Project 2

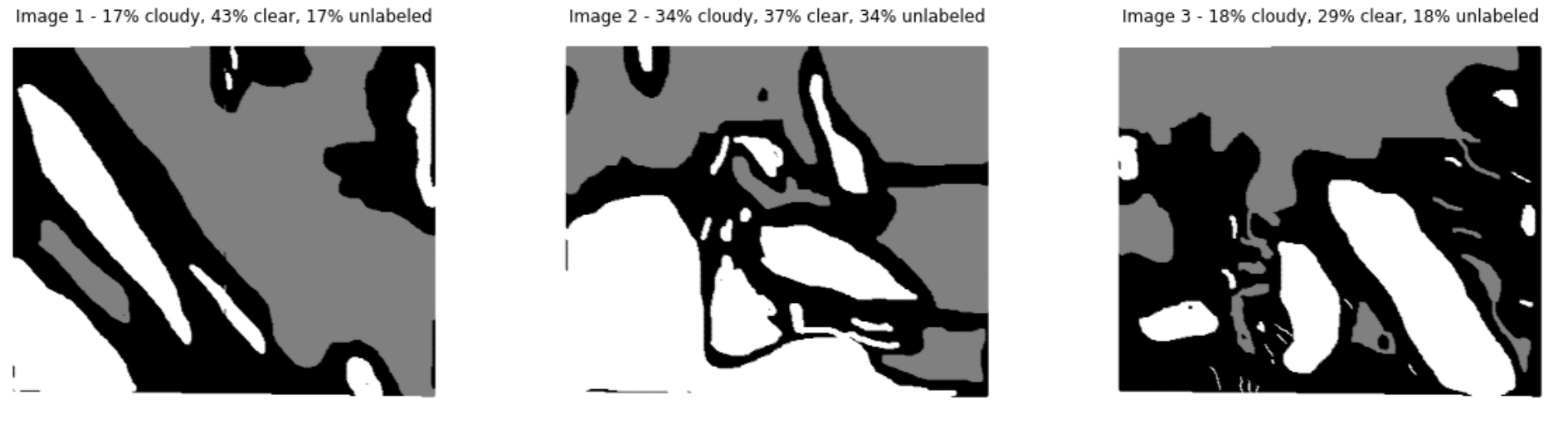
*Zhiling Pan (25712664), Shensheng Chen (25818289)*

**Data Collection and Exploration**

Summary of the Paper

The paper proposes different methods of detecting clouds from satellite data. It uses data collected from the Multiangle Imaging SpectroRadiometer (MISR), specifically images from the 9 onboard cameras of the MISR. Images are taken as the MISR orbits the earth, resulting in 9 images for the same location at different angles, with each MISR pixel covering a 275m × 275m region on the ground. This paper proposes two cloud detection methods to detect clouds from this MISR data. This is significant because there is a massive amount of MISR data that can be used to track the changes in cloud coverage overtime and an efficient algorithm is required to detect the clouds within that data. It introduces three physical features that have high predictive power: the linear correlation of radiation measurements from different camera angles (CORR), the standard deviation of radiation between camera angles (SD), and a normalized difference angular index (NDAI). The concluded that the algorithm provides higher accuracy than the existing MISR operational algorithms for cloud detection, which is significant because it can be used to better study the changes in cloud coverage.

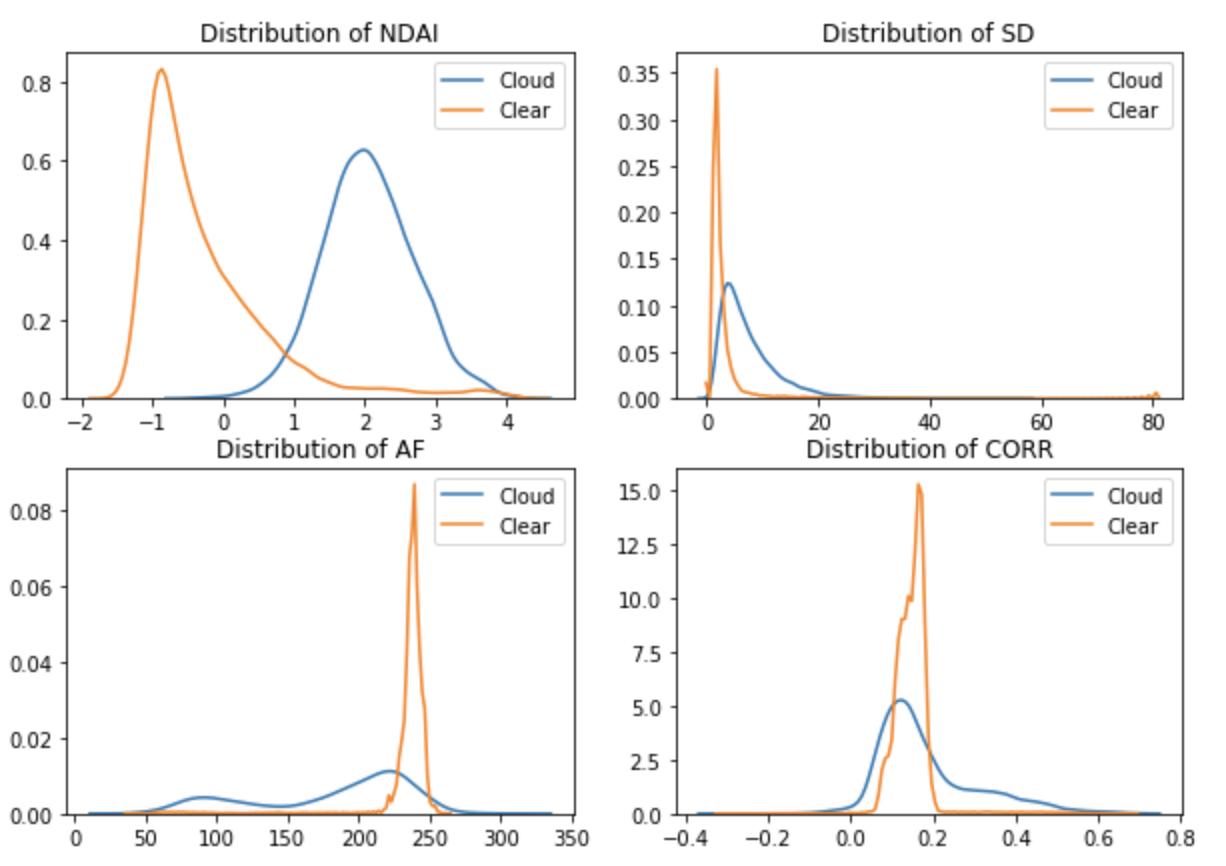
Summary of Data



*Fig1: plot of expert labels (white is cloud, gray is clear, and black is unlabeled)*

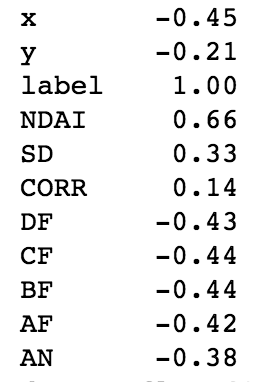
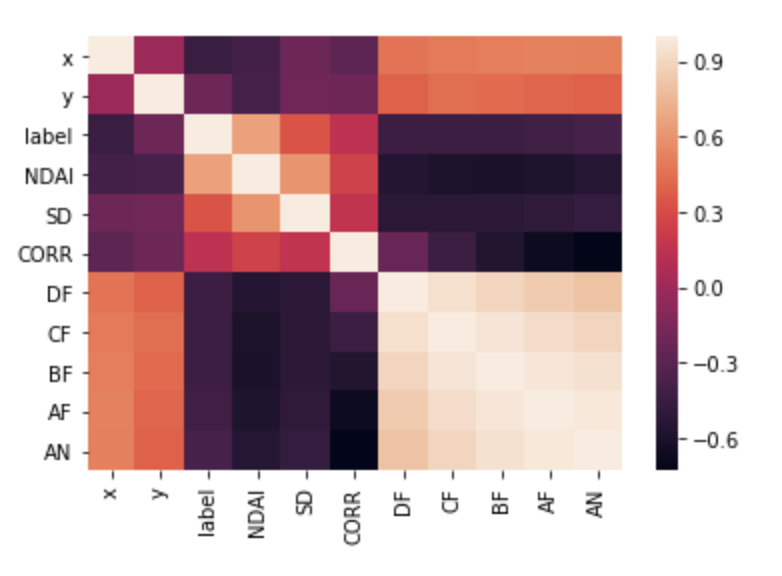
Figure 1 is a plot of the expert labels from the three images in our dataset. As we can see, clouds form in clusters. A pixel surrounded by cloud pixels is more likely to be a cloud pixel itself. Likewise, a pixel surround by clear pixels is more likely to be a clear pixel itself. This means that the data is not i.i.d. In general, pixels in an image are not iid. Creating a training set by uniformly randomly picking pixels will result in a training set that is just a low resolution version of the original image. The validation and testing set will also be a low resolution version of the SAME IMAGE. So in a sense, it’ll just testing/validating on the same image from the training set.

EDA



*Figure 2: distribution of NDAI, SD, CORR, and AF*

Figure 2 is the distribution of 4 key values from the dataset: NDAI, SD, CORR, and AF. Higher values of NDAI corresponds to cloudiness and the NDAI distributions for cloudiness and clear look gaussian and are quite separated. This suggests that a QDA model would work well. The other three values have a less gaussian distribution. There seems to be a tight range of values for the clear pixels and a wider range of values for the cloudy pixels. Finally, there is a lot of overlap between the clear distribution for SD, AF, and CORR, which might reduce the predictive power of these variables.



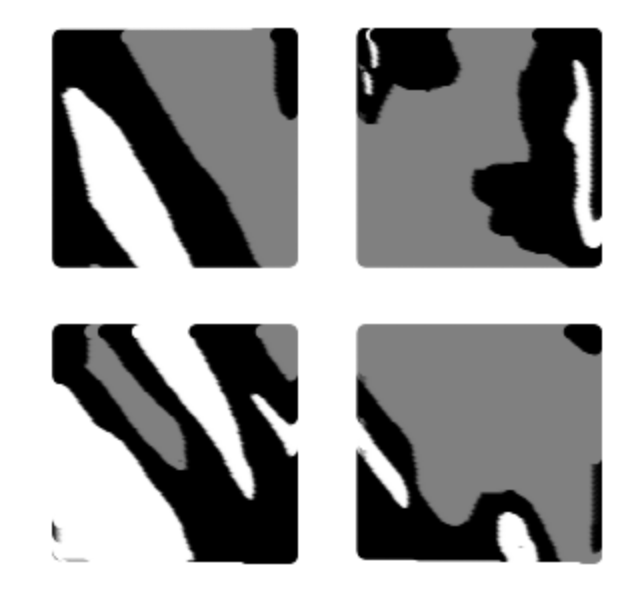
Above on the left, we have the correlation plot of all the variables in the dataset and on the right we have the correlation of each variable with the label. The 5 angles are all highly positively correlated with each other which makes sense since they’re just different angles of the same location. And the angles are correlated with the derived values SD, CORR, and DF, which also makes sense since they’re calculated from the angles. Finally, the angles are negatively correlated with cloudiness, CORR and SD are weakly correlated with cloudiness, and NDAI is strongly correlation with cloudiness. This suggests that NDAI and the angles are good predictors.

**Preparation**

Data Split

We propose two methods of splitting the data. Since the pixels are not i.i.d., choosing at random will create a test set that is similar to the training set and we won’t be able to test how the models generalizes to new examples.

Method 1 is to split each image in the dataset into 4 equal regions. Below is an example of how image 1 would be split. Since we have 3 images, this gives us 12 regions total. We then randomly pick 7 for training, 2 for validation, and 3 for testing. We think that regions are sufficiently different enough from each other that they can be considered i.i.d.



Method 2 is to chose pixels randomly with 75% for training and validation and 25% for testing. Even though the pixels are not i.i.id, we purposefully chose this method because we want to see how much better the model performs when tested on examples similar to the training set. If it performs a lot better, then that means cloud data is very diverse and we will need a lot more training data to improve the performance of the model. If it performs about the same, then that means either the model is not complex enough to detect the patterns or that we need better/new features because the current data does not contain all the information we need to detect clouds.

Baseline classifier

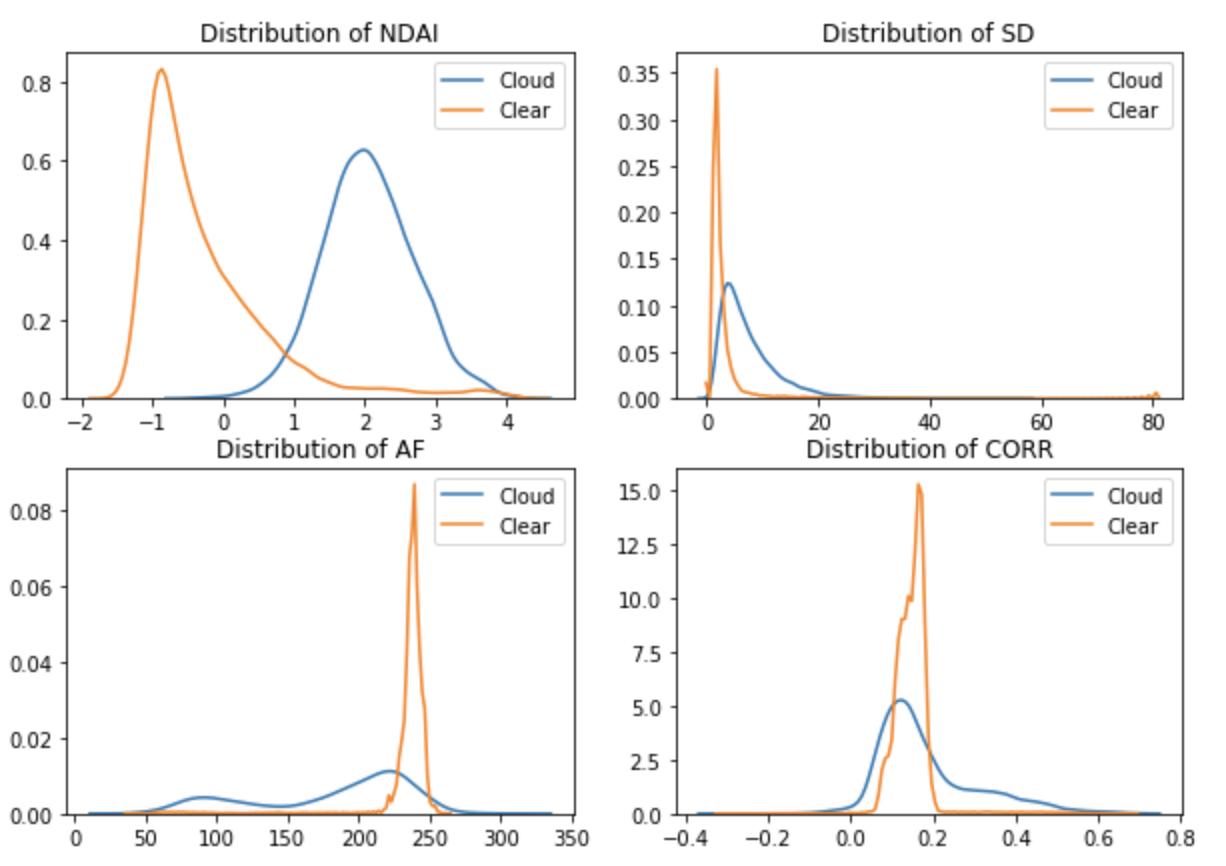
A baseline classifier that classify every pixel as clear gives us the following accuracy:

Method 1: (0.95 on validation, 0.77 on test)

Method 2: (0.61 on validation, 0.61 on test)

The fact that method 1 and method 2 give such different accuracy rates further supports that the data is not i.i.d. The baseline classifier will perform really well if the validation and training set of pixels are mostly clear. In fact, its performance will exactly be the proportion of pixels that are clear in the data. Its performance on method 2 suggests that 61% of pixels in the dataset are clear. This means that the problem is not trivial as there is huge room for improvement and that for our models to be considered predictive, it should have a higher accuracy than 61%.

Feature Importance



From EDA, we found that NDAI and the angles have high correlation with the labels, while SD and CORR have low correlation. The figure above shows that there is little overlap between the distribution of NDAI values for cloud pixels and its distribution for clear pixels, which means NDAI has high predictive power for separating the two classes. For the angles, there is a very tight range where most of its clear pixel values reside. This is very promising for predicting clear pixels. Overall, we choose NDAI, DF, and AN as our three most important features. They all have high correlation and since DF and AN are the further apart so there is less duplicate information between the two.

**Modeling**

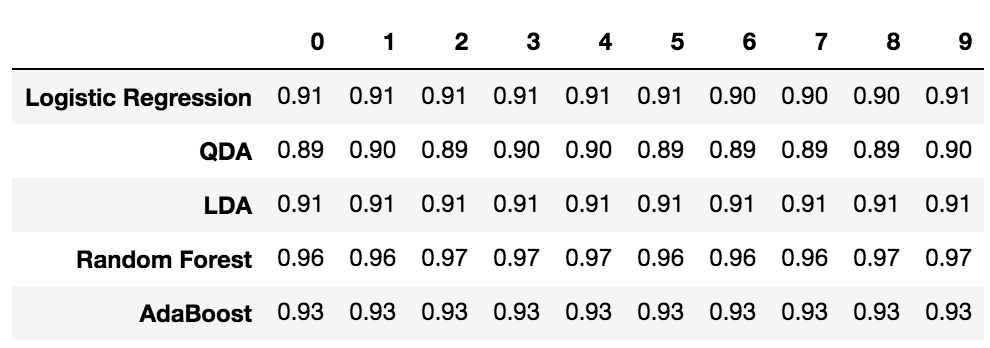
For the feature space we used NDAI, SD, CORR, and all the angles. We left out the x,y coordinates because they’re a random consequence of how the picture was taken and should have no predictive power.

We chose QDA and LDA because the distribution of clear and cloudy NDAI values look gaussian and is sufficiently spaced apart. However, it might be harder for QDA/LDA to pick out the distributions for the other values since there is significant overlap.

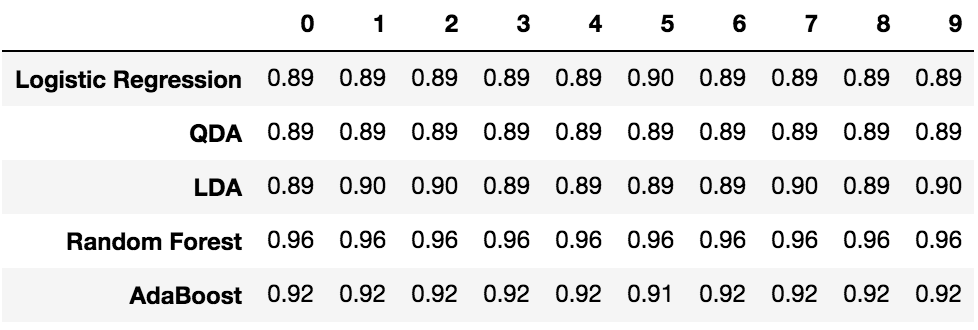
We chose random forest and adaboost because we think the weather patterns are sufficiently complex where there will be weird edge cases that a more general classifier won’t pick up. Since both are ensemble methods, it can better handle the those edge cases in complex weather patterns and should provide good accuracy.

Using 10-fold cross validation, we found that 50 estimators was a good balance between overfitting and underfitting for random forest and adaboost. Specifically, we found that the accuracy dropped significantly after 50 estimators.

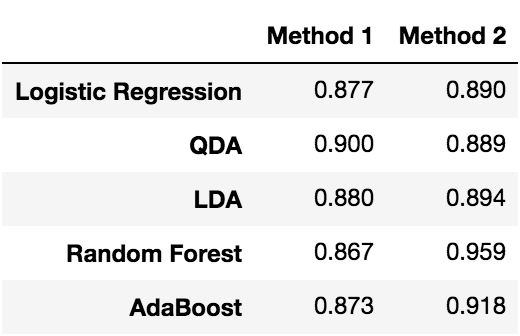
Here is the 10-fold cross validation error for method 1:



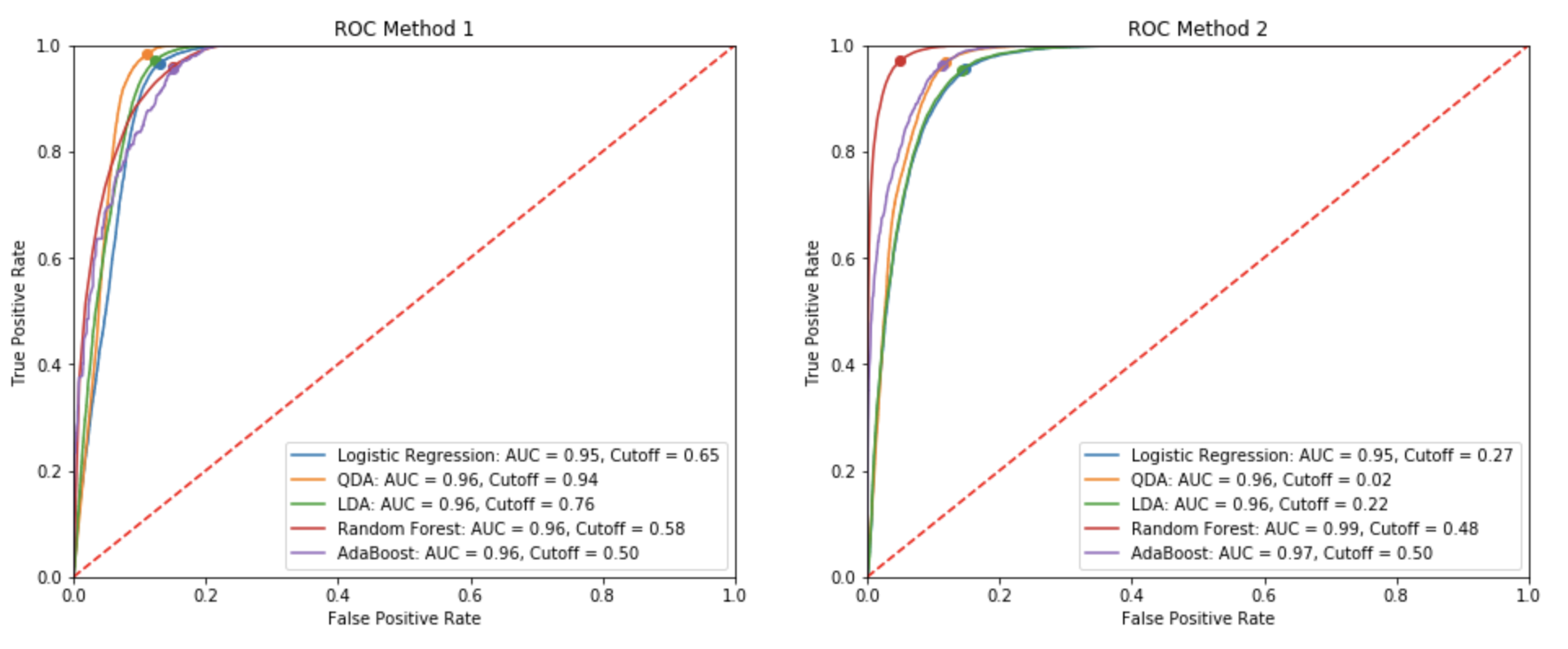
And here is the 10-fold cross validation error for method 2:



The values are pretty stable across folds which suggests that we are not overfitting.



Here we have the test accuracy of the classifiers. Since the data is n method 2 will generate a training set that’s similar to the testing set, we expect the accuracy to be higher. Going forward, we should use the accuracy reported by method 1, as it is a better gauge of how the model will perform given new data.

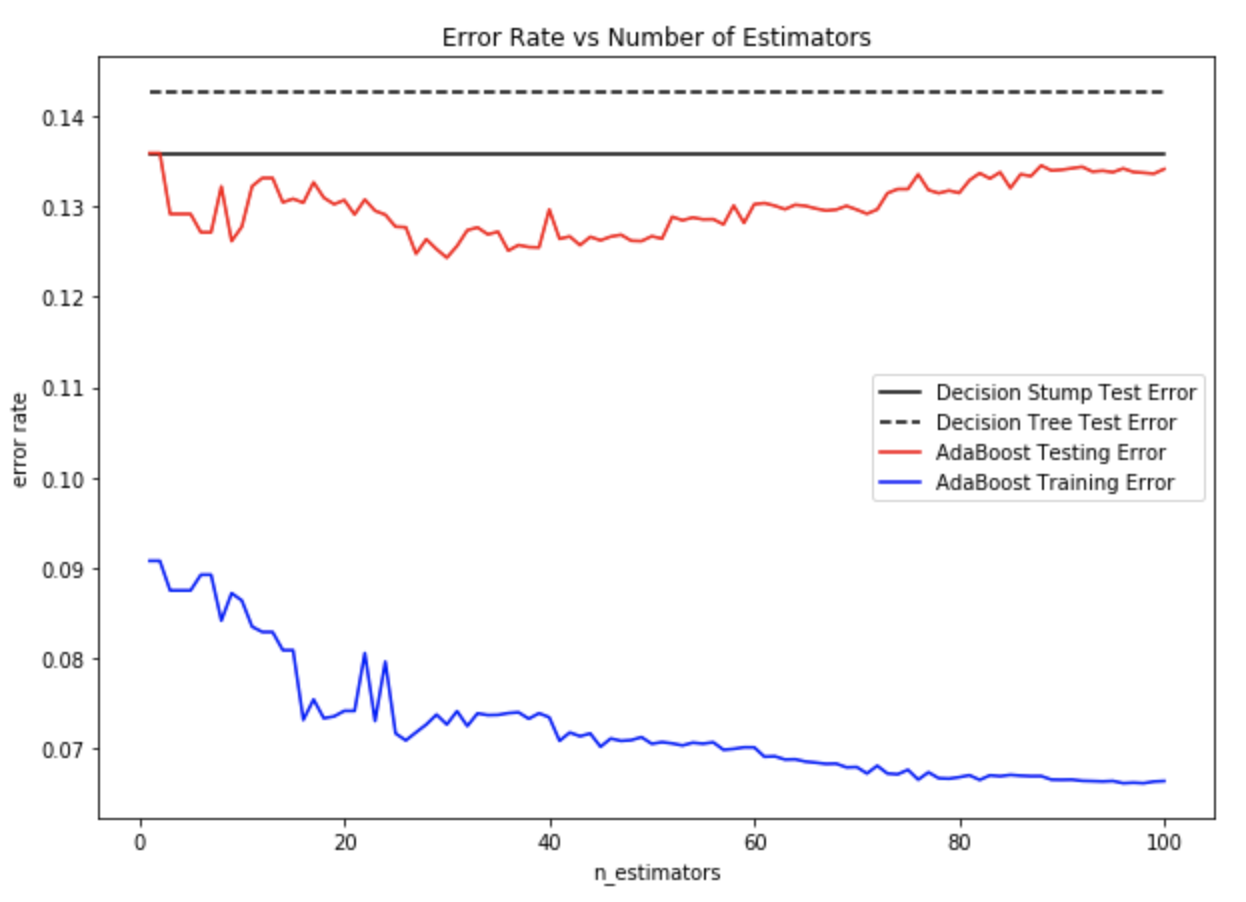


Above we have the ROC curves and cutoff values for method 1 and method 2. Again, we see method 2 performing artificially better than method 1, with generally higher AUC scores. Going forward, we will use the ROC curve and cutoff values from method 1. The purpose of this model is to track the change of cloud coverage over time. This means that we care about the ratio of cloudy to clear pixels so it's important we classify both correctly. Hence, we weigh correctly identifying cloud pixels the same as correctly identifying clear pixels. This is achieved when accuracy is maximized which is when true positive rate is high and false positive rate is low. We find the cutoff threshold to be the threshold that maximizes

True positive rate - false positive rate

**Diagnostics**

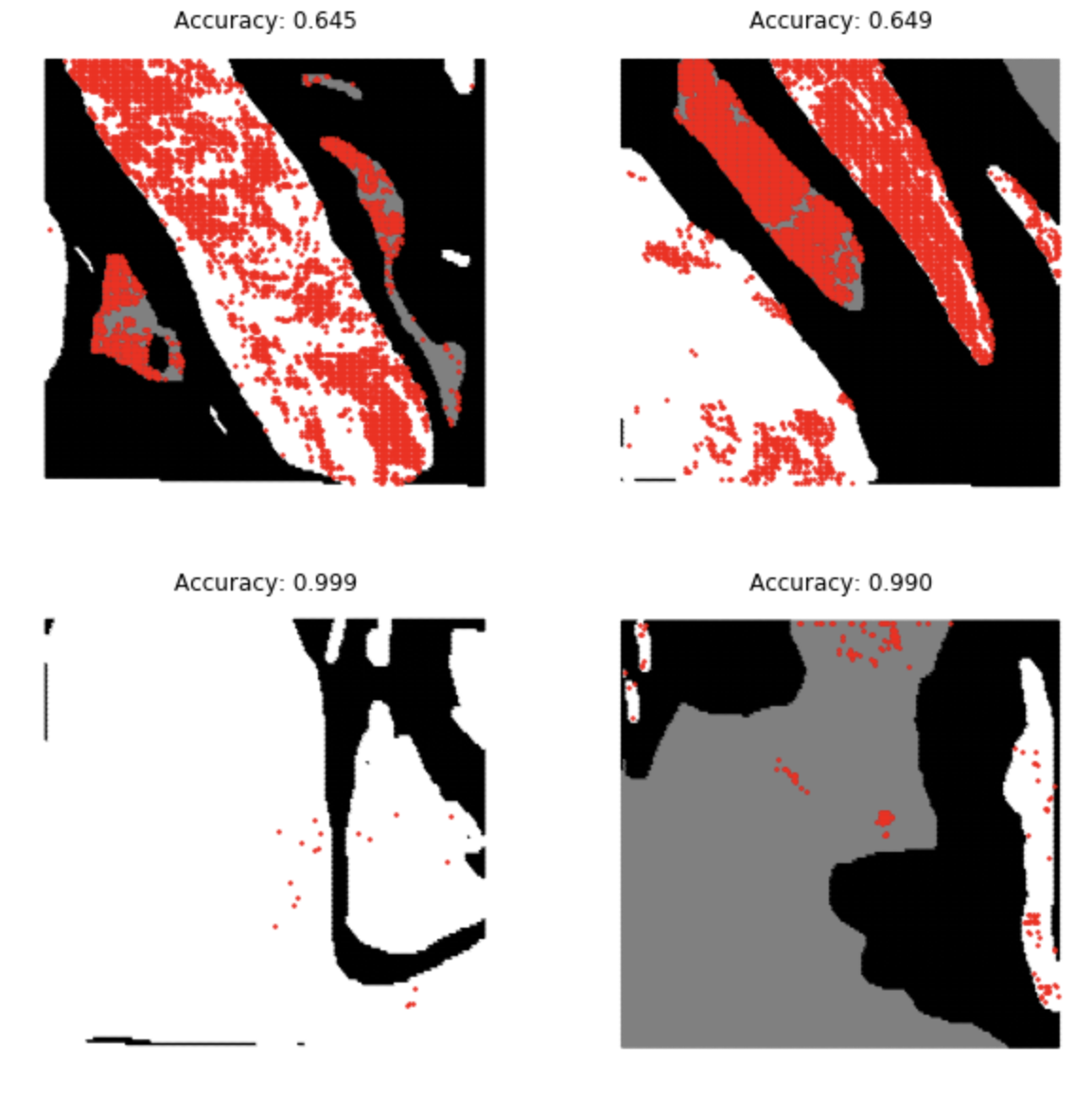
Analysis of AdaBoost



Above is a chart comparing the testing errors of a Decision Stump (max depth = 1), Decision Tree (max depth = 9), and AdaBoost classifier. It's really surprising that the decision stump performed worse than the decision tree. This suggests that this problem is one where it's easy to overfit so it might not be good to use a complex model. This theory is further validated by the testing error of AdaBoost as the number of estimators increases. We can see that the testing error starts to get worse after 40 estimators. Since AdaBoost increases performance by training new classifiers on previous mistakes, the fact that it doesn't perform better after 40 estimators suggests that when the model is wrong, it is wrong in the same way. This suggests that to improve performance, we need better/more data not a better model.

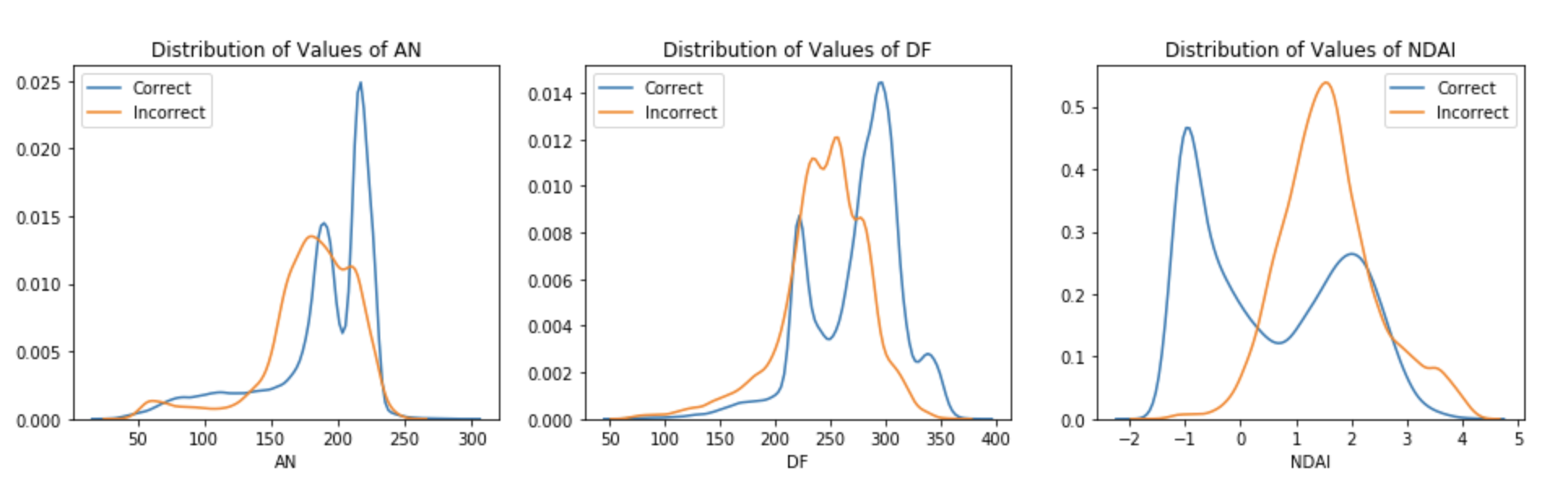
Patterns in Misclassification

Our best classifier is the QDA classifier which achieved an accuracy of 90%. Below are some examples of regions that the classifier performed well on and regions that it performed poorly on. All the pixels that are misclassified are in red.



*Figure 3: classification error of regions*

It appears that the regions the classifier performed well on are uniform--either mostly clear or mostly cloudy. Whereas the regions that the classifier performed poorly on are regions that have more complex cloud coverage. There is a 25%-50% drop in accuracy when classifying complex regions compared to uniform regions. One theory is that uniform regions have values that are more "normal" and have less noise sense the weather pattern is more stable. Complex regions, on the other hand, would have more extreme values / weird value combinations since the weather pattern is more unpredictable.



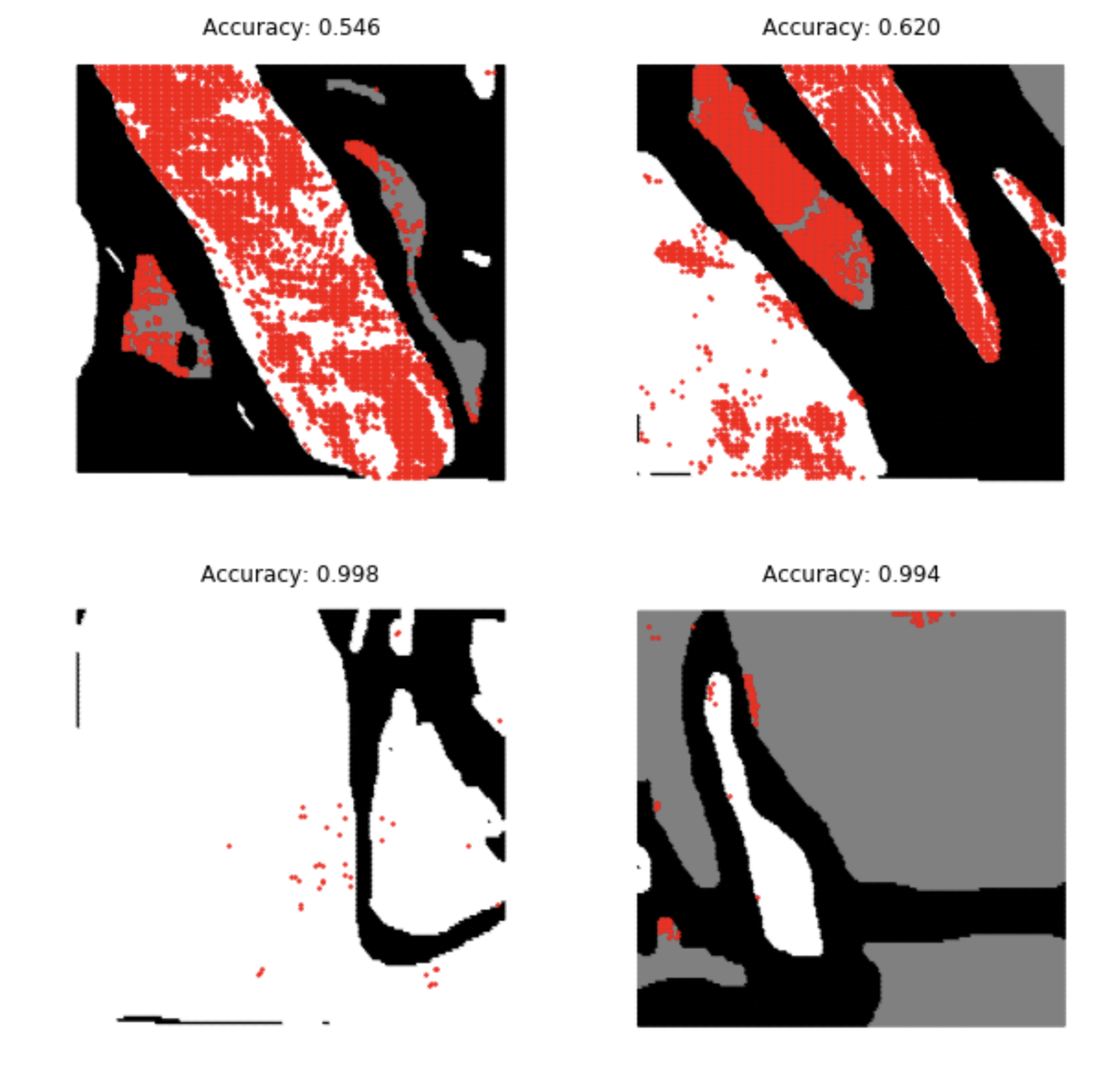
We can explore this theory further by looking at the distribution of our 3 features split by when the classifier is correct and when it's incorrect. The correct predictions tend to have higher values of AN and DF and lower values of NDAI. This makes sense as AN and DF represent the brightness of each pixel, so darker clouds can get misclassified. A cloud pixel could be darker than normal due to lower levels of sunlight or a weird angle of reflection due to its shape. We think that this issue can be solved by taking into account the pixels around it, since it is less likely that all the surrounding pixels suffer from the same problem. Furthermore, the classifier tend to be wrong when the values of NDAI are high. According to the paper, larger NDAI values suggest presence of clouds. Since the distribution of NDAI for correct predictions are lower, NDAI seems to be more useful when predicting clear pixels and adds error when predicting clouds. This is especially present in top right region in figure 3. An entire block of clear pixels were classified as cloudy. This might be a really high mountain peak which was falsely identified as a cloud due to its high NDAI value. In general, we think that NDAI suffers from the same problems as AN and DF and classification can be improved if we also used the surrounding NDAI values during prediction.

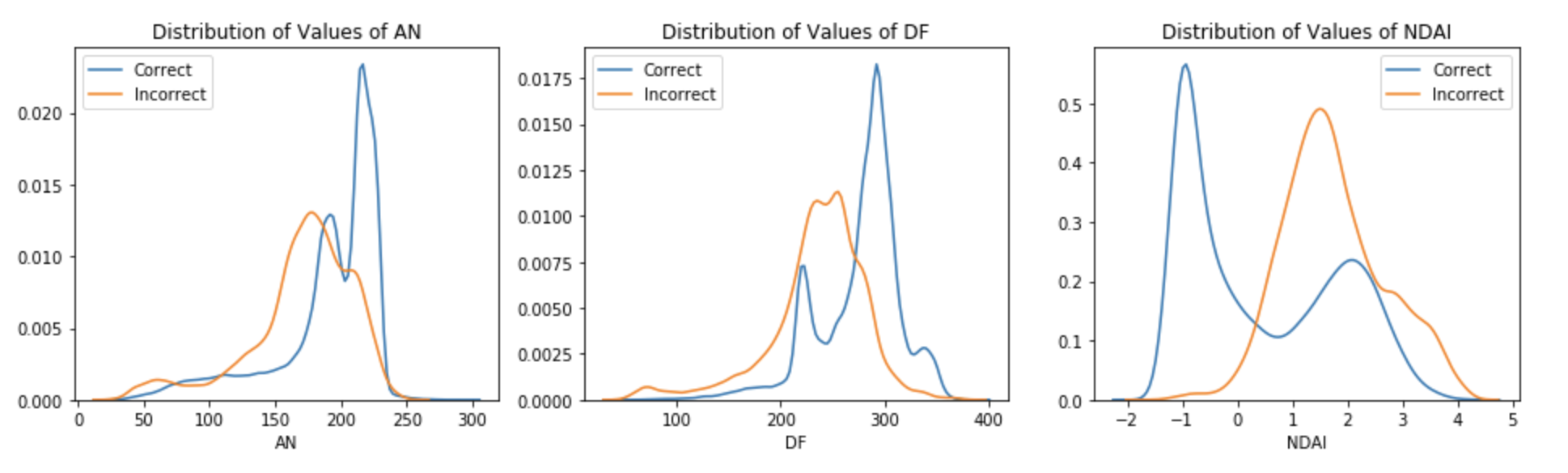
Improvements

We think that a CNN, which takes into account the surround values of each pixel would be a better classifier. However, it might be a lot slower and given the sheer amount of data MISR collect, it might not be the best trade off for slightly better improvement. Both QDA and Adaboost performed fairly well in detecting clouds when tested under method 1 of splitting data. QDA had 90% accuracy and AdaBoost had 87% accuracy on the testing set. Since method 1 tests on entirely new regions, we believe that both models will perform well when given new data.

Comparison to Method 2

Let’s look at the same analysis on method 2:





The data doesn't change when we compare the two methods of splitting. This suggests that the error is systematic. We expected method two, which splits the data randomly to perform better. Since the pixels are not i.i.d., the test set from method 2 is less likely to contain new examples. However, since the performance is the same this means that even when trained on very similar data, the model could not improve its testing accuracy. This suggests that we need more features to improve the accuracy of the model, rather than just a more complex model.

Conclusion

In conclusion, we think we can improve the model’s performance when predicting a pixel by taking into account the values of the surrounding pixels. But overall, both QDA and Adaboost performed fairly well in detecting clouds when tested under method 1 of splitting data. QDA had 90% accuracy and AdaBoost had 87% accuracy on the testing set. Since method 1 tests on entirely new regions, we believe that both models will perform well when given new data.

**Contribution and Reproducibility**

We split the work equally. Here is the github link to our reproduce our work: