Comparison of Classification Algorithms on Arctic Cloud Images

Derek Tao(derek.tao@duke.edu), Minhui Jiang (minhui.jiang@duke.edu)

December 6, 2022

1 Data Collection and Exploration

1.1 (a)

Global climate change and climate models have become an increasingly controversial topic in the scientific and political communities. Of particular interest to researchers are the Arctic regions, where changes in the properties of clouds can result in further surface warming and thus strong sensitivity to increasing amounts of atmospheric carbon dioxide. One of the prominent challenges of studying climate change in the Arctic is the difficulty of assessing the impact that clouds have on future warming, since the similarity of electromagnetic radiation between clouds and snow-covered surfaces makes it difficult to detect clouds in the atmosphere. The current operational algorithm for cloud detection implemented by NASA's MISR technology does not work well over bright polar surface regions. Thus, the purpose of the study is to develop a computationally-efficient algorithm for cloud detection that combines classification and clustering methods. The algorithm will bypass the need for human intervention (expert labeling of pixels as cloudy or not) and perform accurately in polar regions.

The data in the study consists of 10 orbits of MISR over a single 360-km wide path on the Earth's surface, starting in the Arctic and ending in Antarctica. The path is divided into 180 blocks increasing in number labels from north to south, and a set of three blocks can be considered a data unit for this study. For the purposes of this study, only six data units covering the Arctic Ocean, Greenland, and Baffin Bay were used, for a total of 60 data units across the 10 orbits (3 of these were excluded in the analysis). Thus, from 57 data units, over 7 million 1.1-km resolution pixels were analyzed, with 36 radiation measurements collected for each pixel (four spectral bands across nine view zenith angles of the MISR cameras). In our particular data set, only five of the nine radiance angles are recorded, and the data are limited to just three total images.

In addition to pixel coordinates and radiance angles, four other variables were included in the data. Three of these variables are researcher-constructed features that can differentiate surface pixels from cloudy ones. The first is CORR, an average linear correlation of radiation measurements within a data unit at different view angles. The second is standard deviation of radiation measurements within data units, or SD. The third is normalized difference angular index (NDAI), which can be seen as a linear combination difference of the radiation measurements between different view angles. Finally, the fourth variable in the data is the hand-labeled expert label, which takes on a value of 1 for cloudy, -1 for not cloudy, and 0 for ambiguous.

The researchers used an ELCM algorithm (gives explicit predicted labels on pixels) and ELCM-QDA (gives predicted probabilities of cloudiness on pixels) to determine the presence of clouds. The results of these methods, in addition to those of the MISR operational algorithms, were compared against the expert labels, which can be viewed as a label for validation purposes. The results showed that the ELCM algorithm had higher accuracy and better spatial coverage than the MISR algorithms, and the ELCM-QDA algorithm performed strongly for assigning probabilities, particularly for gray areas. The study is significant because it shows the role of statisticians in data processing of Earth Science data. With the improvements of cloud detection algorithms from this study, there is no doubt that statistical analysis will play a large role in future climate models.

percentages	not cloud	unlabeled	cloud
picture1	37%	29%	34%
picture2	44%	38%	18%
picture3	29%	52%	18%
overall	37%	40%	23%

Table 1: Percentage for different expert labels

1.2 (b)

We had the percentages for three different expert labels in **table**[1]. Here we first computed it over each picture and then computed the percentages including all pictures.

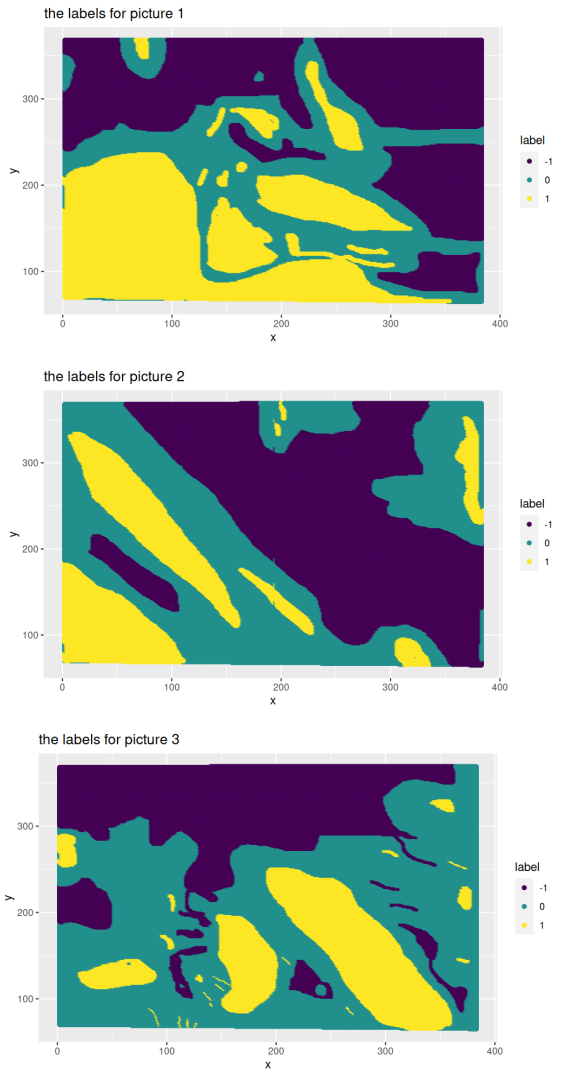


Figure 1: Regions based on expert labels

We think there are some spacial patterns in each of the picture, for example, a chunk of large number of neighboring pixels would be labeled as cloudy by the experts, and no other colors would be inside them, which corresponds to a large cloud covering the region.

We think that an i.i.d assumption for the samples is not appropriate. Since in each image, there is clearly some positive spacial relationship between neighboring pixels: For instance, if one pixel is labeled cloudy, then the neighboring pixels of it are more likely to be labeled cloudy. If the i.i.d assumption holds, these three labeled pictures would be a uniform mixture of the three colors, where no shapes or patterns would be recognized.

1.3 (c)

To check the pair-wise relationships between features, we plotted the Pearson correlation matrix as in **figure**[2]. We notice that CORR has quite high Pearson correlations with BF, AF, and AN. We think the main reason is that CORR is exactly defined by BF, AF, and AN (Shi, Yu, et.al, 2008).

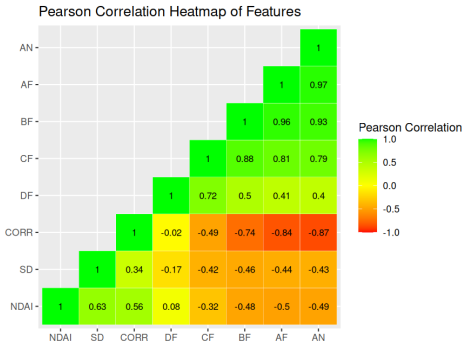


Figure 2: Pairwise comparison for different features

Then we explore the relationships between target labels and different features. We only selected those labels denoted as 1 (**cloud**) and -1(**not cloud**). From **figure**[3], we conclude that the distributions of the three constructed features (NDAI, CORR, and SD) for cloudy and non-cloudy pixels are quite different: the distributions for not-cloud pixels are more concentrated and have smaller variance than those for cloud pixels. Also, the means of these three features are different over cloud and not-cloud pixels.

Moreover, we checked the relationships between target labels and other features (Radiance from different angles). Since these features are essentially the same physical quantity, so we can plot their boxplots together, as in **figure**[4]. We concluded that there are some differences in the boxplots between cloud and not-cloud pixels. For example, not-cloud

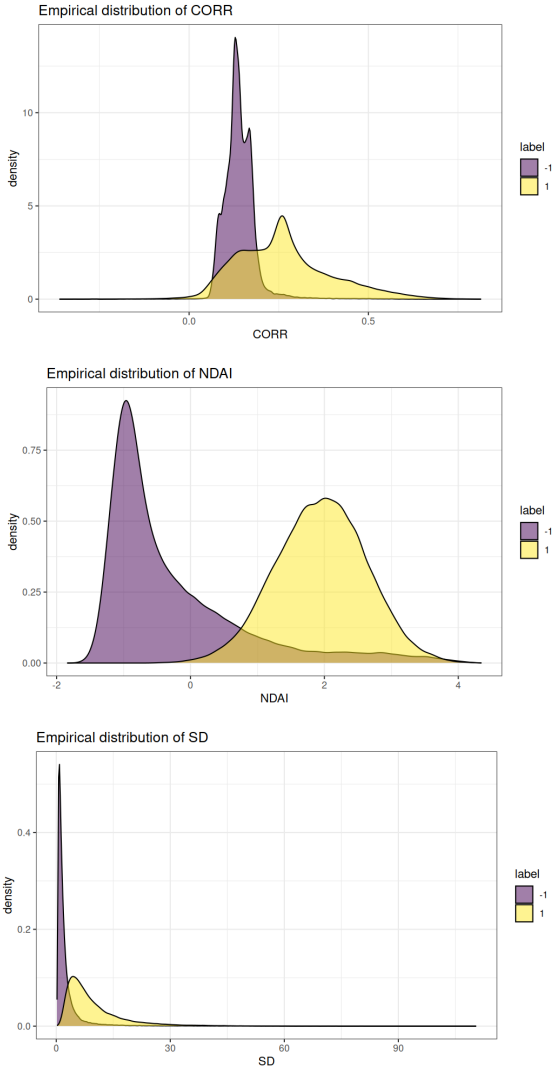


Figure 3: Density plots for CORR, NDAI, and SD

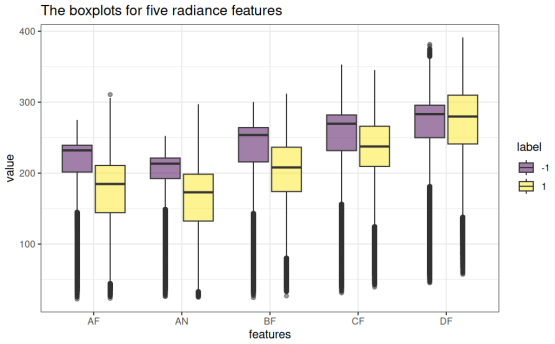


Figure 4: Boxplots for AF, AN, BF, CF,and DF

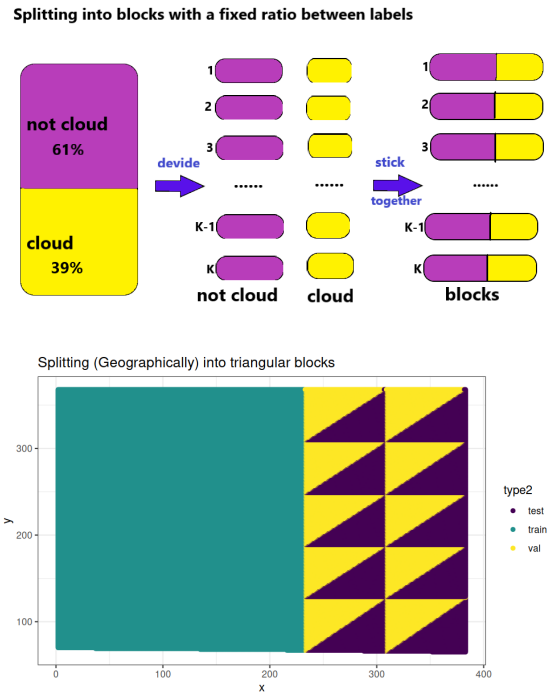


Figure 5: Two splitting methods for validation and testing

pixels are a little more concentrated in the five distributions of radiance, and the medians of not-cloud pixels and cloud pixels also differ to some extent. But generally speaking, these differences are not as obvious as in CORR,NDAI and SD.

2 Preparation

2.1 (a) Data Split

In the whole dataset, after deleting all unlabeled pixels, around 61% pixels are not-cloud, and around 39% pixels are cloud. Our first splitting method ensured that the validation and testing sets have the approximately same ratio between cloud and not-cloud pixels. As displayed in the upper of **figure**[5], we first divide the data into the cloud and not-cloud pixels, then we split both cloud and not-cloud data into K pieces. For every piece of cloud data, we stick it together with a piece of not-cloud data. After this, we had K pieces of data, each with an approximate ratio of 61%:39% between not-cloud and cloud pixels. Here we let $K = 50$, and randomly pick 10 pieces to be the testing set and 10 pieces to be the validation set. Thus the training/testing ratio is around 20%.

As we have observed in 1.2 (b), there exist certain spatial patterns in the pixels from three pictures: Considering data from the same picture, one pixel is likely to have the same expert label as its neighboring pixels, and these adjacent similar pixels form a block of same labels. As a result, the i.i.d assumption does not hold, so we should not ignore the locations of pixels and randomly pick pixels of a certain percentage as a testing set. As a result,

In our second splitting method as in the lower of **figure**[5], we divided the region into $K = 50$ triangular blocks and picked 10 triangles as the testing set and 10 triangles as the validation set. Thus the training/testing ratio is around 20%.

Both of the splitting methods manage to make the classification task non-trivial. The first method ensures that the ratio between labels is around the same level as the original dataset, and this further avoids the situation where all pixels in the testing set are cloud so that the classification problem is trivial. The second method takes the spatial non-i.i.d condition into consideration: testing sets consist of triangular blocks so they are not scattering all over the x-y plane, they also do not have a lot of pixels adjacent to those in the training set so that the prediction would be too trivial. In following problems, we would try both splitting methods and see which one would be better.

2.2 (b) Baseline

Accuracy	splitting method1	splitting method 2
testing	61.3%	72.9%
validation	60.7%	74.8%

Table 2: Accuracy of a trivial classifier over two splitting methods

After the two splitting methods are applied to the dataset, we dropped all pixels with a label of 0, which means that these pixels were not decided whether to be cloud pixels. Then we applied a trivial classifier that classifies all pixels to be -1 (**not cloudy**). The accuracy (defined to be $1-error\ rate$) over testing and validation sets is displayed in **table**[2]. We found that the results are quite different from the two splitting methods. It is understandable that the accuracy in testing and validation sets in the first splitting method is around 61% since we deliberately make this ratio choice. So the first splitting method is clearly non-trivial.

Accuracy	val2	test2
pic1	72%	79%
pic2	98%	89%
pic3	49%	33%

Table 3: Accuracy of a trivial classifier over the second method grouped by pictures

We explored the reason for the results of the second splitting method. It mainly came from geographic positions inside the three pictures. As displayed in **table**[3], the trivial classifier achieves largely differing results on the same split region over different sources of images. For example, in the testing set of the second splitting method, the accuracy is 79%, 89%, and 33% on data from picture 1, picture 2, and picture 3 respectively. This corresponds to the labeled regions in **figure**[1], as we can see, the percentage of cloud pixels in the right region of picture 3 is clearly higher than in picture 1 and picture 2.

As a result, we concluded that if we picked a region that happened to have very low percentages of cloud pixels in all three pictures, the trivial classifier would have very high accuracy on this particular region. Here we claimed that our second splitting method is also non-trivial since the trivial classifier does not achieve very high or very low accuracy. The trivial classifier did not achieve an accuracy higher than 75% under our second splitting method.

2.3 (c) First Order Importance

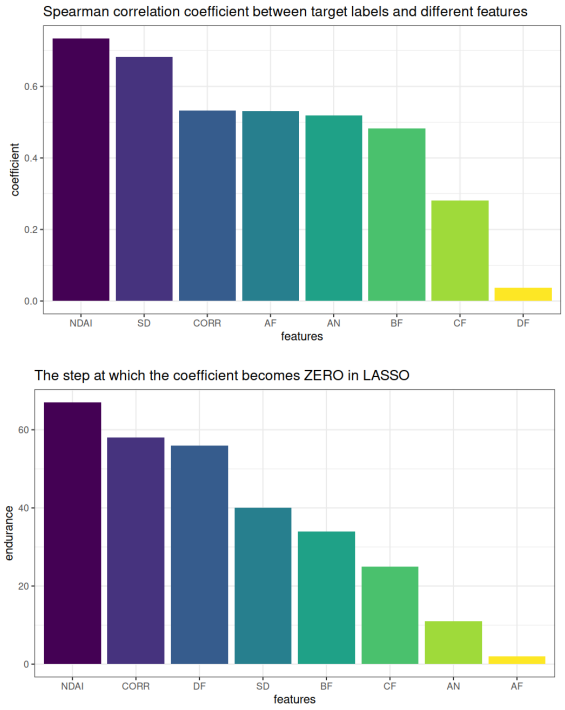


Figure 6: Spearman Correlation Coefficients and LASSO results for different features

Here we used LASSO over logistic regression to pick important variables, the results are as in the

lower of **figure**[6]. We also computed Spearman correlation coefficients between the target label and various features, and the results are as in the upper of **figure**[6]. The Spearman correlation coefficient is a generalization of the Pearson correlation coefficient to non-linear relationships.

From The Spearman correlation coefficients in **Figure**[6] we saw that the most important three variables are **NDAI**,**SD**,**CORR**. The LASSO results for selecting important features are **NDAI**,**CORR**,**DF**. The results seemed to be different, so we further checked the boxplots of DF and SD under different expert labels, it is clear from the **figure**[7] that pixels with cloud and not-cloud labels differ more in terms of SD: The SD distribution for not-cloud pixels is very concentrated, and the interquartile range of SD for two different labeled pixels do not intersect.We also saw from the LASSO results as in **figure**[8] that the two most important features whose coefficients vanished very late are **NDAI** and **CORR**, and all other variables vanished similarly early, so we think the result for the third most important variable from LASSO is not as convincing. In conclusion, we chose **NDAI**,**CORR** and **SD** as most important variables.

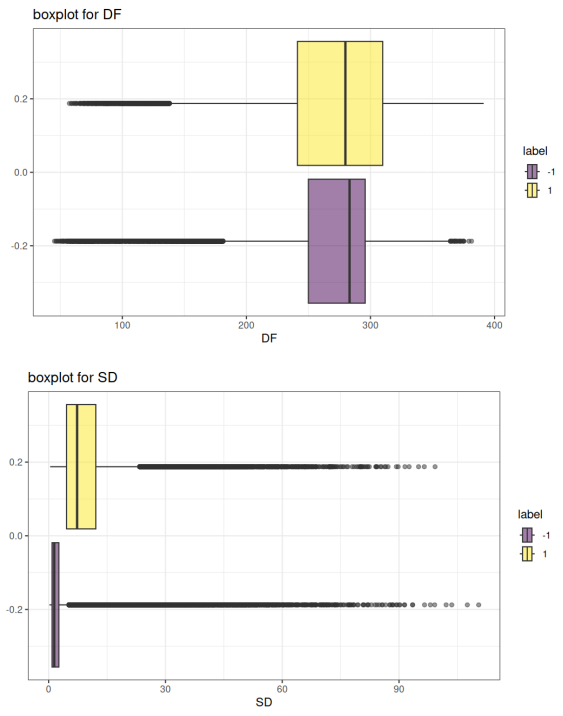


Figure 7: boxplots of DF and SD grouped by different expert labels

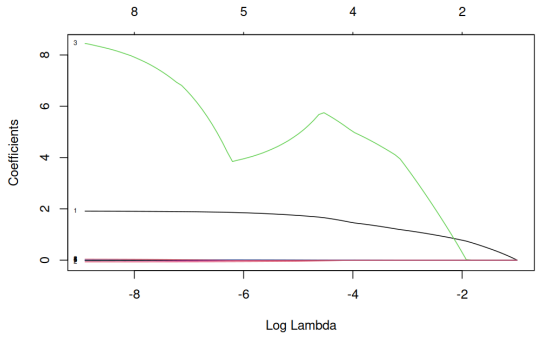


Figure 8: LASSO coefficients with increasing lambda

3 Modelling

3.1 a

In the modeling part, we have tried six models: logistic regression (LR), decision tree (DT), linear discriminant analysis (LDA), quadrant discriminant analysis(QDA), random forests(RF), and gradient boosting(GB).

Model Assumptions: In LR and LDA, it is assumed that the decision boundaries are linear, we think this assumption fits our dataset, since in the research by Yu et.al (2008), an ECLM algorithm that set separate thresholds on constructed features got good fitting results, which is a special case of linear decision boundaries. Moreover, LDA has a stronger assumption than LR, which says that the data are generated in different Gaussian clusters, we think this is appropriate because the pixels are whether cloud or not-cloud, and they are likely to have different data generative mechanisms. QDA further assumes the covariance matrix of different Gaussian clusters are different, which has weaker assumptions than LDA, so it is also appropriate in our opinion.

DT, RF, and GB are all flexible methods. DT assumes that the feature space can be divided into smaller rectangles, and we think this fits our model, the reason is that the ECLM algorithm (Yu, 2008) used thresholds on **SD**,**CORR** and **NDAI** and made predictions based on their logical combinations, which produced good results. This procedure is very similar to DT. RF is a decision-tree based classification method that seeks to minimize variance by building trees independently, and GB is a decision-tree based classification method that seeks to minimize mainly bias by building base

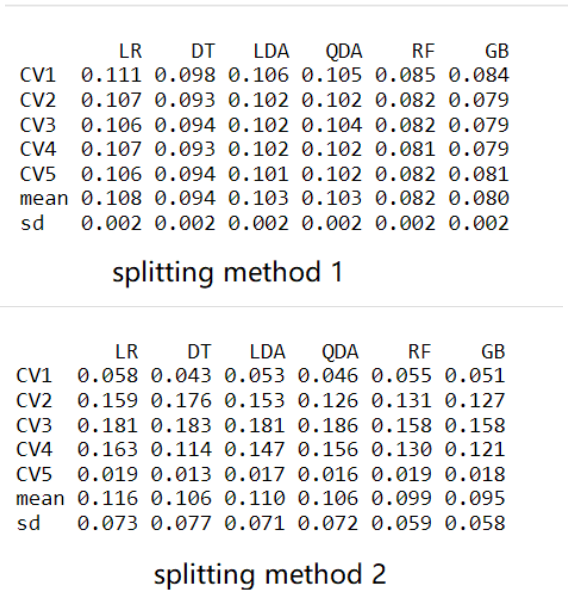
	LR	DT	LDA	QDA	RF	GB
splitting1	10.8%	9.3%	10.3%	10.0%	8.2%	8.5%
splitting2	9.4%	6.8%	8.4%	9.0%	6.3%	5.6%

Table 4: Misclassfication rates on testing sets under both splitting methods

learners (shallow trees) iteratively. Both RF and GB fit our dataset since the base learners (DT) are appropriate, and there are large amount of data for resampling.

Hyperparameters Tuning: In our six methods, only RF and GB required selecting hyperparameters. The mean CV error over 5-folds was used as a benchmark for selecting hyperparameters. The details of hyperparameters tuning are in part **(a)** of **4 Diagnostics**

After selecting hyperparameters, we merged the training data and validation data under both splitting methods, then we divided this merged training data into five folds ($K = 5$), and applied these six learning methods to obtain CV errors, the result is shown in **figure**[9]. After that, we trained the six models on the entire merged training set under both splitting methods, and then test our models on both testing sets under two splitting methods. The result is displayed in **table**[4]



	LR	DT	LDA	QDA	RF	GB
CV1	0.111	0.098	0.106	0.105	0.085	0.084
CV2	0.107	0.093	0.102	0.102	0.082	0.079
CV3	0.106	0.094	0.102	0.104	0.082	0.079
CV4	0.107	0.093	0.102	0.102	0.081	0.079
CV5	0.106	0.094	0.101	0.102	0.082	0.081
mean	0.108	0.094	0.103	0.103	0.082	0.080
sd	0.002	0.002	0.002	0.002	0.002	0.002

splitting method 1

	LR	DT	LDA	QDA	RF	GB
CV1	0.058	0.043	0.053	0.046	0.055	0.051
CV2	0.159	0.176	0.153	0.126	0.131	0.127
CV3	0.181	0.183	0.181	0.186	0.158	0.158
CV4	0.163	0.114	0.147	0.156	0.130	0.121
CV5	0.019	0.013	0.017	0.016	0.019	0.018
mean	0.116	0.106	0.110	0.106	0.099	0.095
sd	0.073	0.077	0.071	0.072	0.059	0.058

splitting method 2

Figure 9: 5-fold CV error rates for two splitting methods

Comments: It is clear that under all six classification algorithms, the means of the 5 CV misclassification rates under the first splitting method are lower than under the second splitting method, and the standard deviations of the 5 CV errors under the first splitting method are lower than under the second splitting method. Also, it is clear that under both splitting methods, RF and GB achieved lower average misclassification rates than LR, DT, LDA and QDA.

	LR	DT	LDA	QDA	RF	GB
splitting1	0.38	0.71	0.38	0.23	0.54	0.48
splitting2	0.36	0.68	0.36	0.25	0.56	0.51

Table 5: cutoffs of classification algorithms under different splitting methods

3.2 b

We employed the R package *ROCR* to plot two ROC curves for two different splitting methods, and the result is as in **figure**[10]. As we can see, under both splitting methods, the ROC curves of LR, LDA and QDA are below the ROC curve of DT. The ROC curve of DT is below the ROC curves of GB and RF. This also corresponds to our conclusion that GB and RF achieve better accuracies than DT, and DT's accuracy is better than LR, LDA and QDA.

We chose the cutoff value of each classification algorithm to be the value so that it minimizes the misclassification rate (achieves highest accuracy) on test sets. This was done for both splitting methods, and the result is shown in **table**[5], the verticle lines in **figure**[10] are chosen cutoff values for different classification algorithms. An interesting finding is that under this criteria for picking up cutoffs, the DT algorithm had an extraordinarily high cutoff value (around 0.7). LR, LDA have similar cutoff values (around 0.36). RF and GB also have similar cutoff values (around 0.45). QDA has a cutoff value about 0.24. This also corresponds to our finding on accuracy in **part**(a)

3.3 c

Since the ROC curves are very close to each other, so we further came up with the area under curves (AUC) of each method under both splitting methods. This measure is independent of different choices of cutoffs. The result is as in **table**[6]. We can see that RF and GB have similar AUCs, and their AUCs

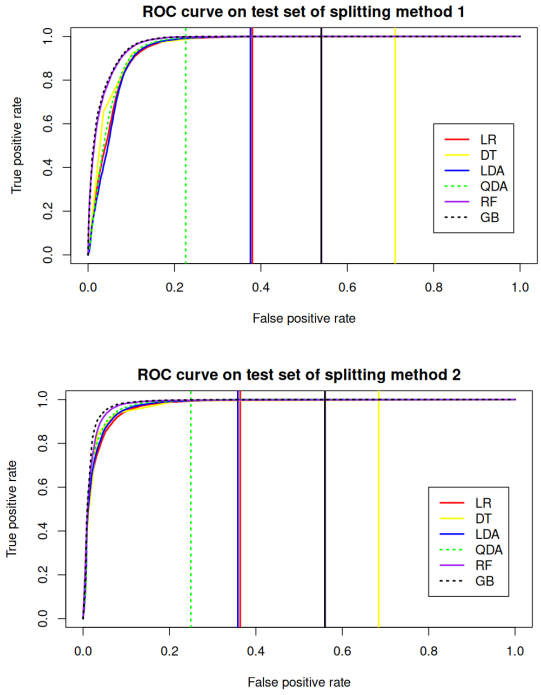


Figure 10: ROC curves for two splitting methods with selected cutoffs

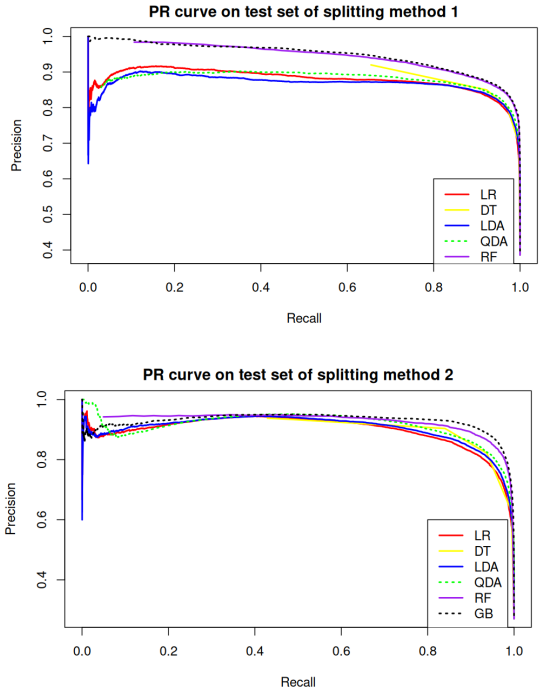


Figure 11: PR curves for classification algorithms under different splitting methods

	LR	DT	LDA	QDA	RF	GB
splitting1	0.949	0.956	0.948	0.953	0.971	0.970
splitting2	0.971	0.972	0.974	0.977	0.980	0.983

Table 6: AUC of classification algorithms under different splitting methods

are higher than other classification algorithms. Also, we employed the concept of Precision/Recall curves and plot PR curves for all six classification algorithms, the result is shown in **figure**[11]. Again, we found that the RF and GB are almost at the same level and are better than all other methods, since at given recalls, the RF and GB had higher precision than other methods.

4 Diagnostics

4.1 a

Variable Importance: We decided to choose the random forest model as our best model because it has the lowest CV error rate for our first splitting method (fixed label ratio). We considered the first splitting method to be more robust in the sense that it avoids picking specific locations which may make the ratio to be extremely large or small. We began by interpreting our random forest using two measures of variable importance. The first is Gini Importance, or mean decrease in impurity, which measures the magnitude by which a tree's node impurity decreases when a feature is included vs. when it is left out, averaged over all trees in the random forest. The second is the mean decrease in accuracy, which is obtained by randomly permuting the values of a given feature and measuring the resulting increase in classification error. We obtained the results for variable importance in **table**[7].

	Mean Decrease Accuracy	Mean Decrease Gini
NDAI	117.25	36709
SD	34.10	22986
CORR	46.16	19606

Table 7: Feature Importance in RF

Based on these metrics in **table**[7], NDAI is the most important feature, followed by SD and CORR. In other words, excluding NDAI from the random forest model results in the greatest increase in error.

Hyperparameters Tuning: Next, we decided to use cross-validation to tune the hyperparameters of RF and GB. In order to obtain the optimal model complexity based on CV misclassification error. The two hyperparameters we are interested in are the

number of trees in the random forest and the number of features chosen at each split of each tree.

In RF, we first select the cardinality of the subset of features that are considered at every splitting step. For the classification task, this cardinality is often chosen to be $\sqrt{p}$ when p is large, where p is the dimensions of the feature space. However, our feature space only has 3 dimensions here, so simply taking $\sqrt{p}$ is not appropriate. We let the cardinality range over $1, 2, 3$. This tuning was done over the first splitting method due to the computational limit. As the result was displayed in the upper of **figure**[12], no matter what is the number of trees in RF, the RF achieved the lowest error rate when the cardinality of the subset of features was set to 1.

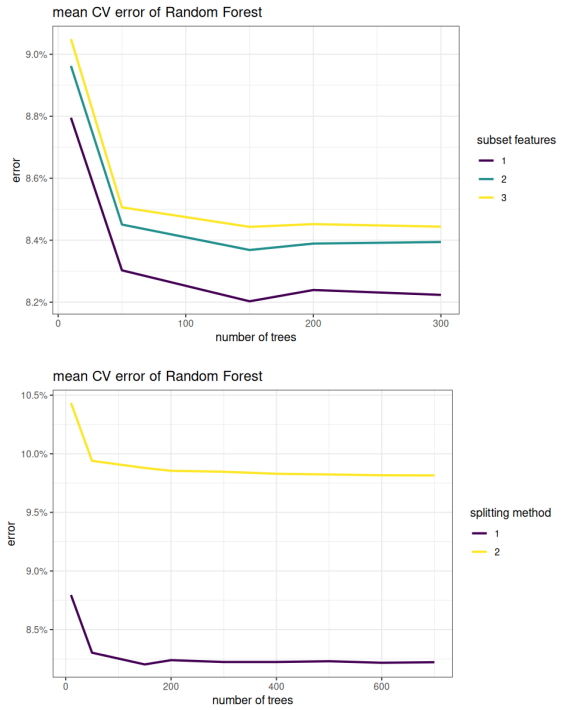


Figure 12: hyperparameters tuning for Random Forests

Then we chose the number of trees in RF, we run RF under different numbers of trees on both splitting methods. The mean 5-fold CV error was displayed on the lower of **figure**[12]. We observed that on both splitting methods, the error decreases at first and stabilized at some level as the number of trees increases. It is pointed out in **Ch.15** of **ref**[2] that increasing the number of trees would not result in overfitting in the RF algorithm. So we could pick any larger number as long as the error stabilized. For computational efficiency, we picked the number of trees to be 200 in RF.

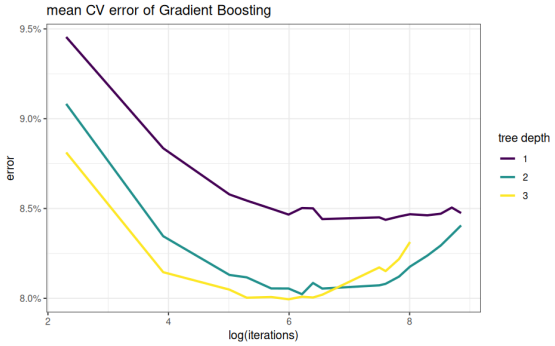


Figure 13: hyperparameters tuning for Gradient Boosting

Last, we tuned the hyperparameters for GB. We selected the complexity of base learners (depth of base decision trees), and the iteration steps. We originally also wanted to tune the learning rate, but as it is pointed in **Ch.8** of **ref**[2], the learning rate has inverse relationship with iteration steps, the smaller learning rate would require larger iteration steps. As a result, we just fixed the learning rate to 0.1, and chose different iteration steps. The result is shown in **figure**[13]. Unlike in RF, too many iterations would result in overfitting in GB. From the figure, we concluded that the results are better for the base tree depth to be 2 or 3 than for the base tree depth to be 1. We also observed that after passing the optimal iteration the error increases quickly when the tree depth is set to 3, which means the overfitting is quickly appearing. In contrast, when the tree depth is set to 2, it not only achieved a smaller error rate than when the tree depth is 1, but also had a slower increase in error rate when overfitting occurred. In conclusion, we choose the base tree depth to be 2, and the iteration steps to be 500, which allows the CV error to be minimal.

single trees in RF: We also wanted to pick out a few trees from RF to see the exact values of decisions being made at tree splits. Extracting the trees revealed that the trees used to build the random forest were extremely complex, with each tree having over 20000 nodes. This is far more complex than the single decision tree we previously implemented, which had five nodes and is shown in **figure**[14].

Robustness: Finally, we wanted to evaluate the robustness of our random forest model by examining its performance on testing data when the training data used to build the model has been perturbed. The most obvious perturbation is when observations are mistakenly given the opposite label of the expert

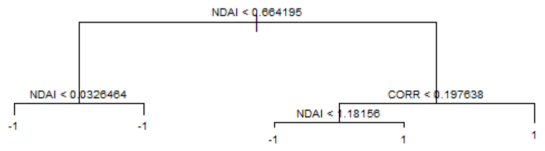


Figure 14: The splitting in the Decision Tree

label. **figure**[15] shows the misclassification error rate on the testing data as a function of the percentage of perturbed training observations.

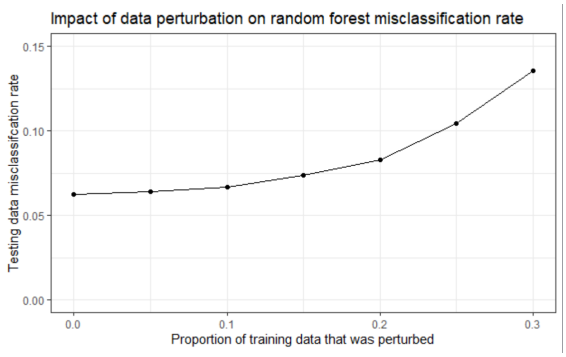


Figure 15: The splitting in the Decision Tree

Since the testing error is not significantly impacted by increasing the proportion of perturbed data, we can conclude that our random forest model is robust. Even at a perturbation rate of 30%, the testing error remains below 15%.

4.2 b

We examined patterns among misclassification errors for our gradient boosting model and our random forest model. For both models, we generated plots comparing feature distributions for misclassified and correctly classified data. We found that the distributions for the two types of data across all features were very similar between the two models. Thus, we will only discuss the plots for our random forest model, since we analyzed this model in part a. For the coordinate plots in **figure**[16], we used combined data on testing errors from 5 different training-testing fold splits, so that we could obtain testing misclassifications over the whole geographic space. For all other feature plots, we used only training-testing data generated from our first splitting method. For the random forest model, we obtained the following **figure**[17] for the features exhibiting the most significant difference in distribution between the two groups.

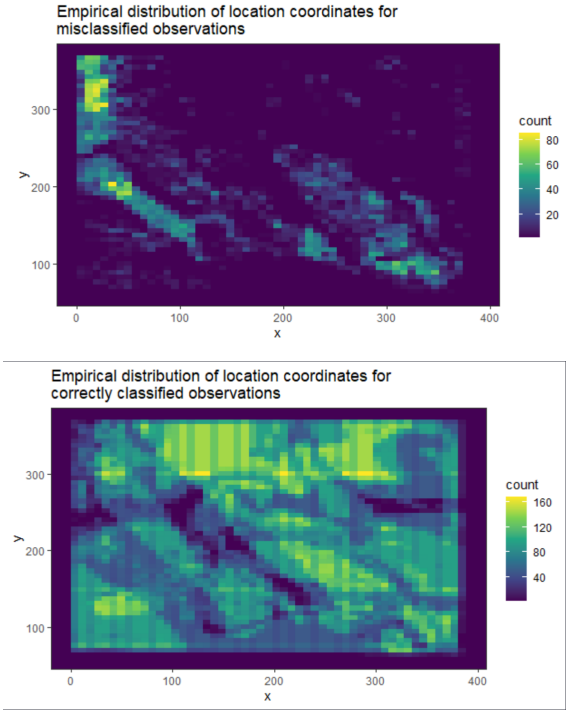


Figure 16: Geographic Patterns for Misclassification

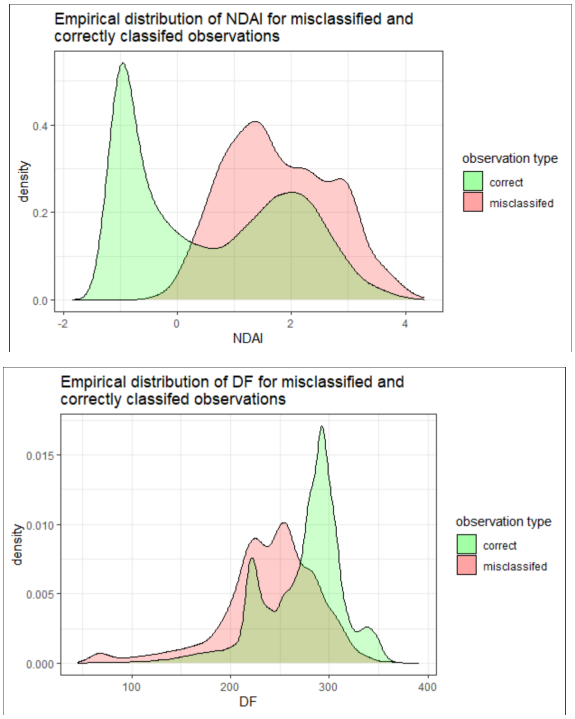


Figure 17: Feature Patterns for Misclassification

There are some geographic patterns among the misclassification pixels: they are not randomly distributed over the x-y plane, but they are very concentrated in some specific region. Based on the distributions of location coordinates, we can see that misclassified points tend to be concentrated near the top left regions of the image, while correctly

classified points are distributed relatively uniformly across the image space.

As for our three features of interest, the only one with a significant difference in the distributions across misclassified and correctly classified points is NDAI. The density of NDAI values for misclassified points is more concentrated in positive ranges between 1 and 3, whereas the density of NDAI values for correctly classified points is more concentrated in negative ranges between -1 and 0. Since larger NDAI values are associated with cloudier regions, it would seem that, based on the NDAI feature alone, our model may tend to misclassify cloudy observations more often than non-cloudy observations.

Finally, we noticed that all five radiance angles exhibit very similar distribution shapes and spreads between misclassified and correctly classified points. Thus, we only display a single plot for the DF radiance angle above for brevity. It appears that, generally speaking, observations with larger values of radiance angles (closer to 300) are more likely to be correctly classified than observations with smaller values of radiance angles (closer to 200).

We also examined plots comparing the distribution of features for points misclassified as cloudy and points misclassified as non-cloudy. We have the following **figure**[18] for the features exhibiting a significant difference in these distributions. The "label" group indicates the expert label of the original observations, corresponding to the actual value.

We notice that the distributions of location coordinates for observations which were misclassified as cloudy tend to be concentrated in the top left regions of the image, while the distributions for observations which were misclassfied as non-cloudy tend to be concentrated in the bottom right regions of the image. From the NDAI plot, we can see that observations misclassified as non-cloudy tend to have lower values of NDAI than observations misclassified as cloudy. This finding is consistent with our knowledge that larger NDAI values are associated with cloudy regions. Specifically, our model may have trouble classifying cloudy observations correctly if their NDAI values are too low, and vice versa for non-cloudy observations.

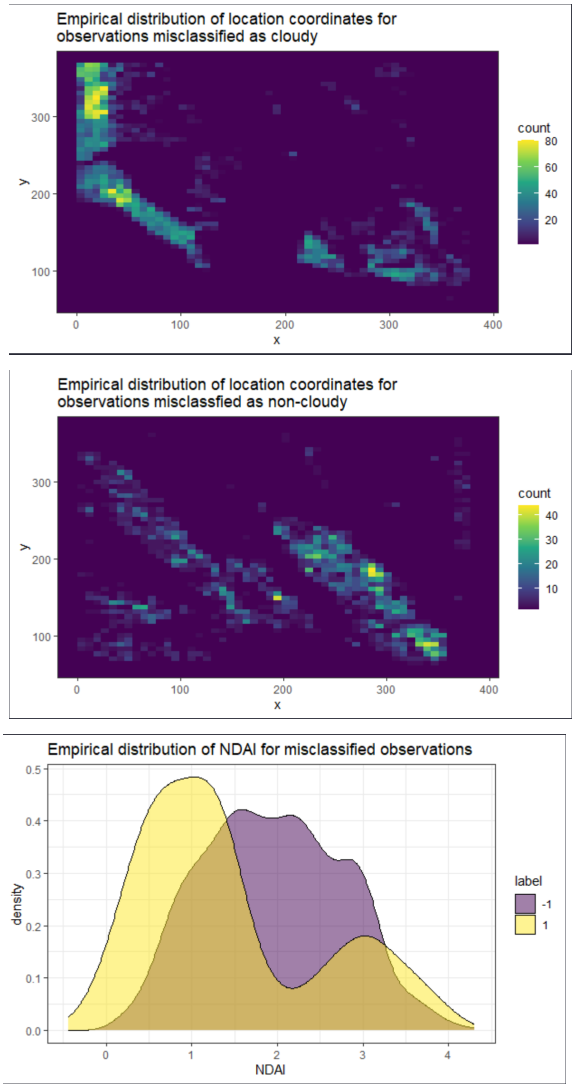


Figure 18: Cloudy or not for Misclassification

4.3 c

Based on our findings from the previous parts, we wanted to improve our random forest classifier by setting a limit on the number of nodes that the individual trees can have. The hope was that the new classifier would achieve similar classification accuracy while reducing complexity and increasing computational efficiency. Initially, we set the node limit to be 5, since this is equal to the number of terminal nodes in our single decision tree previously. However, this did not result in a better test error than our original model. After some tuning, we found that setting the limit to 20 nodes per tree achieved a lower test error than our previous model. Specifically, for splitting method 1, the test error remained approximately the same, but for splitting method 2, the test error decreased by 1.2%. The computational efficiency of our new algorithm also improved significantly over our original algorithm, as the runtime was shortened by about 70 seconds.

To assess how well our model might perform on future data, we opted to train our model using data

from image 1 and 3, and then evaluate test accuracy on image 2. In this case, image 2 can be considered as "future data" that was collected after images 1 and 3. Using this method, we found the test error on image 2 to be 7.09%, which is close to the test errors obtained from our previous two splitting methods. Thus, we believe that our model will perform well on future unlabeled data.

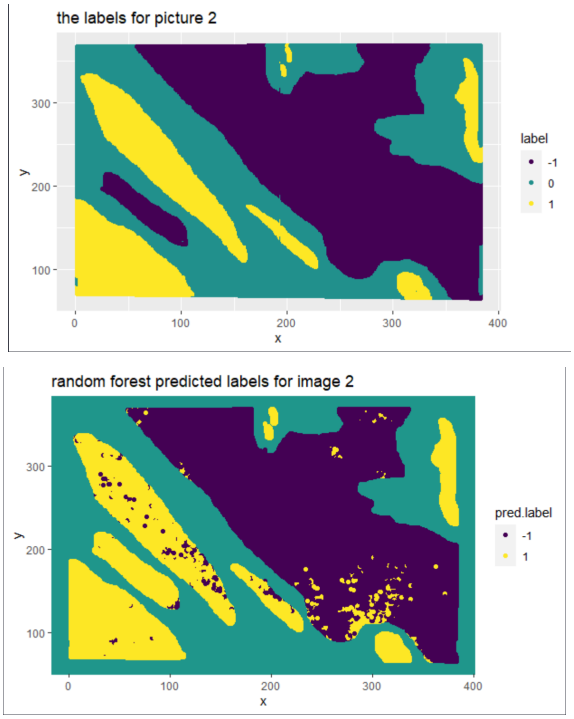


Figure 19: Predicted labels for image2, compared with true labels

The predicted labels are shown in **figure**[19], compared with true labels. Overall, the random forest performs well, with the exception of a small region of points that it nearly entirely misclassified as cloudy.

4.4 d

Finally, we checked for differences in model diagnostics and misclassification error trends when our second splitting method (geographical splitting) was applied to the data instead of the first splitting method. We used the exact same model that was used for splitting method 1, which does not include the 20-node limit on individual trees but does include hyperparameter optimization on the number of trees used and the number of features considered at each split.

In terms of variable importance, we found that NDAI remained the most important feature of the three, but CORR was now more important than SD under both metrics of variable importance. All other diagnostic metrics used in part a, such as hyperparameter tuning and data perturbation analysis, remained the same or very similar for splitting method 2 compared to splitting method 1.

When examining trends in misclassified observations, checking for disparities in coordinate positions is irrelevant because splitting method 2 is done geographically, so not all x-coordinates are represented in the testing data. As for the rest of the features, comparing distributions of misclassified and correctly classified points yields similar results to splitting method 1. NDAI remains the only feature of interest that exhibits a significant difference in density between the two groups. As for the radiance angles, all 5 angles continued to exhibit similar behavior as in splitting method 1, but the difference in the density centers for misclassified and correctly classified points is much more defined (ie. more concentrated towards a smaller range of values). Next, when plotting differences in distributions for points misclassified as cloudy and points misclassified as non-cloudy, both NDAI and CORR show clear differences in density between the two groups, as opposed to splitting method 1 where only NDAI was observed to be different. As before, observations misclassified as non-cloudy tend to have lower values of NDAI than observations misclassified as cloudy, though this disparity is slightly less obvious than in splitting method 1. On the other hand, the disparity in density for CORR indicates that observations misclassified as cloudy tend to have larger values of CORR than observations misclassified as non-cloudy.

4.5 e

Overall, we conclude that our random forest is accurate, robust, and well-optimized. Hyperparameter tuning allowed us to optimize the number of trees used (200) and the number of predictors selected at each split (1) in the random forest. This resulted in a model with relatively low test error and high robustness to data perturbation. Furthermore, using metrics of mean decrease in node impurity and mean decrease in accuracy, we found NDAI to be the most important feature, followed by SD and CORR. Then, we were able to consider an additional hyperparameter that limited the maximum number of

terminal nodes in individual trees within the random forest. Not only did this improve our testing error, but it also drastically improved the computational efficiency of our model. This was supported by the fact that our model performed strongly on our "future" data set. The main shortcoming of our model is that it tends to misclassify cloudy observations as non-cloudy if the NDAI values are lower than normal. A more in-depth analysis of specific observations within the testing data, as well as the implementation of new testing sets, should be done to more confidently support this claim.

5 Acknowledgement

Derek Tao worked on the summary in **Task1.a**, and Minhui Jiang did plotting and EDA in **Task1.b/c**. Minhui Jiang mainly did coding and write-up in **Task2** and **Task3**, wrote the **CVMaster** function. Derek Tao contributed to **Task1.c** and did coding and write-up in **Task4**, Derek also wrote the **README** in our codes folder. And we asked and referred to answers by Nancy Huang in ED discussion forum.

References

[1] Tao Shi, Bin Yu, et al. (2008) Daytime Arctic Cloud Detection Based on Multi-Angle Satellite Data With Case Studies,*Journal of the American Statistical Association*.

[2] T Hastie, R Tibshirani, J Friedman. (2009) *The Elements of Statistical Learning*.

[3] T Hastie, R Tibshirani, et al. (2017)*An Introduction to Statistical Learning*.