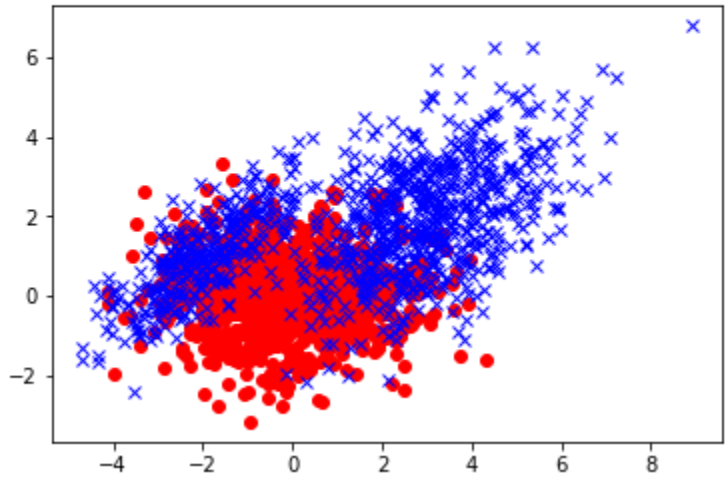
UNIVERSITY OF CALIFORNIA, SANTA BARBARA
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Name: Ashish Vyas
Perm: 6931570
Course: ECE 283 Machine Learning
Homework: 2 Classification Using Neural Networks

Visualization of Dataset:

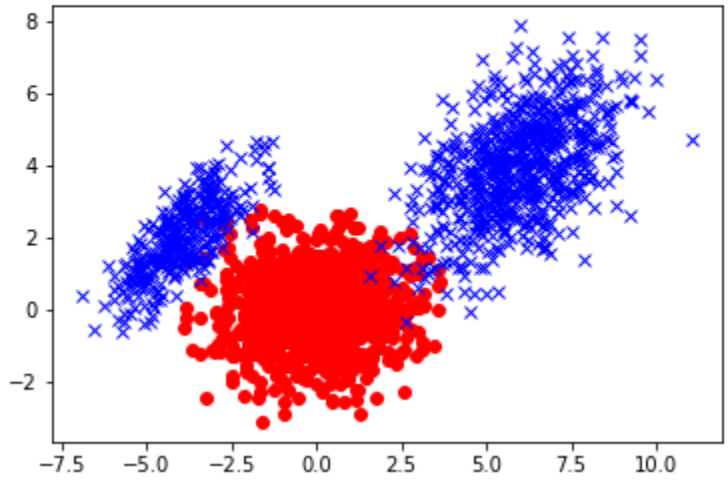
Dataset 1:

```
Training Data Size = 1280
Testing  Data Size = 400
Validation Data Size = 320
```



Dataset 2: Mean under Class 1 in doubled

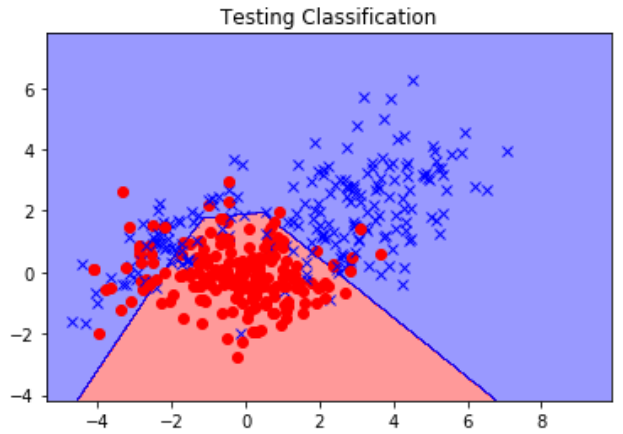
```
Training Data Size = 1280
Testing  Data Size = 400
Validation Data Size = 320
```

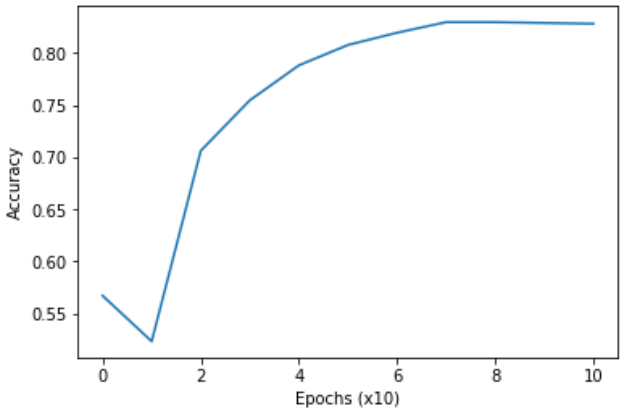


1.

- **Using Dataset 1**
- **Fully Connected with Single Hidden Layer**
    - **Neurons in Layer 1 = 5**
- **ReLu Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

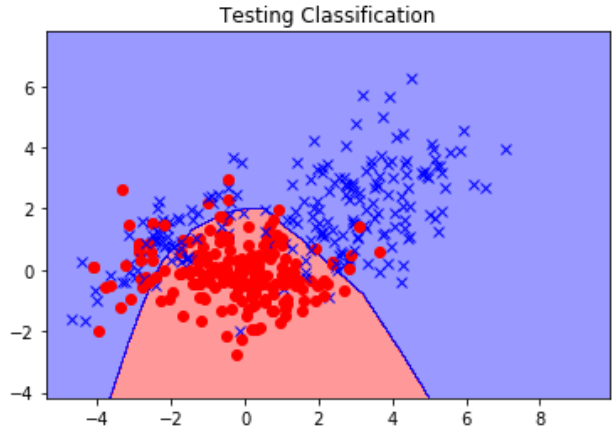
```
Step 1, Training Loss= 2.1068 & Accuracy= 0.567 || Validation Loss= 1.9625 & Accuracy= 0.572
Step 10, Training Loss= 0.7271 & Accuracy= 0.523 || Validation Loss= 0.7262 & Accuracy= 0.550
Step 20, Training Loss= 0.6153 & Accuracy= 0.706 || Validation Loss= 0.6231 & Accuracy= 0.703
Step 30, Training Loss= 0.5305 & Accuracy= 0.755 || Validation Loss= 0.5348 & Accuracy= 0.756
Step 40, Training Loss= 0.4816 & Accuracy= 0.788 || Validation Loss= 0.4735 & Accuracy= 0.828
Step 50, Training Loss= 0.4460 & Accuracy= 0.808 || Validation Loss= 0.4452 & Accuracy= 0.844
Step 60, Training Loss= 0.4253 & Accuracy= 0.820 || Validation Loss= 0.4198 & Accuracy= 0.866
Step 70, Training Loss= 0.4120 & Accuracy= 0.830 || Validation Loss= 0.3987 & Accuracy= 0.866
Step 80, Training Loss= 0.4062 & Accuracy= 0.830 || Validation Loss= 0.3899 & Accuracy= 0.856
Step 90, Training Loss= 0.4024 & Accuracy= 0.829 || Validation Loss= 0.3862 & Accuracy= 0.856
Step 100, Training Loss= 0.3992 & Accuracy= 0.828 || Validation Loss= 0.3836 & Accuracy= 0.856
Optimization Finished!
Testing Accuracy : 0.84249997139
```

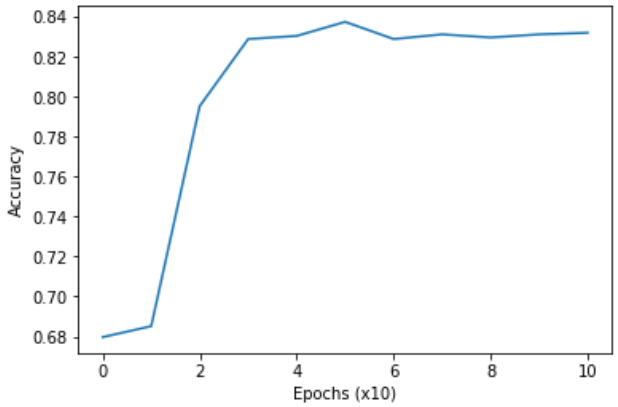




Increased no of neurons to 10

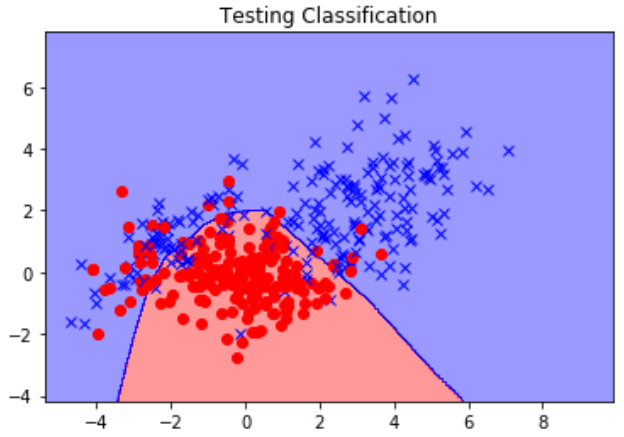
```
Step 1, Training Loss= 10.4544 & Accuracy= 0.680 || Validation Loss= 9.5025 & Accuracy= 0.656
Step 10, Training Loss= 0.9254 & Accuracy= 0.685 || Validation Loss= 1.0036 & Accuracy= 0.663
Step 20, Training Loss= 0.5886 & Accuracy= 0.795 || Validation Loss= 0.6097 & Accuracy= 0.819
Step 30, Training Loss= 0.5298 & Accuracy= 0.829 || Validation Loss= 0.5296 & Accuracy= 0.853
Step 40, Training Loss= 0.4911 & Accuracy= 0.830 || Validation Loss= 0.4701 & Accuracy= 0.850
Step 50, Training Loss= 0.4598 & Accuracy= 0.837 || Validation Loss= 0.4379 & Accuracy= 0.859
Step 60, Training Loss= 0.4381 & Accuracy= 0.829 || Validation Loss= 0.4193 & Accuracy= 0.853
Step 70, Training Loss= 0.4218 & Accuracy= 0.831 || Validation Loss= 0.4041 & Accuracy= 0.850
Step 80, Training Loss= 0.4094 & Accuracy= 0.830 || Validation Loss= 0.3971 & Accuracy= 0.856
Step 90, Training Loss= 0.4019 & Accuracy= 0.831 || Validation Loss= 0.3886 & Accuracy= 0.853
Step 100, Training Loss= 0.3963 & Accuracy= 0.832 || Validation Loss= 0.3843 & Accuracy= 0.856
Optimization Finished!
Testing Accuracy : 0.84500002861
```

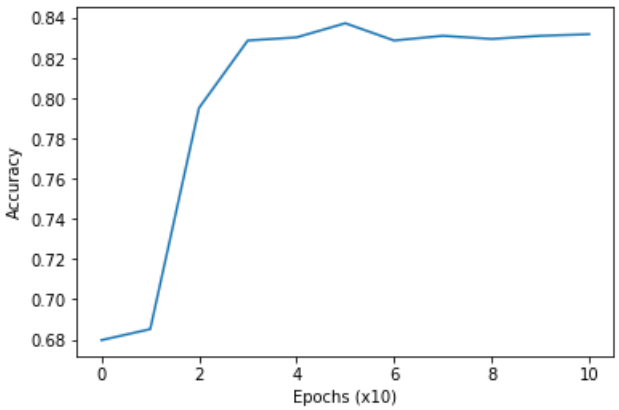




Increased no of neurons to 100

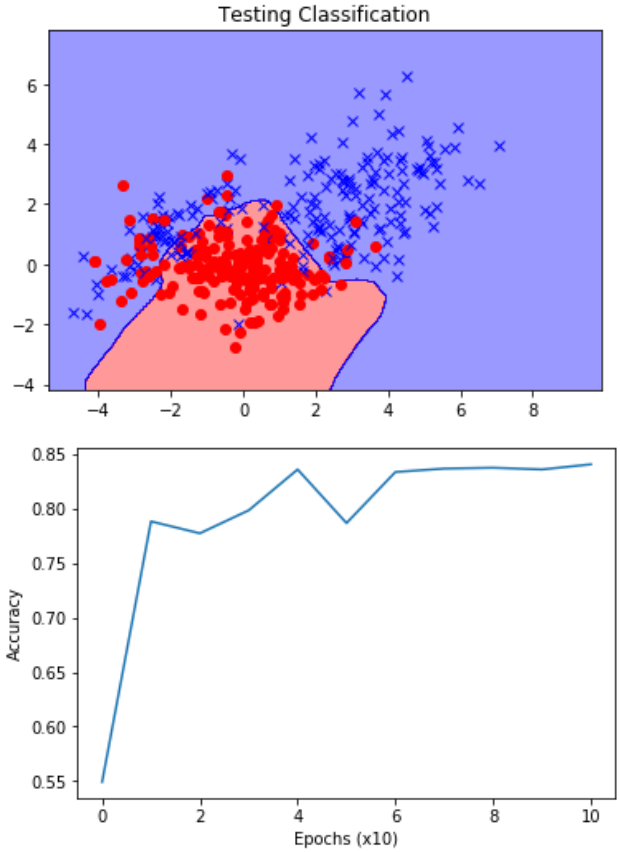
```
Step 1, Training Loss= 23.2502 & Accuracy= 0.533 || Validation Loss= 21.0845 & Accuracy= 0.575
Step 10, Training Loss= 2.1137 & Accuracy= 0.634 || Validation Loss= 2.2512 & Accuracy= 0.603
Step 20, Training Loss= 0.6839 & Accuracy= 0.807 || Validation Loss= 0.7117 & Accuracy= 0.812
Step 30, Training Loss= 0.6341 & Accuracy= 0.805 || Validation Loss= 0.6193 & Accuracy= 0.834
Step 40, Training Loss= 0.5945 & Accuracy= 0.820 || Validation Loss= 0.5851 & Accuracy= 0.837
Step 50, Training Loss= 0.5385 & Accuracy= 0.819 || Validation Loss= 0.5189 & Accuracy= 0.856
Step 60, Training Loss= 0.4988 & Accuracy= 0.834 || Validation Loss= 0.4817 & Accuracy= 0.863
Step 70, Training Loss= 0.4673 & Accuracy= 0.833 || Validation Loss= 0.4466 & Accuracy= 0.859
Step 80, Training Loss= 0.4439 & Accuracy= 0.830 || Validation Loss= 0.4261 & Accuracy= 0.856
Step 90, Training Loss= 0.4270 & Accuracy= 0.832 || Validation Loss= 0.4097 & Accuracy= 0.853
Step 100, Training Loss= 0.4151 & Accuracy= 0.834 || Validation Loss= 0.4001 & Accuracy= 0.856
Optimization Finished!
Testing Accuracy : 0.84500002861
```





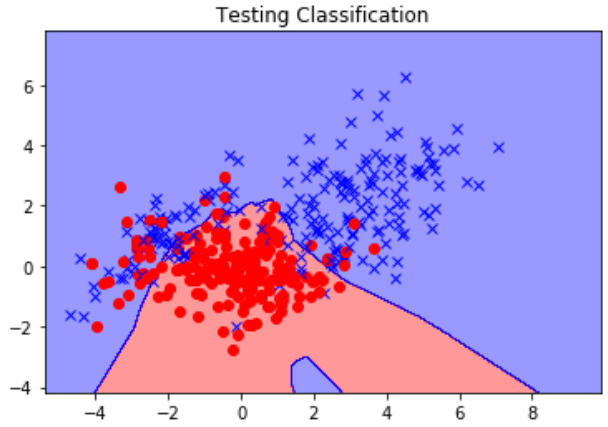
- **Using Dataset 1**
- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **ReLu Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs =100**

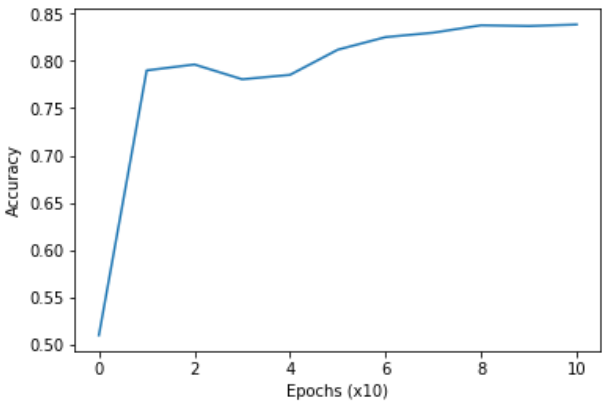
```
Step 1, Training Loss= 295.7397 & Accuracy= 0.549 || Validation Loss= 277.0569 & Accuracy= 0.587
Step 10, Training Loss= 65.1159 & Accuracy= 0.788 || Validation Loss= 63.3038 & Accuracy= 0.803
Step 20, Training Loss= 38.6312 & Accuracy= 0.777 || Validation Loss= 38.5744 & Accuracy= 0.788
Step 30, Training Loss= 26.8459 & Accuracy= 0.798 || Validation Loss= 26.8716 & Accuracy= 0.825
Step 40, Training Loss= 19.4214 & Accuracy= 0.836 || Validation Loss= 19.4543 & Accuracy= 0.853
Step 50, Training Loss= 14.3582 & Accuracy= 0.787 || Validation Loss= 14.3886 & Accuracy= 0.809
Step 60, Training Loss= 10.8608 & Accuracy= 0.834 || Validation Loss= 10.8983 & Accuracy= 0.853
Step 70, Training Loss= 8.4316 & Accuracy= 0.837 || Validation Loss= 8.4727 & Accuracy= 0.859
Step 80, Training Loss= 6.6901 & Accuracy= 0.837 || Validation Loss= 6.7261 & Accuracy= 0.853
Step 90, Training Loss= 5.3872 & Accuracy= 0.836 || Validation Loss= 5.4191 & Accuracy= 0.850
Step 100, Training Loss= 4.4045 & Accuracy= 0.841 || Validation Loss= 4.4210 & Accuracy= 0.853
Optimization Finished!
Testing Accuracy : 0.847500026226
```



Increased no. of neurons in both layers to 10

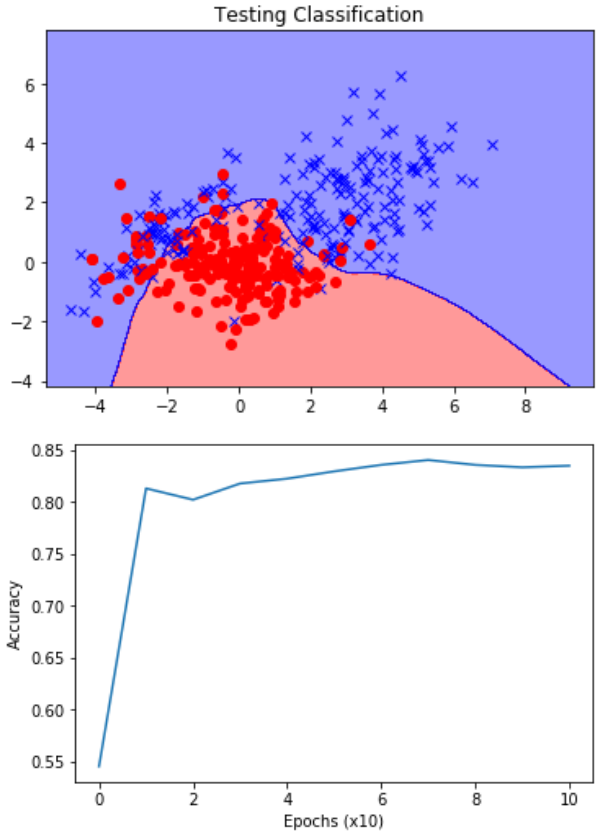
```
Step 1, Training Loss= 39.0193 & Accuracy= 0.510 || Validation Loss= 38.3361 & Accuracy= 0.544
Step 10, Training Loss= 12.7058 & Accuracy= 0.790 || Validation Loss= 11.6950 & Accuracy= 0.831
Step 20, Training Loss= 7.2938 & Accuracy= 0.796 || Validation Loss= 7.0948 & Accuracy= 0.791
Step 30, Training Loss= 4.8715 & Accuracy= 0.780 || Validation Loss= 4.7746 & Accuracy= 0.809
Step 40, Training Loss= 3.5752 & Accuracy= 0.785 || Validation Loss= 3.4881 & Accuracy= 0.809
Step 50, Training Loss= 2.6541 & Accuracy= 0.812 || Validation Loss= 2.6477 & Accuracy= 0.831
Step 60, Training Loss= 2.1046 & Accuracy= 0.825 || Validation Loss= 2.0957 & Accuracy= 0.859
Step 70, Training Loss= 1.7351 & Accuracy= 0.830 || Validation Loss= 1.7210 & Accuracy= 0.853
Step 80, Training Loss= 1.4690 & Accuracy= 0.837 || Validation Loss= 1.4529 & Accuracy= 0.859
Step 90, Training Loss= 1.2729 & Accuracy= 0.837 || Validation Loss= 1.2562 & Accuracy= 0.844
Step 100, Training Loss= 1.1249 & Accuracy= 0.838 || Validation Loss= 1.1075 & Accuracy= 0.847
Optimization Finished!
Testing Accuracy : 0.852500021458
```





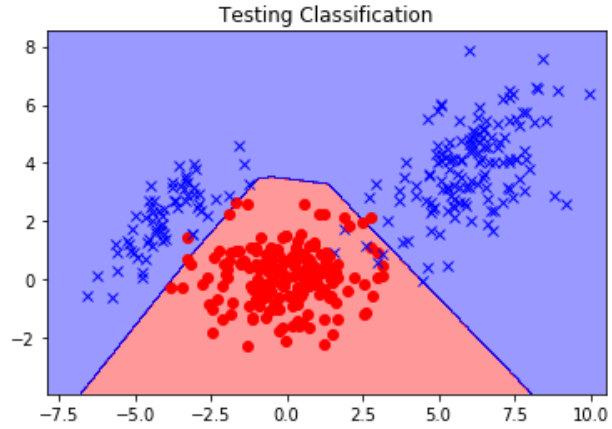
Increased no. of neurons in both layers to 100

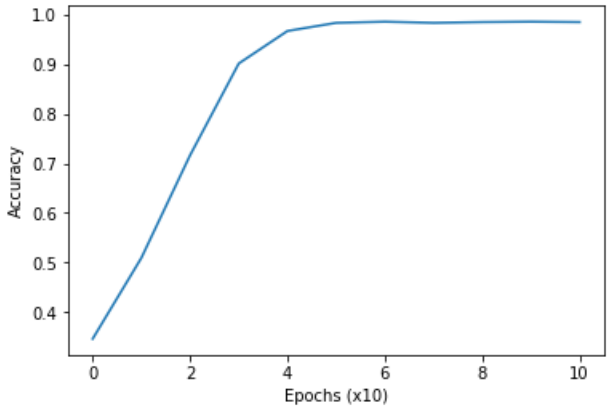
```
Step 1, Training Loss= 139.5102 & Accuracy= 0.545 || Validation Loss= 140.7211 & Accuracy= 0.572
Step 10, Training Loss= 36.0786 & Accuracy= 0.813 || Validation Loss= 35.5128 & Accuracy= 0.819
Step 20, Training Loss= 20.0917 & Accuracy= 0.802 || Validation Loss= 20.2741 & Accuracy= 0.819
Step 30, Training Loss= 13.1015 & Accuracy= 0.818 || Validation Loss= 13.1150 & Accuracy= 0.841
Step 40, Training Loss= 9.2767 & Accuracy= 0.823 || Validation Loss= 9.3095 & Accuracy= 0.847
Step 50, Training Loss= 6.7179 & Accuracy= 0.830 || Validation Loss= 6.7404 & Accuracy= 0.859
Step 60, Training Loss= 5.0001 & Accuracy= 0.836 || Validation Loss= 5.0148 & Accuracy= 0.847
Step 70, Training Loss= 3.8133 & Accuracy= 0.841 || Validation Loss= 3.8219 & Accuracy= 0.863
Step 80, Training Loss= 2.9684 & Accuracy= 0.836 || Validation Loss= 2.9764 & Accuracy= 0.856
Step 90, Training Loss= 2.3550 & Accuracy= 0.834 || Validation Loss= 2.3560 & Accuracy= 0.853
Step 100, Training Loss= 1.9023 & Accuracy= 0.835 || Validation Loss= 1.9018 & Accuracy= 0.847
Optimization Finished!
Testing Accuracy : 0.84500002861
```



- **Using Dataset 2**
- **Fully Connected with Single Hidden Layer**
    - **Neurons in Layer 1 = 5**
- **ReLu Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

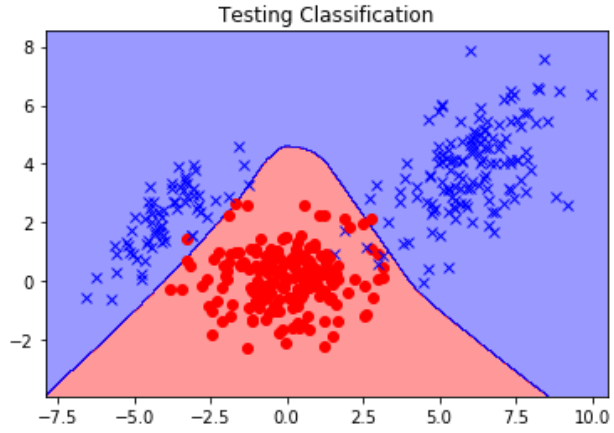
```
Step 1, Training Loss= 3.7209 & Accuracy= 0.346 || Validation Loss= 3.4042 & Accuracy= 0.363
Step 10, Training Loss= 0.6942 & Accuracy= 0.509 || Validation Loss= 0.7007 & Accuracy= 0.484
Step 20, Training Loss= 0.5122 & Accuracy= 0.716 || Validation Loss= 0.5302 & Accuracy= 0.728
Step 30, Training Loss= 0.3537 & Accuracy= 0.902 || Validation Loss= 0.3577 & Accuracy= 0.928
Step 40, Training Loss= 0.2337 & Accuracy= 0.967 || Validation Loss= 0.2336 & Accuracy= 0.978
Step 50, Training Loss= 0.1591 & Accuracy= 0.984 || Validation Loss= 0.1618 & Accuracy= 0.984
Step 60, Training Loss= 0.1222 & Accuracy= 0.986 || Validation Loss= 0.1269 & Accuracy= 0.988
Step 70, Training Loss= 0.0981 & Accuracy= 0.984 || Validation Loss= 0.1026 & Accuracy= 0.988
Step 80, Training Loss= 0.0854 & Accuracy= 0.985 || Validation Loss= 0.0911 & Accuracy= 0.988
Step 90, Training Loss= 0.0779 & Accuracy= 0.986 || Validation Loss= 0.0856 & Accuracy= 0.988
Step 100, Training Loss= 0.0722 & Accuracy= 0.985 || Validation Loss= 0.0816 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.977500021458
```

