CSC 730 Assignment 04
Semi-supervised Learning

Mangesh Sakordekar

Overview

The aim of this assignment is to:

- Pick a supervised classifier model type and then generate 3 classifiers.
- The first classifier will just use all the data, with all labels.
- The second classifier will use only a single point from each class.
- The third classifier will perform an iterative procedure of self-training semi-supervised wrapper methods.
- Show the decision boundaries for all three classifiers.

Picking the Model

For this assignment, I used a neural net model from scikitlearn module. Using the MLPClassifier I wrote a function to fit the model and plot the decision surface.

```
from sklearn.neural_network import MLPClassifier
def classify(X_data, y_lbls, X_U, plot=False):

    clf = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1)
    history = clf.fit(X_data, y_lbls)

    if plot:
        plot2dgraph(clf, X_data, y_lbls, X_U)

    return clf
```

```
def plot2dgraph(model, X_L, y_L, X_U = None):
    fig, axs = plt.subplots(1,1)

    # plt.figure()
    if X_U is not None:
        plt.plot(X_U[:,0],X_U[:,1],'ko',alpha = 0.2)

    c_inds = np.where(y_L == 1)[0]
    plt.plot(X_L[c_inds,0],X_L[c_inds,1],'rs',label = f'class {str(1)}',markersize = 10)
    c_inds = np.where(y_L == 2)[0]
    plt.plot(X_L[c_inds,0],X_L[c_inds,1],'bd',label = f'class {str(2)}',markersize = 10)

    axs.set(xlim=(-5, 5), ylim=(-5, 5))
    axs.set_aspect('equal', 'box')

    plt.grid()
    plt.legend()
    xx, yy = np.meshgrid(np.linspace(-5, 5, 50), np.linspace(-5, 5, 50), indexing="xy")
    xy = np.dstack((xx, yy))
    zz = np.ndarray((50 , 50))
    for i in range(0, 50):
        for j in range(0, 50):
            zz[i][j] = model.predict(xy[i][j].reshape(1,-1))

    plt.contourf(xx, yy, zz, levels=10, cmap="RdBu", alpha=0.4)
    plt.xlabel('x0')
    plt.ylabel('x1')
    plt.title('Model Surface')
    plt.colorbar()
    plt.show()
```

Figure 1: Model and Plotting Function

Generating Data

The data was generated and and two points from each cluster were selected at random to separate the data into known and unknown datasets.

