Problem Set #9

51 points (3 hours) -> counts for 7.5 points on your final grade.

Due: Monday, 03/31/2022, 11.59 PM.

1. **(51 points; ~3 hours) Comparing classification algorithms**
You are given three .mat files containing data. Each file contains one training dataset and two testing datasets. The data are organized as points (**x**) in 2-D feature space and scalar class labels y (1 or 2). With each of the three .mat files, do the following. By the end of this problem, you will hopefully appreciate (through first person experience), the difference between linear and nonlinear classifiers, the concepts of generalization (two different flavors), and the potential limitations of even an 'excellent' algorithm (such as Random forests) in learning from examples, when compared to what we, as humans, can accomplish.

A. ***Visualizing the given data*** (3 points; 10 min)

This is completed in the problem set code.

i. Plot in **subplot(2,2,1)**, $\mathbf{x}_{\mathbf{training}}$ (from the training dataset) in the 2D plane. Use **circles** (markersize=**3**), to plot $\mathbf{x}_{\mathbf{training}}$; use red circles for points belonging to class 1, and green circles for points belonging to class 2.

ii. In the same subplot, plot $\mathbf{x}_{\mathbf{testing-1}}$ (from the testing dataset). Use **circles** (markersize=**15**) and use red circles for points belonging to class 1, and green circles for points belonging to class 2.

iii. Finally, also in the same subplot, plot $\mathbf{x}_{\mathbf{testing-2}}$. Use the **square symbol** (markersize=**15**) and use red squares for points belonging to class 1, and green squares for points belonging to class 2.

B. ***LDA*** (6 points; 20 min) *(Use the included LDA.m function. It was downloaded from http://www.mathworks.com/matlabcentral/fileexchange/29673-lda--linear-discriminant-analysis)*

Notes: *(a) >> help LDA.m will tell you how to use the LDA function and obtain predicted ("learned") class labels.*
(b) The way this function implements LDA is slightly different from the approach I described in the slides/video. Instead, it is essentially the 'Fisher's linear discriminant' implementation. As I mentioned in the slides/video, the two implementations are conceptually isomorphic.

i. Apply LDA to the training dataset and "learn" the classification boundary (use code at the link above). In **subplot(2,2,2)**, plot $\mathbf{x}_{\mathbf{training}}$ again (as circles, markersize=3), but now the colors should correspond to the "learned" class labels (red-1/green-2). Compare this plot with the previous one, and comment on how well LDA has done on this training dataset. Compute the accuracy (% correct) on the **training** dataset and report it.

We see that our LDA weights yielded a top part of the graph that is entirely labeled as class 1 in red with the bottom portion green as class 2! Its vital to note that our training dataset appears to not have any data illustrating how points are distributed in the upper half of the plot (from >0). On the training data, we have a relatively successful classification accuracy of 83.54%. Visually, it does appear that the classifier did okay by labeling points close to 0 with respect to feature 2 as class 1 and those below as class 2.

ii. Apply this learned classification boundary to $\mathbf{x}_{\mathbf{testing-1}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**),** plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the learned LDA boundary **generalize** to previously unseen data?

The classifier did extremely poorly on this dataset! It only got a score of 34%. We can see in the original dataset that there are many class 2 labels in the upper portion of the plot (feature 2 > 0.5) while the training data has most data less than zero as class 2. The classifier therefore predicted every single point in test 1 to be a member of class 1! This demonstrates that, in some cases, the learned LDA boundary generalizes extremely poorly to previously unseen data or distributions!

iii. Finally, apply the learned classification boundary to $\mathbf{x}_{\mathbf{testing-2}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**),** plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "LDA". How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the LDA boundary generalize to previously unseen data that are similar to the training set (let's call this '**interpolation'**)?

This classifier did somewhat better with an accuracy of 60.78% likely because, as the problem set mentions, this training data is similar to our training data. Thus, the learned weights were capable of being interpolated to this new dataset somewhat well. The classifier correctly labeled the data near zero with a horizontal line as the boundary as Class 1 and things below this line as Class 2.

C. ***Random forests*** (6 points; 25 min) *(https://code.google.com/p/randomforest-matlab/*)*
**This code works on Windows machines. If it does not work on your Mac, see the included PS9_RFCodeAlternatives.pdf*

i. Now apply RF to the same training dataset and learn the classification boundary (use code at the link above). In **subplot(2,2,3)**, plot $\mathbf{x}_{\mathbf{training}}$ (as circles, markersize=3), using colors that correspond to the learned class labels (red-1/green-2). Comment on how well RF has done on this **training** dataset. What is the accuracy (% correct)?

RF performed perfectly on the training data! It has an accuracy of 100%! That's an incredible performance! We must be aware, however, that our dataset is not very large and thus the classifier could almost memorize the dataset or risk overfitting.

ii. Apply this learned classification boundary to $\mathbf{x}_{\mathbf{testing\text{-}1}}$. In the same subplot as in (Ci), i.e, subplot(2,2,3), plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does the RF boundary **generalize** to previously unseen data?

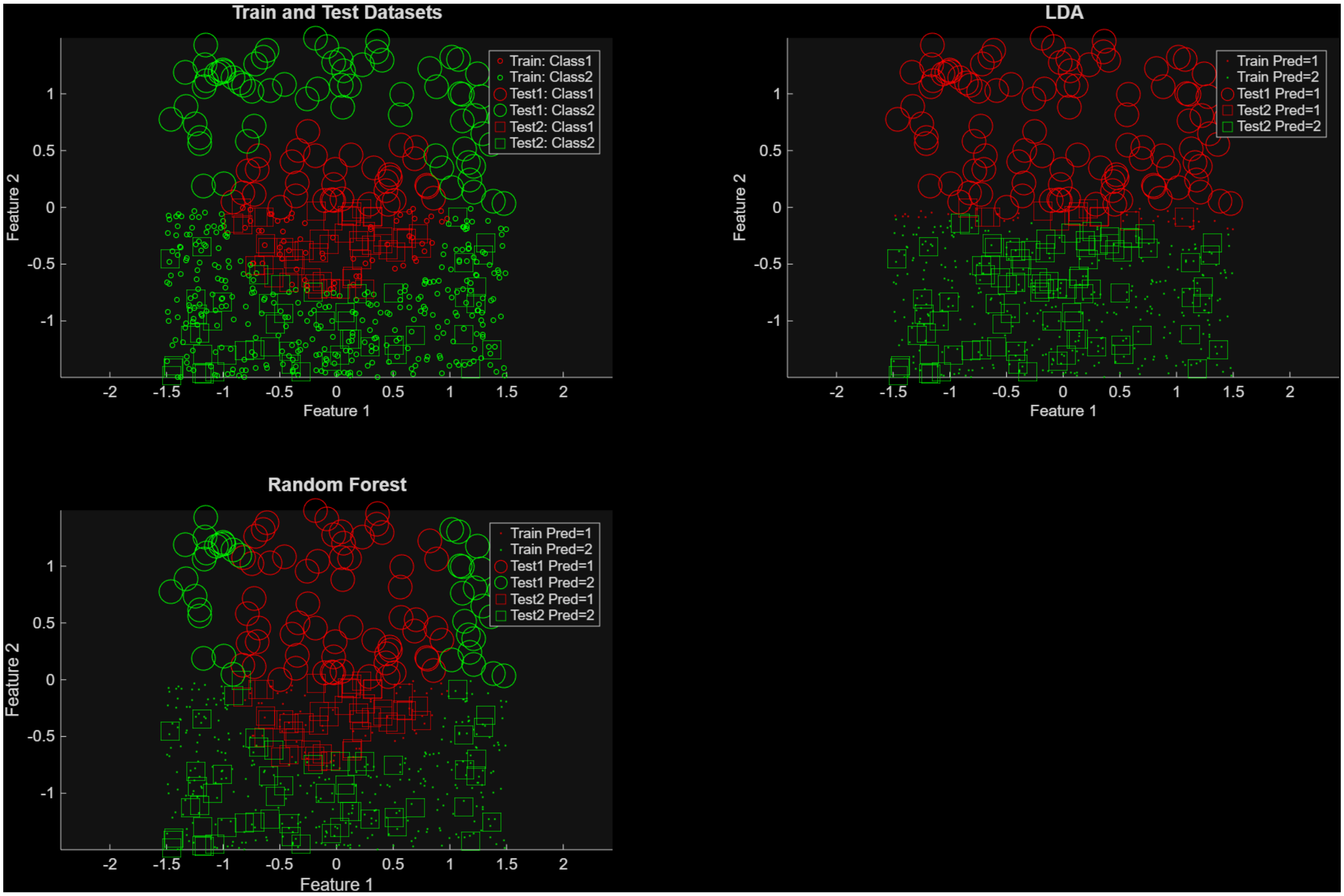
In the Test1 dataset, the model achieved an accuracy of 70% which is still a solid performance especially compared to the LDA model's attempt! This generalizes much better to previously unseen data interestingly.

iii. Finally, apply the learned classification boundary to $\mathbf{x}_{\mathbf{testing\text{-}2}}$. In the same subplot as in (Ci), i.e**, subplot(2,2,3),** plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "Random forests". How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does RF generalize to previously unseen data that are similar to the training set ('interpolation')?

The RF Classifier in the interpolation case performs exceptionally well with an accuracy of 93.14%! This shows that RF performs quite well when the unseen data is similar to the training dataset.

D. ***Introspection*** (2 points; 5 min). Comment (in a sentence or two) on how well you think you might have done in predicting the class labels for $\mathbf{x}_{\mathbf{testing\text{-}1}}$ and $\mathbf{x}_{\mathbf{testing\text{-}2}}$, given only $\mathbf{x}_{\mathbf{training}}$ and y: Do you think you could have done better than LDA and RF?

After conducting this exercise, I believe that I probably would have only done as well as LDA if I only had access to the training data and was restricted to making a simple linear line to classify the data. I would have placed the threshold slightly lower potentially if I were to naively guess where the classification boundary with the best performance lies. If I could draw a non-linear line, I believe I would have performed slightly better than LDA, but that doesn't seem like a fair comparison given LDA's constraints. I don't think I would have necessarily classified the edges of the test datasets as well as random forest given that I had no knowledge that the pattern seen in the training data would continue into the test dataset. I suppose I would be more conservative in how I classify things without additional knowledge.



data_2.mat

A. ***Visualizing the given data*** (3 points; 10 min)

This is completed in the problem set code.

B. ***LDA*** (6 points; 20 min)

i. Apply LDA to the training dataset and "learn" the classification boundary (use code at the link above). In **subplot(2,2,2)**, plot $\mathbf{x}_{\mathbf{training}}$ again (as circles, markersize=3), but now the colors should correspond to the "learned" class labels (red-1/green-2). Compare this plot with the previous one, and comment on how well LDA has done on this training dataset. Compute the accuracy (% correct) on the **training** dataset and report it.

Here, we see that our LDA classifier has essentially chosen to label the entirety of the training set as class two! Interestingly, it still has a higher accuracy score of 74.94%! However, we know that this value is quite misleading as it fails spectacularly at classifying class 1. Further, this shows a failure of a linear discriminator when there appears to be unbalanced datasets that are inherently non-linear in character!

iv. Apply this learned classification boundary to $\mathbf{x}_{\mathbf{testing-1}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**),** plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the learned LDA boundary **generalize** to previously unseen data?

The classifier had a similar performance on the test 1 dataset! I found an accuracy score of 75%. Again, however, this score alone does not describe the performance adequately. It failed severely to classify any of the values in class 1! Thus, we see further evidence that the LDA boundary failed to generalize to unseen data.

v. Finally, apply the learned classification boundary to $\mathbf{x}_{\mathbf{testing-2}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**)**, plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "LDA". How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the LDA boundary generalize to previously unseen data that are similar to the training set (let's call this '**interpolation'**)?

In this case, the LDA algorithm performed poorly and received an accuracy of only 50%. It has failed again to properly label any points as class 1! In this case, it was only capable of interpolating somewhat successfully. However, I would struggle to say that it really did interpolate correctly because it has simply learned to classify everything as class 2 seemingly no matter what!

E. ***Random forests*** (6 points; 25 min) *(https://code.google.com/p/randomforest-matlab/*)*
**This code works on Windows machines. If it does not work on your Mac, see the included PS9_RFCodeAlternatives.pdf*

iv. Now apply RF to the same training dataset and learn the classification boundary (use code at the link above). In **subplot(2,2,3)**, plot $\mathbf{x}_{\mathbf{training}}$ (as circles, markersize=3), using colors that correspond to the learned class labels (red-1/green-2). Comment on how well RF has done on this **training** dataset. What is the accuracy (% correct)?

Again, RF performed perfectly on the training data! It has an accuracy of 100%! Another amazing performance! We must be aware again, however, that our dataset is not very large and thus the classifier could almost memorize the dataset or risk overfitting.

v. Apply this learned classification boundary to $\mathbf{x}_{\mathbf{testing-1}}$. In the same subplot as in (Ci), i.e, subplot(2,2,3), plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does the RF boundary **generalize** to previously unseen data?

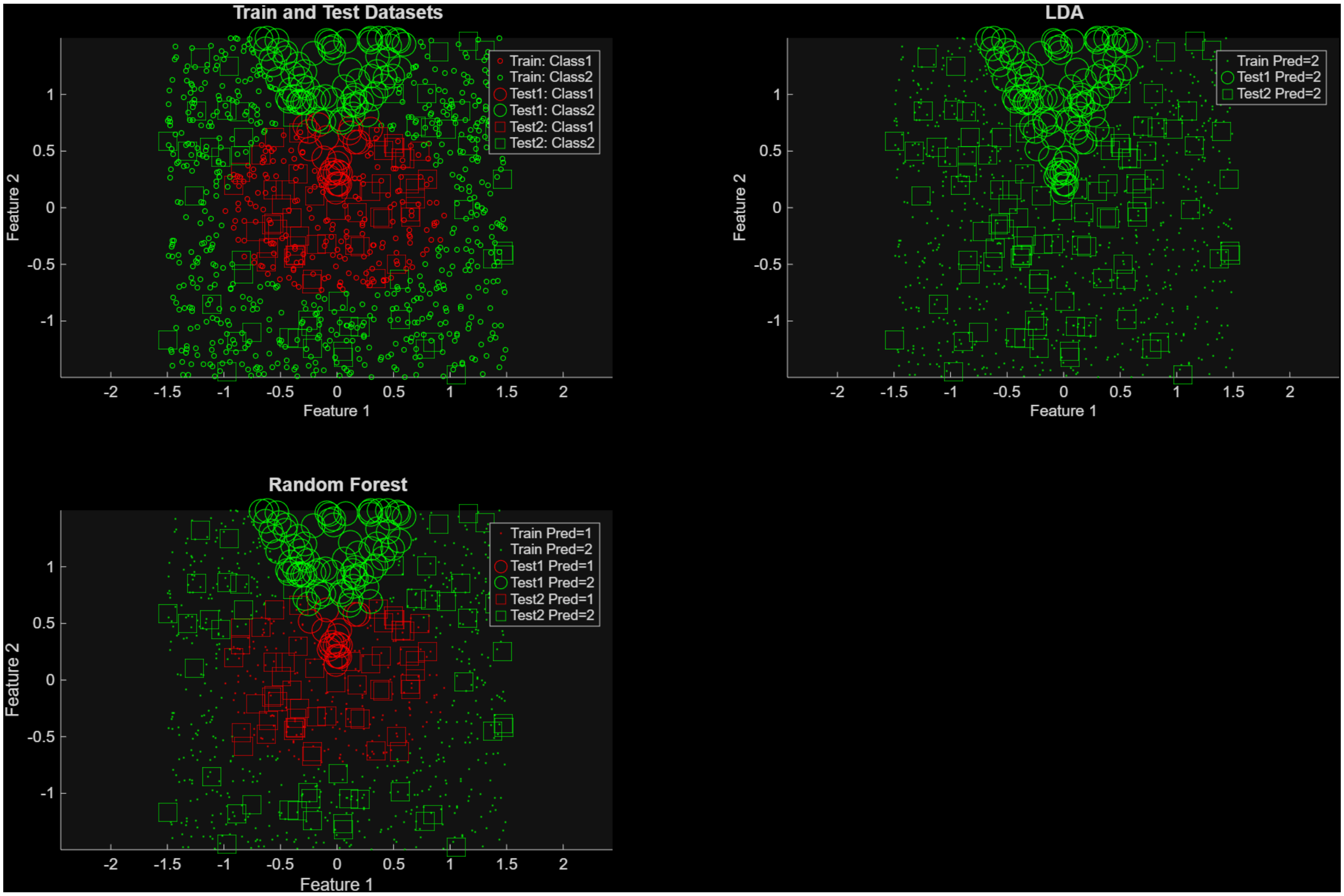
In the Test1 dataset, the model achieved an accuracy of 93.48% which is an excellent result! Visually, the dataset appears to be very well classified. This generalizes much better to previously unseen data. The top of the circle with class 1 appears to be the problem region.

vi. Finally, apply the learned classification boundary to $\mathbf{x}_{\mathbf{testing-2}}$. In the same subplot as in (Ci), i.e**, subplot(2,2,3),** plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "Random forests". How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does RF generalize to previously unseen data that are similar to the training set ('interpolation')?

The RF Classifier in the interpolation case performs exceptionally well with an accuracy of 96.08%! It appears that the RF performs exceptionally well in the interpolation case as well!

F. ***Introspection*** (2 points; 5 min). Comment (in a sentence or two) on how well you think you might have done in predicting the class labels for $\mathbf{x}_{\text{testing-1}}$ and $\mathbf{x}_{\text{testing-2}}$, given only $\mathbf{x}_{\text{training}}$ and y: Do you think you could have done better than LDA and RF?

After working through this dataset, I believe that I would have done probably just as poorly as the LDA classifier if I was restricted to a simple linear separation. There does not appear to be a cleanly linearly separable space for our training data. I believe if I could draw a nonlinear classification, I could have done about as well as the random forest because it appears that the training data fills the entire space of the two features and that I would have drawn a circle in the middle to segregate the two classes.



data_3.mat

C. ***Visualizing the given data*** (3 points; 10 min)

This is completed in the problem set code.

D. ***LDA*** (6 points; 20 min)

ii. Apply LDA to the training dataset and "learn" the classification boundary (use code at the link above). In **subplot(2,2,2)**, plot $\mathbf{x}_{\text{training}}$ again (as circles, markersize=3), but now the colors should correspond to the "learned" class labels (red-1/green-2). Compare this plot with the previous one, and comment on how well LDA has done on this training dataset. Compute the accuracy (% correct) on the **training** dataset and report it.

Here, again, we see that our LDA classifier has labeled the entirety of the training set as class two! It achieves an accuracy of 76.55%. This value is, again, quite misleading as it does not classify class 1.

vi. Apply this learned classification boundary to $\mathbf{x}_{\text{testing-1}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**)**, plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the learned LDA boundary **generalize** to previously unseen data?

The classifier had a similar performance on the test 1 dataset! I found an accuracy score of 70.73%. Again, however, this score alone does not describe the performance adequately. It, again, failed severely to classify any of the values in class 1! Thus, we see further evidence that the LDA boundary failed to generalize to unseen data even though it achieved a success of 70.73%.

vii. Finally, apply the learned classification boundary to $\mathbf{x}_{\textbf{testing-2}}$. In the same subplot as in (Bi), i.e, subplot(2,2,2**)**, plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "LDA". How accurate is the LDA classifier on this testing dataset (what is the % correct)? In other words, how well does the LDA boundary generalize to previously unseen data that are similar to the training set (let's call this '**interpolation'**)?

In this case, the LDA algorithm performed poorly and received an accuracy of only 50%. It has failed again to properly label any points as class 1! In this case, it was only capable of interpolating somewhat successfully. However, I would struggle to say that it really did interpolate correctly because it has simply learned to classify everything as class 2.

G. ***Random forests*** (6 points; 25 min) *(https://code.google.com/p/randomforest-matlab/*)*
**This code works on Windows machines. If it does not work on your Mac, see the included PS9_RFCodeAlternatives.pdf*

vii. Now apply RF to the same training dataset and learn the classification boundary (use code at the link above). In **subplot(2,2,3)**, plot $\mathbf{x}_{\textbf{training}}$ (as circles, markersize=3), using colors that correspond to the learned class labels (red-1/green-2). Comment on how well RF has done on this **training** dataset. What is the accuracy (% correct)?

Once again, RF performed perfectly on the training data! It has an accuracy of 100%. We must be aware again, however, that our dataset is not very large and thus the classifier could almost memorize the dataset or risk overfitting.

viii. Apply this learned classification boundary to $\mathbf{x}_{\textbf{testing-1}}$. In the same subplot as in (Ci), i.e, subplot(2,2,3), plot $\mathbf{x}_{\text{testing-1}}$ (using circles, markersize=15), with colors corresponding to the "predicted" class labels. How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does the RF boundary **generalize** to previously unseen data?

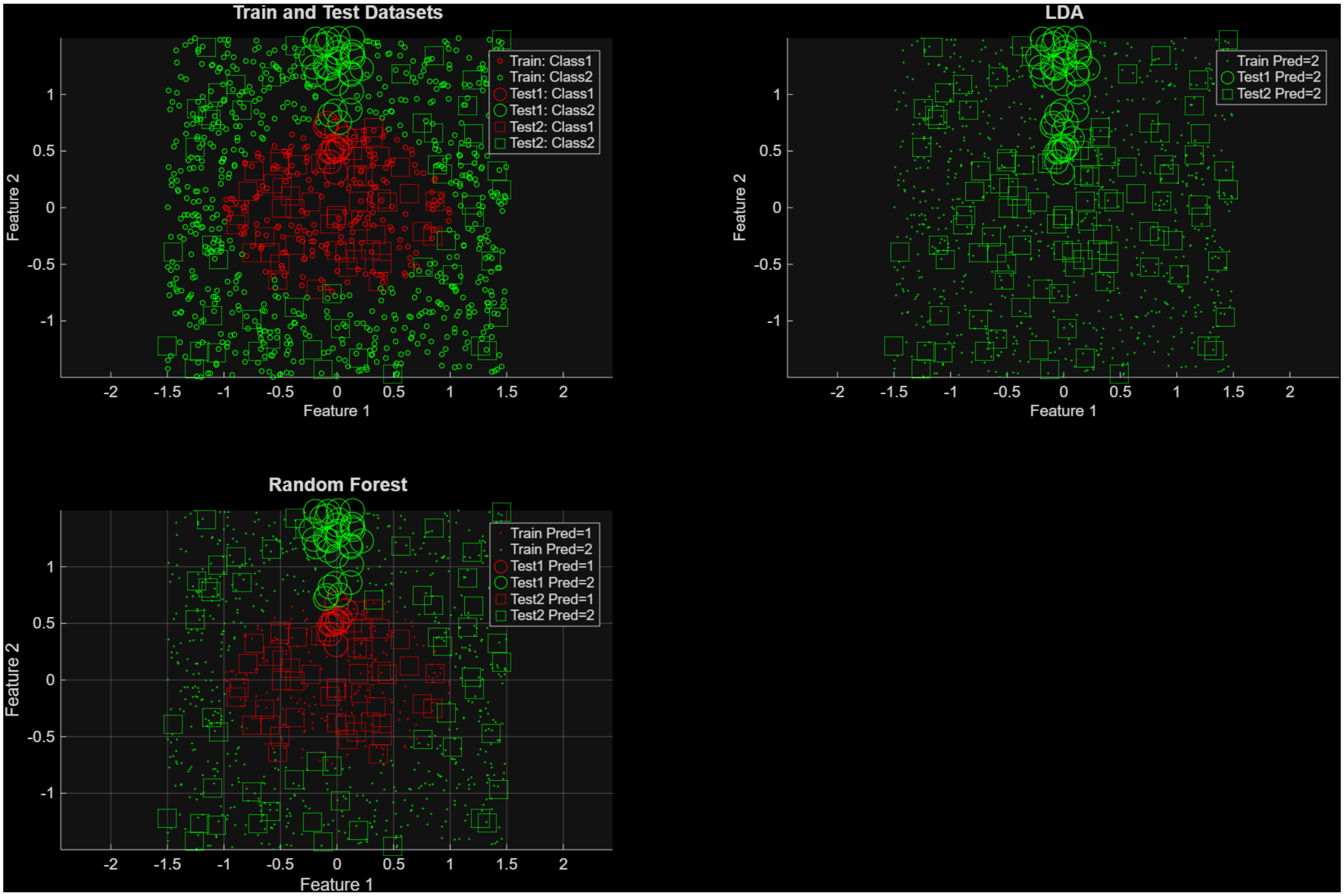
In the Test1 dataset, the model achieved an accuracy of 95.12%! Visually, the dataset appears to be very well classified. This generalizes much better to previously unseen data. The top of the circle with class 1 appears to be the problem region again.

ix. Finally, apply the learned classification boundary to $\mathbf{x}_{\textbf{testing-2}}$. In the same subplot as in (Ci), i.e**, subplot(2,2,3),** plot $\mathbf{x}_{\text{testing-2}}$ (using the square symbol, markersize=15), with colors corresponding to the "predicted" class labels. Title this plot as "Random forests". How accurate is the RF classifier on this testing dataset (what is the % correct)? In other words, how well does RF generalize to previously unseen data that are similar to the training set ('interpolation')?

The RF Classifier in the interpolation case performs exceptionally well with an accuracy of 98.04%! It appears that the RF performs exceptionally well in the interpolation case as well!

H. ***Introspection*** (2 points; 5 min). Comment (in a sentence or two) on how well you think you might have done in predicting the class labels for $\mathbf{x}_{\textbf{testing-1}}$ and $\mathbf{x}_{\textbf{testing-2}}$, given only $\mathbf{x}_{\textbf{training}}$ and y: Do you think you could have done better than LDA and RF?

In this dataset, I believe that I would have done probably just as poorly as the LDA classifier if I was restricted to a simple linear separation. There does not appear to be a cleanly linearly separable space for our training data. I believe if I could draw a nonlinear classification, I could have done about as well as the random forest because it appears that the training data fills the entire space of the two features and that I would have drawn a circle in the middle to segregate the two classes. This dataset appears to be extremely similar to the second dataset.



Train and Test Datasets

LDA

Random Forest