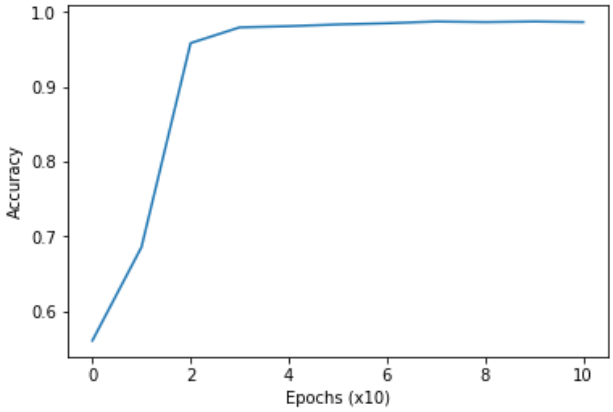




Increased no of neurons to 10

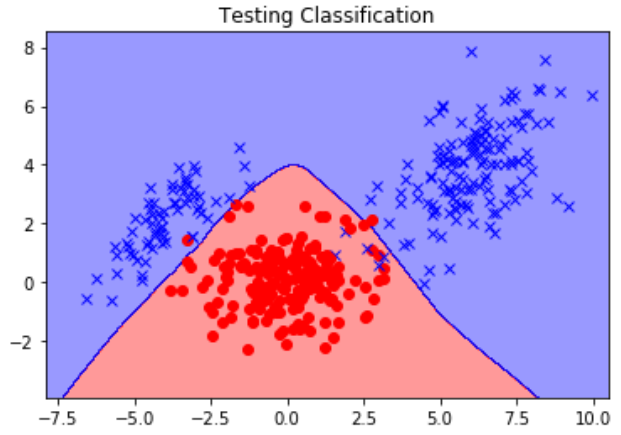
```
Step 1, Training Loss= 14.7172 & Accuracy= 0.561 || Validation Loss= 13.4115 & Accuracy= 0.528
Step 10, Training Loss= 1.1130 & Accuracy= 0.686 || Validation Loss= 1.0831 & Accuracy= 0.678
Step 20, Training Loss= 0.3690 & Accuracy= 0.959 || Validation Loss= 0.3871 & Accuracy= 0.962
Step 30, Training Loss= 0.2522 & Accuracy= 0.980 || Validation Loss= 0.2514 & Accuracy= 0.988
Step 40, Training Loss= 0.2105 & Accuracy= 0.981 || Validation Loss= 0.2155 & Accuracy= 0.988
Step 50, Training Loss= 0.1696 & Accuracy= 0.984 || Validation Loss= 0.1743 & Accuracy= 0.988
Step 60, Training Loss= 0.1413 & Accuracy= 0.985 || Validation Loss= 0.1508 & Accuracy= 0.988
Step 70, Training Loss= 0.1191 & Accuracy= 0.988 || Validation Loss= 0.1307 & Accuracy= 0.988
Step 80, Training Loss= 0.1016 & Accuracy= 0.987 || Validation Loss= 0.1172 & Accuracy= 0.988
Step 90, Training Loss= 0.0889 & Accuracy= 0.988 || Validation Loss= 0.1043 & Accuracy= 0.988
Step 100, Training Loss= 0.0802 & Accuracy= 0.987 || Validation Loss= 0.0989 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.980000019073
```

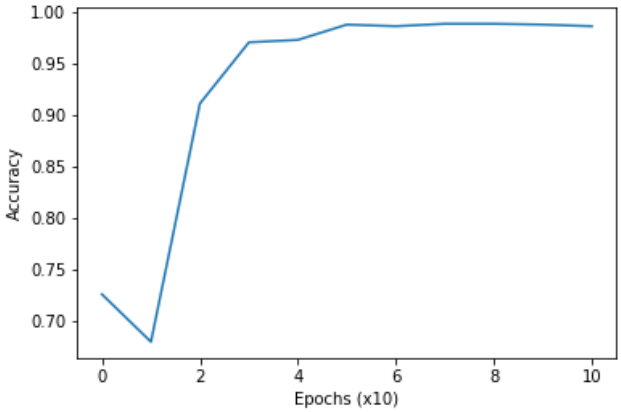




Increased no of neurons to 100

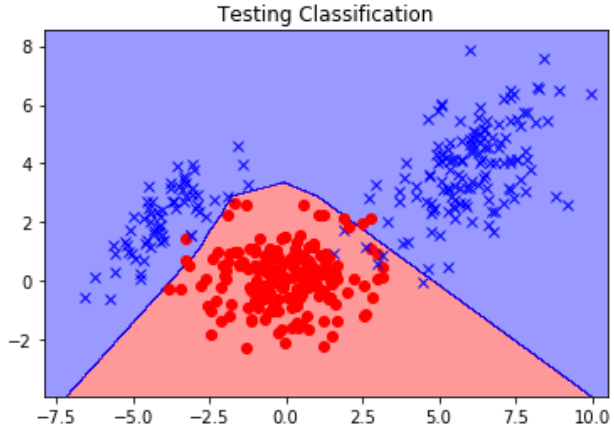
```
Step 1, Training Loss= 34.9933 & Accuracy= 0.726 || Validation Loss= 31.7208 & Accuracy= 0.706
Step 10, Training Loss= 1.9363 & Accuracy= 0.680 || Validation Loss= 1.8498 & Accuracy= 0.678
Step 20, Training Loss= 0.5688 & Accuracy= 0.911 || Validation Loss= 0.6248 & Accuracy= 0.931
Step 30, Training Loss= 0.3382 & Accuracy= 0.970 || Validation Loss= 0.3365 & Accuracy= 0.981
Step 40, Training Loss= 0.2770 & Accuracy= 0.973 || Validation Loss= 0.2835 & Accuracy= 0.984
Step 50, Training Loss= 0.2176 & Accuracy= 0.988 || Validation Loss= 0.2271 & Accuracy= 0.984
Step 60, Training Loss= 0.1827 & Accuracy= 0.986 || Validation Loss= 0.1963 & Accuracy= 0.988
Step 70, Training Loss= 0.1560 & Accuracy= 0.988 || Validation Loss= 0.1670 & Accuracy= 0.988
Step 80, Training Loss= 0.1361 & Accuracy= 0.988 || Validation Loss= 0.1510 & Accuracy= 0.988
Step 90, Training Loss= 0.1195 & Accuracy= 0.988 || Validation Loss= 0.1356 & Accuracy= 0.988
Step 100, Training Loss= 0.1056 & Accuracy= 0.986 || Validation Loss= 0.1233 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```

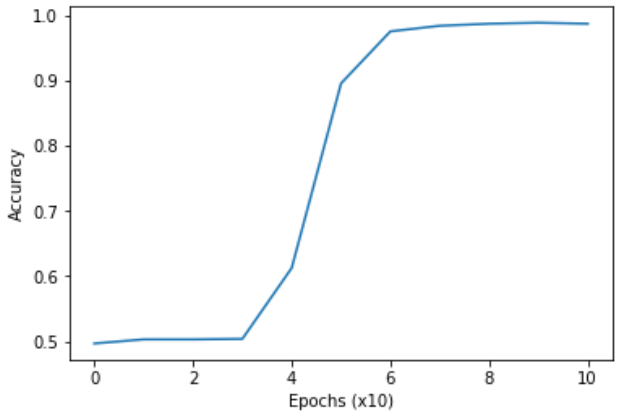




- **Using Dataset 2**
- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **ReLu Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

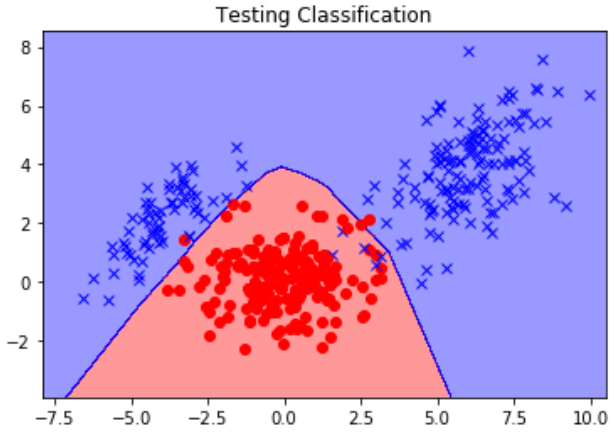
```
Step 1, Training Loss= 7.9087 & Accuracy= 0.497 || Validation Loss= 7.1416 & Accuracy= 0.531
Step 10, Training Loss= 0.9367 & Accuracy= 0.503 || Validation Loss= 0.9395 & Accuracy= 0.469
Step 20, Training Loss= 0.9216 & Accuracy= 0.503 || Validation Loss= 0.9275 & Accuracy= 0.469
Step 30, Training Loss= 0.8674 & Accuracy= 0.504 || Validation Loss= 0.8614 & Accuracy= 0.534
Step 40, Training Loss= 0.8085 & Accuracy= 0.613 || Validation Loss= 0.7941 & Accuracy= 0.659
Step 50, Training Loss= 0.5164 & Accuracy= 0.895 || Validation Loss= 0.5154 & Accuracy= 0.913
Step 60, Training Loss= 0.3172 & Accuracy= 0.975 || Validation Loss= 0.3315 & Accuracy= 0.981
Step 70, Training Loss= 0.2398 & Accuracy= 0.984 || Validation Loss= 0.2473 & Accuracy= 0.984
Step 80, Training Loss= 0.1994 & Accuracy= 0.987 || Validation Loss= 0.2064 & Accuracy= 0.988
Step 90, Training Loss= 0.1766 & Accuracy= 0.988 || Validation Loss= 0.1837 & Accuracy= 0.988
Step 100, Training Loss= 0.1587 & Accuracy= 0.987 || Validation Loss= 0.1677 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.975000023842
```

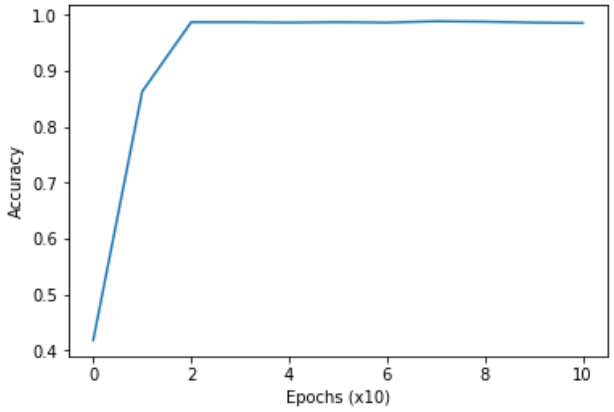




Increased no. of neurons in both layers to 10

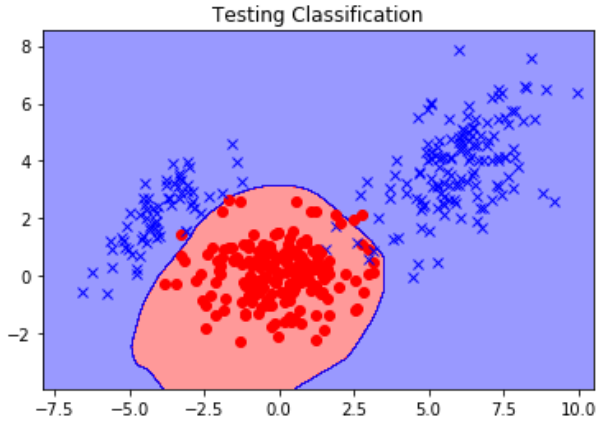
```
Step 1, Training Loss= 2.6889 & Accuracy= 0.419 || Validation Loss= 2.8836 & Accuracy= 0.359
Step 10, Training Loss= 0.7214 & Accuracy= 0.863 || Validation Loss= 0.7138 & Accuracy= 0.881
Step 20, Training Loss= 0.4016 & Accuracy= 0.988 || Validation Loss= 0.4261 & Accuracy= 0.984
Step 30, Training Loss= 0.3079 & Accuracy= 0.988 || Validation Loss= 0.3543 & Accuracy= 0.981
Step 40, Training Loss= 0.2429 & Accuracy= 0.987 || Validation Loss= 0.2788 & Accuracy= 0.984
Step 50, Training Loss= 0.1940 & Accuracy= 0.988 || Validation Loss= 0.2250 & Accuracy= 0.984
Step 60, Training Loss= 0.1579 & Accuracy= 0.987 || Validation Loss= 0.1858 & Accuracy= 0.988
Step 70, Training Loss= 0.1308 & Accuracy= 0.989 || Validation Loss= 0.1596 & Accuracy= 0.984
Step 80, Training Loss= 0.1105 & Accuracy= 0.988 || Validation Loss= 0.1391 & Accuracy= 0.988
Step 90, Training Loss= 0.0954 & Accuracy= 0.987 || Validation Loss= 0.1227 & Accuracy= 0.988
Step 100, Training Loss= 0.0841 & Accuracy= 0.986 || Validation Loss= 0.1103 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```

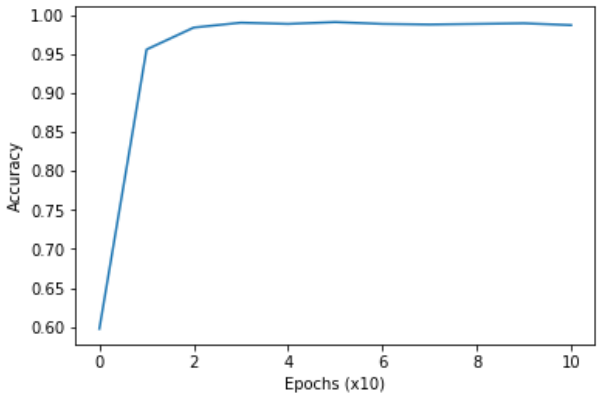




Increased no. of neurons in both layers to 100

```
Step 1, Training Loss= 110.7521 & Accuracy= 0.598 || Validation Loss= 106.1545 & Accuracy= 0.584
Step 10, Training Loss= 38.7742 & Accuracy= 0.955 || Validation Loss= 38.9516 & Accuracy= 0.966
Step 20, Training Loss= 28.2481 & Accuracy= 0.984 || Validation Loss= 30.0321 & Accuracy= 0.984
Step 30, Training Loss= 21.4308 & Accuracy= 0.990 || Validation Loss= 22.1530 & Accuracy= 0.984
Step 40, Training Loss= 16.4856 & Accuracy= 0.988 || Validation Loss= 16.9937 & Accuracy= 0.981
Step 50, Training Loss= 12.8281 & Accuracy= 0.991 || Validation Loss= 13.3919 & Accuracy= 0.988
Step 60, Training Loss= 10.1819 & Accuracy= 0.988 || Validation Loss= 10.7336 & Accuracy= 0.988
Step 70, Training Loss= 8.2186 & Accuracy= 0.988 || Validation Loss= 8.7100 & Accuracy= 0.988
Step 80, Training Loss= 6.7335 & Accuracy= 0.988 || Validation Loss= 7.1435 & Accuracy= 0.988
Step 90, Training Loss= 5.5867 & Accuracy= 0.989 || Validation Loss= 5.9211 & Accuracy= 0.988
Step 100, Training Loss= 4.6951 & Accuracy= 0.987 || Validation Loss= 4.9903 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.980000019073
```

