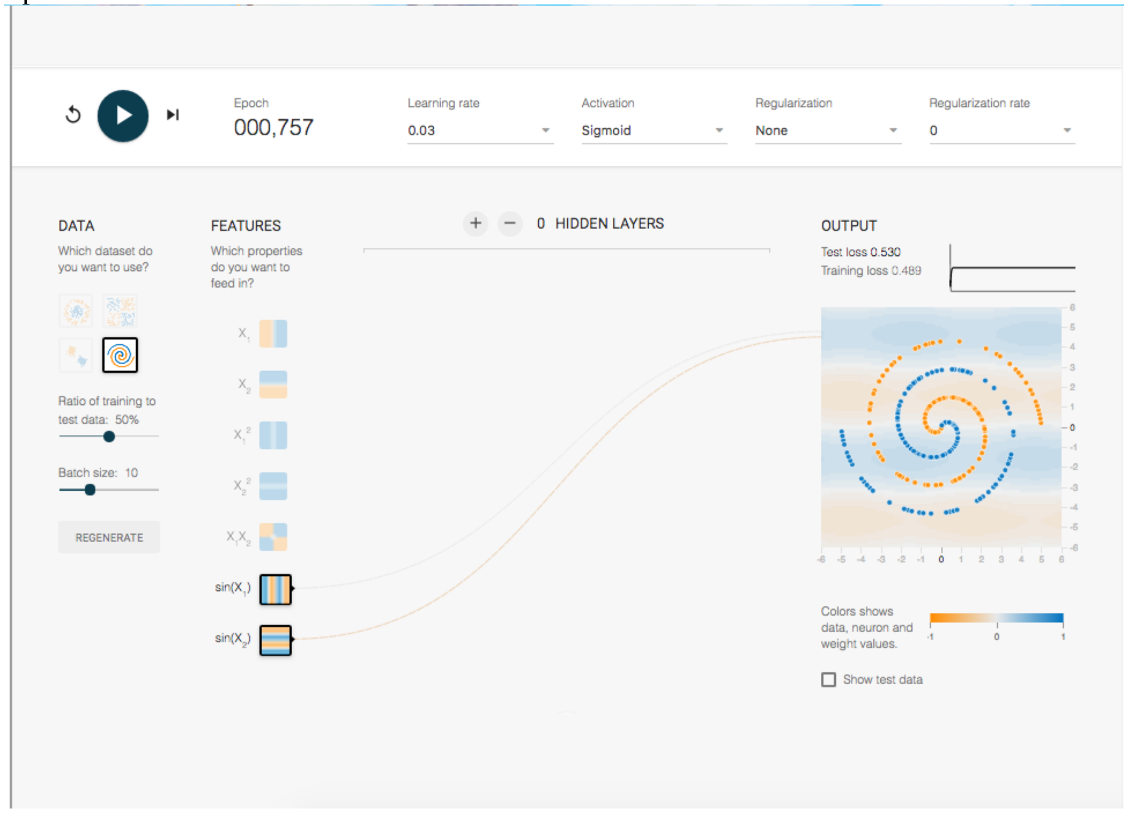
Colors shows data, neuron and weight values.

Show test data

Intuition:

From this configuration we can see that the X_1 and X_2 perceptrons create the best model across the Gaussian distribution. Since there is a clear dividing line across the data the equation $X_1 = c*X_2$, where in this case c is negative, creates an accurate model.

Spiral:



Epoch 000,757

Learning rate 0.03

Activation Sigmoid

Regularization None

Regularization rate 0

Test loss 0.530

Training loss 0.489

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
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
$\sin(X_1)$
$\sin(X_2)$
0 HIDDEN LAYERS
OUTPUT
Colors shows data, neuron and weight values.
Show test data

