Machine Learning Homework 3

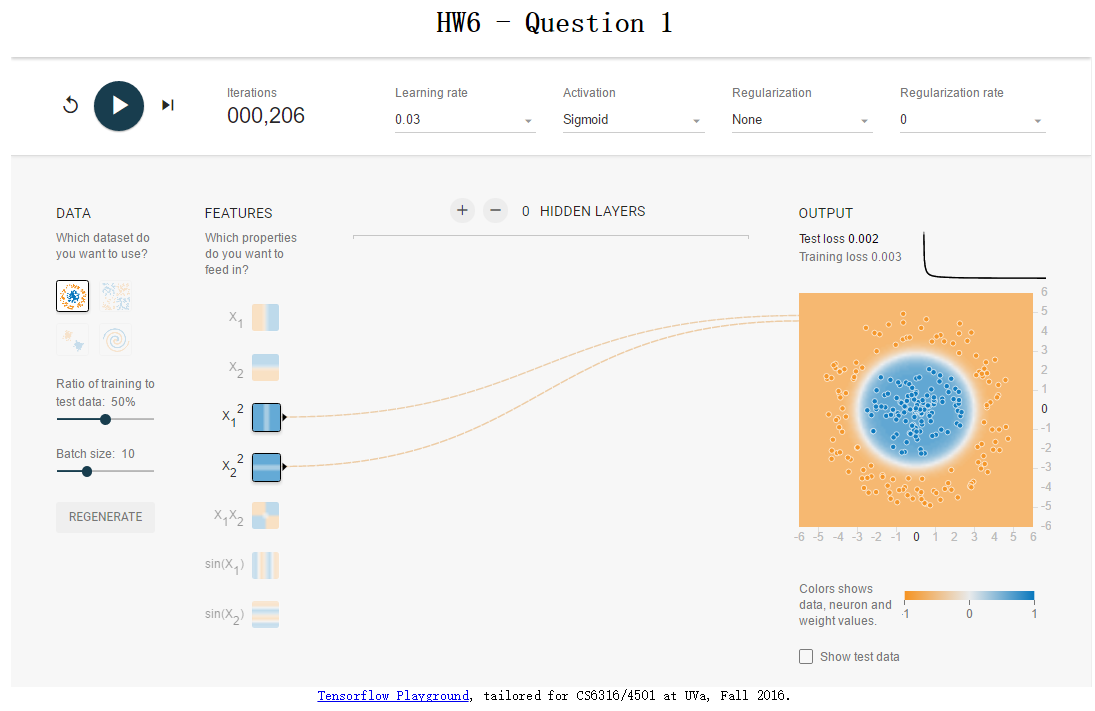
**Question 1:**

* 1. Hand craft Feature Engineering:

**Circle:**

**Feature selected:** X12, X22

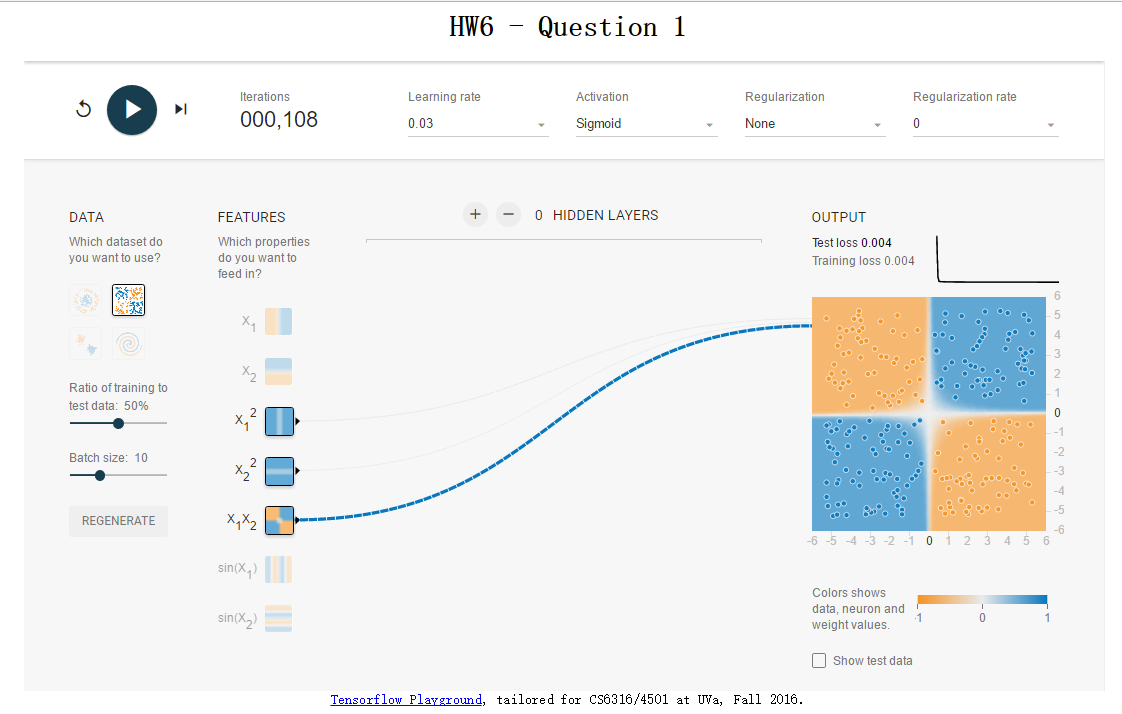
**Explanation:** The data samples can be distinguished by a circle. Thus X12, X22 are used to classify the data samples.