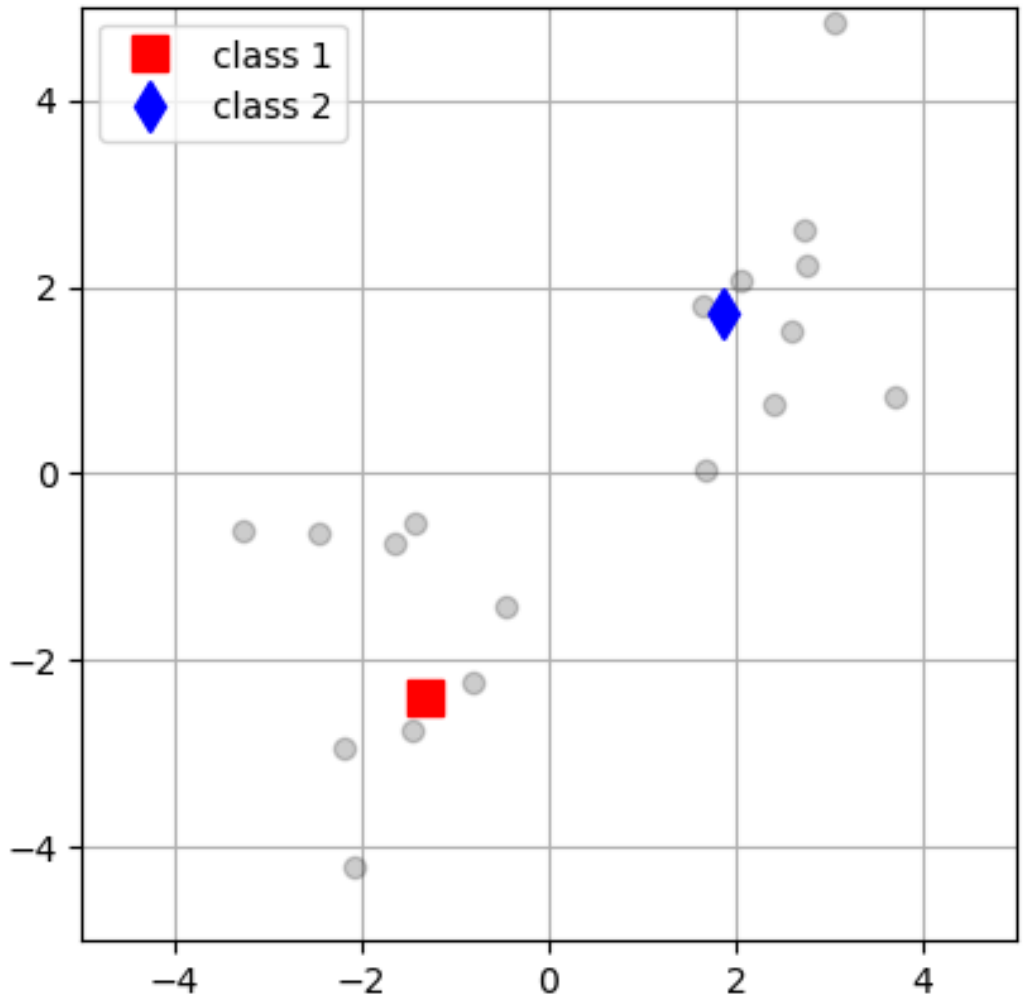

Figure 2: Data

Using Labelled Data

Model was trained using the labelled data. The decision surface was plotted and can be seen below.

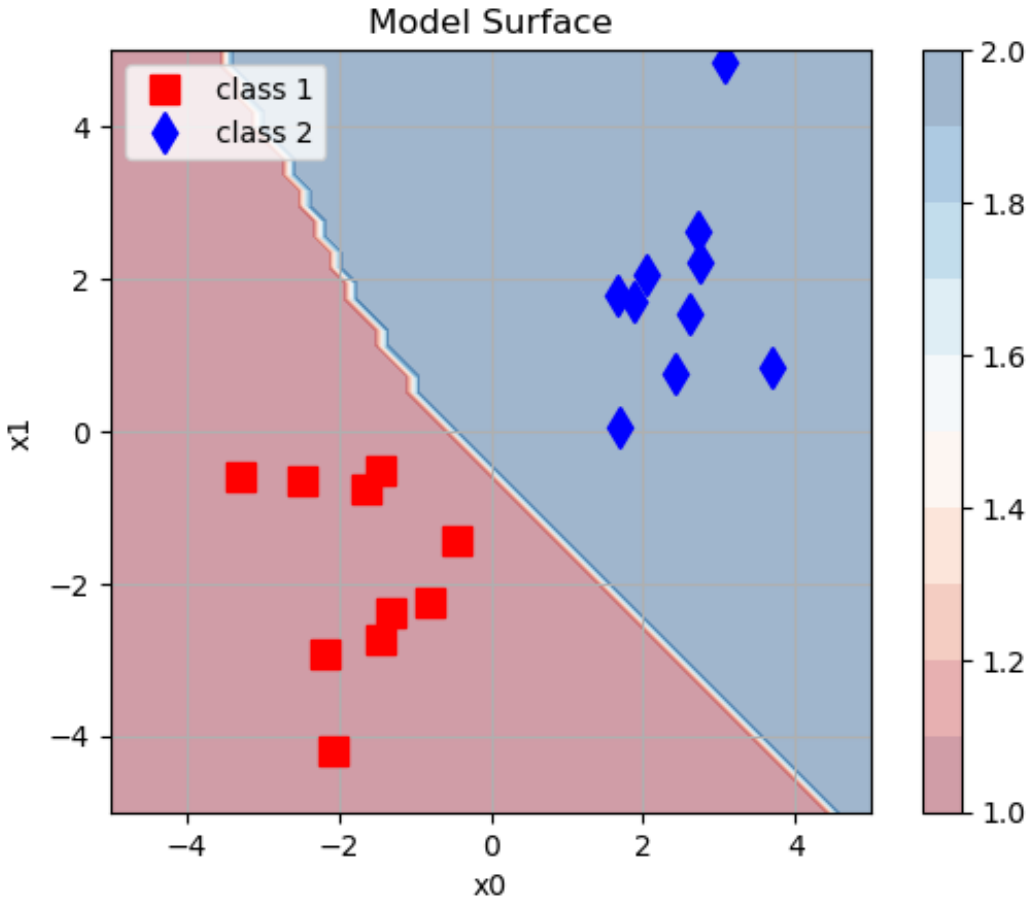

Figure 3: Model Surface for Labelled Data

Using Single Point from Each Class

Model was trained using just a single point from each class. The decision surface was plotted and can be seen below.