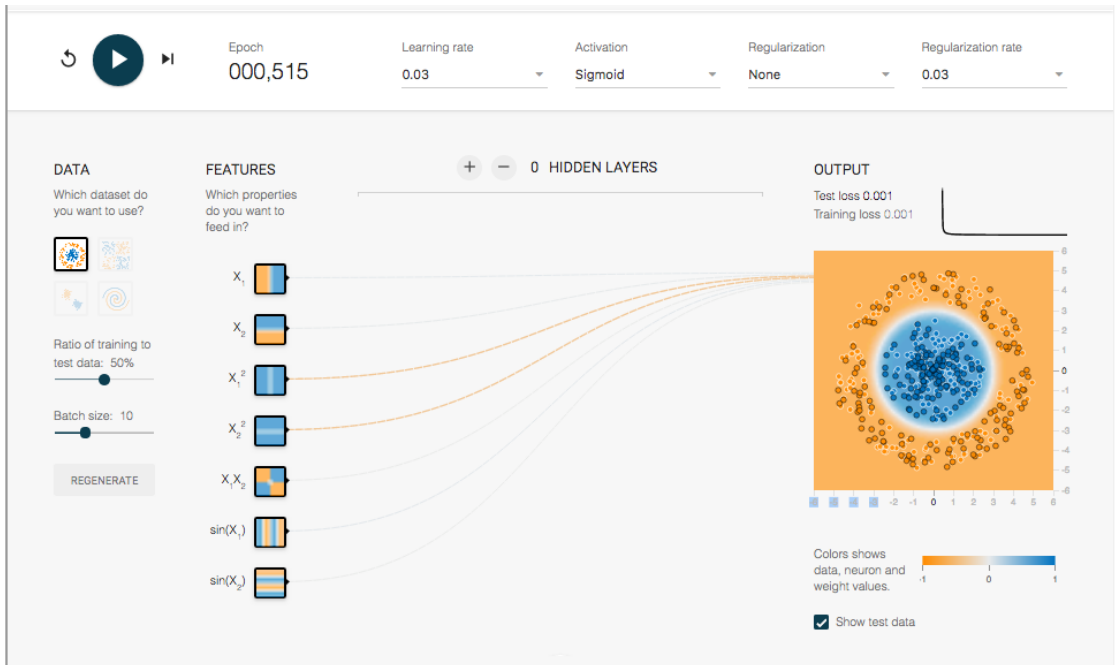
Intuition:

Modeling this set of data points is not as straight forward since a function following the spiral is not continuous. Because the data is distributed in “waves” across both axes, I chose to create a model with $\sin(X_1)$ $\sin(X_2)$. However the Test loss of the model is too poor to say the model is accurate so I cannot conclude some equation with $\sin(X_1)$ and $\sin(X_2)$ models the data.

1.2
Task A:
Evaluations were performed with Epochs = 500, Regularization Rate = 0.03, Batch Size = 10, Test:Train = 1:1

	Estimated Decision Boundary	Test Loss
Circle	None: $x^2+y^2=9$ L1: $x^2+y^2=9$ L2: $x^2+y^2=9$	**None: 0.001** L1: 0.005 L2: 0.002
Exclusive OR	None: (x=.2, y=0) L1: (x=0.2, y=0) L2: (x=0.2, y=0)	**None: 0.003** L1: 0.013 L2: 0.012
Gaussian	None: x=0 y=-2 L1: y = -x L2: y = -x	**None: 0.000** L1: 0.006 L2: 0.002

Circle:



Epoch 000,515
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0.03

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?

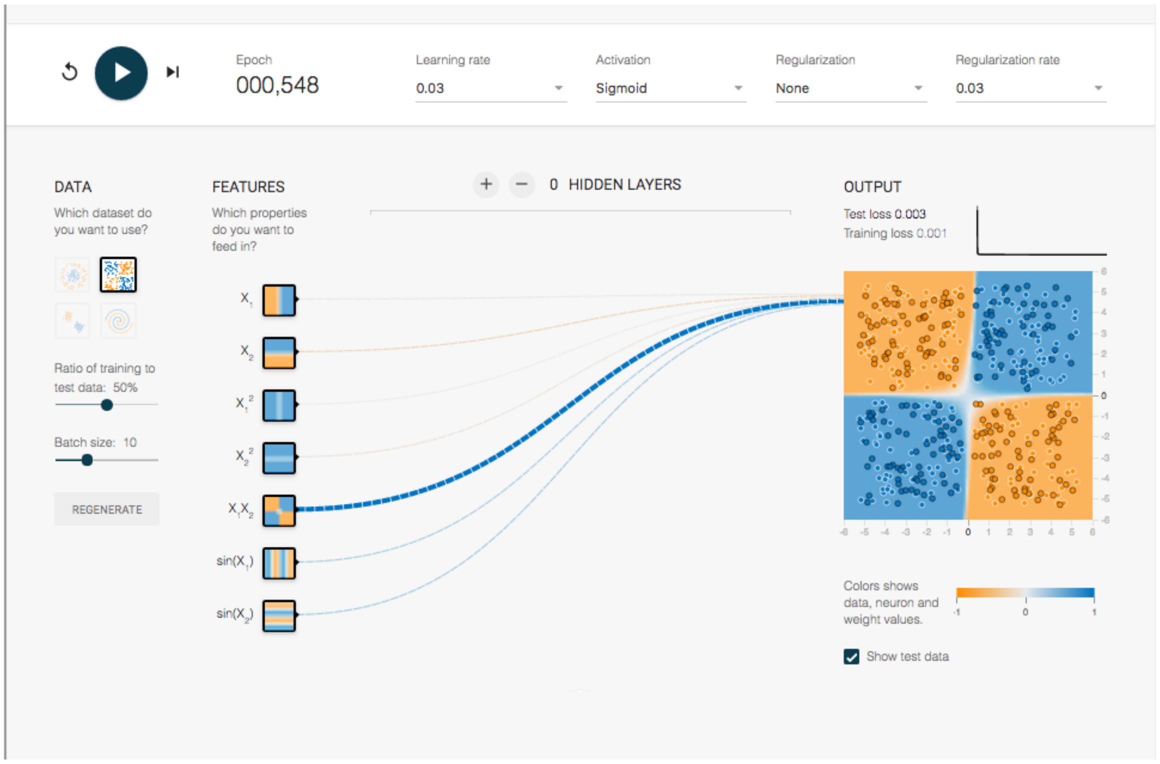
0 HIDDEN LAYERS

OUTPUT
Test loss 0.001
Training loss 0.001

Colors shows data, neuron and weight values.
✓ Show test data

Intuition:
X_1^2 and X_2^2 with no regularization formed the decision boundary around the circle data points. As stated in the previous problem it is clear that these two perceptrons create the best models around the data because they can be trained to set the decision boundary around the data to a circle. The most successful model was the one without L1 or L2 regularization because the test and train data was so similar that a well fitted model to the train data accurately model the test data.

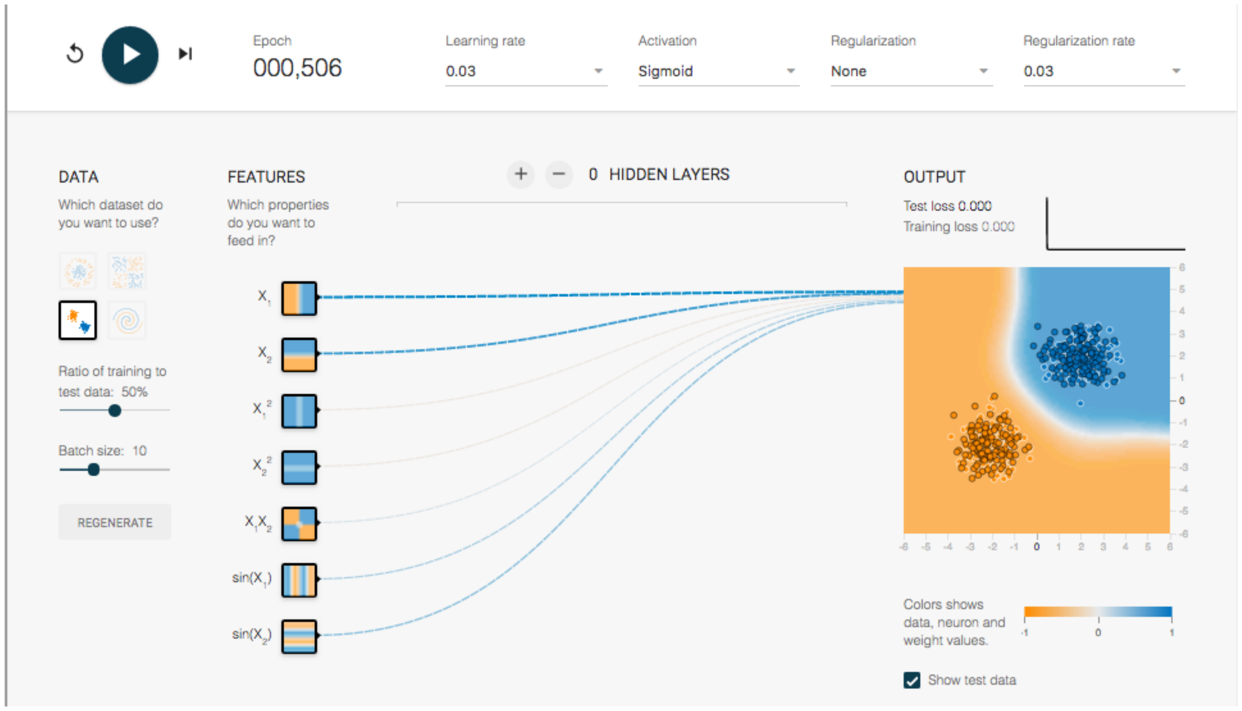
Exclusive OR:


Epoch 000,548
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0.03
Ratio of training to test data: 50%
Batch size: 10
Test loss 0.003
Training loss 0.001
DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
0 HIDDEN LAYERS
OUTPUT
Colors shows data, neuron and weight values.
Show test data

Intuition:

The X_1X_2 perceptron has the highest weights as predicted in the previous question because of its ability to create a decision boundary modeling the 1/X equation. This model also uses the sine function across both the vertical and horizontal axes which we can assume allow the model to be less straight. Again, the most successful model was the one without L1 or L2 regularization because the test and train data was so similar that a well fitted model to the train data accurately model the test data.

Gaussian:



Epoch 000,506
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0.03

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Batch size: 10

FEATURES
Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT
Test loss 0.000
Training loss 0.000

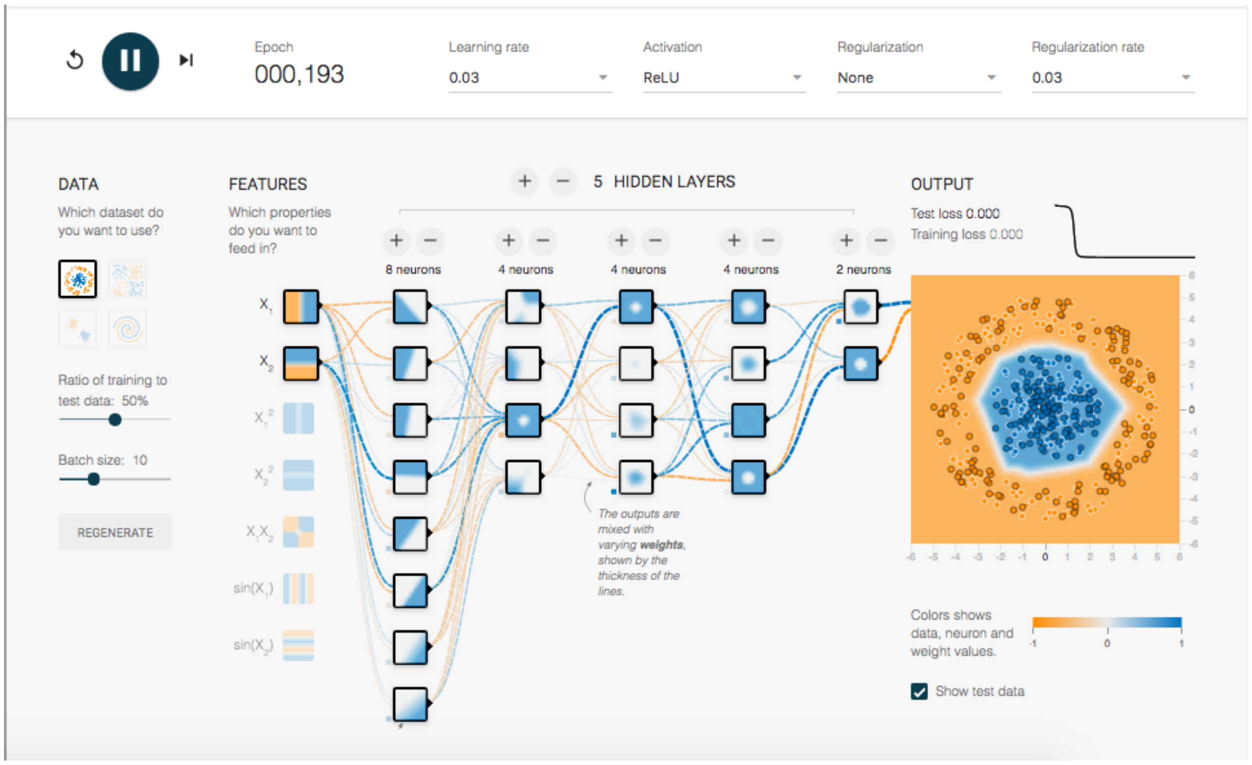
Colors shows data, neuron and weight values.
✓ Show test data

Intuition:
Like in the previous question we see that X_1 and X_2 have the highest weights contributing to the decision boundary. In this model, sine equations are used as well, creating a closer fit to the training data.
Like in the previous questions, the most successful model was the one without L1 or L2 regularization because the test and train data was so similar that a well fitted model to the train data accurately model the test data. However, if the distribution had more outlier points we could presume that a regularized model could create a more accurate decision boundary around the points.

Task B:
The features with higher weights were the ones used in the previous question. Below is a table of the feature weights for each data distribution. I bolded the most relevant features for each set of data.

	X_1	X_2	X_1^2	X_2^2	$X_1 X_2$	$\sin(X_1)$	$\sin(X_2)$
Circle	-.011	-.016	**.29**	**.27**	.0023	0	.05
XOR	.011	-.13	-.015	-.01	**1.6**	.083	.14
Gaussian	**.41**	**.53**	-.045	.012	-.047	.47	**.71**

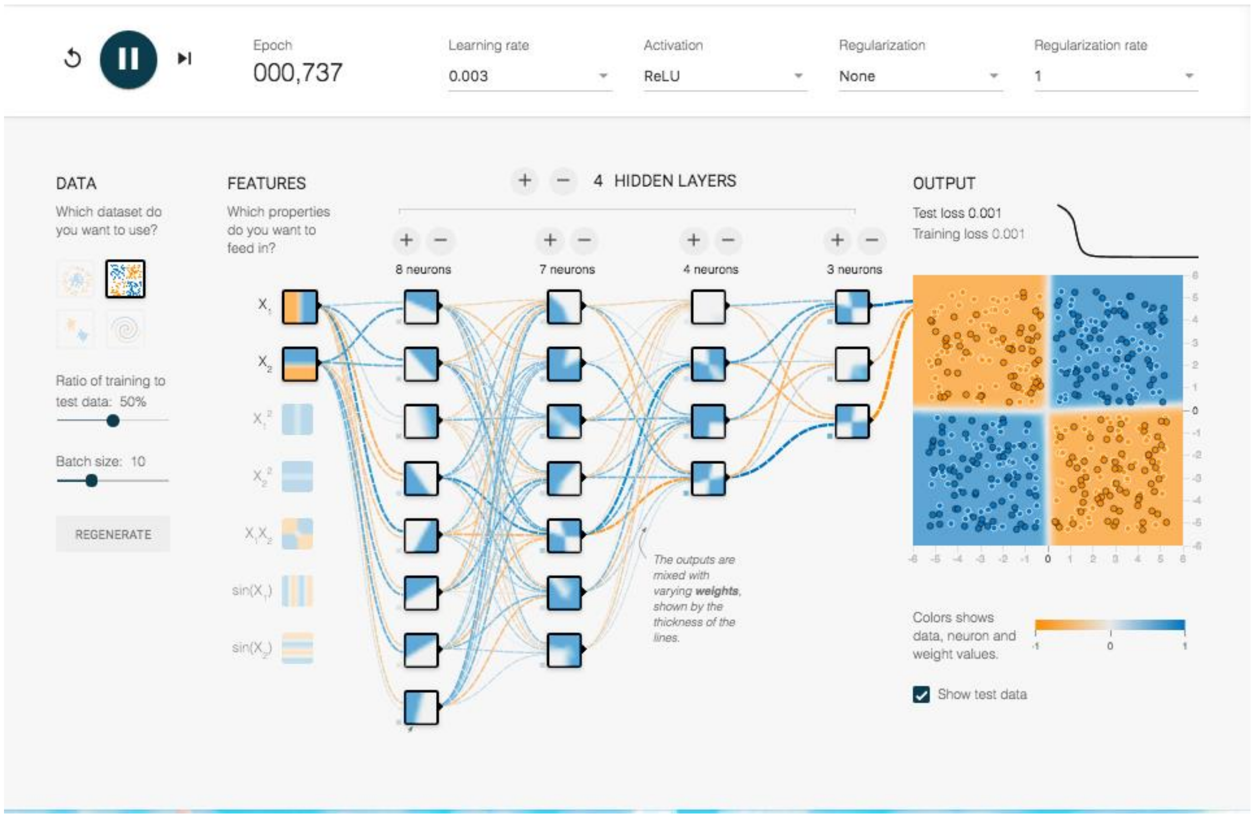
1.3
For this question I changed the activation function to reLU since this does not change the gradient for large input values, creating potentially more accurate and quicker learning. To find an accurate model I started with a large number of neurons and depth and decreased the number of neurons in a layer if I saw the layer contained a neuron that had a very low weight. Removing more neurons also appeared to increase the bias of the model which I found helpful when modeling the Exclusive Or data.
Circle:


Epoch 000,193
Learning rate 0.03
Activation ReLU
Regularization None
Regularization rate 0.03
Ratio of training to test data: 50%
Batch size: 10
5 HIDDEN LAYERS
8 neurons, 4 neurons, 4 neurons, 4 neurons, 2 neurons
Test loss 0.000
Training loss 0.000
Show test data ✓

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
The outputs are mixed with varying **weights**, shown by the thickness of the lines.
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1

The test loss for this is 0 and the model was created in only 193 iterations. I used the reLU activation function. This beats the .01 test loss I had for question 1.1. I believe this model works because multiple hidden layers allow for single reLU regressions to be combine to create a circular model around the data.

Exclusive OR:



Epoch 000,737

Learning rate 0.003

Activation ReLU

Regularization None

Regularization rate 1

Ratio of training to test data: 50%

Batch size: 10

4 HIDDEN LAYERS

Test loss 0.001

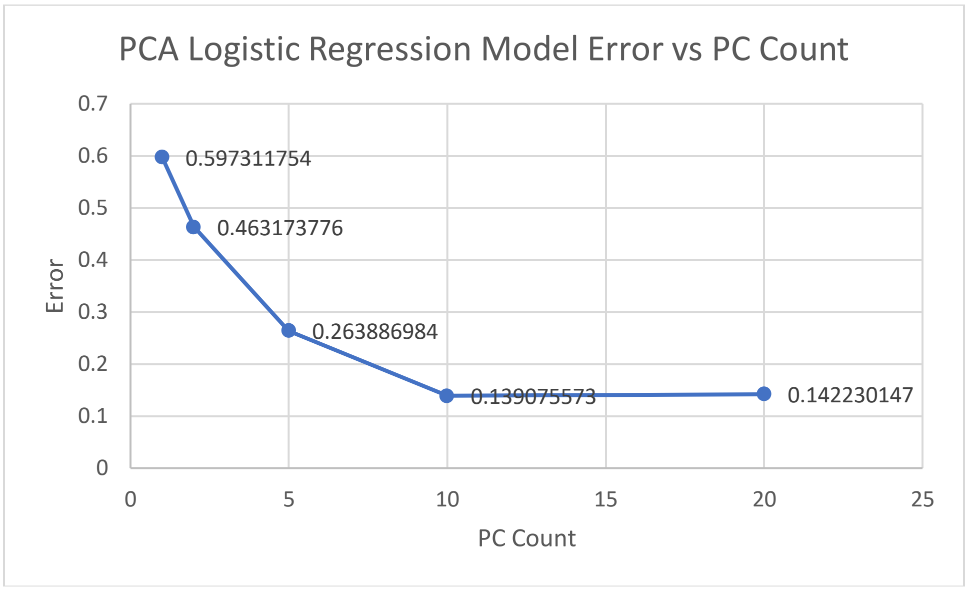
Training loss 0.001

Show test data ✓

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
8 neurons
7 neurons
4 neurons
3 neurons
The outputs are mixed with varying **weights**, shown by the thickness of the lines.
OUTPUT
Colors shows data, neuron and weight values.

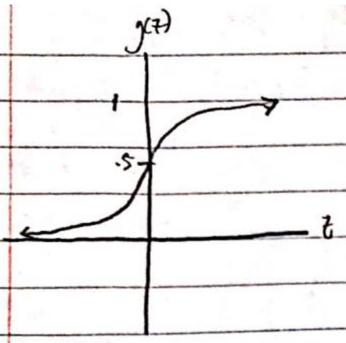
The test loss for this is .01 and the model was created in only 193 iterations. I used the reLU activation function. This ties the .01 test loss I had for question 1.1. I believe this model works because multiple hidden layers allow for single reLU regressions to combine to model the 1/x equation from question 1.1 across several layers of depth.

2.5



Since I was not given the labels for the test data, I scored the regression model on its train data. While this isn't going to show the actual accuracy of the model, it does still show how the accuracy changes with reduced PCs. The error follows a logistic curve with respect to the PC count, where the error decreases as the number of PCs increases. This seems fairly intuitive as more features creates a model that better fits the data.
Error without PCs is .98 which is accurate since the model is scored using its train data. Since it's a logistic regression model it is intuitive to think that the accuracy will not be 1, as the regression does not exactly fit the data.

3)
1.
(a)
We know that g(z) creates a logistic function where g(z) > 0 at z = 0.



$z = -w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$

When $x_1 = x_2 = 1$ $z \geq .5$
when $x_3 = 1$ $z \geq .5$

If $w_1 = w_2 = .5$
for $x_1 = x_2 = 1$ $w_1 x_1 + w_2 x_2 = 1$
for $x_1 \neq x_2$ $0 + .5 = .5$
for $x_1 = x_2 = 0$ $0 + 0 = 0$

If $w_3 = 1$
for $x_3 = 1$ $w_3 x_3 = 1$
for $x_3 = 0$ $w_3 x_3 = 0$ as desired

Since for $x_1 \neq x_2$ with $w_1 = w_2 = .5$ and $x_1 \neq x_2$ must be $< .5$
determine $-1 < w_0 < .5$

Choose $w_0 = -.75$ $w_1 = .5$ $w_2 = .5$ $w_3 = 1$
$z = -.75 + .5x_1 + .5x_2 + x_3$

(b)

i) True. Linear regression can use matrix algebra by using the normal equation theta = $(X^TX)^{-1}X^Ty$ which we derived in class. Linear regression can be implemented with stochastic gradient descent by repeatedly updating the theta coefficient based on reducing error across data records.

ii) True. The error surface is created from the gradient of the sum of squared error. Since this sum changes with the data the gradient surface will change.

iii) False. Since stochastic gradient descent looks at random individual points, if the points chosen to do the regression with are an accurate representation of the distribution, SGD can be much faster than batch gradient descent while maintaining model accuracy.

a. True

   i. If we use some modified function other than the g(z) given by the problem, negating the weights could negate the output.

b. False

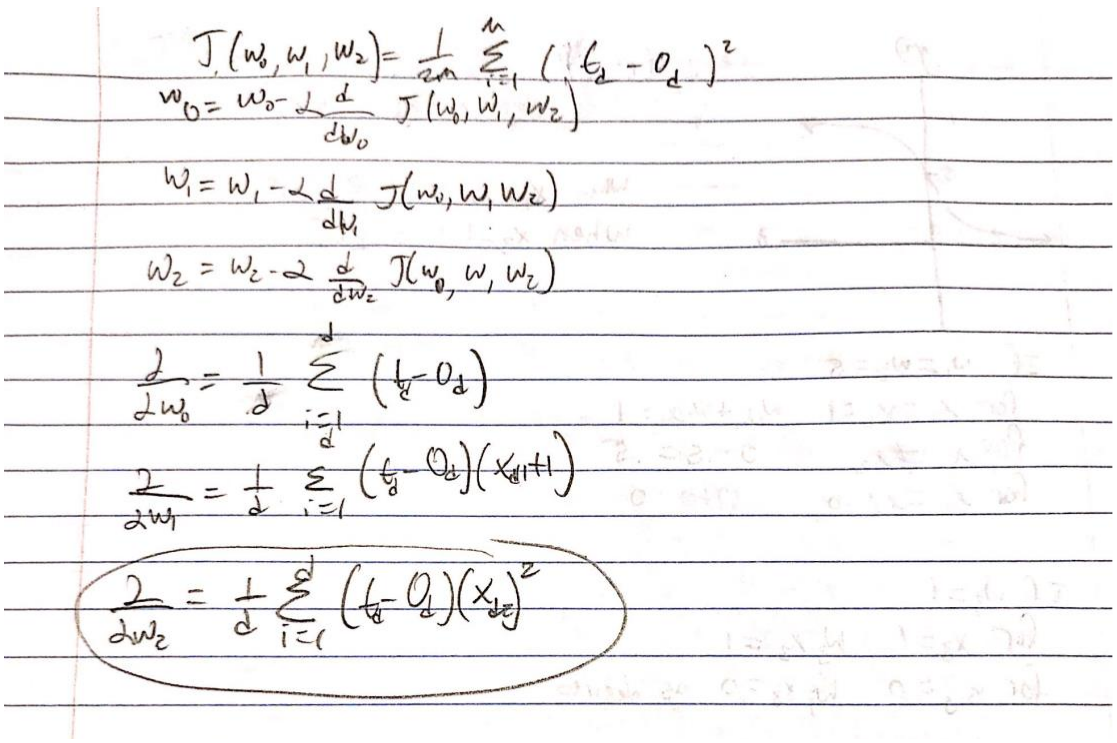
$$g(z) = \frac{1}{1+\exp(-z)}$$
$$z = w_0 + w_1x_1 + w_2x_2$$

Negating the coefficients we get:

$$-w_0 - w_1x_1 - w_2x_2 = -1(w_0 + w_1x_1 + w_2x_2)$$
$$= -z$$
$$\frac{1}{1+\exp(z)}$$

This flips our sigmoid function graph across the z axis, so it's clear negating the weights may not change the sign of the unit output

iv) False. Since when calculating delta w_2 we are we are squaring the x_2 value and the update equation is the sum of



$$J(w_0, w_1, w_2) = \frac{1}{2m}\sum_{i=1}^{m}(t_d - o_d)^2$$

$$w_0 = w_0 - \alpha \frac{d}{dw_0} J(w_0, w_1, w_2)$$

$$w_1 = w_1 - \alpha \frac{d}{dw_1} J(w_0, w_1, w_2)$$

$$w_2 = w_2 - \alpha \frac{d}{dw_2} J(w_0, w, w_2)$$

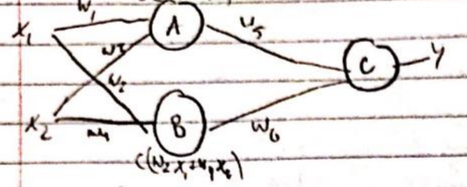
$$\frac{d}{dw_0} = \frac{1}{d}\sum_{i=1}^{d}(t_d - o_d)$$

$$\frac{d}{dw_1} = \frac{1}{d}\sum_{i=1}^{d}(t_d - o_d)(x_{d1}+1)$$

$$\boxed{\frac{d}{dw_2} = \frac{1}{d}\sum_{i=1}^{d}(t_d - o_d)(x_{d2})^2}$$

2)

(a) $A = L \quad B = L \quad C = L$



$$y = cw_5(cw_1 x_1 + cw_2 x_2) + cw_6(cw_1 x_1 + cw_2 x_2)$$

$$= c^2 w_5 w_1 x_1 + c^2 w_5 w_2 x_2 + c^2 w_6 w_1 x_1 + c^2 w_6 w_2 x_2$$
$$= x_1(c^2 w_5 w_1 + c^2 w_6 w_1) + x_2(c^2 w_5 w_2 + c^2 w_6 w_2)$$
$$= B_1 x_1 + B_2 x_2$$

(b)

$$a = w_5(c w_1 x_1 + c w_3 x_2) + w_6(c w_2 x_1 + c w_4 x_2)$$

$$= c w_1 w_5 x_1 + c w_3 w_5 x_2 + c w_6 w_2 x_1 + c w_4 w_6 x_2$$
$$= x_1(c w_1 w_5 + c w_6 w_2) + x_2(c w_3 w_5 + c w_4 w_6)$$
$$B_1 = c(w_1 w_5 + w_2 w_6) \quad B_2 = c(w_3 w_5 + w_4 w_6)$$
$$= B_1 x_1 + B_2 x_2$$

$$S(a) = \text{Sign}\left(\frac{1}{1+\exp(-(B_1 x_1 + B_2 x_2))} - .5\right)$$

for $B_1 x_1 + B_2 x_2 < 0$ $\quad y = -1$ as desired
for $B_1 x_1 + B_2 x_2 > 0$ $\quad y = 1$ as desired

c) Derived above
$$B_1 = c(w_1 w_5 + w_2 w_6)$$
$$B_2 = c(w_3 w_5 + w_4 w_6)$$

3)
Graph 1:
Since the sigmoid function inputs a linear equation with respect to X and W we know that the relationship between X and W create a decision boundary used by the second layer of the network.
Setting N_1 to zero we get

$$w_{10} + w_{11}X_1 + w_{12}X_2 = 0$$
$$X_2 = \frac{-w_{11}X_1 - w_{10}}{w_{12}}$$

We know that the X_1 coefficient is negative, creating negative linear equation over the data justifying the solution given.
It follows that N_2 will have a positive line separating the data in the opposite direction.
V_1 and V_2 are the output of the N_1 N_2 equations entered in their respective sigmoid functions.
Since N_1 has a negative weighted coefficient we know that putting a negative value in the sigmoid function will look like an inverted sigmoid graph. The normal sigmoid graph will be the output of the N_2 equation. We can take points like (2,3) on the graph, look that they will be above the N_1 decision boundary, and know that they will input a larger value into V_1 so will be at a higher value in the resulting graph. Since we also know that values on the lower side of the N_1 decision boundary will be smaller and therefore less to the extreme of the sigmoid graph they will be smaller on the V_1 graph. This reasoning validates how the points are displayed on the N_3 graph. It's clear to see how a linear decision boundary will be formed on the N_3 graph using the reasoning above.