**Exclusive Or:**

**Feature selected:** X1X2

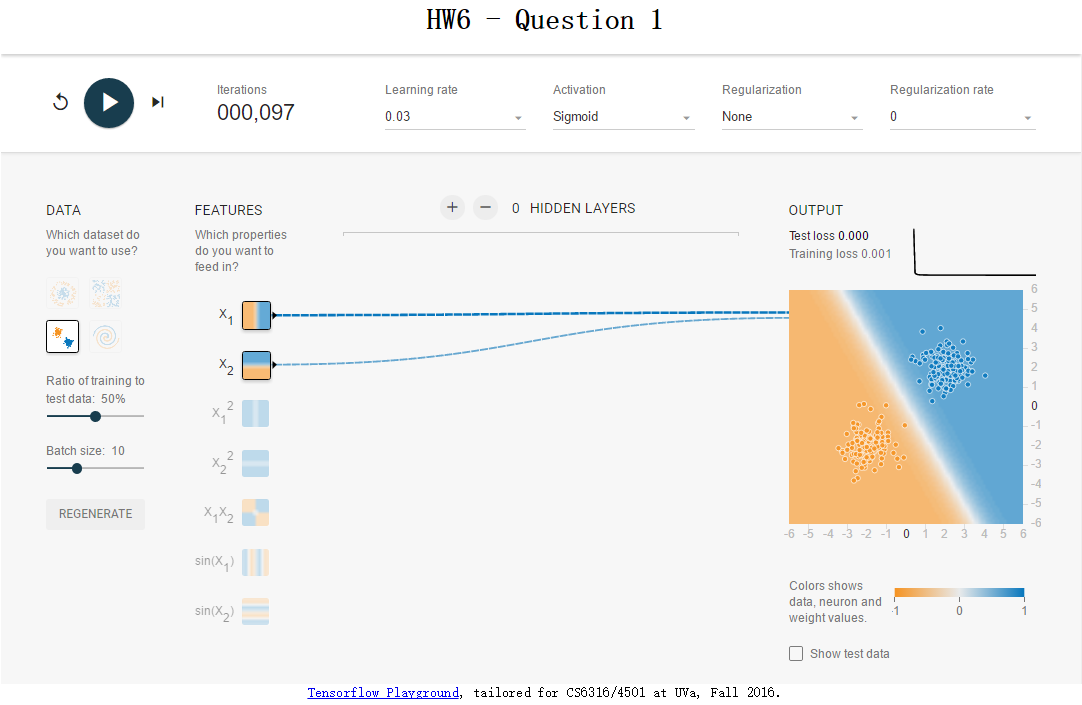
**Explanation:** X1X2 forms an inverse proportional function, which could classify the data samples easily.