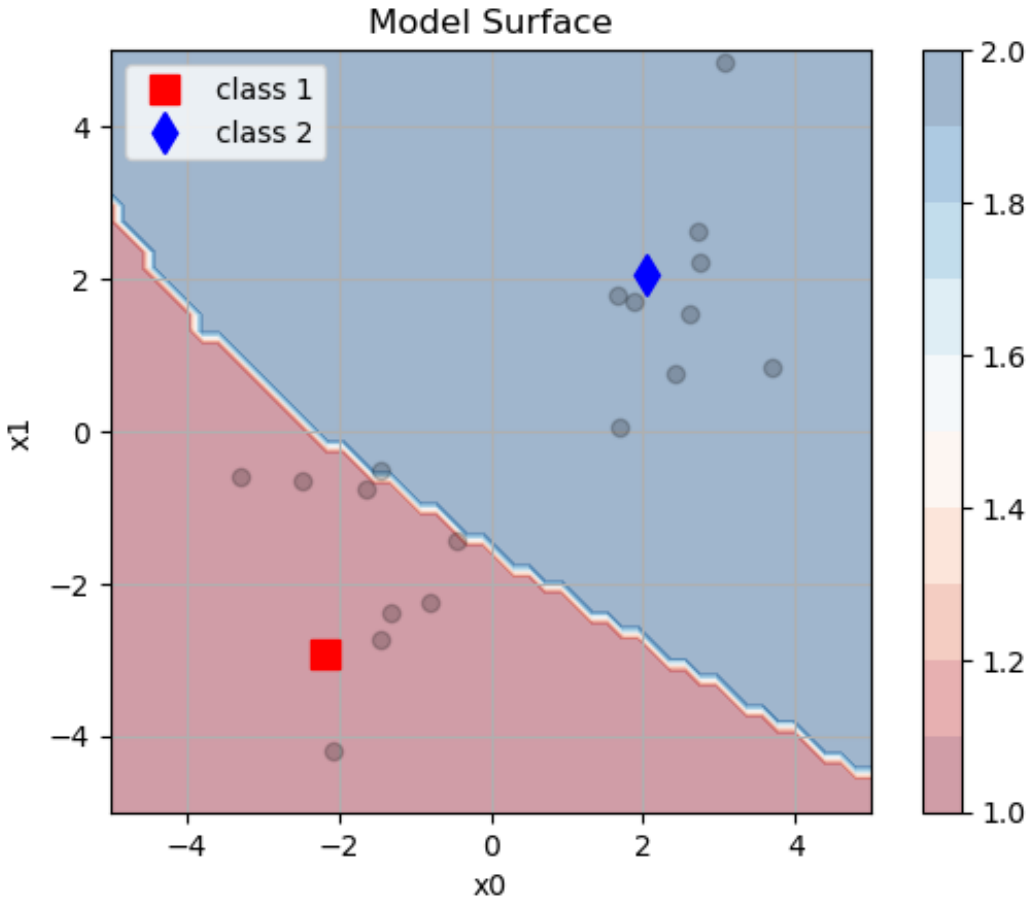

Figure 4: Model Surface for Single Points

Using Semi-Supervised Learning

Model was trained using just the original labeled points, that model was used to predict the classes for the unlabeled points, and those exceeding a confidence threshold, set to 0.95 in this case, were selected for belonging to a particular class. Those points with "pseudo-labels" were then included in the training set. Then the model was re-trained and this process was iterated until stabilization.

```python
# Find classifier using semi-supervised wrapper method
threshold = 0.95
model3 = classify(X_L, y_L, X_U, True)
while X_U.size:
    confidence = np.max(model3.predict_proba(X_U), axis=1)
    lbls = model3.predict(X_U)
    lst = [i for i in range(0,len(confidence)) if confidence[i]>= threshold]

    if len(lst):
        y_add = lbls[lst]
        X_add = X_U[lst,:]
        X_L = np.concatenate((X_L,X_add))
        y_L = np.concatenate((y_L, y_add))
        X_U = np.delete(X_U, lst, axis=0)
        model3 = classify(X_L, y_L, X_U, True)
    else :
        break

if len(X_U):
    print("Failed to assign " + str(len(X_U)) + " points to a class")
```

Figure 5: Semi-Supervised learning implementation

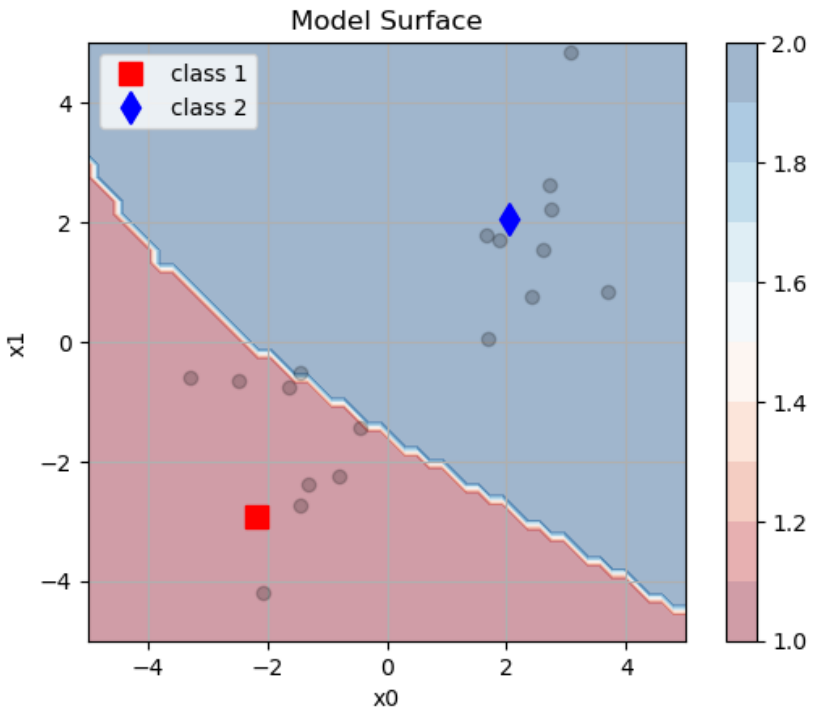

Figure 6: First iteration surface

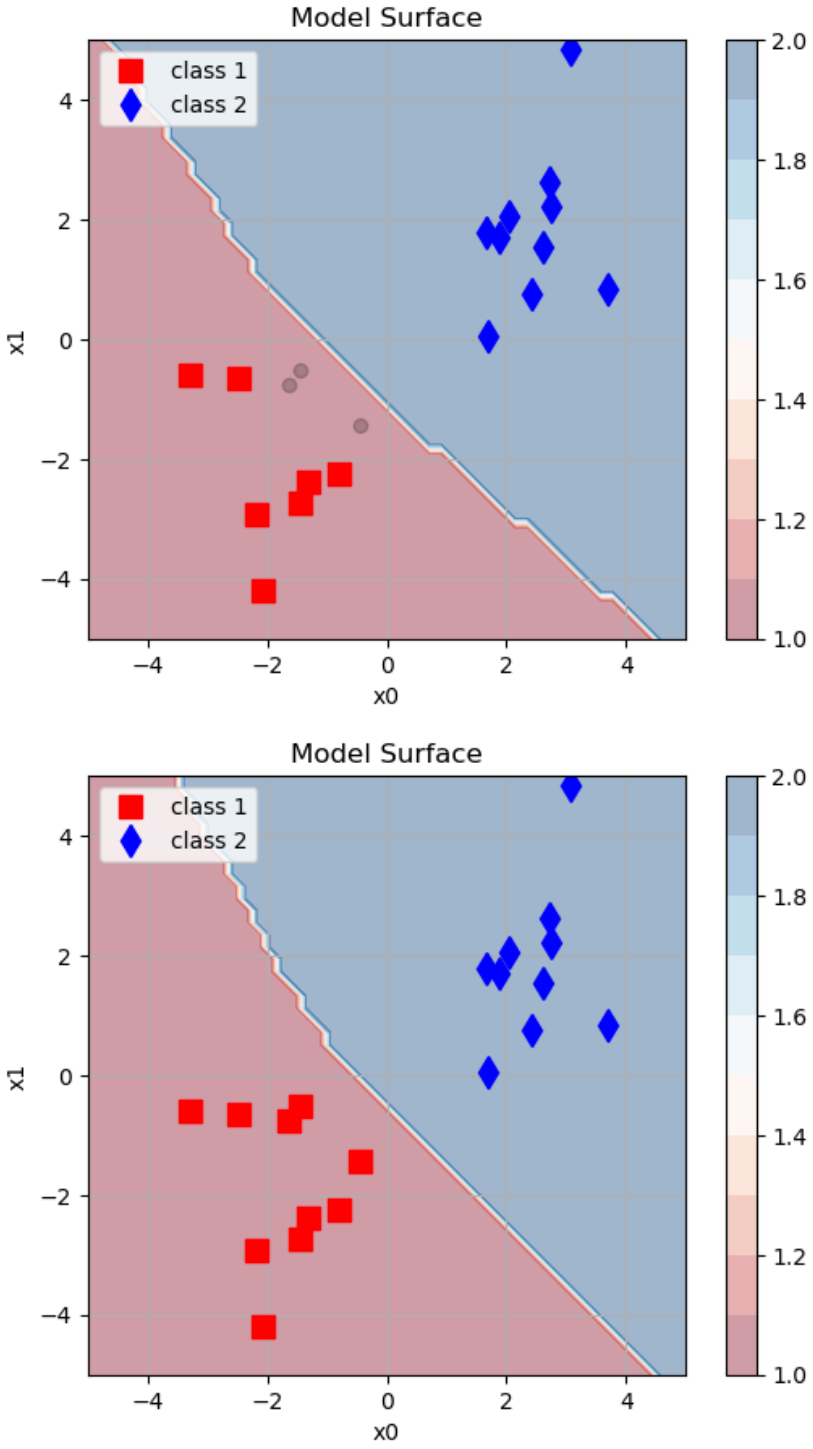

Figure 7: Top: Second iteration surface Bottom: Final Model surface

Conclusion

As we can see in the plots, during semi supervised learning the decision surface is generated using only the two labelled points at the start and every iteration more of the unlabelled points are assigned a class and the final model boundary is the same as the one obtained using fully labelled data. This indicates the model worked well for this particular dataset and starting known data.

Depending on the distribution of data and which points are labelled at the start of the process, the results can vary massively. Below is a plot for the same dataset with different labelled points to start with.

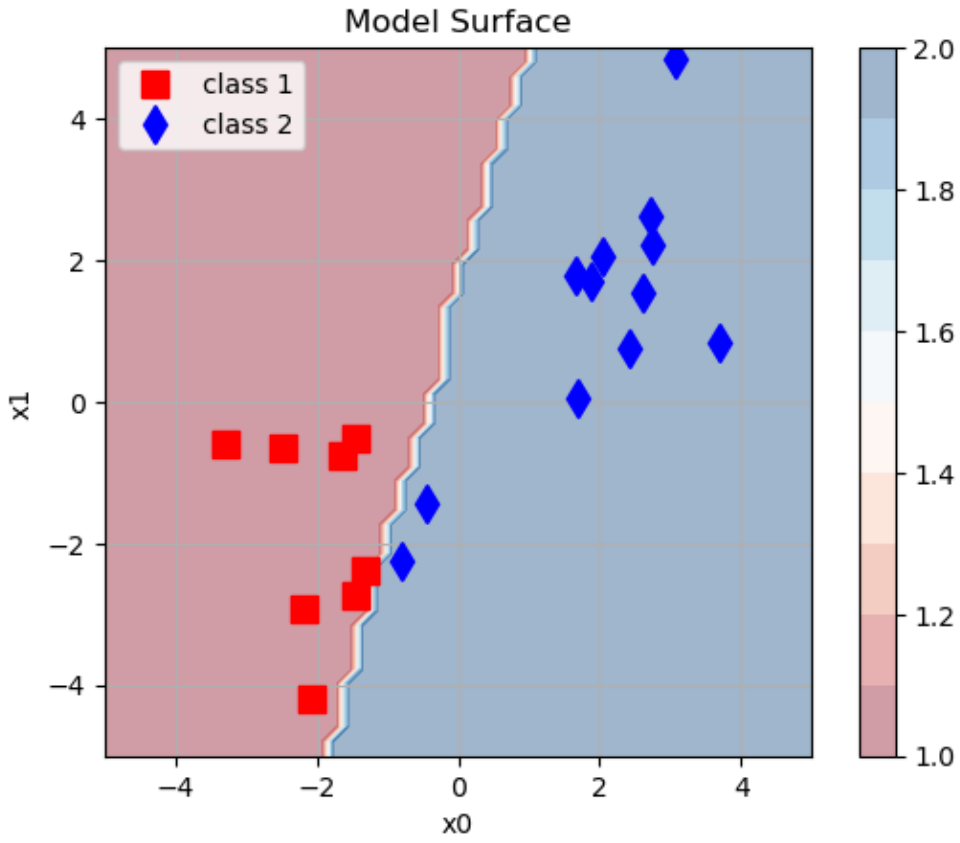

Figure 8: Incorrect Clustering Example