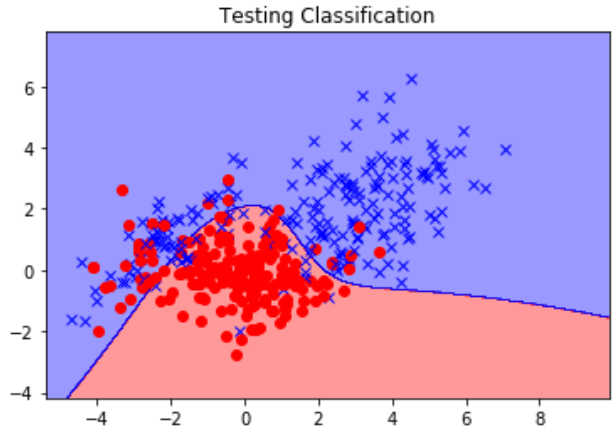


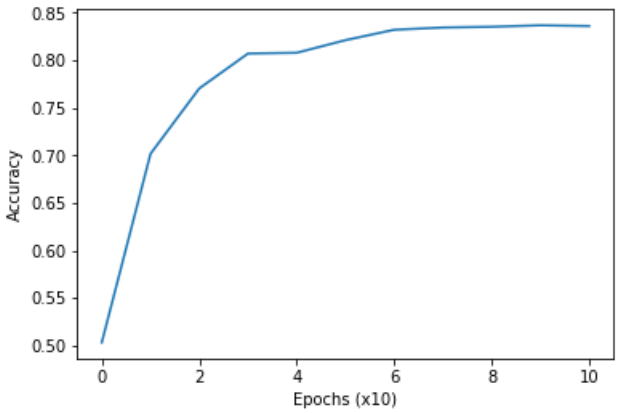


2.

- **Using Dataset 1**
- **Fully Connected with Single Hidden Layer**
    - **Neurons in Layer 1 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

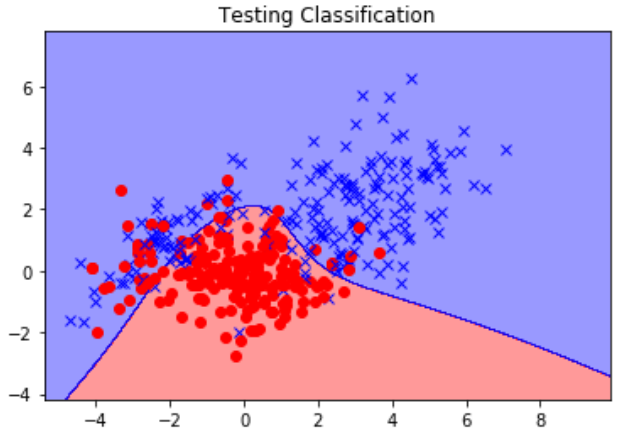
```
Step 1, Training Loss= 0.8822 & Accuracy= 0.503 || Validation Loss= 0.8781 & Accuracy= 0.469
Step 10, Training Loss= 0.5833 & Accuracy= 0.702 || Validation Loss= 0.5976 & Accuracy= 0.681
Step 20, Training Loss= 0.5054 & Accuracy= 0.770 || Validation Loss= 0.4906 & Accuracy= 0.791
Step 30, Training Loss= 0.4605 & Accuracy= 0.807 || Validation Loss= 0.4324 & Accuracy= 0.828
Step 40, Training Loss= 0.4350 & Accuracy= 0.808 || Validation Loss= 0.4179 & Accuracy= 0.819
Step 50, Training Loss= 0.4167 & Accuracy= 0.821 || Validation Loss= 0.3909 & Accuracy= 0.853
Step 60, Training Loss= 0.4053 & Accuracy= 0.832 || Validation Loss= 0.3874 & Accuracy= 0.850
Step 70, Training Loss= 0.3954 & Accuracy= 0.834 || Validation Loss= 0.3768 & Accuracy= 0.853
Step 80, Training Loss= 0.3883 & Accuracy= 0.835 || Validation Loss= 0.3779 & Accuracy= 0.856
Step 90, Training Loss= 0.3850 & Accuracy= 0.837 || Validation Loss= 0.3757 & Accuracy= 0.856
Step 100, Training Loss= 0.3831 & Accuracy= 0.836 || Validation Loss= 0.3741 & Accuracy= 0.856
Optimization Finished!
Testing Accuracy : 0.847500026226
```

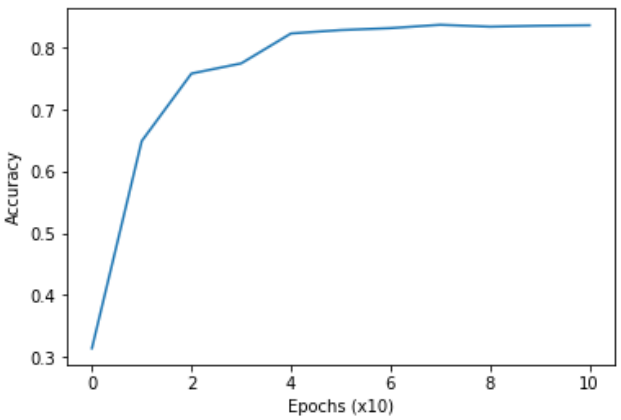




Increased no. of neurons in both layers to 10

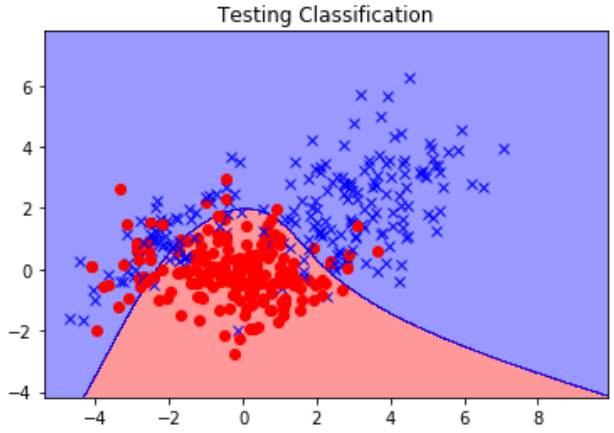
```
Step 1, Training Loss= 1.8776 & Accuracy= 0.314 || Validation Loss= 1.8092 & Accuracy= 0.328
Step 10, Training Loss= 0.6591 & Accuracy= 0.649 || Validation Loss= 0.6808 & Accuracy= 0.628
Step 20, Training Loss= 0.5576 & Accuracy= 0.759 || Validation Loss= 0.5612 & Accuracy= 0.766
Step 30, Training Loss= 0.4931 & Accuracy= 0.775 || Validation Loss= 0.4984 & Accuracy= 0.794
Step 40, Training Loss= 0.4460 & Accuracy= 0.823 || Validation Loss= 0.4319 & Accuracy= 0.834
Step 50, Training Loss= 0.4213 & Accuracy= 0.829 || Validation Loss= 0.3997 & Accuracy= 0.844
Step 60, Training Loss= 0.4066 & Accuracy= 0.832 || Validation Loss= 0.3849 & Accuracy= 0.856
Step 70, Training Loss= 0.3967 & Accuracy= 0.837 || Validation Loss= 0.3804 & Accuracy= 0.863
Step 80, Training Loss= 0.3909 & Accuracy= 0.834 || Validation Loss= 0.3794 & Accuracy= 0.859
Step 90, Training Loss= 0.3875 & Accuracy= 0.836 || Validation Loss= 0.3760 & Accuracy= 0.866
Step 100, Training Loss= 0.3851 & Accuracy= 0.837 || Validation Loss= 0.3728 & Accuracy= 0.863
Optimization Finished!
Testing Accuracy : 0.84500002861
```

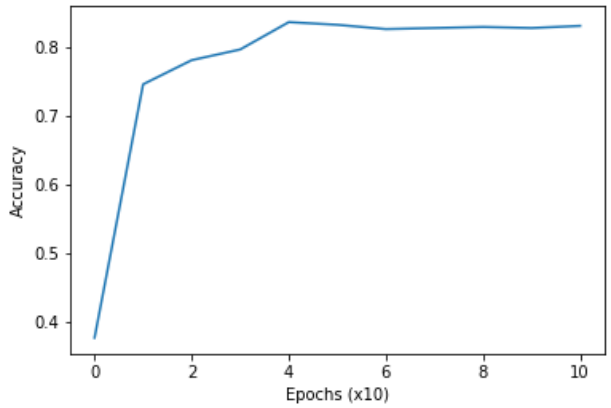




Increased no. of neurons in both layers to 100

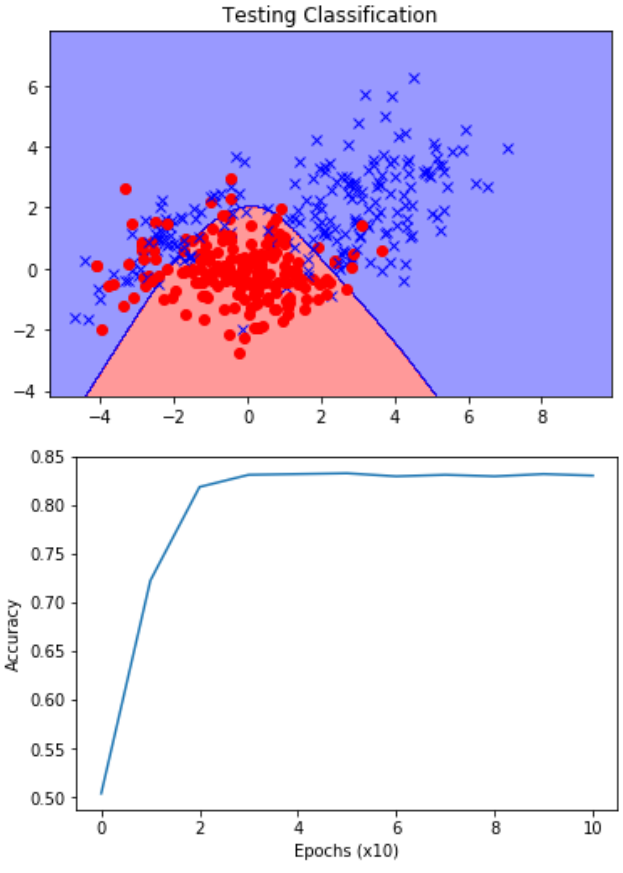
```
Step 1, Training Loss= 12.0445 & Accuracy= 0.377 || Validation Loss= 11.8503 & Accuracy= 0.425
Step 10, Training Loss= 1.4269 & Accuracy= 0.746 || Validation Loss= 1.3614 & Accuracy= 0.750
Step 20, Training Loss= 0.7979 & Accuracy= 0.781 || Validation Loss= 0.7385 & Accuracy= 0.809
Step 30, Training Loss= 0.6550 & Accuracy= 0.797 || Validation Loss= 0.5861 & Accuracy= 0.831
Step 40, Training Loss= 0.5447 & Accuracy= 0.837 || Validation Loss= 0.4999 & Accuracy= 0.844
Step 50, Training Loss= 0.4705 & Accuracy= 0.833 || Validation Loss= 0.4438 & Accuracy= 0.847
Step 60, Training Loss= 0.4374 & Accuracy= 0.827 || Validation Loss= 0.4169 & Accuracy= 0.853
Step 70, Training Loss= 0.4202 & Accuracy= 0.828 || Validation Loss= 0.3979 & Accuracy= 0.856
Step 80, Training Loss= 0.4115 & Accuracy= 0.830 || Validation Loss= 0.3887 & Accuracy= 0.853
Step 90, Training Loss= 0.4056 & Accuracy= 0.828 || Validation Loss= 0.3847 & Accuracy= 0.856
Step 100, Training Loss= 0.4012 & Accuracy= 0.831 || Validation Loss= 0.3821 & Accuracy= 0.856
Optimization Finished!
Testing Accuracy : 0.850000023842
```





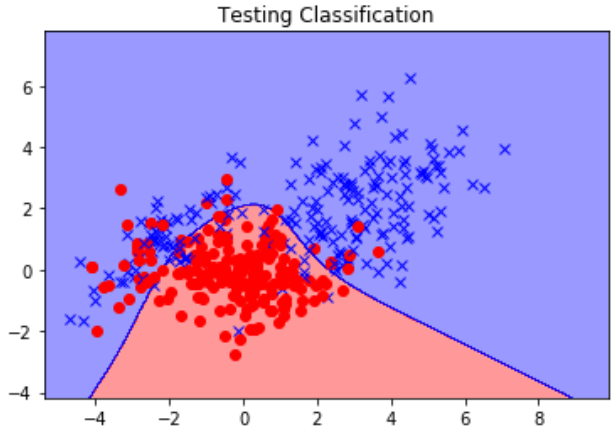
- **Using Dataset 1**
- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 500**

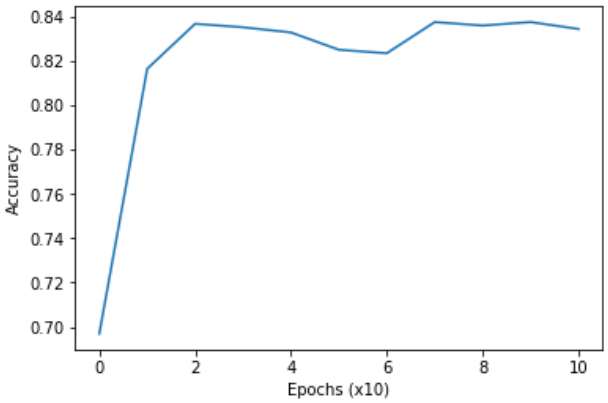
```
Step 1, Training Loss= 1.6411 & Accuracy= 0.504 || Validation Loss= 1.7097 & Accuracy= 0.478
Step 10, Training Loss= 0.6815 & Accuracy= 0.723 || Validation Loss= 0.6712 & Accuracy= 0.716
Step 20, Training Loss= 0.5554 & Accuracy= 0.819 || Validation Loss= 0.5259 & Accuracy= 0.834
Step 30, Training Loss= 0.4839 & Accuracy= 0.831 || Validation Loss= 0.4548 & Accuracy= 0.856
Step 40, Training Loss= 0.4331 & Accuracy= 0.832 || Validation Loss= 0.4123 & Accuracy= 0.856
Step 50, Training Loss= 0.4100 & Accuracy= 0.833 || Validation Loss= 0.3937 & Accuracy= 0.856
Step 60, Training Loss= 0.4019 & Accuracy= 0.830 || Validation Loss= 0.3828 & Accuracy= 0.856
Step 70, Training Loss= 0.3967 & Accuracy= 0.831 || Validation Loss= 0.3781 & Accuracy= 0.853
Step 80, Training Loss= 0.3935 & Accuracy= 0.830 || Validation Loss= 0.3746 & Accuracy= 0.856
Step 90, Training Loss= 0.3914 & Accuracy= 0.832 || Validation Loss= 0.3740 & Accuracy= 0.856
Step 100, Training Loss= 0.3900 & Accuracy= 0.830 || Validation Loss= 0.3764 & Accuracy= 0.859
Optimization Finished!
Testing Accuracy : 0.84249997139
```



Increased no. of neurons in both layers to 10

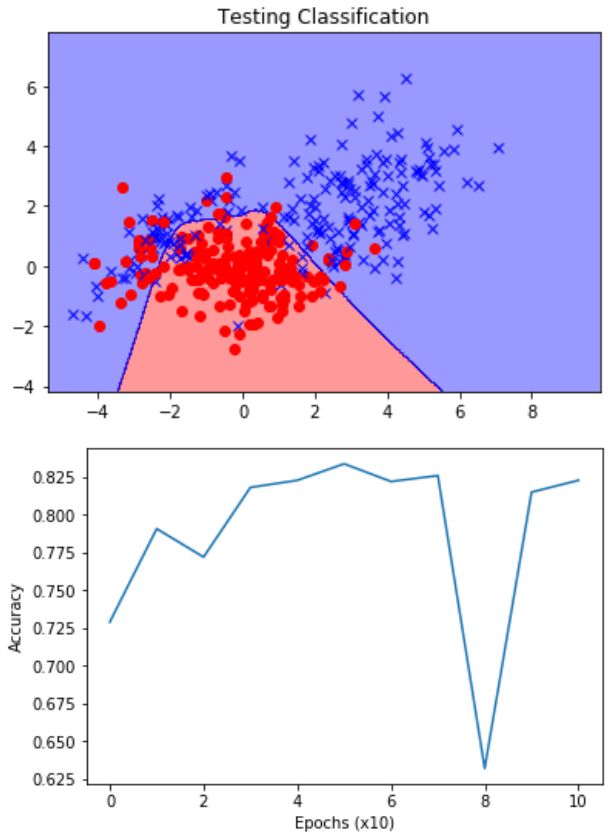
```
Step 1, Training Loss= 1.3575 & Accuracy= 0.697 || Validation Loss= 1.3259 & Accuracy= 0.722
Step 10, Training Loss= 0.7237 & Accuracy= 0.816 || Validation Loss= 0.7150 & Accuracy= 0.803
Step 20, Training Loss= 0.4825 & Accuracy= 0.837 || Validation Loss= 0.4677 & Accuracy= 0.856
Step 30, Training Loss= 0.4242 & Accuracy= 0.835 || Validation Loss= 0.4022 & Accuracy= 0.853
Step 40, Training Loss= 0.4031 & Accuracy= 0.833 || Validation Loss= 0.3876 & Accuracy= 0.866
Step 50, Training Loss= 0.4040 & Accuracy= 0.825 || Validation Loss= 0.3955 & Accuracy= 0.863
Step 60, Training Loss= 0.4069 & Accuracy= 0.823 || Validation Loss= 0.4058 & Accuracy= 0.850
Step 70, Training Loss= 0.3927 & Accuracy= 0.837 || Validation Loss= 0.3794 & Accuracy= 0.863
Step 80, Training Loss= 0.3897 & Accuracy= 0.836 || Validation Loss= 0.3786 & Accuracy= 0.872
Step 90, Training Loss= 0.3876 & Accuracy= 0.837 || Validation Loss= 0.3752 & Accuracy= 0.869
Step 100, Training Loss= 0.3862 & Accuracy= 0.834 || Validation Loss= 0.3712 & Accuracy= 0.866
Optimization Finished!
Testing Accuracy : 0.84249997139
```





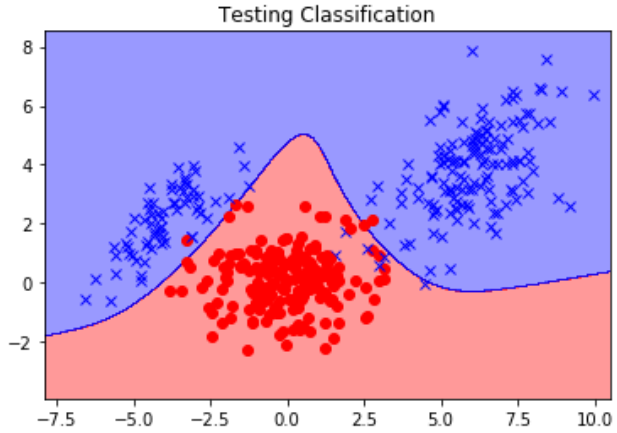
Increased no. of neurons in both layers to 100

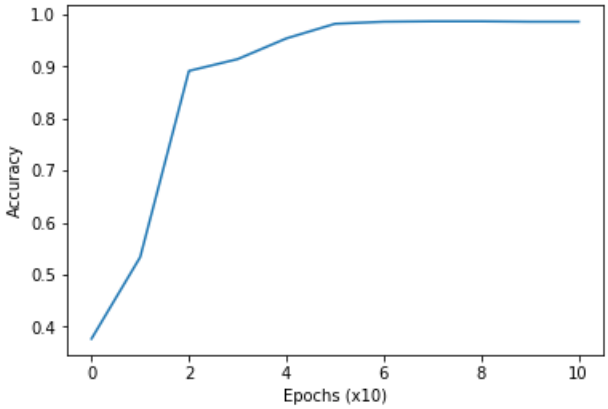
```
Step 1, Training Loss= 46.9722 & Accuracy= 0.729 || Validation Loss= 46.7823 & Accuracy= 0.744
Step 10, Training Loss= 12.8303 & Accuracy= 0.791 || Validation Loss= 12.8450 & Accuracy= 0.784
Step 20, Training Loss= 3.7687 & Accuracy= 0.772 || Validation Loss= 3.7931 & Accuracy= 0.762
Step 30, Training Loss= 1.7437 & Accuracy= 0.818 || Validation Loss= 1.7108 & Accuracy= 0.844
Step 40, Training Loss= 1.2194 & Accuracy= 0.823 || Validation Loss= 1.1827 & Accuracy= 0.834
Step 50, Training Loss= 0.7931 & Accuracy= 0.834 || Validation Loss= 0.7803 & Accuracy= 0.847
Step 60, Training Loss= 0.7207 & Accuracy= 0.822 || Validation Loss= 0.6847 & Accuracy= 0.847
Step 70, Training Loss= 0.7331 & Accuracy= 0.826 || Validation Loss= 0.7258 & Accuracy= 0.856
Step 80, Training Loss= 1.1561 & Accuracy= 0.632 || Validation Loss= 1.1518 & Accuracy= 0.650
Step 90, Training Loss= 1.0484 & Accuracy= 0.815 || Validation Loss= 1.0530 & Accuracy= 0.809
Step 100, Training Loss= 0.7474 & Accuracy= 0.823 || Validation Loss= 0.7129 & Accuracy= 0.850
Optimization Finished!
Testing Accuracy : 0.834999978542
```



- **Using Dataset 2**
- **Fully Connected with Single Hidden Layer**
    - **Neurons in Layer 1 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

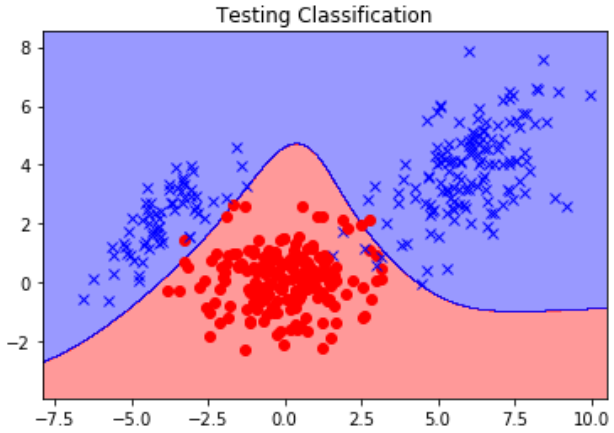
```
Step 1, Training Loss= 0.9939 & Accuracy= 0.377 || Validation Loss= 0.9649 & Accuracy= 0.384
Step 10, Training Loss= 0.6773 & Accuracy= 0.534 || Validation Loss= 0.6882 & Accuracy= 0.506
Step 20, Training Loss= 0.3362 & Accuracy= 0.891 || Validation Loss= 0.3323 & Accuracy= 0.903
Step 30, Training Loss= 0.2467 & Accuracy= 0.914 || Validation Loss= 0.2432 & Accuracy= 0.913
Step 40, Training Loss= 0.1969 & Accuracy= 0.954 || Validation Loss= 0.1858 & Accuracy= 0.962
Step 50, Training Loss= 0.1228 & Accuracy= 0.982 || Validation Loss= 0.1186 & Accuracy= 0.984
Step 60, Training Loss= 0.0867 & Accuracy= 0.986 || Validation Loss= 0.0873 & Accuracy= 0.988
Step 70, Training Loss= 0.0738 & Accuracy= 0.987 || Validation Loss= 0.0783 & Accuracy= 0.988
Step 80, Training Loss= 0.0671 & Accuracy= 0.987 || Validation Loss= 0.0757 & Accuracy= 0.988
Step 90, Training Loss= 0.0624 & Accuracy= 0.986 || Validation Loss= 0.0722 & Accuracy= 0.988
Step 100, Training Loss= 0.0591 & Accuracy= 0.986 || Validation Loss= 0.0698 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.980000019073
```

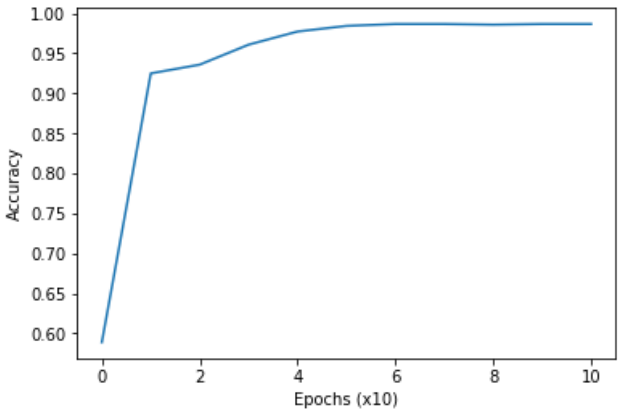




Increased no. of neurons in both layers to 10

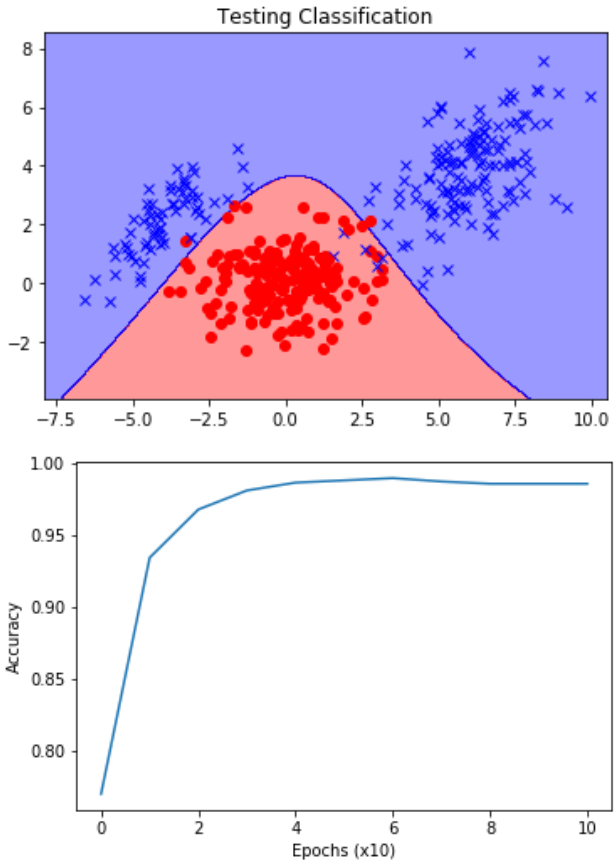
```
Step 1, Training Loss= 1.4809 & Accuracy= 0.589 || Validation Loss= 1.4314 & Accuracy= 0.572
Step 10, Training Loss= 0.3883 & Accuracy= 0.925 || Validation Loss= 0.3702 & Accuracy= 0.931
Step 20, Training Loss= 0.2091 & Accuracy= 0.936 || Validation Loss= 0.2010 & Accuracy= 0.931
Step 30, Training Loss= 0.1558 & Accuracy= 0.961 || Validation Loss= 0.1472 & Accuracy= 0.972
Step 40, Training Loss= 0.1117 & Accuracy= 0.977 || Validation Loss= 0.1131 & Accuracy= 0.978
Step 50, Training Loss= 0.0876 & Accuracy= 0.984 || Validation Loss= 0.0936 & Accuracy= 0.984
Step 60, Training Loss= 0.0728 & Accuracy= 0.987 || Validation Loss= 0.0807 & Accuracy= 0.988
Step 70, Training Loss= 0.0663 & Accuracy= 0.987 || Validation Loss= 0.0746 & Accuracy= 0.988
Step 80, Training Loss= 0.0620 & Accuracy= 0.986 || Validation Loss= 0.0722 & Accuracy= 0.988
Step 90, Training Loss= 0.0587 & Accuracy= 0.987 || Validation Loss= 0.0703 & Accuracy= 0.988
Step 100, Training Loss= 0.0563 & Accuracy= 0.987 || Validation Loss= 0.0688 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.980000019073
```





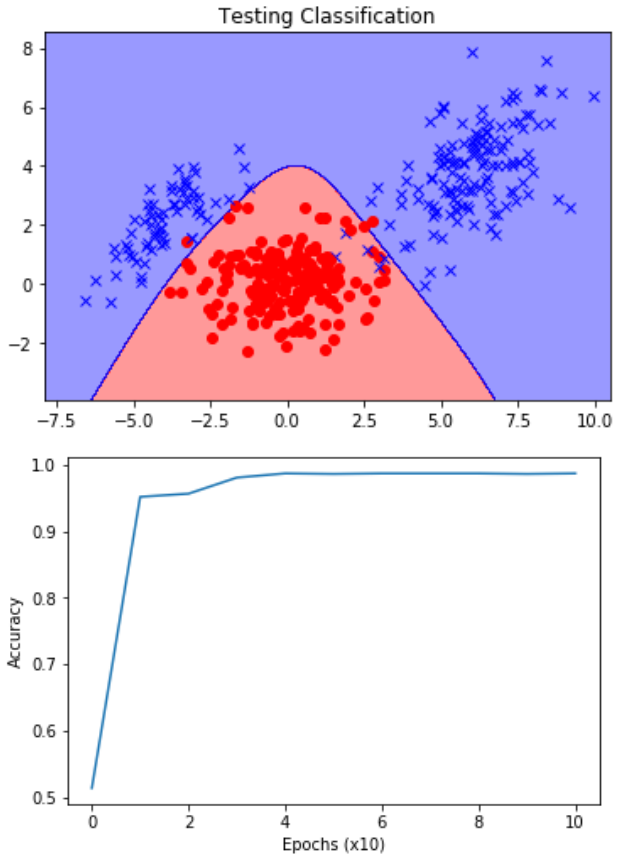
Increased no. of neurons in both layers to 100

```
Step 1, Training Loss= 11.2095 & Accuracy= 0.770 || Validation Loss= 10.4469 & Accuracy= 0.784
Step 10, Training Loss= 0.5703 & Accuracy= 0.934 || Validation Loss= 0.5611 & Accuracy= 0.941
Step 20, Training Loss= 0.3915 & Accuracy= 0.968 || Validation Loss= 0.4175 & Accuracy= 0.975
Step 30, Training Loss= 0.3105 & Accuracy= 0.981 || Validation Loss= 0.3362 & Accuracy= 0.984
Step 40, Training Loss= 0.2252 & Accuracy= 0.987 || Validation Loss= 0.2442 & Accuracy= 0.984
Step 50, Training Loss= 0.1564 & Accuracy= 0.988 || Validation Loss= 0.1755 & Accuracy= 0.984
Step 60, Training Loss= 0.1143 & Accuracy= 0.990 || Validation Loss= 0.1313 & Accuracy= 0.984
Step 70, Training Loss= 0.0887 & Accuracy= 0.988 || Validation Loss= 0.1040 & Accuracy= 0.988
Step 80, Training Loss= 0.0747 & Accuracy= 0.986 || Validation Loss= 0.0894 & Accuracy= 0.988
Step 90, Training Loss= 0.0669 & Accuracy= 0.986 || Validation Loss= 0.0812 & Accuracy= 0.988
Step 100, Training Loss= 0.0618 & Accuracy= 0.986 || Validation Loss= 0.0763 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.980000019073
```



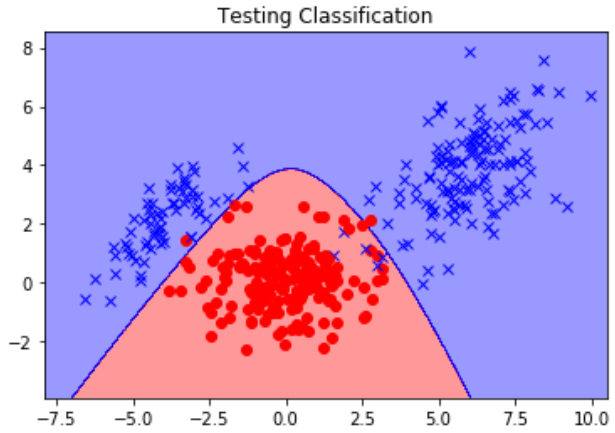
- **Using Dataset 2**
- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

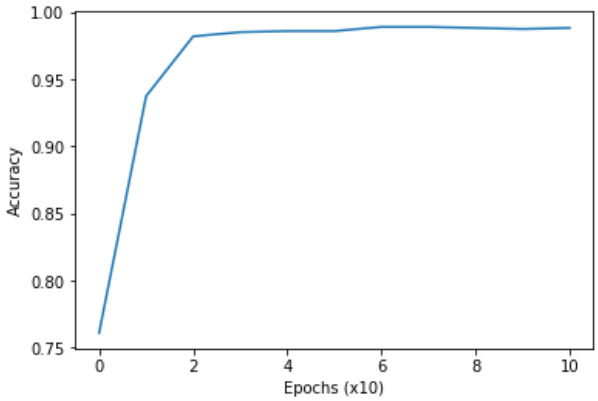
```
Step 1, Training Loss= 1.1622 & Accuracy= 0.514 || Validation Loss= 1.1789 & Accuracy= 0.484
Step 10, Training Loss= 0.3409 & Accuracy= 0.952 || Validation Loss= 0.3291 & Accuracy= 0.947
Step 20, Training Loss= 0.2385 & Accuracy= 0.956 || Validation Loss= 0.2379 & Accuracy= 0.953
Step 30, Training Loss= 0.1491 & Accuracy= 0.980 || Validation Loss= 0.1492 & Accuracy= 0.984
Step 40, Training Loss= 0.0953 & Accuracy= 0.987 || Validation Loss= 0.1050 & Accuracy= 0.988
Step 50, Training Loss= 0.0702 & Accuracy= 0.986 || Validation Loss= 0.0820 & Accuracy= 0.988
Step 60, Training Loss= 0.0599 & Accuracy= 0.987 || Validation Loss= 0.0760 & Accuracy= 0.988
Step 70, Training Loss= 0.0540 & Accuracy= 0.987 || Validation Loss= 0.0742 & Accuracy= 0.988
Step 80, Training Loss= 0.0512 & Accuracy= 0.987 || Validation Loss= 0.0722 & Accuracy= 0.988
Step 90, Training Loss= 0.0492 & Accuracy= 0.986 || Validation Loss= 0.0714 & Accuracy= 0.988
Step 100, Training Loss= 0.0477 & Accuracy= 0.987 || Validation Loss= 0.0706 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```



Increased no. of neurons in both layers to 10

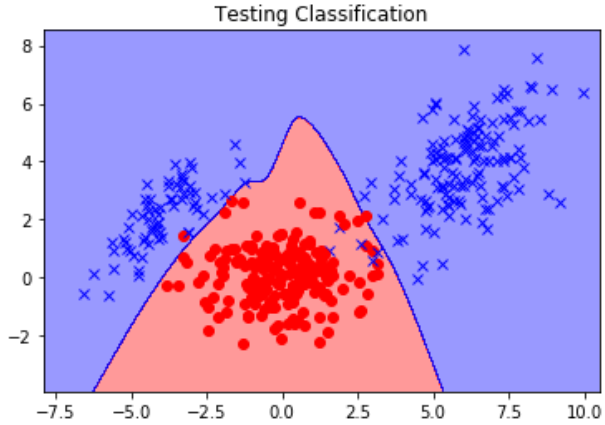
```
Step 1, Training Loss= 1.3976 & Accuracy= 0.761 || Validation Loss= 1.4180 & Accuracy= 0.762
Step 10, Training Loss= 0.4712 & Accuracy= 0.938 || Validation Loss= 0.4613 & Accuracy= 0.944
Step 20, Training Loss= 0.1846 & Accuracy= 0.982 || Validation Loss= 0.1902 & Accuracy= 0.984
Step 30, Training Loss= 0.1023 & Accuracy= 0.985 || Validation Loss= 0.1105 & Accuracy= 0.984
Step 40, Training Loss= 0.0773 & Accuracy= 0.986 || Validation Loss= 0.0934 & Accuracy= 0.988
Step 50, Training Loss= 0.0636 & Accuracy= 0.986 || Validation Loss= 0.0824 & Accuracy= 0.988
Step 60, Training Loss= 0.0560 & Accuracy= 0.989 || Validation Loss= 0.0764 & Accuracy= 0.988
Step 70, Training Loss= 0.0527 & Accuracy= 0.989 || Validation Loss= 0.0738 & Accuracy= 0.988
Step 80, Training Loss= 0.0504 & Accuracy= 0.988 || Validation Loss= 0.0727 & Accuracy= 0.988
Step 90, Training Loss= 0.0485 & Accuracy= 0.988 || Validation Loss= 0.0725 & Accuracy= 0.988
Step 100, Training Loss= 0.0468 & Accuracy= 0.988 || Validation Loss= 0.0721 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```

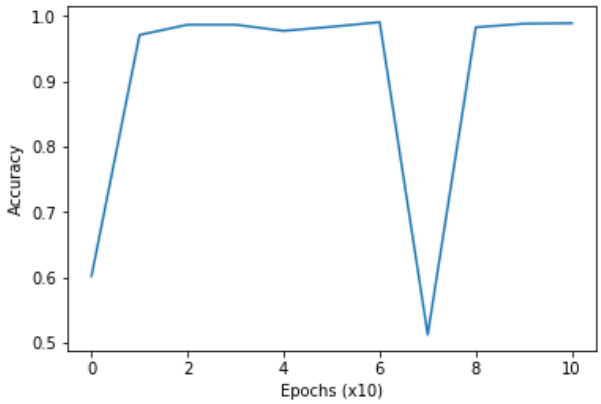




Increased no. of neurons in both layers to 100

```
Step 1, Training Loss= 52.8564 & Accuracy= 0.602 || Validation Loss= 52.1224 & Accuracy= 0.650
Step 10, Training Loss= 15.6936 & Accuracy= 0.971 || Validation Loss= 15.6815 & Accuracy= 0.984
Step 20, Training Loss= 4.2397 & Accuracy= 0.987 || Validation Loss= 4.2752 & Accuracy= 0.981
Step 30, Training Loss= 1.8780 & Accuracy= 0.987 || Validation Loss= 1.8942 & Accuracy= 0.978
Step 40, Training Loss= 0.9176 & Accuracy= 0.977 || Validation Loss= 0.9704 & Accuracy= 0.984
Step 50, Training Loss= 1.3429 & Accuracy= 0.984 || Validation Loss= 1.4155 & Accuracy= 0.975
Step 60, Training Loss= 0.7317 & Accuracy= 0.991 || Validation Loss= 0.7813 & Accuracy= 0.984
Step 70, Training Loss= 4.6183 & Accuracy= 0.512 || Validation Loss= 4.7477 & Accuracy= 0.466
Step 80, Training Loss= 3.5350 & Accuracy= 0.983 || Validation Loss= 3.5391 & Accuracy= 0.988
Step 90, Training Loss= 2.5123 & Accuracy= 0.988 || Validation Loss= 2.5551 & Accuracy= 0.988
Step 100, Training Loss= 1.0097 & Accuracy= 0.989 || Validation Loss= 1.0381 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```





COMMENTS ON 1 & 2

- When using Dataset 2 I observed better accuracy (98 %) in general as compared to dataset 1 (85%). This was due to the fact that classes are more separated.
- I observed no. of neurons does not really affect the network performance. (Unless it is too low, like 1 or 2, in which I got accuracy all over the place).
- When using 2 layers, initial losses were greater as compared to single layer. But network quickly learnt and reduced losses.
- Sigmoid maps input between 0 & 1 while tanh maps input between -1 & 1. Sigmoid can cause the network to get stuck during training, when big negative numbers are provided sigmoid outputs nearly 0. While tanh avoids this problem as it maps big negative numbers close to -1.
  I observed this when training on dataset 2 with 2 hidden layers and 128 neurons. Network got stuck on 65% accuracy and could not adapt.

3. All the models shown above were initialized using uniform weights.

```
weights = {
    'h1': tf.Variable(tf.random_uniform([num_input, n_hidden_1])),
    'h2': tf.Variable(tf.random_uniform([n_hidden_1, n_hidden_2])),
    'out': tf.Variable(tf.random_uniform([n_hidden_2, num_classes]))
}
biases = {
    'b1': tf.Variable(tf.random_uniform([n_hidden_1])),
    'b2': tf.Variable(tf.random_uniform([n_hidden_2])),
    'out': tf.Variable(tf.random_uniform([num_classes]))
}
```

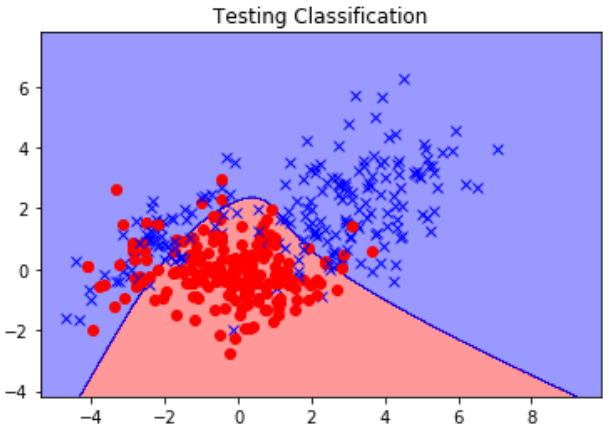
Initializing Gaussian weights.

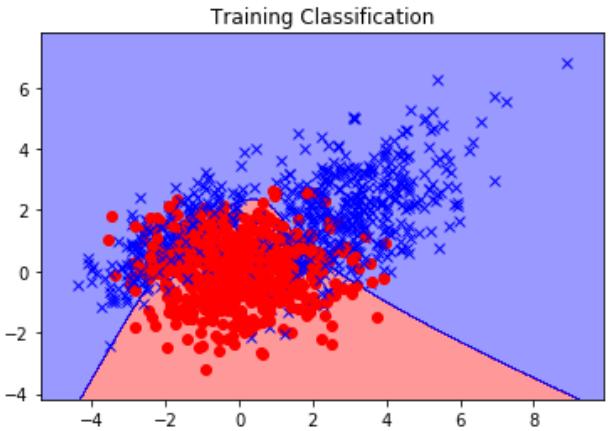
```
weights = {
    'h1': tf.Variable(tf.random_normal([num_input, n_hidden_1])),
    'h2': tf.Variable(tf.random_normal([n_hidden_1, n_hidden_2])),
    'out': tf.Variable(tf.random_normal([n_hidden_2, num_classes]))
}
biases = {
    'b1': tf.Variable(tf.random_normal([n_hidden_1])),
    'b2': tf.Variable(tf.random_normal([n_hidden_2])),
    'out': tf.Variable(tf.random_normal([num_classes]))
}
```

Using Dataset 1

- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

```
Step 1, Training Loss= 82.0656 & Accuracy= 0.688 || Validation Loss= 82.2811 & Accuracy= 0.672
Step 50, Training Loss= 1.3689 & Accuracy= 0.820 || Validation Loss= 1.3342 & Accuracy= 0.822
Step 100, Training Loss= 0.8226 & Accuracy= 0.821 || Validation Loss= 0.8059 & Accuracy= 0.847
Step 150, Training Loss= 0.8505 & Accuracy= 0.784 || Validation Loss= 0.8639 & Accuracy= 0.772
Step 200, Training Loss= 0.4446 & Accuracy= 0.827 || Validation Loss= 0.4244 & Accuracy= 0.847
Step 250, Training Loss= 0.4544 & Accuracy= 0.828 || Validation Loss= 0.4154 & Accuracy= 0.841
Step 300, Training Loss= 0.4647 & Accuracy= 0.818 || Validation Loss= 0.4118 & Accuracy= 0.828
Step 350, Training Loss= 0.4106 & Accuracy= 0.826 || Validation Loss= 0.3991 & Accuracy= 0.850
Step 400, Training Loss= 0.5130 & Accuracy= 0.780 || Validation Loss= 0.5329 & Accuracy= 0.762
Step 450, Training Loss= 0.4148 & Accuracy= 0.820 || Validation Loss= 0.4107 & Accuracy= 0.841
Step 500, Training Loss= 0.4000 & Accuracy= 0.834 || Validation Loss= 0.3838 & Accuracy= 0.859
Optimization Finished!
Testing Accuracy : 0.847500026226
```





Comparison

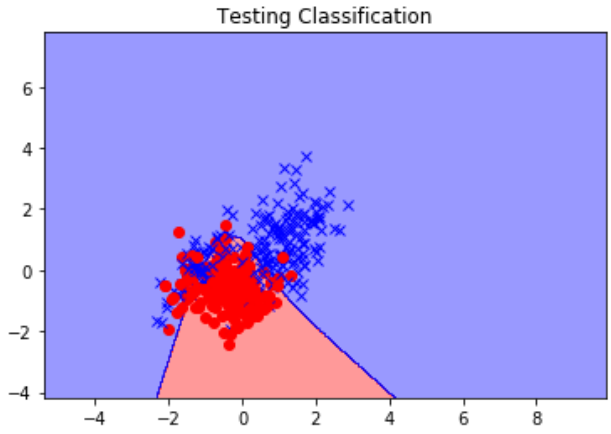
	Testing Accuracy	Training Accuracy	Validation Accuracy
With Uniform Wts	0.8424	0.821	0847
With Gaussian Wts	0.8475	0.834	0.859

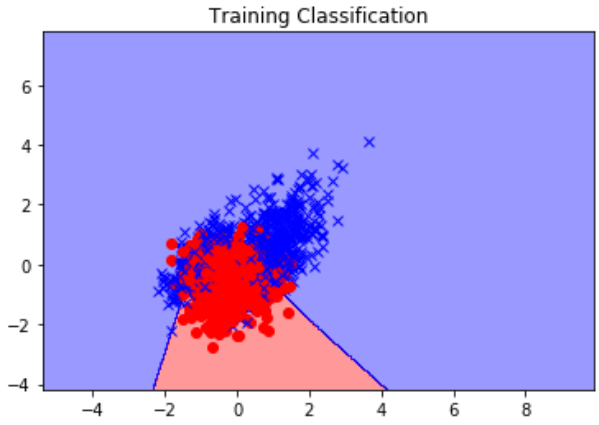
I observed initial weights do not have any impact on the results, as these weights are “initial” they soon get overwritten by new weights as the network adopts epoch after epoch.

Scaled Data to Zero Mean & Unit Variance

- **Fully Connected with 2 Hidden Layers**
    - **Neurons in Layer 1 = 5**
    - **Neurons in Layer 2 = 5**
- **Tanh Non-linearities**
- **Sigmoid at output**
- **L2 Weight Regularization**
- **Epochs = 100**

```
Step 1, Training Loss= 80.9128 & Accuracy= 0.759 || Validation Loss= 81.0546 & Accuracy= 0.762
Step 50, Training Loss= 1.6537 & Accuracy= 0.766 || Validation Loss= 1.6825 & Accuracy= 0.747
Step 100, Training Loss= 0.7059 & Accuracy= 0.834 || Validation Loss= 0.6893 & Accuracy= 0.866
Step 150, Training Loss= 0.7290 & Accuracy= 0.795 || Validation Loss= 0.7292 & Accuracy= 0.800
Step 200, Training Loss= 0.4264 & Accuracy= 0.833 || Validation Loss= 0.4039 & Accuracy= 0.863
Step 250, Training Loss= 0.4350 & Accuracy= 0.819 || Validation Loss= 0.4313 & Accuracy= 0.837
Step 300, Training Loss= 0.4431 & Accuracy= 0.823 || Validation Loss= 0.4301 & Accuracy= 0.847
Step 350, Training Loss= 0.4146 & Accuracy= 0.830 || Validation Loss= 0.3954 & Accuracy= 0.859
Step 400, Training Loss= 0.4796 & Accuracy= 0.806 || Validation Loss= 0.4303 & Accuracy= 0.806
Step 450, Training Loss= 0.4168 & Accuracy= 0.833 || Validation Loss= 0.3918 & Accuracy= 0.834
Step 500, Training Loss= 0.4279 & Accuracy= 0.822 || Validation Loss= 0.3974 & Accuracy= 0.834
Optimization Finished!
Testing Accuracy : 0.839999973774
```





Comparison

	Testing Accuracy	Training Accuracy	Validation Accuracy
Scaled	0.8399	0.822	0.834
Un-scaled	0.8424	0.821	0.847

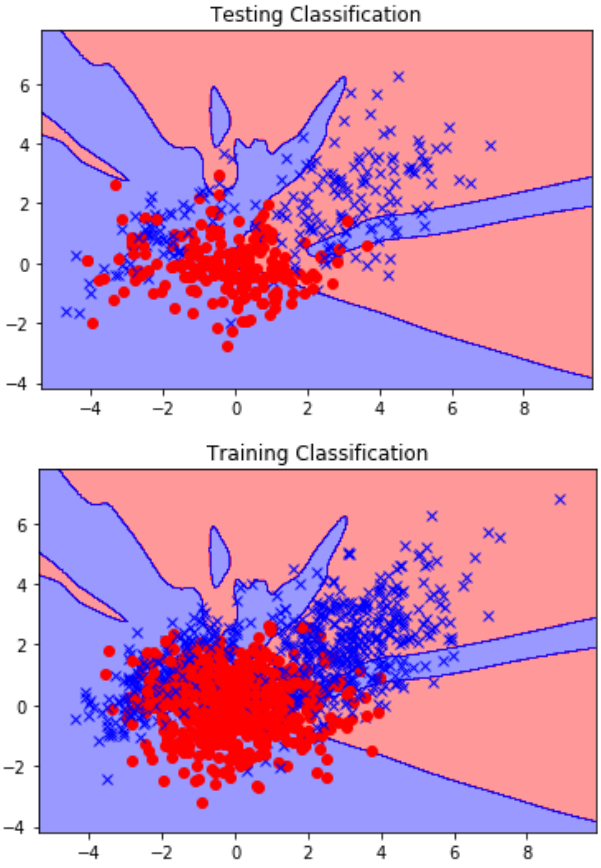
Scaling to zero mean and unit variance did not really impact the results in this dataset. The inputs are already Gaussian, which means most of the data points (68.26 %) are within 1 deviation and some points (27.18%) between 1 & 2 deviations.

But this could be very helpful preprocessing technique if the data points are far apart from each other.

In all the previous models, learning rate was 0.1

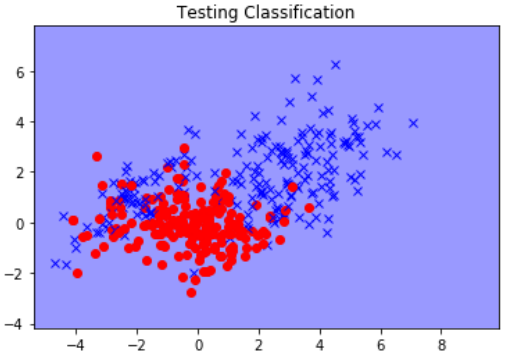
Learning rate = 0.000001

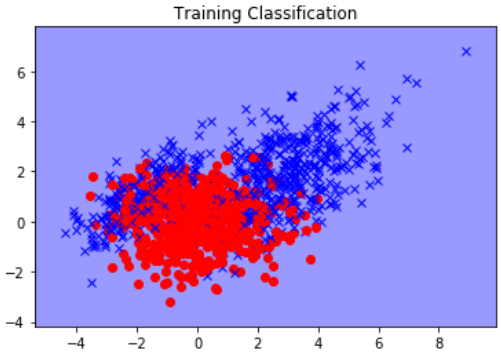
```
Step 1, Training Loss= 87.1085 & Accuracy= 0.355 || Validation Loss= 87.3992 & Accuracy= 0.350
Step 50, Training Loss= 87.0908 & Accuracy= 0.355 || Validation Loss= 87.3817 & Accuracy= 0.350
Step 100, Training Loss= 87.0728 & Accuracy= 0.355 || Validation Loss= 87.3638 & Accuracy= 0.350
Step 150, Training Loss= 87.0548 & Accuracy= 0.355 || Validation Loss= 87.3459 & Accuracy= 0.350
Step 200, Training Loss= 87.0368 & Accuracy= 0.355 || Validation Loss= 87.3281 & Accuracy= 0.350
Step 250, Training Loss= 87.0188 & Accuracy= 0.355 || Validation Loss= 87.3103 & Accuracy= 0.353
Step 300, Training Loss= 87.0009 & Accuracy= 0.356 || Validation Loss= 87.2925 & Accuracy= 0.353
Step 350, Training Loss= 86.9830 & Accuracy= 0.357 || Validation Loss= 87.2747 & Accuracy= 0.353
Step 400, Training Loss= 86.9650 & Accuracy= 0.357 || Validation Loss= 87.2570 & Accuracy= 0.353
Step 450, Training Loss= 86.9471 & Accuracy= 0.358 || Validation Loss= 87.2392 & Accuracy= 0.353
Step 500, Training Loss= 86.9293 & Accuracy= 0.357 || Validation Loss= 87.2215 & Accuracy= 0.356
Optimization Finished!
Testing Accuracy : 0.354999989271
```



Learning rate =10

