Lab 4 - Stat 215A, Fall 2018

Cloud Detection in the Arctic

Zihao Chen, Yutong Wang, and Todd Faulkenberry

November 9, 2018

1 Introduction

In the age of climate change, the increasing concentration of carbon dioxide in the Earth's atmosphere has important, far-reaching impacts on the temperature of our environment and thus how our environment functions. Understanding these effects is of the utmost importance for scholars, scientists, and policymakers as we strive to create a more sustainable society. This understanding is especially important in the Arctic Circle, where many scientific models expect some of the sharpest increases in atmospheric carbon dioxide levels and subsequently surface air temperatures. By measuring Arctic cloud cover – effectively, a barrier that helps mitigate surface air temperatures – scientists can gain a more comprehensive understanding of the state of the Arctic climate. The value of such an understanding inspired Yu et. al's 2008 on paper on cloud detection, which uses a handful of both natural and artificial measurements to predict which sections of Arctic satellite images correspond to clouds and which to ice.

This paper expands on Yu et. al.'s work by exploring some of the data they collected to create their model. We immediately apply our analysis by creating our own classification model to determine from the variables provided whether a surface is cloud-based or ice-based. We constructed three types of models for cloud detection: Logistic Regression, Random Forest, and AdaBoost. We found our Random Forest model to be our most effective classifier in terms of cross-validation accuracy and variance.

2 Exploratory Data Analysis

2.1 The Data

The data analyzed in this paper was collected in 1999 from the Multiangle Imaging SpectroRadiometer (MISR), a sensor on board NASA's Terra satellite. MISR's sensor contains nine slightly overlapping cameras, with each camera recording the Earth's surface from a different angle across four spectral bands. While MISR pioneered a new type of remote sensing, the sheer amount of data it creates presents a logistical problem for scientists. In aggregate, the nine MISR cameras cover a 360 kilometer stretch of the Earth's surface. This global coverage creates a lot of data: MISR collects up to 9.0 megabits/second, with an mean rate of 3.3. Using an expert to label this information is inefficient and ineffective. Developing an algorithm that can accurately identify the type of surface from the angular radiances in MISR data would help scientists make sense of how the earth's surface air temperature is changing in a much more effective manner.

While the original data contained nine different angular radiances, the data in this paper contains five, with the four left out being those facing the aft direction. In addition to the five included angles (DF, CF, BF, AF, AN), the data contains coordinates, as well as a label designated from an expert stating whether the observation is a cloud or ice-based surface. The data also contains three measures created by scientists to predict surface. The first, CORR, represents the average correlation of radiation measurements between two different angles (AF/AN and BF/AN.) CORR is supposed to help identify either cloud-free surfaces or very low altitude clouds that blend in with the ice. The second, SD, represents the standard deviation within groups of MISR and

helps identify smooth surfaces. In a prediction model, CORR and SD are intended to work in tandem: While CORR may generally identify clouded surfaces, SD helps distinguish between low-cloud and actual ice surfaces. The last variable here, NDAI, measures the difference in average radiation measurements from the DF and AN angles. This is because, over clouds, the DF angle is expected to records higher radiation than AN. This means the gap between DF and AN increases over clouds, so large values of NDAI suggest a clouded surface. Ultimately, these three features are the basis for the prediction model described in the paper.

While the data here presents many challenges, cleanliness was not one of them. We received the data perfectly clean, with no missing values or other irregularities. This structure allowed us to focus our efforts on this project entirely around creating the best model.

2.2 Plotting Expert Labels and Main Features

We began our EDA by looking at CORR, SD, and NDAI to investigate their respective relationships with the expert labels provided in the data. Given how the authors described the functions of the features in the paper, we expected to observe some unique qualities. For CORR, we expected it to be able to identify ice but to struggle between separating ice from low clouds. For SD, we expected it to ably identify edges (i.e. where ice begins and clouds end or vice versa.) For NDAI, we expect high values for clouded surfaces. The below graphs show the label (Figure 1), CORR (Figure 2), SD (Figure 3), and NDAI (Figure 4) for each of the three images we received:

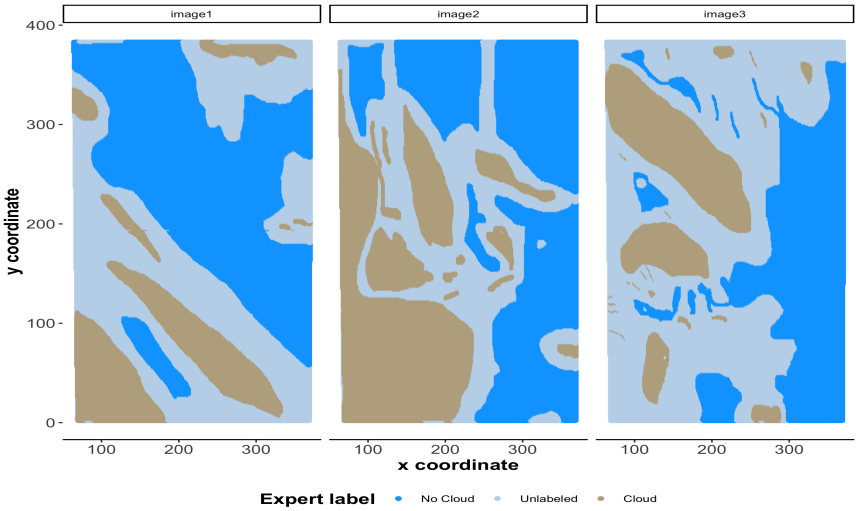


Figure 1: Expert Labeling of Clouds

When comparing the expert labels to the three features, all three of our priors hold. It becomes clear that CORR does not clearly discern ice and clouds. We see that SD clearly defines the edges of the clouds in the expert labels, but does a bad job distinguishing the body of both clouds and ice. In addition, we see high values of NDAI on clouded surfaces. One clear trend emerges from these visualizations: NDAI appears to be the best general predictor of cloud surface. NDAI gets the shape and structure of the clouds in each image much more clearly than either SD or CORR. While NDAI certainly has its deficiencies – for example, it doesn't capture the nuances of clouds (especially clouds with holes) particularly well – its shortcomings are much less pronounced than CORR (which cannot distinguish well between low-hanging clouds and ice) and SD (which cannot identify bodies of clouds and ice well.)

An additional benefit of NDAI in comparison to SD or CORR is its clearer separation of values. The density graphs below show the distributions for NDAI, SD, and CORR between expert label values equal to -1 (corresponding to ice) and 1 (corresponding to cloud) for all three images combined. As we can see, distributions for SD and CORR overlap much more heavily than the one for NDAI. This overlap makes it difficult to distinguish which CORR values and which SD

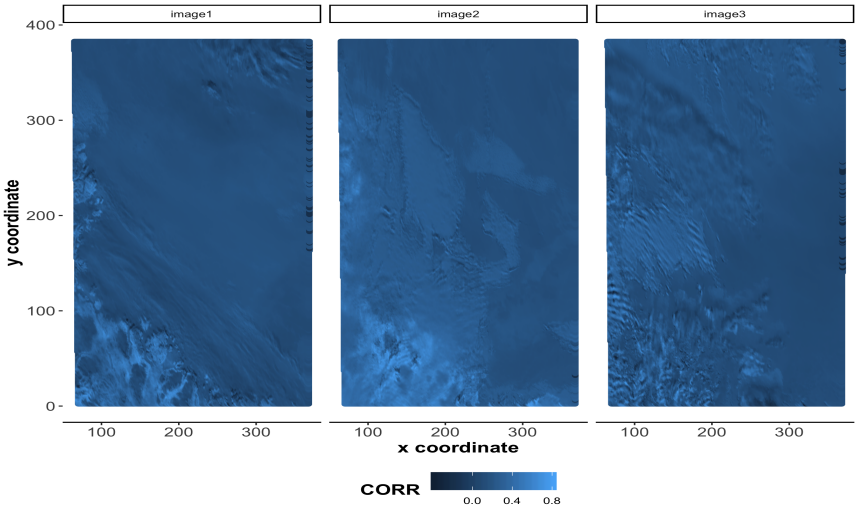


Figure 2: Corr

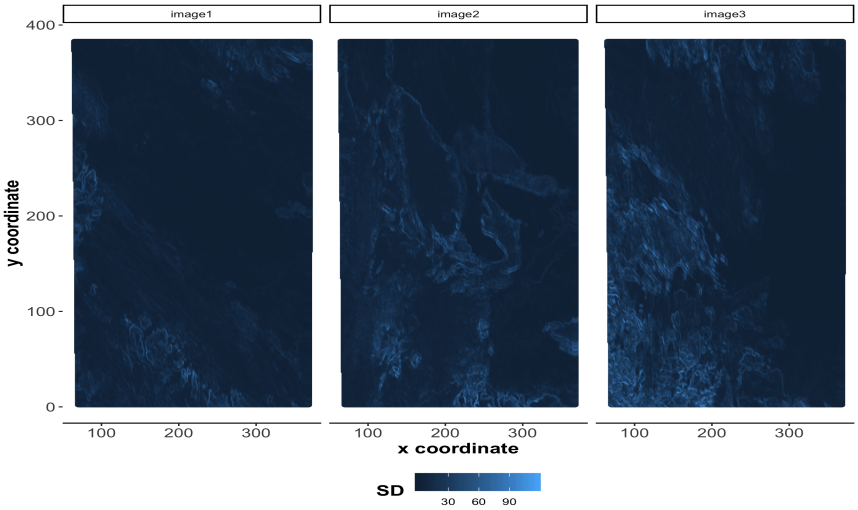


Figure 3: SD

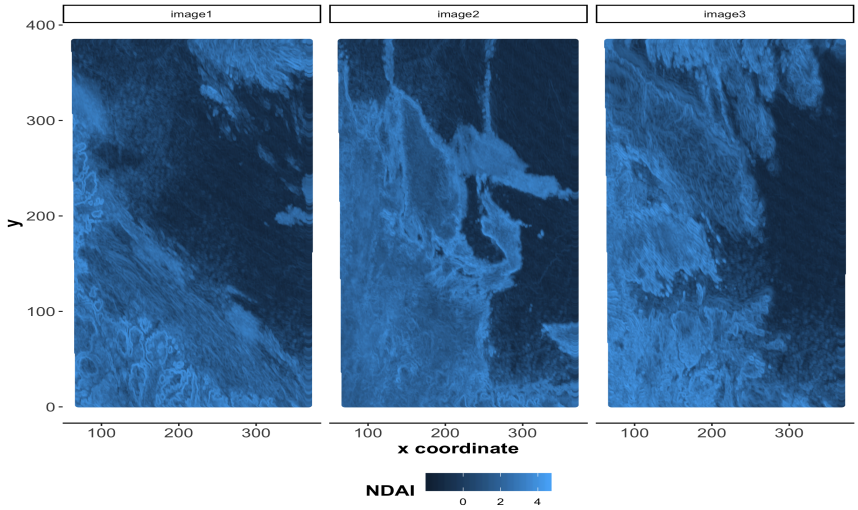
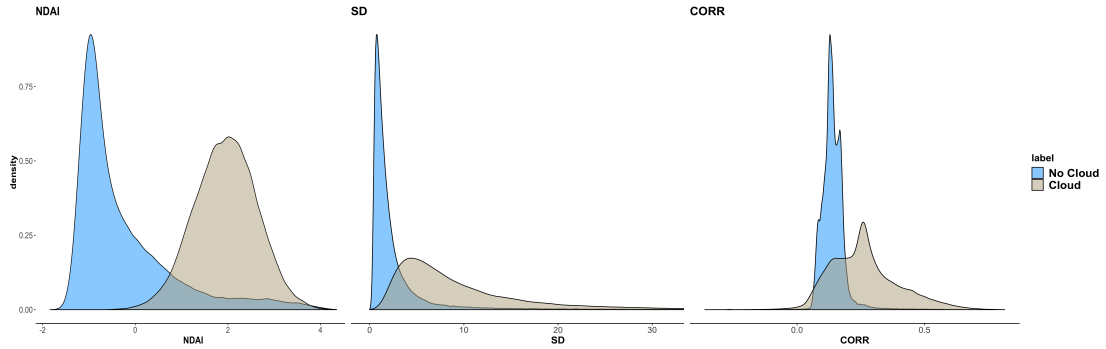


Figure 4: NDAI

values suggest specific surfaces. With NDAI, however, the minimal overlap makes it much easier to identify how NDAI reflects the type of surface.



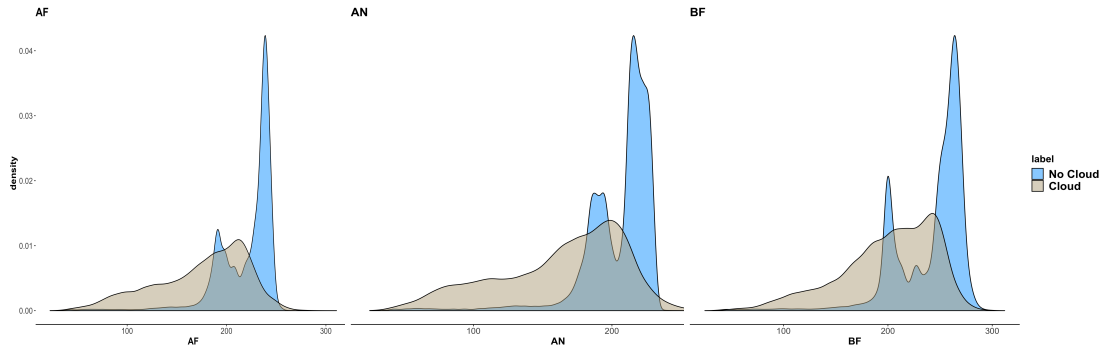
(a) NDAI (b) SD (c) CORR

Figure 5: density plot of NDAI, sd, corr

At the same time, while the above graphs do suggest that NDAI is a particularly good indicator, they simultaneously do not suggest that CORR or SD would be particularly bad. In both, we see some separation of distributions. For SD, the distribution for ice is much more compressed around the central tendency and has a lower average value. For CORR, the fat tails of the cloud distribution and the separate mode means it could be used to identify surfaces of a certain CORR value with strong accuracy. At the same time, however, the fact that SD and CORR aren't quite as definitive as NDAI in their predictions led us to then investigate the various angular radiances to see which would pair well with NDAI in our model.

2.3 Considering Other Factors

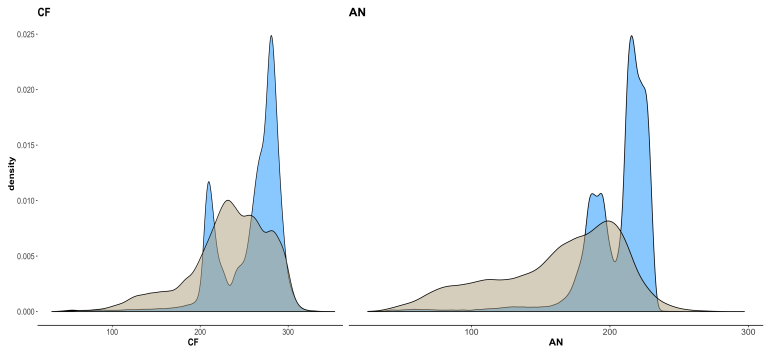
To begin determining which angles would work best in our model, we initially visualized them the same way we visualized NDAI, SD, and CORR – by plotting all points for each individual image on an x-y coordinate map. This approach was not as helpful here. Unlike the original measures, it was not obvious from these visualizations which angles performed best. As such, in order to save space, we have omitted those visualizations in this paper. We have included density graphs similar to the ones above. As a reminder, to create these graphs, we combined data from all three images into one data set, removed all ambiguous observations (i.e. label = 0), and plotted their densities. The results are in Figure 6 and Figure 7:



(a) AF (b) AN (c) BF

Figure 6: Density Plot of AF, AN, BF

Some of the angles have strong overlapping densities, making them difficult candidates to include in a model. For example, CF and DF both have the lion's share of their observations between 200 and 300 units. This reality makes it initially unclear how these angles could be effectively



(a) CF (b) AN

Figure 7: Density Plot of CF, AN

	label	NDAI	SD	CORR	DF	CF	BF	AF	AN
label	1								
NDAI	0.758	1							
SD	0.436	0.647	1						
CORR	0.551	0.535	0.407	1					
DF	0.011	-0.164	-0.197	0.148	1				
CF	-0.288	-0.438	-0.407	-0.229	0.850	1			
BF	-0.448	-0.571	-0.491	-0.518	0.670	0.919	1		
AF	-0.507	-0.612	-0.514	-0.684	0.538	0.826	0.962	1	
AN	-0.505	-0.609	-0.507	-0.746	0.489	0.780	0.926	0.982	1

Table 1: Correlation Matrix of Features

used by themselves in a model without some adjustment to the measurement or attachment to a separate variable. BF suffers from some of the same problems that CF and DF do. The overlap here, however, is not as dramatic. In fact, BF's mode for ice surface, as well as a good portion of the ice distribution, exist more or less outside of the cloud distribution. AF and AN fall closer to BF than they do to CF and DF in terms of separated values. While both the ice and cloud distributions certainly overlap, the cloud distribution is much more spread out and contains plenty of low values. Meanwhile, the ice distribution for both has most of its values at the end of the cloud distribution, meaning plenty of the values could be identified with relatively few misclassifcation errors.

To further flesh out which variables to ultimately use, we then looked at a correlation matrix of the variables. We did this for a handful of reasons. First, we wanted to see if the numbers corroborated our visual hunches, i.e. see if we can easily identify which variables most closely correspond to label. Because label is a discrete variable, running a correlation basically tells us what percentage of each variable is cloud . This means if the correlation is negative, that tells us what percentage of each variable is ice. Second, we wanted to know how the variables correlated with each other. This would be useful for our model to keep us from considering variables that are highly correlated, thus introducing problems of collinearity.

The correlation matrix in Table 1 backs up some our initial claims while also producing some mildly surprising results. First, our expectation that DF and CF would be bad predictors is confirmed - they correctly identify the surface only 10% and 28% of the the time, respectively. Additionally, it is clear that, like we expected, NDAI is the best predictor, getting it right over 75% of the time. The rest of the results are relatively close together, with SD, AF, AN, and CORR all falling between 43% and 55%. CORR has a higher correlation than SD, something we expected given the more spread out distributions in the CORR density graphs. AF and AN both having better correlation than SD is a mild surprise, but one that makes sense when we revisit the graphs given this information.

In terms of correlations with each other, we should take notice of a few variables. AF and AN have almost an identical correlation with label, suggesting that only one of them needs to be included in a model. In terms of the artificial features, none of them have very high correlations with any of the angles, with the highest coming in around 60%. This is good for the model - the differences suggest that the angles reveal somewhat different information from the other features. One thing to keep in mind, however, is the high correlation between angles. For example, AF and AN have a correlation of .98, AF and BF have a correlation of .96, CF and CF have a correlation of .91, etc. This means we need to be careful with the angles. We likely do not need to include all of them in our model in order to gain the insights we want, as some of them display strong collinearity with others.

3 Modeling

Following our EDA, we began to build our classification using the combined image data. In creating the infrastructure necessary to construct and run our models, we grappled with some tough questions around constructing our test set, designing our cross-validation procedure.

Test Set Before we begin constructing our model, we must separate our data into a training set and a test set. This split is necessary for us so that we can later test our created model on data with which it has not interacted. This will ultimately allow us to gauge our model's effectiveness. As such, the primary goal while separating your data into a training and test set is to create a test set that will best reflect the classification model's performance on unseen or unknown data. Initially, given the three images provided, one may think it natural to simply put two images in your training set and one in your test set. This type of separation can negatively impact your test set, however, if there exists a biased distribution of data in the image. This bias may interfere with your classification model, leading to an analysis that doesn't accurately reflect your model's performance. Due to the problems this approach can introduce, we opted instead to divide all three images into many, many blocks and then randomly choose some blocks from each image. We believed this the best way to not introduce any bias into our test set so we can have a better understanding of our model in the end.

Cross-validation Before we discuss our cross-validation procedure, let's clarify the terminology we will use throughout our discussion. K-fold cross validation means dividing the data into K folds. Each time we run cross-validation, we fit the model using $K-1$ folds. We refer to these folds as the *training* set. We then use the one remaining fold to evaluate the performance on our training set. We refer to the remaining fold as the *validation* set). The training set and validation set from cross-validation are separate from the aforementioned training set and test set of our data. In short, cross-validation is something we do within the overall training set.

In designing cross-validation, there are multiple concerns to consider. First, we considered whether we wanted to create a validation set on which we can eventually test our training sets to see how strongly they predict label. Keeping a validation set is not required, however, and deciding to use one depends heavily on one's ultimate goal. If our goal is to build the best possible classifier from the available data, we should maximize the amount of data we feed into our cross-validation. This would mean using all our data and forgoing a validation set. At the same time, if our goal is to build a classifier with an accurate and tested performance estimate, then we should create a validation set and leave it untouched until the end, after we find the best classification model from the training set. This method is particularly useful if, for example, we only plan to deploy our classifier if its performance meets a certain threshold. Ultimately, we decided it would be best to have a validation set so we can test our results and a measurable estimate of our model's performance instead of simply trusting that a model built on the maximum amount of available would work out best.

Next, we had to consider and establish our classification methodology. First, we decided to treat each pixel as a sample, with the variables (NDAI, CORR, SD, DF, CF, BF, AF, AN) as features. This does create a large dependence among some samples, which violates some of the assumptions our model. This has a hugely negative impact theoretically, as this means our

accuracy is not theoretically guaranteed. In practice, however, we did not find this violation to be a problem. From an algorithmic viewpoint, this approach works fine due to the fact that the algorithms are just fitting the training data and thus can apply to dependent data. Regardless of this lack of impact, the theoretical implication of our decision should be noted. Additionally, we made the decision to include only the information about each particular pixel for each data sample, i.e. each pixel.Ideally, we would include, in addition to each pixel's information, the information of neighboring pixels that share the same label. This additional information would allow us to execute more sophisticated methods. For example, the introduction of spatial data like this would allow for a linear model that builds upon a convolution filter. Due to time constraints, however, we did not have the ability to execute this more complex and time-consuming model. Models like these could be subjects of future analyses.

Last, we needed to decide how to divide our data into different folds. A simple way would be to, of course, randomly divide data into the desired amount of folds without any other considerations. This approach can be problematic, however, because it causes the validation accuracy to be an incorrect estimate of the test accuracy during cross-validation. This is because simple random sampling will lead to a validation set with pixels that, in the original data set, will be adjacent to pixels in the training set. Because adjacent pixels tend to have similar features and labels, this leads to a training set that, in some sense, already contains the validation set. Because of the similarities shared between the training and validation set in this simply randomization, the validation accuracy of our training set will be overly optimistic. In order to avoid this predicament, we decided to divide the data set into four blocks according to quadrant so that adjacent pixels exist in the same folds. From there, we performed 12-fold cross-validation.

3.1 Logistic Regression

First, we ran logistic regression with NDAI, CORR, SD. Because these features were used heavily by the authors of the paper, we thought it would be a useful baseline for us to begin our analysis. We ran logistic regressions at different cutoff values, and then used cross-validation to pick the best cutoff value, i.e. the one with the highest cross-validation accuracy. The relationship between cross-validation accuracy and cutoff value for our logistic regressions is displayed in the following plot:

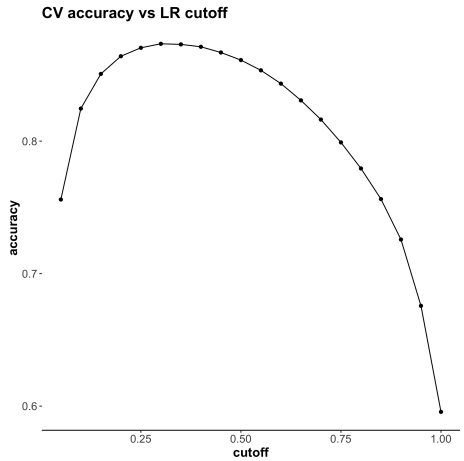


Figure 8: Logistic regression CV accuracy and cutoff values

As the graph above clearly shows, 0.3 is has the highest cross-validation accuracy and thus serves as the best choice of cutoff value. By choosing a cutoff value as 0.3, we obtained the mean training accuracy 0.897 and standard deviation 0.010. When we run cross-validation, we obtain a cross-validation accuracy of 0.873 and standard deviation 0.113. This will be our threshold to hopefully surpass in our upcoming classification models.

3.2 Random Forest

We then ran a random forest with the three original features, in addition to DF, one of the angular radiances. We did to this to see whether introducing the readings from one of the angles would produce a more accurate classification model. We ran our random forests with a different number of trees. Additionally, for the decision trees in our random forest, we randomly selected 2 features to grow them. We used cross-validation in order to tune the number of trees. The relationship between cross validation accuracy and the number of decision trees is as follows.

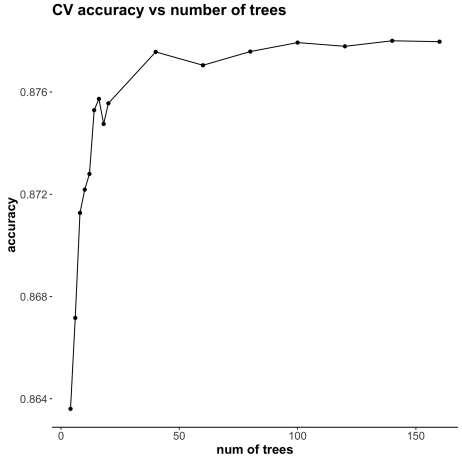


Figure 9: Random forest CV accuracy and the number of trees

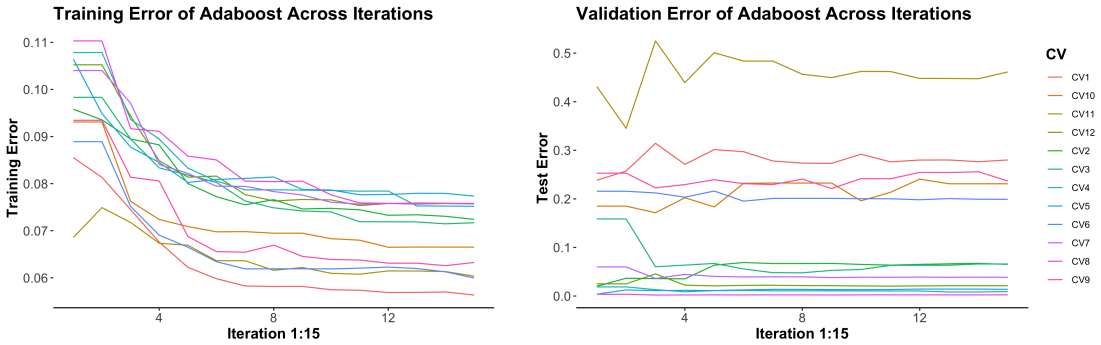
As the graph shows, we effectively reach our maximum cross-validation accuracy around 40 decision trees. While we see a marginal increase in cross-validation accuracy around 100 and 150 trees, the difference is very small and not worth the extra time and computation. Choosing the number of decision trees as 40, we obtain a cross-validation accuracy of 0.878 and standard deviation 0.112. This slightly outperforms the cross-validation accuracy we witnessed from our logistic regression, making random forest the better of the two models. With unlimited time, we would have fine-tuned some of the hyper-parameters in our random forest, such as the set of features or the the number of randomly selected features. Given time constraints, however, we decided to conduct another classification model to set if we could top our random forest cross-validation accuracy instead of spending time tweaking our random forest.

3.3 AdaBoost

Next, we used Adaboost with the same features that we used for random forest (NDAI, CORR, SD, and DF.) We thought Adaboost could be a useful classification model because of its natural ability to adapt adjust our "weak learners," thus leading to more accurate classification. At the same time, however, we recognize the problems Adaboost has with noise and outliers. Given Adaboost's problems with noise and outliers, we must keep in mind that it requires early stopping in order to be an effective algorithm. Because of this reality, the iteration number of Adaboost is a hyper-parameter.

Below are three graphs. The first and second are the training error for Adaboost across iterations for, respectively, the training set and the validation set. Additionally, the third graph plots the convergence of training error and test error in cross validation, and calculated mean cross validation error for all iterations. We created this graph in order to determine the best iteration number for our Adaboost.

As the third graph shows, AdaBoost achieves the best cross validation performance when the number of iterations is 4. Choosing the number of iteration as 4, we obtain the cross validation accuracy 0.872 and standard deviation 0.139. This accuracy falls short of the cross-validation accuracy for both our logistic regression and our random forest. This makes Adaboost our least effective classification model (by a small margin). In turn, this means that, between the three



(a) Training Error (b) Validation Error

Figure 10: Training and validation error convergence of AdaBoost

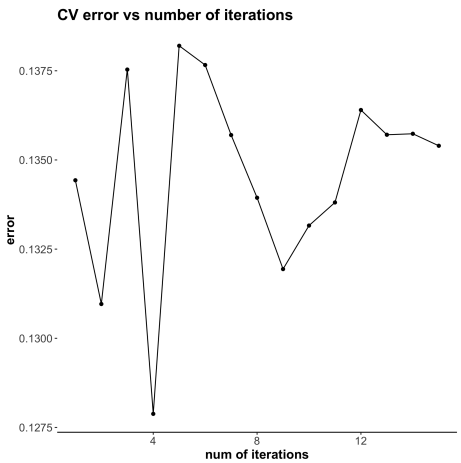


Figure 11: AdaBoost CV accuracy and the number of iterations

classification models we constructed, the random forest achieved the highest accuracy. We are confident that this accuracy could have been improved even further given some time to think about and improve on some of the hyper-parameters apparent in the model.

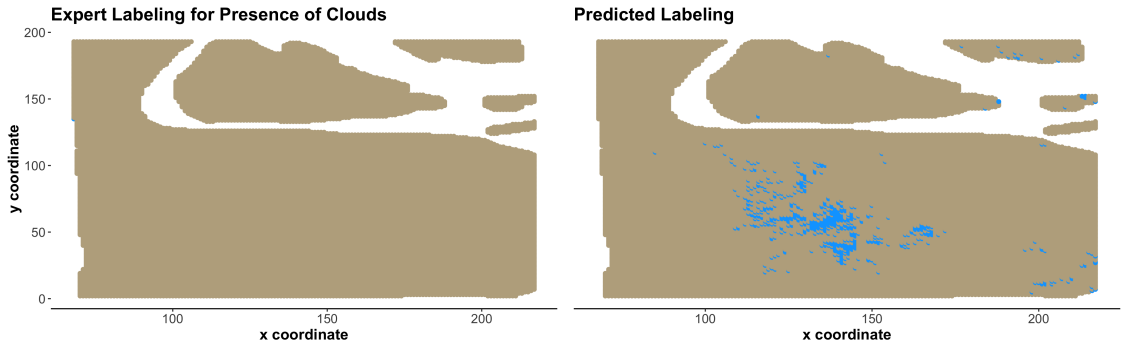
3.4 Misclassification Pattern of Random Forest

CV1	CV2	CV3	CV4	CV5	CV6	CV7	CV8	CV9	CV10	CV11	CV12
0.265	0.074	0.083	0.015	0.025	0.218	0.041	0.002	0.248	0.165	0.318	0.024

Table 2: Random forest misclassification Error of 12-fold Cross Validation

Now that random forest has been determined as our best model, we now must take a deeper look into its misclassification errors to see if we see anything systemic or part of a larger pattern. The misclassification test error of 12-fold cross validation is shown in table 2 above. There, we can see that the average misclassfication error is 0.122. In order to investigate where this error is occuring, we created graphs for the 12 quadrants on which we ran cross-validation. For each quadrant, there exists two graphs: One showing the actual label, and the other showing the predicted. This visualizations help us laser in on where the model is inaccurately classifying data. For example, Figure 12 shows the misclassification error in the bottom left quadrant in Image 2. As we can see there, some data points in the center of the plots are misclassified. Additionally, comparing Figure 12 with Figure 3, we can see the misclassified area has a much higher SD compared with other areas in image 2. Similarly, if we compare Figure 13b with Figure 2, we once again see a higher correlation in the misclassified area as well.

Figure 13 shows the misclassification error in the bottom right quadrant in Image 3, and the misclassified data points concentrated in the bottom of Figure 13b. Comparing it with Figure 4, it's difficult to distinguish cloud from non-cloud in the bottom of Image 3 in Figure 4. The high similarity of NDAI might contribute to the misclassification in Figure 13b. Therefore, we conclude that our classification model does a good job in making decision rules according to features. There may not exist, however, a satisfying cut-off method for binary classification in this problem.

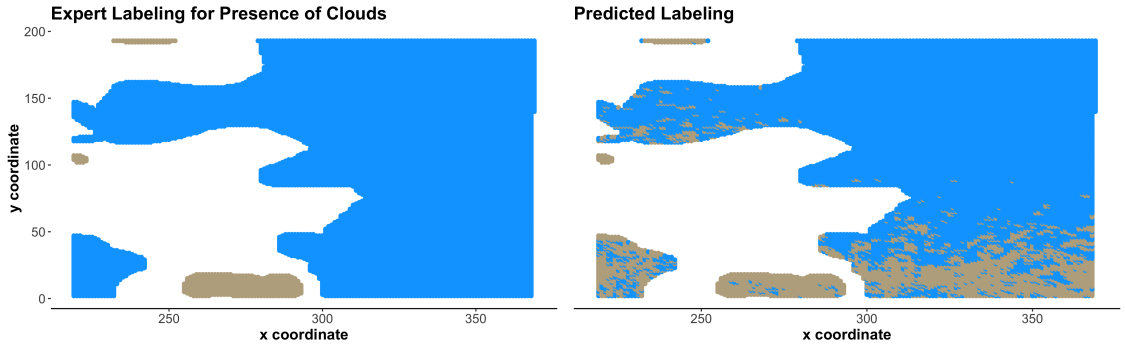


(a) Expert Label (b) Predicted Label

Figure 12: Misclassification: bottom left quadrant in Image 2

4 Conclusion

In this report we have used logistic regression, random forest, and AdaBoost to classify the cloud detection data from the Arctic. Between our three models, we found random forest to have the highest accuracy, even though all three models had a misclassification error within the same general range. Given that all three models perform similarly, this might suggest better methods should move beyond simply using NDAI, CORR and SD as features for each pixel. Possible



(a) Expert Label (b) Predicted Label

Figure 13: Misclassification: bottom right quadrant in Image 3

future directions include utilizing the spatially connected nature of the images and designing new features.

Another interesting quality to note is that the training accuracy has a smaller variance than the validation accuracy in cross validation. This arises given the significant differences in the training and validation sets. Because we created our cross-validation folds from image quadrants, it is unsurprising that each quadrant of the images can be quite different from others. Lastly, one limitation of our procedure is that the scarcity of the training data prevents us from learning models that are robust to the distribution shift. Given this constraint, we do expect that even our best classifier will be negatively impacted by unseen data and will have a drop in accuracy.