**Gaussian:**

**Feature selected:** X1, X2

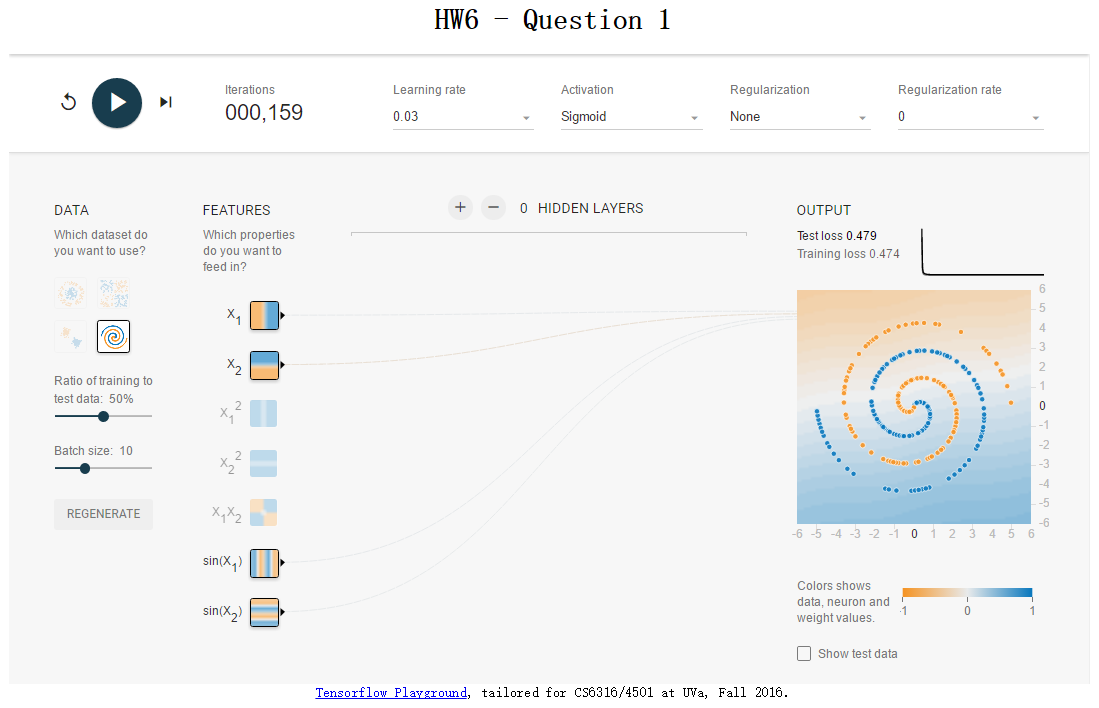
**Explanation:** The data samples can be linearly classified, thus we can choose only X1, X2 to classify them.



**Spiral:**

**Feature selected:** X1, X2, sin(X1), sin(X2)

**Explanation:** Cannot distinguish data samples using linear expressions.

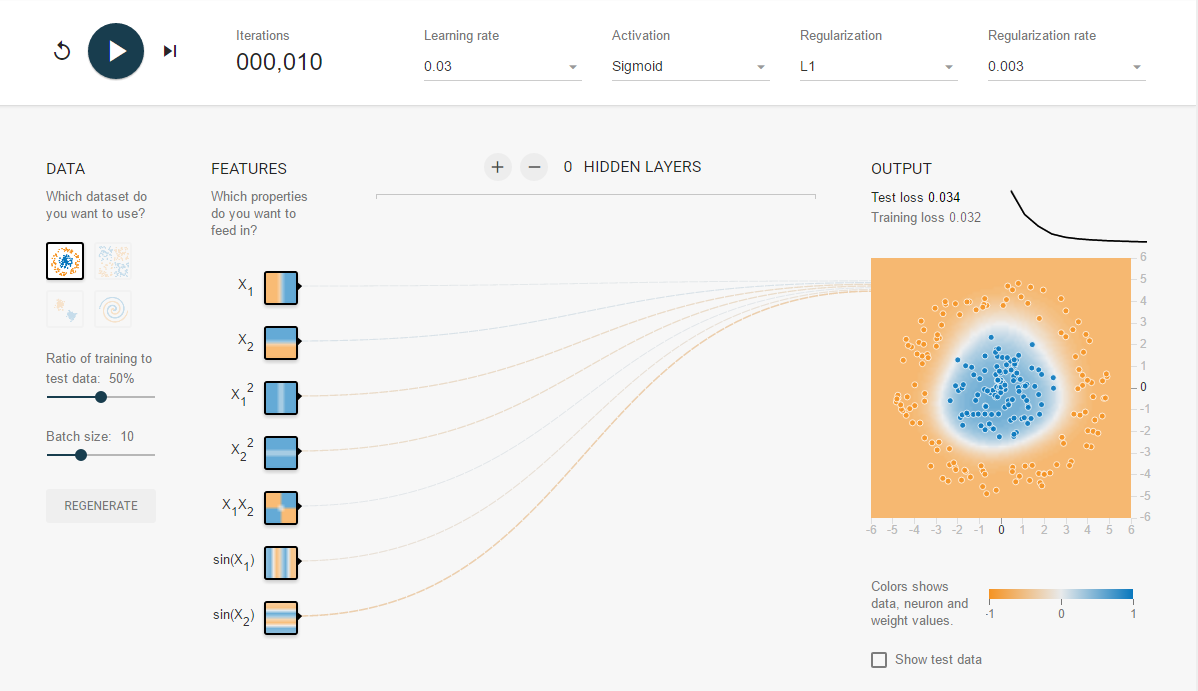


* 1. **Regularization:**

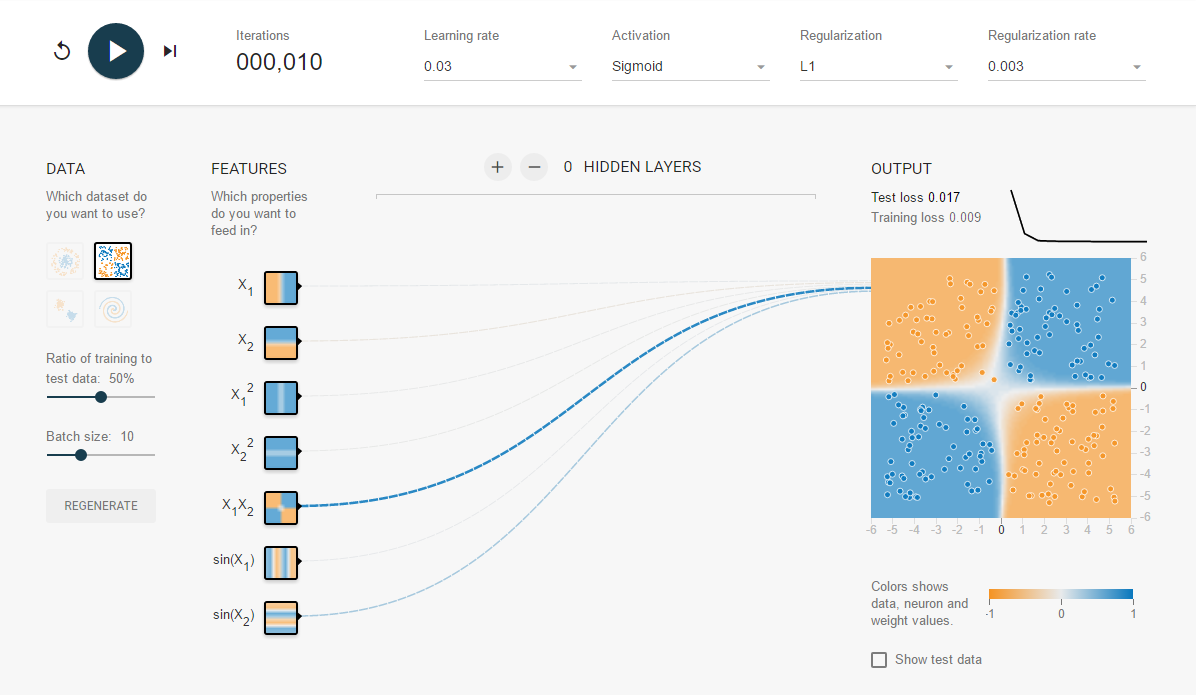
**TASK A:**

In this section, 10 iterations and 3 different regularization rates are used for every different regularization form. Below is the test error table for every regularization rate and regularization form. As shown in the table, these three data samples converge quickly and in general, L1 is better than the other two regularization form.

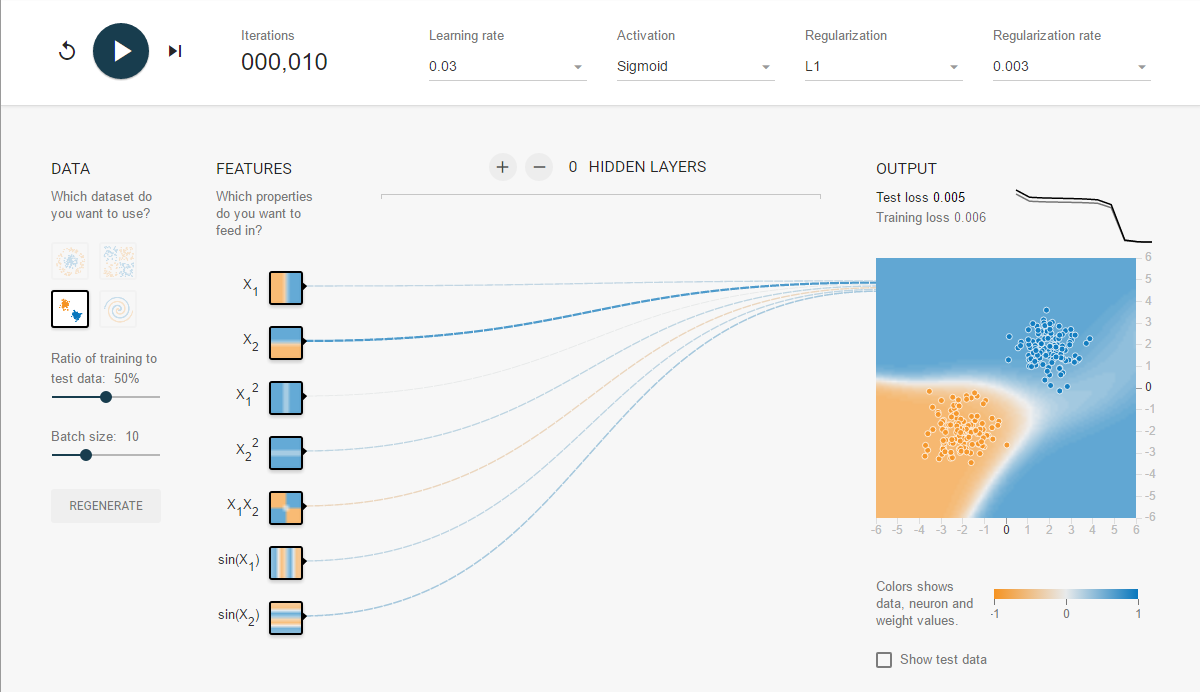
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | None | | | L1 regularization | | | L2 regularization | | |
| 0 | 0.03 | 0.1 | 0 | 0.03 | 0.1 | 0 | 0.03 | 0.1 |
| Circle | 0.037 | 0.039 | 0.038 | 0.038 | 0.037 | 0.037 | 0.036 | 0.037 | 0.039 |
| Gaussian | 0.006 | 0.002 | 0.003 | 0.004 | 0.003 | 0.008 | 0.003 | 0.005 | 0.003 |
| Exclusive Or | 0.008 | 0.008 | 0.007 | 0.009 | 0.007 | 0.008 | 0.009 | 0.007 | 0.008 |



Circle, 10 iterations, L1 regularization, 0.003 regularization rate



Exclusive Or, 10 iterations, L1 regularization, 0.003 regularization rate



Gaussian, 10 iterations, L1 regularization, 0.003 regularization rate

**TASK B:**

By checking the weight of all input features in L1 regularization, we found that the features with significantly high weights are the same with the ones that I chose in 1.1, which proves my conjecture is right.

* 1. **Automated Feature Engineering with Neural Network**

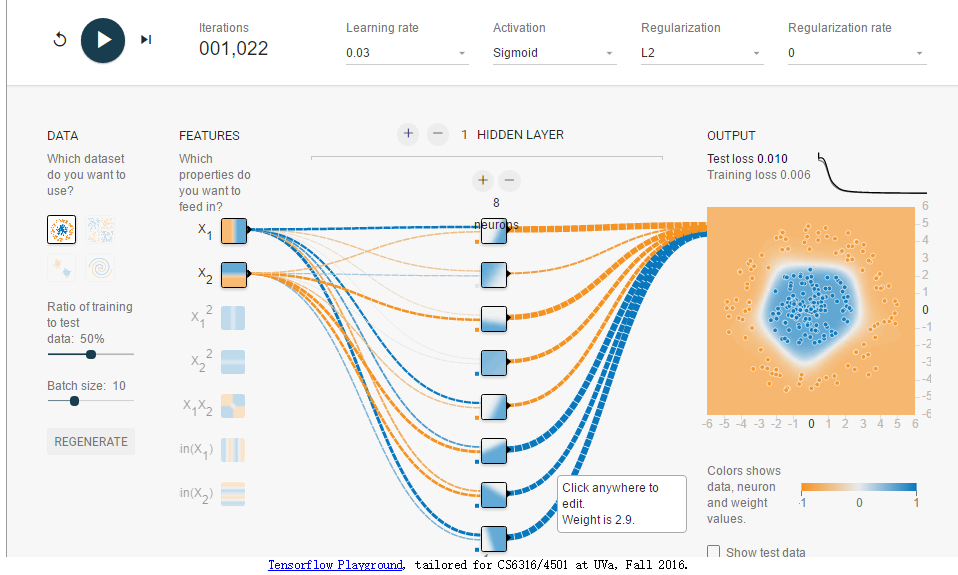
**By using** neural network to train the data, users can no longer be worried about the difficulty of choosing the best parameters, as the model can train it-self. Below is the exact information between circle data and Exclusive Or.

|  |  |  |
| --- | --- | --- |
|  | Test loss | Iterations |
| Circle | 0.010 | 1022 |
| Exclusive OR | 0.032 | 577 |

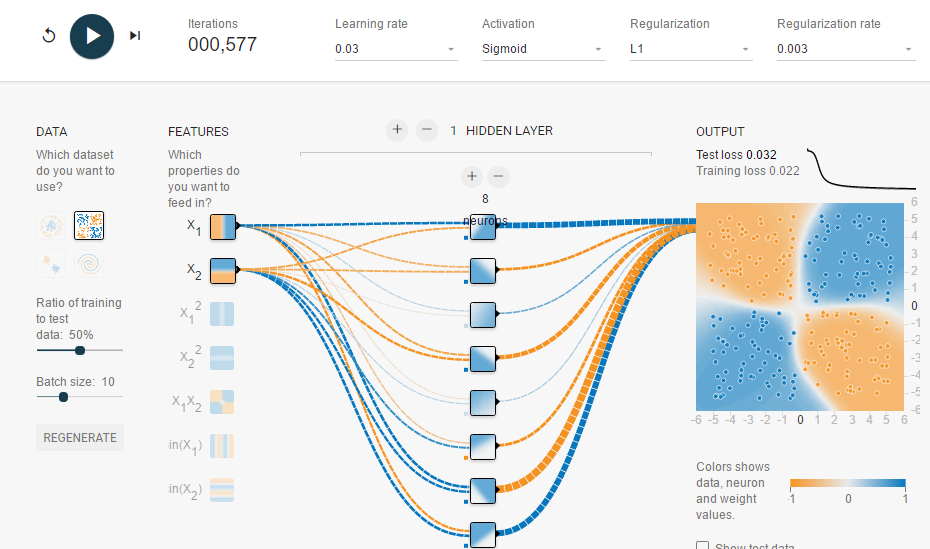
Model:

Circle: 1 hidden layer, 8 neurons; L1 regularization; Regularization rate – 0.

Exclusive OR: 1 hidden layer, 8 neurons; L1 regularization; Regularization rate – 0.03



Circle, 1 hidden layer 8 neurons



Exclusive OR, 1 hidden layer 8 neurons

As shown above, by using neural network, the model can train itself and find a proper model to classify the data. However, compared to hand-crafted classifier, the neural network is much slower. The hand-crafted classifier would converge no more than 50 iterations. While neural network would take up to 1000 iterations to converge. I think the reason is that neural network has to fit parameters randomly and choose the best ones, which is time consuming.