```
Step 1, Training Loss= 7566.0967 & Accuracy= 0.444 || Validation Loss= 7564.1577 & Accuracy= 0.469
Step 50, Training Loss= 3156.6045 & Accuracy= 0.503 || Validation Loss= 3169.4666 & Accuracy= 0.469
Step 100, Training Loss= 10826.1934 & Accuracy= 0.503 || Validation Loss= 10842.0967 & Accuracy= 0.469
Step 150, Training Loss= 15184.4121 & Accuracy= 0.497 || Validation Loss= 15173.1836 & Accuracy= 0.531
Step 200, Training Loss= 14364.5186 & Accuracy= 0.503 || Validation Loss= 14388.4141 & Accuracy= 0.469
Step 250, Training Loss= 18575.1270 & Accuracy= 0.497 || Validation Loss= 18572.6348 & Accuracy= 0.531
Step 300, Training Loss= 7592.6836 & Accuracy= 0.503 || Validation Loss= 7607.0767 & Accuracy= 0.469
Step 350, Training Loss= 525.2292 & Accuracy= 0.503 || Validation Loss= 529.5817 & Accuracy= 0.469
Step 400, Training Loss= 251.7755 & Accuracy= 0.497 || Validation Loss= 251.2874 & Accuracy= 0.531
Step 450, Training Loss= 188.7163 & Accuracy= 0.497 || Validation Loss= 187.4189 & Accuracy= 0.531
Step 500, Training Loss= 109.1351 & Accuracy= 0.503 || Validation Loss= 116.6049 & Accuracy= 0.469
Optimization Finished!
Testing Accuracy : 0.514999985695
```





Comparison

Learning Rate	Testing Accuracy	Training Accuracy	Validation Accuracy
0.000001	0.3549	0.357	0.356
0.1	0.8424	0.821	0.847
10	0.5149	0.503	0.469

When learning rate is too small, the model may take many epochs to converge, sometimes it gets stuck and does not move towards local/global minimum, like in our case when rate was (0.000001) negligible.
When learning is too big, the model takes big steps towards increasing accuracy that it misses local/ global minimum. If rate is too big, like in our case (10), it may never be able to increase accuracy and gets stuck in a cycle of jumping around.

4.

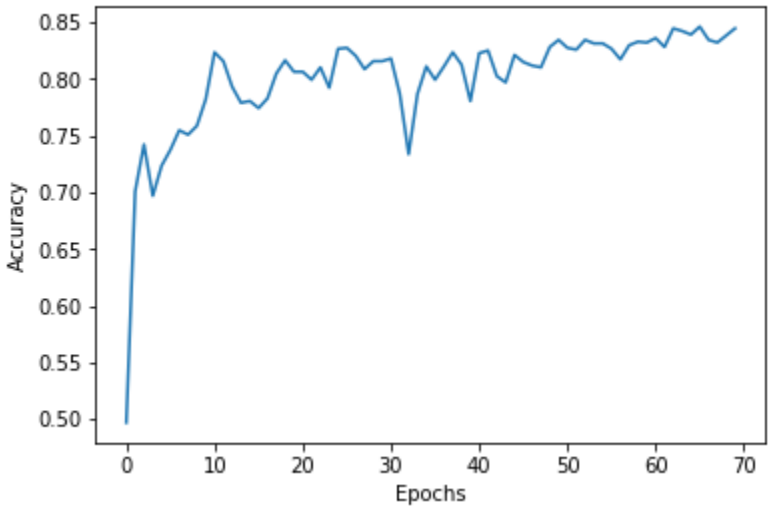
Training set = 1280
Testing set = 400
Validation set = 320

I used 2000 points in total of which 20 % was split into testing set and from the remaining another 20% was split in validation set. Remaining was used as training.

400 & 320 seemed to be good number of data points to test and validate on as it would not impair accuracy. If number of data points in testing and validation set is small, there's a good chance to get very high accuracy. If number is high, there would be less data points to train on.

I observed very low accuracy initially but it jumped to 70 % in first few epochs. It seemed to reach 80 % at about 60 epoch, and then model started to made minor improvement towards 85 %. The same was reflected on validation set.

After training, all the accuracies matched.



5. Comparision with MAP, kernelized logistic regression, and logistic regression with explicit feature engineering

Out of all the models, neural network has performed better with about 85% accuracy, without overfitting and with better classification regions.

MAP = 79%

Kernelized Logistic Regression = 83%
It was observed that as we decreased *'l'*, accuracy increased but that was due to overfitting.

Non-Kernalized Logistic Regression = 83%.

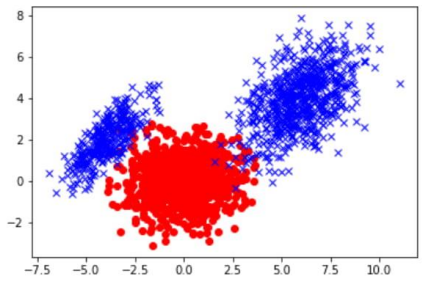
In [13]:
```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn import preprocessing
```

In [14]:
```
# Cov matrix
def cov(lam1, lam2, theta):
    d = np.matrix([[lam1, 0], [0, lam2]])
    p = np.matrix([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])
    invp = np.linalg.inv(p)
    return np.linalg.multi_dot((p,d,invp))

# Generating Data
def gendata(data_pts):

    data_pts = data_pts/2

    # Class 0
    cov0 = cov(2, 1, 0)
    mean0 = [0,0]

    data0 = np.random.multivariate_normal(mean0, cov0, data_pts)

    # Class 1

    cov1a = cov(2, 0.25, -3*np.pi/4)
    mean1a = [-2,1]
    cov1b = cov(3, 1, np.pi/4)
    mean1b = [3,2]

    data1a = np.random.multivariate_normal(mean1a, cov1a, (data_pts/3))
    data1b = np.random.multivariate_normal(mean1b, cov1b, (data_pts - data_pts/3))
    data1 = np.concatenate((data1a, data1b), axis=0)

    data = np.concatenate((data0,data1), axis = 0)
    labels = np.concatenate((np.zeros(data_pts), np.ones(data_pts)))

    return data,labels

def plotdata(x,y):
    for i in range(y.shape[0]):
        if(y[i] == 0):
            plt.plot(x[i,0], x[i,1], 'o', color = 'red')
        else:
            plt.plot(x[i,0], x[i,1], 'x', color = 'blue')
```

In [15]:
```
data_size = 2000
# data, labels = gendata(data_size)
# np.save('data_2', data)
# np.save('labels_2',Labels)
```

In [132]:
```
data = np.load('data_2.npy')
labels = np.load('labels_2.npy')

# data_scaled = preprocessing.scale(data)

x_train, x_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=15)
x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2, random_state=25)

# x_train_scaled, x_test_scaled, y_train, y_test = train_test_split(data_scaled, labels, test_size=0.2, random
_state=15)
# x_train_scaled, x_valid_scaled, y_train, y_valid = train_test_split(x_train_scaled, y_train, test_size=0.2,
 random_state=25)


onehot_encoder = OneHotEncoder(sparse=False)
y_train_onehot = onehot_encoder.fit_transform(y_train.reshape((-1,1)))
y_test_onehot = onehot_encoder.fit_transform(y_test.reshape((-1,1)))
y_valid_onehot = onehot_encoder.fit_transform(y_valid.reshape((-1,1)))

print("Training Data Size = {}".format(y_train.shape[0]))
print("Testing  Data Size = {}".format(y_test.shape[0]))
print("Validation Data Size = {}".format(y_valid.shape[0]))

plotdata(data,labels)
```

```
Training Data Size = 1280
Testing  Data Size = 400
Validation Data Size = 320
```



In [150]:
```
# Parameters
learning_rate = 0.1
epochs = 100
batch_size = data_size/100
display_step = 10

# Network Parameters
n_hidden_1 = 100 # 1st layer number of neurons
n_hidden_2 = 100 # 2nd layer number of neurons
num_input = x_train.shape[1] # MNIST data input (img shape: 28*28)
num_classes = 2 # MNIST total classes (0-9 digits)

# tf Graph input
X = tf.placeholder("float", [None, num_input])
Y = tf.placeholder("float", [None, num_classes])
```

In [151]:
```
weights = {
    'h1': tf.Variable(tf.random_normal([num_input, n_hidden_1])),
    'h2': tf.Variable(tf.random_normal([n_hidden_1, n_hidden_2])),
    'out': tf.Variable(tf.random_normal([n_hidden_2, num_classes]))
}
biases = {
    'b1': tf.Variable(tf.random_normal([n_hidden_1])),
    'b2': tf.Variable(tf.random_normal([n_hidden_2])),
    'out': tf.Variable(tf.random_normal([num_classes]))
}
```

In [152]:
```
def neural_net(x, nl):

    layer_1 = tf.add(tf.matmul(x, weights['h1']), biases['b1'])
    if(nl == 'relu'):
        layer_1 = tf.nn.relu(layer_1)
        print("Using RELU")
    else:
        layer_1 = tf.nn.tanh(layer_1)
        print("Using TanH")

    layer_2 = tf.add(tf.matmul(layer_1, weights['h2']), biases['b2'])
    if(nl == 'relu'):
        layer_2 = tf.nn.relu(layer_2)
    else:
        layer_2 = tf.nn.tanh(layer_2)

    # Output fully connected layer with a neuron for each class
    out_layer = tf.matmul(layer_2, weights['out']) + biases['out']
    return out_layer
```

In [153]:
```
logits = neural_net(X, nl='tanh')

# Define loss and optimizer
loss_op = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(logits=logits, labels=Y))

#L2 Regularization
regularizers = tf.nn.l2_loss(weights['h1']) + tf.nn.l2_loss(weights['h2'])
loss_op = tf.reduce_mean(loss_op + 0.01 * regularizers)

optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate).minimize(loss_op)

# Evaluate model (with test logits, for dropout to be disabled)
correct_pred = tf.equal(tf.argmax(logits, 1), tf.argmax(Y, 1))
accuracy = tf.reduce_mean(tf.cast(correct_pred, tf.float32))

# Initialize the variables (i.e. assign their default value)
init = tf.global_variables_initializer()
```

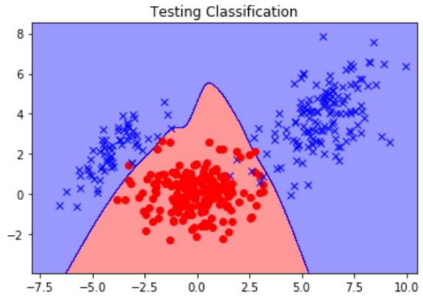
```
Using TanH
```

In [154]:
```
l = [] #to store accuracy.
```

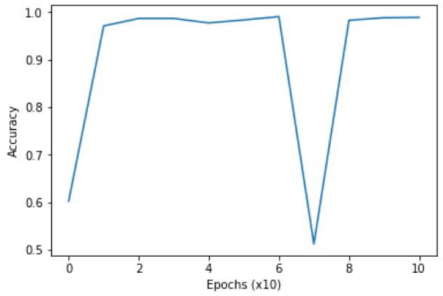
In [155]:

```
with tf.Session() as sess:

    # Run the initializer
    sess.run(init)

    for step in range(1, epochs+1):
        batch_x, batch_y = x_train, y_train_onehot
        # Run optimization op (backprop)
        sess.run(optimizer, feed_dict={X: batch_x, Y: batch_y})
        if step % display_step == 0 or step == 1:
            # Calculate batch loss and accuracy
            loss, acc = sess.run([loss_op, accuracy], feed_dict={X: batch_x,
                                                                 Y: batch_y})
            l.append(acc)
            valid_loss, valid_acc = sess.run([loss_op, accuracy], feed_dict={X: x_valid,
                                                                             Y: y_valid_onehot})
            print("Step " + str(step) + ", Training Loss= " + \
                  "{:.4f}".format(loss) + " & Accuracy= " + \
                  "{:.3f}".format(acc) + " || Validation Loss= " + \
                  "{:.4f}".format(valid_loss) + " & Accuracy= " + \
                  "{:.3f}".format(valid_acc))

    print("Optimization Finished!")

    # Calculate accuracy for MNIST test images
    test_acc = sess.run(accuracy, feed_dict={X: x_test,Y: y_test_onehot})
    print("Testing Accuracy : {}".format(test_acc))

    predictions = sess.run(tf.argmax(logits,1), feed_dict = {X: x_test})

    colors = ('red', 'blue', 'lightgreen', 'gray', 'cyan')
    cmap = ListedColormap(colors[:len(np.unique(y_train))])
    resolution = 0.02
    # plot the decision surface
    x1_min, x1_max = x_train[:, 0].min() - 1, x_train[:, 0].max() + 1
    x2_min, x2_max = x_train[:, 1].min() - 1, x_train[:, 1].max() + 1
    xx1, xx2 = np.meshgrid(np.arange(x1_min, x1_max, resolution),
                           np.arange(x2_min, x2_max, resolution))
    Z = sess.run(tf.argmax(logits,1), feed_dict = {X: (np.array([xx1.ravel(), xx2.ravel()]).T)})
    Z = Z.reshape(xx1.shape)
    plt.contourf(xx1, xx2, Z, alpha=0.4, cmap=cmap) # Displays classification region
    plt.xlim(xx1.min(), xx1.max())
    plt.ylim(xx2.min(), xx2.max())
    plotdata(x_test,y_test)
    plt.title("Testing Classification")
```

```
Step 1, Training Loss= 52.8564 & Accuracy= 0.602 || Validation Loss= 52.1224 & Accuracy= 0.650
Step 10, Training Loss= 15.6936 & Accuracy= 0.971 || Validation Loss= 15.6815 & Accuracy= 0.984
Step 20, Training Loss= 4.2397 & Accuracy= 0.987 || Validation Loss= 4.2752 & Accuracy= 0.981
Step 30, Training Loss= 1.8780 & Accuracy= 0.987 || Validation Loss= 1.8942 & Accuracy= 0.978
Step 40, Training Loss= 0.9176 & Accuracy= 0.977 || Validation Loss= 0.9704 & Accuracy= 0.984
Step 50, Training Loss= 1.3429 & Accuracy= 0.984 || Validation Loss= 1.4155 & Accuracy= 0.975
Step 60, Training Loss= 0.7317 & Accuracy= 0.991 || Validation Loss= 0.7813 & Accuracy= 0.984
Step 70, Training Loss= 4.6183 & Accuracy= 0.512 || Validation Loss= 4.7477 & Accuracy= 0.466
Step 80, Training Loss= 3.5350 & Accuracy= 0.983 || Validation Loss= 3.5391 & Accuracy= 0.988
Step 90, Training Loss= 2.5123 & Accuracy= 0.988 || Validation Loss= 2.5551 & Accuracy= 0.988
Step 100, Training Loss= 1.0097 & Accuracy= 0.989 || Validation Loss= 1.0381 & Accuracy= 0.988
Optimization Finished!
Testing Accuracy : 0.982500016689
```



In [158]:
```
plt.plot(l)
plt.xlabel("Epochs (x10)")
plt.ylabel("Accuracy")
plt.show()
```



In [ ]: