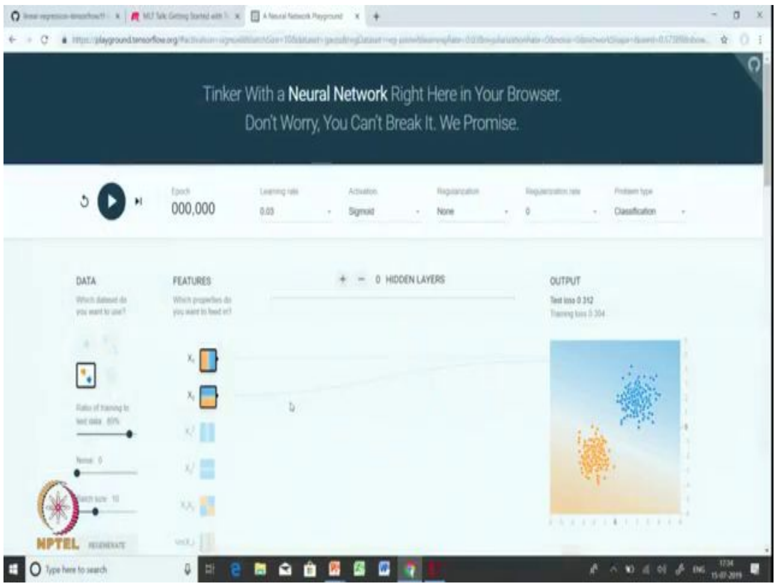
Practical Machine Learning with Tensor Flow

Dr. Ashish Tendulkar Google
Department of Computer Science Engineering
Indian Institute of Technology, Bombay

Lecture – 08
Machine Learning Visualization

[FL] Welcome to the next session of our course on Practical Machine Learning. In this session we will go through some of the machine learning concepts visually, so that it is easier for you to understand or get intuition for this concepts. We will use a neural network playground as a tool to visualize machine learning algorithms.

(Refer Slide Time: 00:42)



Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.

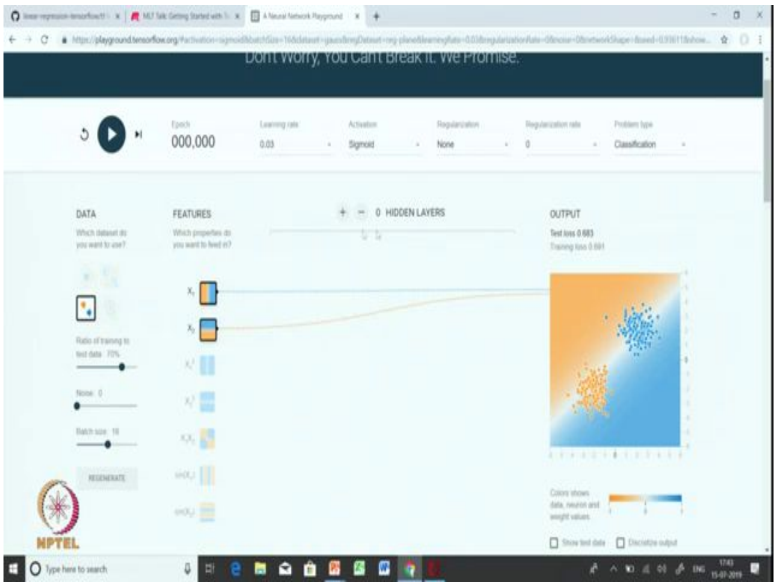
Epoch 000,000 · Learning rate 0.03 · Activation Sigmoid · Regularization None · Regularization rate 0 · Problem type Classification

DATA · FEATURES · 0 HIDDEN LAYERS · OUTPUT

Test loss 0.312
Training loss 0.304

So, we have a tool called neural network playground with a very interesting tagline, "tinker with neural network right here in browser do not worry you cannot break anything". So, let us let explore bravely without worrying about things getting broken up.

(Refer Slide Time: 01:08)

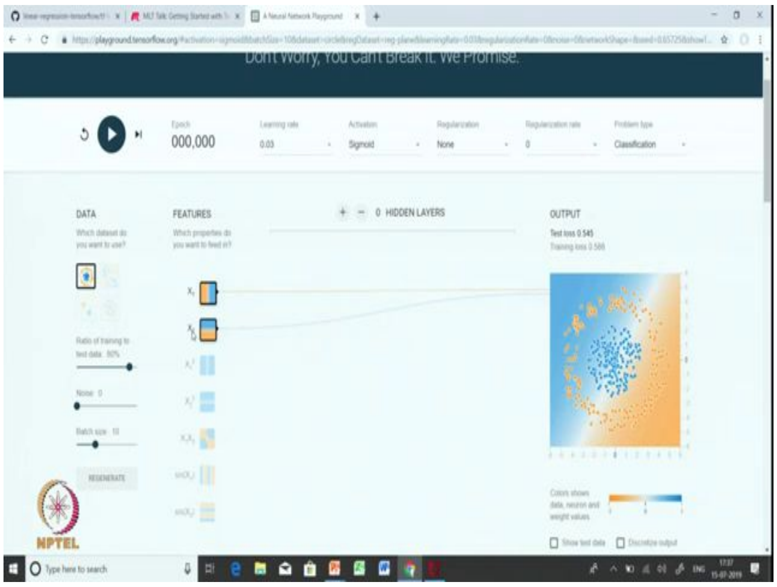

Epoch 000,000
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification

Don't Worry, You Can't Break It. We Promise.
DATA
Which dataset do you want to use?
Ratio of training to test data: 70%
Noise: 0
Batch size: 16
REGENERATE
FEATURES
Which properties do you want to feed in?
0 HIDDEN LAYERS
OUTPUT
Test loss 0.683
Training loss 0.691
NPTEL

So, before we begin, let me tell you a few things about this particular tool since most of you are new to this. So, in the previous session, we look at we talked about componentized view of machine learning model where we say that we need training data when we set up the model, we train the model and we evaluate the model performance. In training the model, we define the loss function and we also define what kind of algorithm will be used. So, if you are using gradient descent or any of its variant, you have to set up things like learning rate and we might also have to set up regularization in order to control the model complexity.

So, we studied all these things in the previous sessions. Let us try to use those things in practice and see how they affect the training of our machine learning models. So, here on your left, you see a pen where we can define different kinds of data. So, you can think of this a simulative data set, so the one that is highlighted here is the data which has got clear linear separation between two classes. So, we have a positive class which is denoted by blue color and negative class denoted by the orange color.

(Refer Slide Time: 02:57)



Don't Worry, You Can't Break It. We Promise.

Epoch 000,000 | Learning rate 0.03 | Activation Sigmoid | Regularization None | Regularization rate 0 | Problem type Classification

DATA Which dataset do you want to use?

FEATURES Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT Test loss 0.545 Training loss 0.586

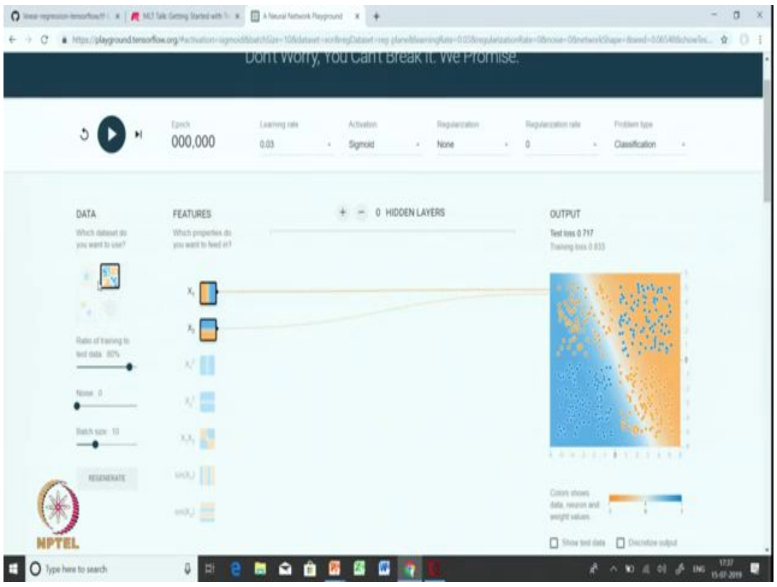
Ratio of training to test data: 80%

Noise: 0

Batch size: 10

So, we have four different data set one which is linearly separable, one which is not linearly separable.

(Refer Slide Time: 03:01)



Don't Worry, You Can't Break It. We Promise.

Epoch 000,000 | Learning rate 0.03 | Activation Sigmoid | Regularization None | Regularization rate 0 | Problem type Classification

DATA Which dataset do you want to use?

FEATURES Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT Test loss 0.717 Training loss 0.633

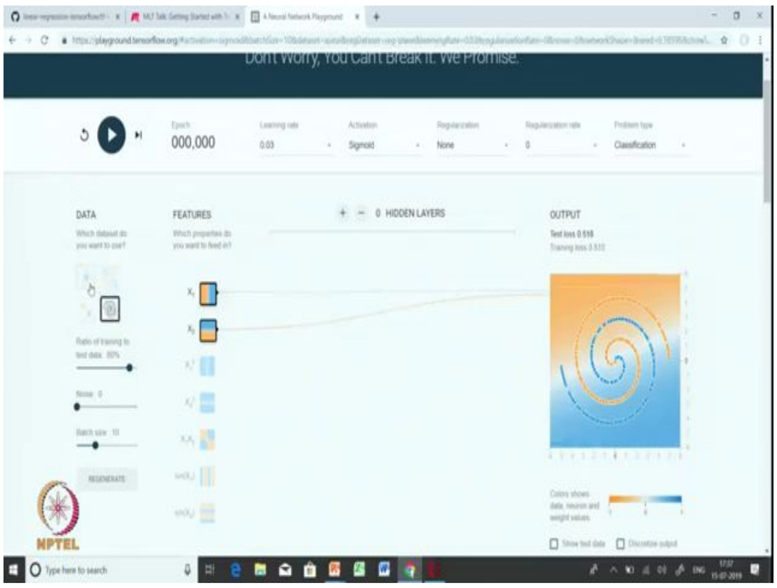
Ratio of training to test data: 80%

Noise: 0

Batch size: 10

Or rather the remaining will be data sets are not linearly separable but have varying degrees of complexities.

(Refer Slide Time: 03:04)


Epoch 000,000
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification

Don't Worry, You Can't Break It. We Promise.
DATA
Which dataset do you want to use?
Ratio of training to test data: 80%
Noise: 0
REGENERATE
FEATURES
Which properties do you want to feed in?
0 HIDDEN LAYERS
OUTPUT
Test loss 0.518
NPTEL

We will see how to make use of techniques like future crosses to fit a model over here. We will be using this neural network playground later with neural networks, where you will see how some of the things that we construct by hand in traditional machine learning algorithms have it is kind of taken away by the use of neural network. So, we can choose the data set here in this paint, we can specify what is the ratio of training and test. So, remember if we talked about training test split and you can specify in what, what percentage you want to split training and test data.

So, we have a slider here that defines how much data is used for training and how much data is used for testing. So, here you know we are going to use let us say 70 percent data for training and 30 percent data for testing. Then we can also add noise to this data set. So, currently in its current form of the screen, you can see that this does not have any noise, it has got 0 noise.

As you add more and more noise you will see that the points will so the classes gets polluted with point from the other class. So, this is like 45 percent noise and 50 percent noise you will see that there are some of these negative points that are present among positive class and vice versa. So, noise is; so with noise we can actually simulate the real life data set, which generally contain some noisy labels.

And finally, since this neural network playground uses mini batch gradient descent, we also get to see set the batch size. So, let us so we can you can also expand by setting the batch size, let us say it to 16, then you can simply press the generate button that will generate a data for us, then you know here. So, each of these data points has two features which is X_1 and X_2 and you know this is the part where we build a model. So, currently we are using a logistic regression model and logistic regression model is set up by having one output layer which has got sigmoid activation. So, this will be clear to you, once we get into neural network. We had briefly seen neural network so, you kind of know what is activation and what is output layer.

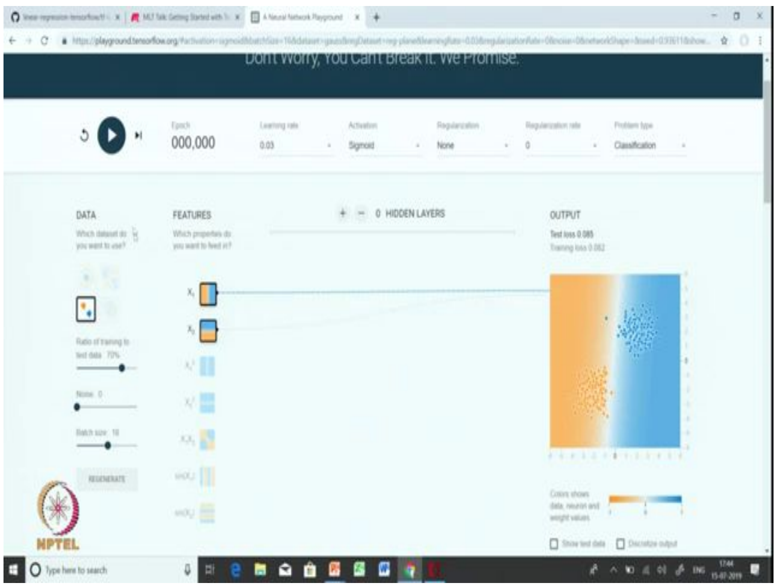
So, here we use one output layer and we use sigmoid activation over there. We can also specify a bunch of other parameters like learning rate, regularization, right now we are not using any regularization, but you can use either L1 or L2 as regularization. We can also you know fix up the rate of regularization which was denoted by parameter lambda as we saw in the previous class and we can define the problem type. Here we are defining classification as a problem type

So, you can think of this particular part defining the hyper parameters of our model. And this is where we visualize you know the prediction, how the model is what kind of prediction it is giving. And this particular part you will see learning course appearing when we start training and you will be able to see test loss and a training loss. And this is the button where; this is the part where we control or where we so, if you press the play button, the model starts training, you can see that. See this you can stop the training, you can revert to the initial situation. And you can use this particular button to see what happens how model train stepwise. So, you can see what happens in the first step, second step and so on.

So, let us try to solve this problem. With a setting where we use 70 percent data for training and 30 percent data for testing we have a data set without any noise and we are using batch size of 16, and let us see we are using learning rate of 0.3. We can explore is we can see how learning rate affects the loss or the convergence. We saw that learning rate affects the convergence. If you have two small learning rate, it takes longer to reach the minima, and if you use very high learning rate. There is a possibility that you will never converge because you might be oscillating across minima's and you will never reach the convergence.

So, we will have to you know find a sweet spot between two extremes, we do not want to have too low learning rate or not too high learning rate, but we want to have sufficient a large learning rate, so that we can safely train our model in an efficient manner. By safety I mean it does not do oscillations or anything like that.

(Refer Slide Time: 06:40)



Don't Worry, You Can't Break It. We Promise.

Epoch 000,000 | Learning rate 0.03 | Activation Sigmoid | Regularization None | Regularization rate 0 | Problem type Classification

DATA — Which dataset do you want to use?

FEATURES — Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT

Test loss 0.085

Training loss 0.082

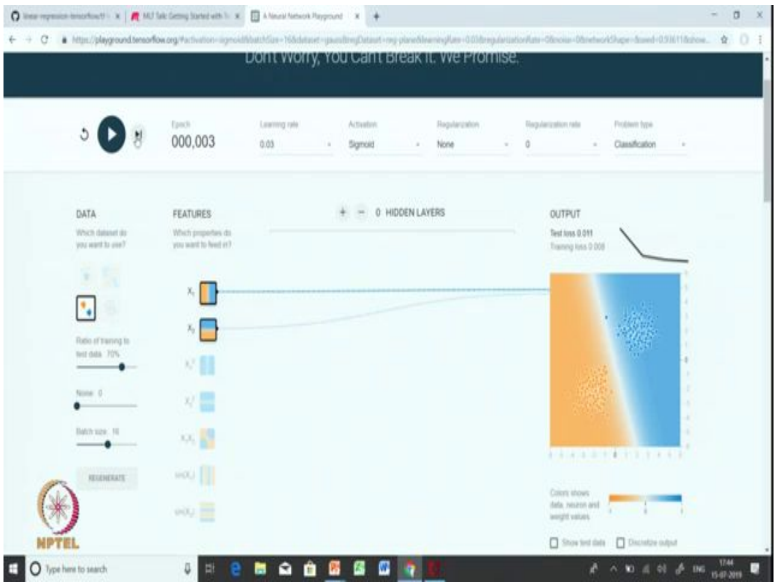
Ratio of training to test data: 70%

Noise: 0

Batch size: 10

We would also you know try bit different regularization along with regularization rates. So, let us see what happens. So, so what do you see is, so we have then we have you know reset everything. And now what you see is if the model is initialized with some weights, so there is a weight of 0.46 on X_1, and there is a weight of; there is a negative weight on W_2 on X_2, this is, it is W_2 and this is W_1. And you can see that the width of the line defines the strength of the weight. So, you can see that as things appear right now X_1 seems to be a stronger predictor over X_2; X_2 seems to be weak predictor because the line is quite faint and it has got and yeah. So, and the color of the line tells you the sign of the weight. So, this color is slightly orangish, so this will have negative, this is the weight will be negative and this weight will be positive.

(Refer Slide Time: 10:00)


Epoch 000,003
Learning rate 0.03
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 70%
Noise: 0
Batch size: 16

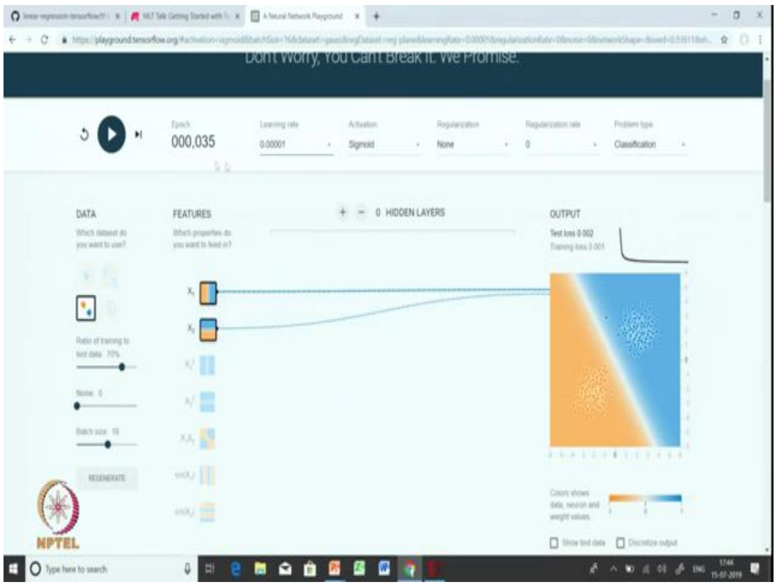
FEATURES
Which properties do you want to feed in?

0 HIDDEN LAYERS

OUTPUT
Test loss 0.011
Training loss 0.008

Let us start stepping through the model. We can see that the error reduces gradually on training loss and test loss.

(Refer Slide Time: 10:06)


Epoch 000,035
Learning rate 0.00001
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 70%
Noise: 0
Batch size: 16

FEATURES
Which properties do you want to feed in?

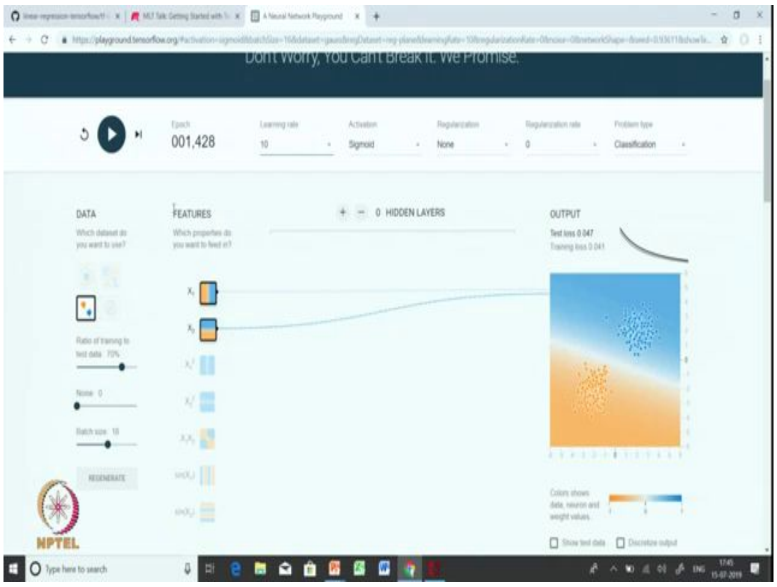
0 HIDDEN LAYERS

OUTPUT
Test loss 0.002
Training loss 0.001

Now, it has slow down. You can see that training loss is moving towards 0 as we go through more epochs. So, you can see that after 27 epochs, the training loss is 0.001 and test loss is

0.002. Now, so we achieve this state after 24 epochs, if you go if you reduce your learning rate let us reset and see what happens.

(Refer Slide Time: 10:32)


Don't Worry, You Can't Break It. We Promise.

Epoch 001,428 | Learning rate 10 | Activation Sigmoid | Regularization None | Regularization rate 0 | Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 70%
Noise: 0
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?

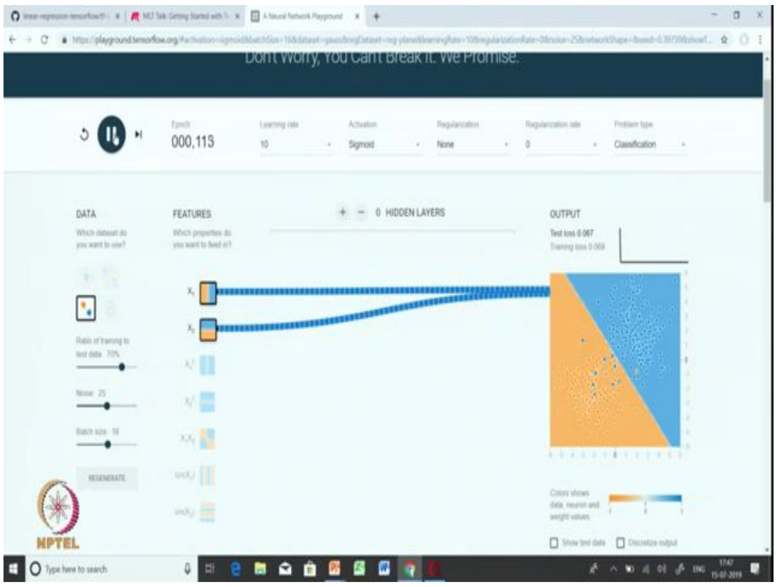
0 HIDDEN LAYERS

OUTPUT
Test loss 0.047
Training loss 0.041

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

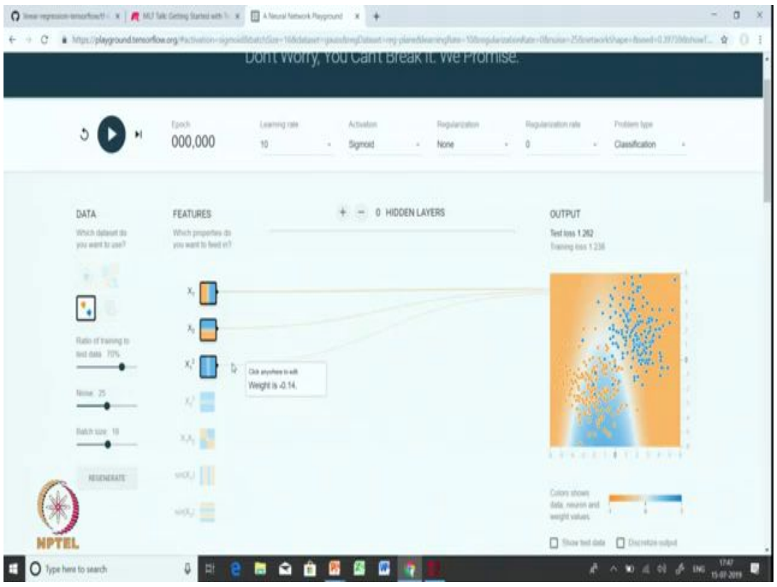
You can see that the points are you know the weights are randomly initialized, now there is a high weight on X_2 and low weight on X_1. So, this is the initial stage, let us start stepping through. You can see that it is kind of learning very slowly. If you play this further, you can see that it takes longer; it has already taken more than 500 epochs, and it is not even near to any of the numbers that we got from our first experiment. You can see that it is training very slowly let us go to the other exchange.

(Refer Slide Time: 11:30)



Let us use the learning rate of ten, which is very high. And you can see that it in one epoch we manage to get to the zero loss. If you reduce it to 1, you can see that pretty much 1 epoch we are able to achieve very low training and test loss. You can see that when we use 0.1, in first seven epochs we reach very close to zero loss. So, now let us try to increase the noise level and see how algorithm response to this. Now, we got loss which is much higher than what we were getting previously and it is obvious because we have some of the points that are misclassified. And you can also check the weights here on each one of them. If we increase learning rate, we are able to train faster.

(Refer Slide Time: 13:05)

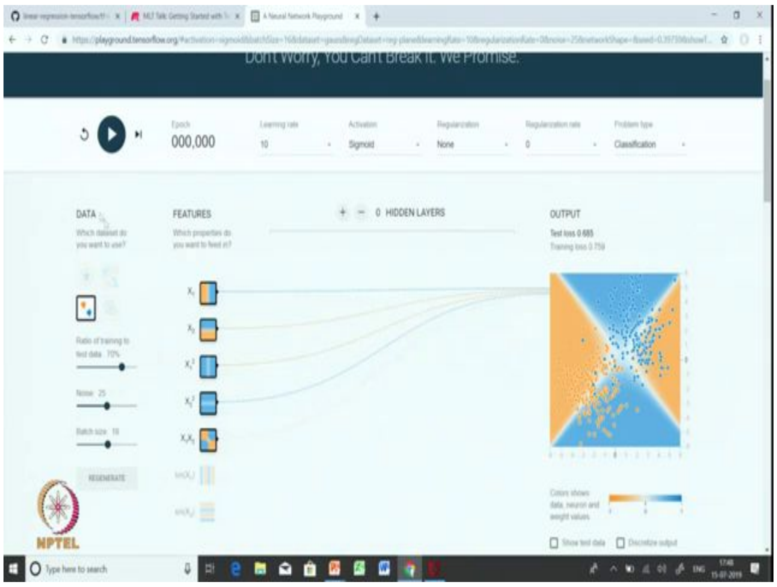

Epoch 000,000 Learning rate 10 Activation Sigmoid Regularization None Regularization rate 0 Problem type Classification

OUTPUT
Test loss 1.262
Training loss 1.238

Weight is -0.14.

So, we are not able to get a zero error rate so, what we can do is we can increase the complexity of the model, and try to learn some more complex representation, so that our losses are reduced. So, one way of increasing the complexity is by raising the polynomial degree of the original input features. So, here what we will do is, we will raise the power of X_1 and X_2 and also add an interaction term and see how it affects the performance.

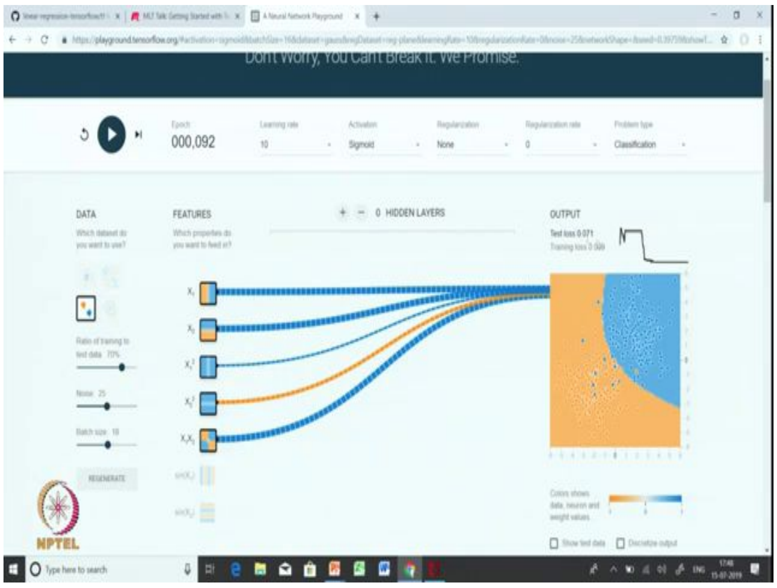
(Refer Slide Time: 13:30)


Epoch 000,000 Learning rate 10 Activation Sigmoid Regularization None Regularization rate 0 Problem type Classification

OUTPUT
Test loss 0.685
Training loss 0.759

So, you can simply click on this to add the second the square of X_1, we can add we can click on this to get a square of X_2, and we can click on this to get interaction term. And now let us try to train again.

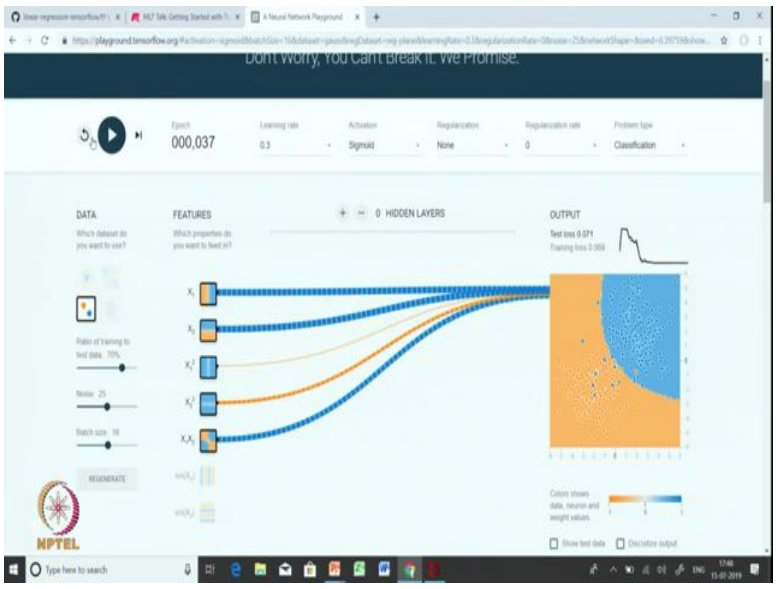
(Refer Slide Time: 13:53)



Epoch 000,092
Learning rate 10
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification
Test loss 0.071
Training loss 0.0

You see now when I used learning rate of 10, we can see that there are oscillations let us play it back and see how the oscillation plays out, we can see that it is oscillating.

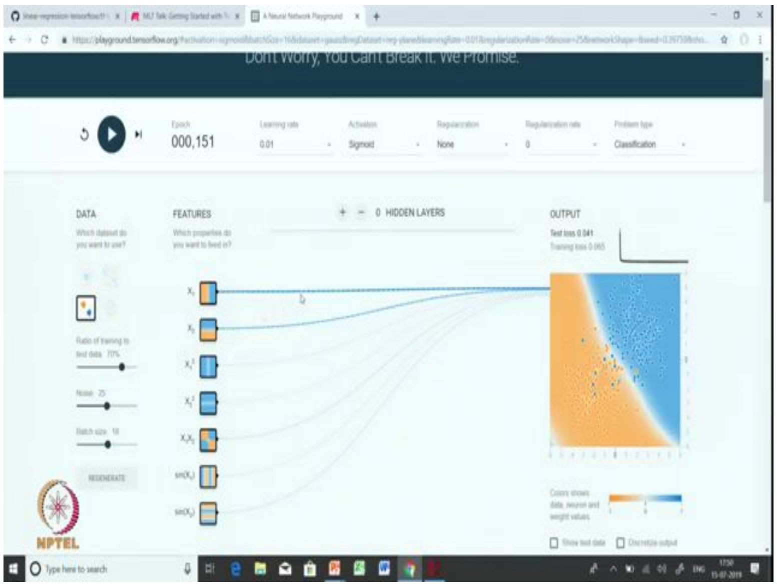
(Refer Slide Time: 14:10)



Epoch 000,037
Learning rate 0.3
Activation Sigmoid
Regularization None
Regularization rate 0
Problem type Classification
Test loss 0.071
Training loss 0.0

So, of course, this is the very high learning rate, we will bring it down to 0.3 and see what happens

(Refer Slide Time: 14:19)


Don't Worry, You Can't Break It. We Promise.

Epoch 000,151 | Learning rate 0.01 | Activation Sigmoid | Regularization None | Regularization rate 0 | Problem type Classification

Ratio of training to test data: 70%

Noise: 25

Batch size: 10

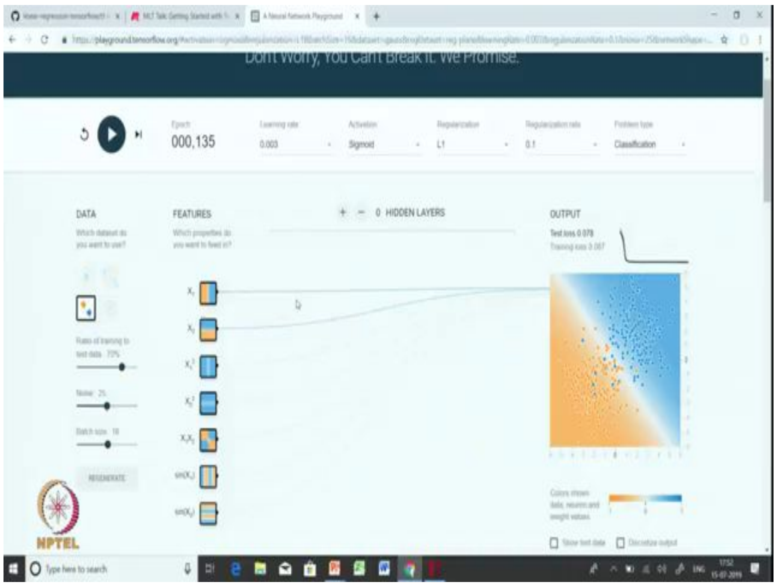
Test loss 0.041

Training loss 0.065

So, now you can observe that the; so, in the initial two experiments, we had a boundary which was linear, but now you can see that we have boundary which is non-linear. We have boundary which is slightly curved and this kind of complex boundary we were able to add, because we added these interaction features.

The overall separator is you know linear combination of all the individual separators and you can see different weights here. So, let us try to train even more slowly and see what happens. You know which seem to have reached I mean we are not able to get it below that. So, let us add couple of more terms which are signs of the original features and we will try to. You can see that now we have boundary which is even more interesting even more complex than the previous ones.

(Refer Slide Time: 16:15)



Epoch 000,135 · Learning rate 0.003 · Activation Sigmoid · Regularization L1 · Regularization rate 0.1 · Problem type Classification

0 HIDDEN LAYERS

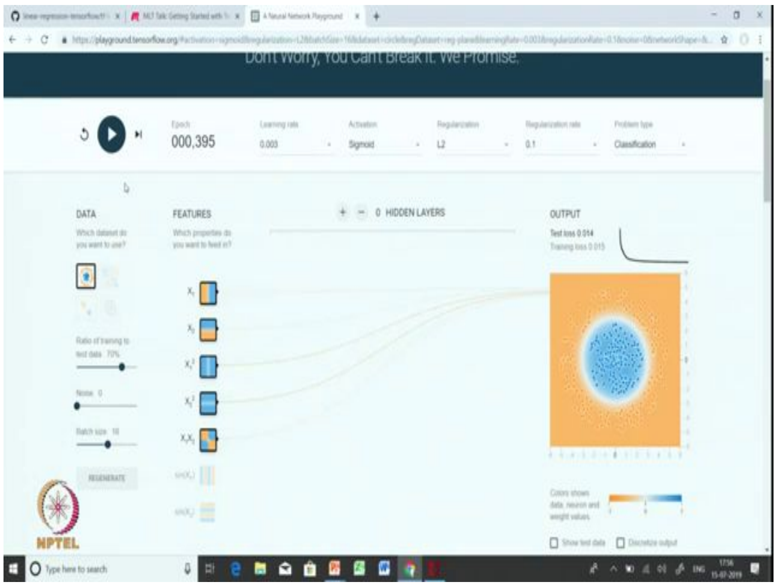
Test loss 0.078

So, let us try to add regularization and see the effect of regularization. We will start with L2 regularization. Before doing that, let us once run the whole thing and note down the weights. So, you can see that X_1 and X_2 are strong positive weights, and then there is some weight on the sin(X). And then all other elements do not have that high weights because they are quiet faint, you can make it out based on their width. So, let us say we use L2 regularization with regularization coefficient of 0.1, let us see what happens now.

You can so probably learning rate is a bit higher. Yeah, you can see more complex boundary getting learnt and it is very interesting to see that it is using now small weights everywhere else except for these two features. If we use L1 regularization here, let us see what happens. If I use L1 regularization, L1 regularization has tendency to put zero weight to the features that are not important. And you can see that all these features got zero weights so, only features that are important here is X_1 and X_2. So, so you can see that L1 regularization can also be used for feature selection for getting the features which are probably more important in the classification task.

And indeed L1 regularization is used for model feature selections. One of the way in which you can build machine learning models if you have enough competition power, is take your features, raise it to some degree of polynomial and use L1 regularization with sufficient regularization rate, to get feature selection in the process of training. So, the training will happen and you know most important features will get picked up. Later we will see that, we do not have to construct the feature crosses by hand and neural network takes care of that automatically.

(Refer Slide Time: 18:45)

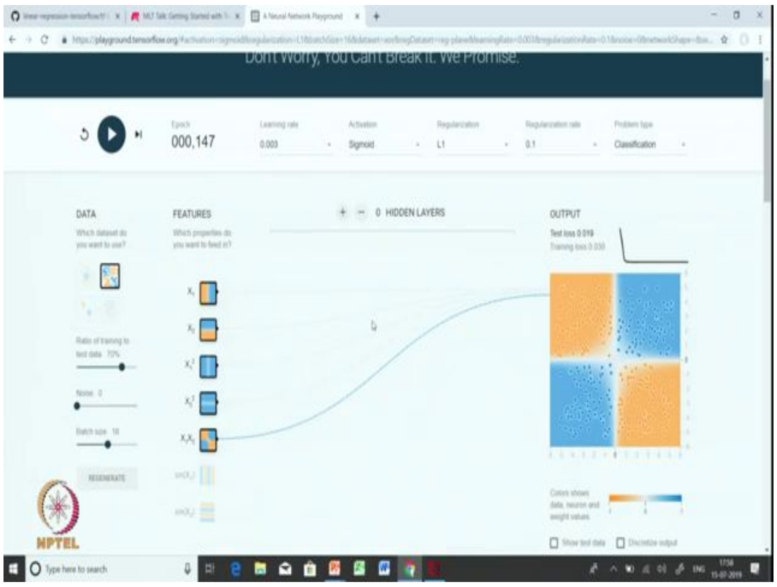

Epoch 000,395
Learning rate 0.003
Activation Sigmoid
Regularization L2
Regularization rate 0.1
Problem type Classification
0 HIDDEN LAYERS
Test loss 0.014
Training loss 0.015
Ratio of training to test data 70%
Noise 0
Batch size 16
Don't Worry, You Can't Break It. We Promise.
DATA
Which dataset do you want to use?
FEATURES
Which properties do you want to feed in?
OUTPUT
REGENERATE
NPTEL

So, let us try to go to another data set and try to see. So, unlike our previous data set, this data set is non-linearly separable. So, even if you have a clean data without any noise, we cannot just use the original features X1 and X2, because they simply do not have the capacity to learn the complex boundary which is a circle in this case separating both the classes. So, what we will do is, we will right away, we will actually train it once and see where we reach. We can see that the training error is 0.49, let us add the interaction features and see where we reach.

And now let us retrain it again. You can see that within, so we are getting very low training and test error. So, we are getting almost a perfect classifier which is a circle which is separating two classes. So, we can stop it, and let us try to use L 1 regularization here with a point one regularization rate and see what happens if you retrain this. So, you can see that we have again achieved fairly low training and test error. And now you can see that only features that are important are the squared features, which is called obvious because it is a circular decision boundary. So, squared features obviously, will have larger say. So, yeah so you can clearly see that you know here, we, we did feature crosses raised it to degree of two polynomial and we simply apply L1 regularization which give us these two features.

Let us try to apply L2 regularization and see what happens. Now, L2 regularization also got us fairly similar training and test error, but you can see that L2 regularization does not assigns zero weights to the features, instead it assigns weights which are very small. So, this is one of the differences that you can note that you can observe in L1 and L2. I would suggest not to change the activation type here because we are solving this is the classification problem sigmoid is a right activation type, but I would suggest you I would strongly encourage you to change the learning rate and regularization rate, try to add more noise in the data and see whether you can feed the model and how the model looks like after getting you know fairly low training and test error.

(Refer Slide Time: 22:03)


Don't Worry, You Can't Break It. We Promise.

Epoch 000,147 | Learning rate 0.003 | Activation Sigmoid | Regularization L1 | Regularization rate 0.1 | Problem type Classification

DATA | FEATURES | 0 HIDDEN LAYERS | OUTPUT

Test loss 0.019
Training loss 0.030

Which dataset do you want to use?
Ratio of training to test data: 70%
Noise: 0
Batch size: 10
REGENERATE
Which properties do you want to feed in?
NPTEL

So, let us try on the final data set which is XOR data. This is the one more interesting data set you can see that the classes are in the XOR situation. So, let us hope with a simple linear classify. So, we add interaction features or the second order polynomial features. And we can do the training and see what happens. So, quickly it went down to reasonably low error and we can see that you know we have a complex decision boundary. And the most important feature is the interaction feature that helps us predict this particular thing, all other features have very small weights around zero.

So, now if you apply, if we do not regularization, still we see that this is the most dominating feature. If you use L1 regularization, we can see that all the features having driven to weight of 0, only one feature which is the most important feature which is the interaction feature has got a strong positive weight, and we are able to separate the two classes. So, this was a nice visual way of intuitively learning how machine learning algorithms perform under different data sets and different noise added to the data set.

So, in this session, we looked at linearly separable data set, a non-linearly separable data set and XOR data set, and applied classification technique on them to classify points into the correct classes. We also studied how we can use the interaction features and L1 and L2 regularization in the context to you know to control the model complexity.

Hope you enjoyed learning this session with us. This brings us to an end of machine learning refresher using neural network playground. In the upcoming session, we are going to do a similar refresher for deep neural networks. We will start with the basic primer on deep neural networks. We will follow that with some mathematical foundations of deep learning through coding and we will also visualize some of the concepts of neural networks and their application to different data set to neural network playground.