* 1. **Spiral Challenge**

Tried to use ReLU activation function to train a neural network, which converge very fast, compared to sigmoid activation function.

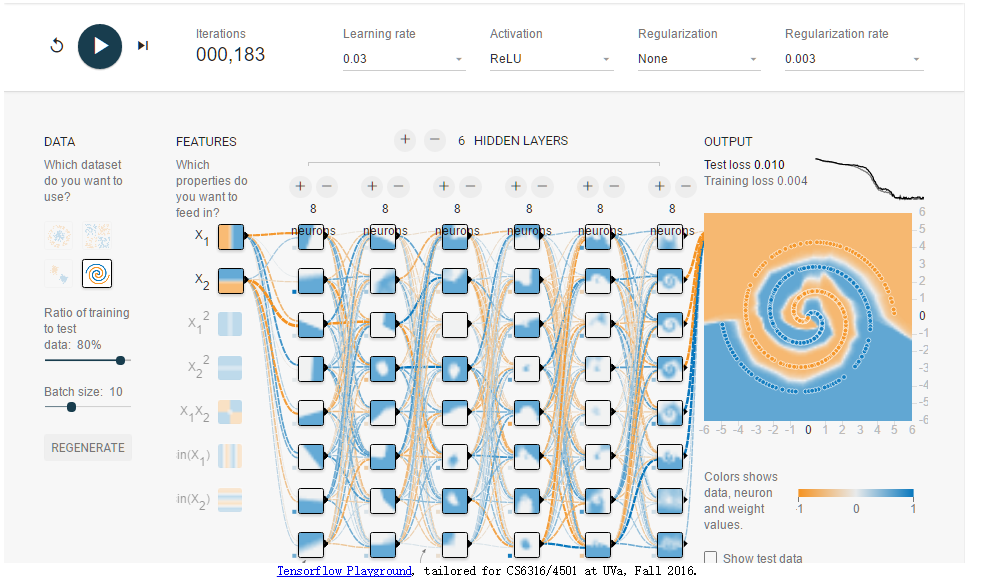
Input feature: X1, X2;

Network architecture: ReLU; 8 hidden layer with 8 neurons.

Hyper parameters: learning rate 0.03; None Regularization; Regularization rate 0.003;

Iteration: 183

Test loss: 0.010



**Question 2:**

**2.1 Decision Tree:**

In this section, I tried three parameters to customize the decision tree

1. Default; Accuracy = 82.461
2. criterion="entropy", max\_features=20, random\_state=10; Accuracy = 82.660
3. presort=True, random\_state=5, class\_weight="balanced"; Accuracy = 83.859

The best model is not the default one, but the third one. Basically, the reason for its higher performance is because the data is presorted under the third situation. As for some other feature, e.g. random\_state, is the seed used by the random number generator, to determine whether a random number is used to generate the decision tree.

**2.2 K Nearest Neighbors:**

In this section, I tried 5 ks to test the performance of K-nearest neighbors.

|  |  |
| --- | --- |
| **K** | **Accuracy** |
| 1 | 94.369 |
| 5 | 94.469 |
| 10 | 93.572 |
| 20 | 91.778 |
| 200 | 82.810 |

The value of K with highest accuracy is 5. When the k is too smal (e.g. 1) it may overfit the model. It may gain a high train-accuracy, but the test loss may raise. While when the k is too big (e.g. 200), the model may include too many noise point, thus it influent the accuracy of the model. While distance weighting may improve the performance of the model with big k. As the further the points, get the smaller weight, which means the weight of noise points may decrease.

**2.3 Support Vector Machine:**

I tested the RBF, Linear, and Polynomial kernels for SVM. The RBF’s accuracy is the highest. The parameters and accuracy details are shown below:

|  |  |
| --- | --- |
| Kernel | Accuracy |
| RBF | 95.316 |
| Linear | 93.173 |
| Polynomial | 93.173 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Kernel** | **C** | **Degree** | **Gamma** |
| RBF | 100 | 3 | 0.01 |
| Linear | 0.01 | 3 | 0.01 |
| Polynomial | 1 | 1 | 0.01 |

Theoretically, I think SVM is suitable to this problem. Because these data can be classified clearly using hyper plains. And SVM could find the best margin to classify the data.

**2.4 PCA**

In this section, I tried 5 principal components and 20 principal components, using KNN and SVM. The performances are shown below:

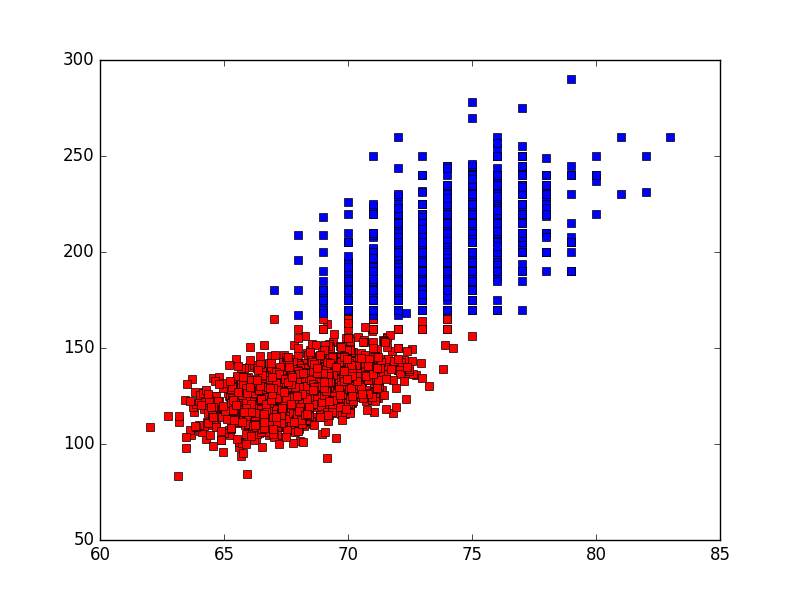
|  |  |  |
| --- | --- | --- |
| Strategy | PC | Accuracy |
| KNN | 5 | 77.77 |
| 20 | 93.37 |
| SVM | 5 | 79.97 |
| 20 | 94.17 |

When we choose an extreme low principle component, the performance is really bed, but when we raise the principle components to 20, the performance is much better, and even similar to that under 256 components. However, 20 components is much smaller than 256, which make the fit process much faster.

**Question 3. Unsupervised Learning with Clustering**

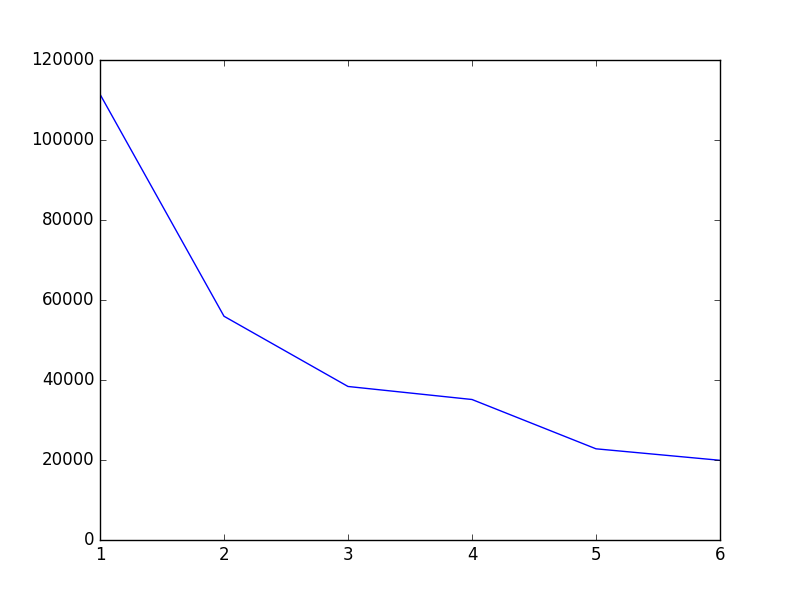
Q2: I use 100 iterations in this project.

Q3: Plot the clustering result using k-means:



The red points represent cluster of 1, while the blue points represents the cluster of 0.

Q4: The knee-finding picture. As can be seen in the picture, when set cluster number as 2, the submission of distance has a highest changing rate. Thus, cluster these data into two clusters is suitable.



Q5: The purity of the two cluster is 97% and 99%, which I think is ideal.