Cloudy Region Analysis

Team Name: Sunny Squad
Jincen Li 3033312349
Yi Xu 3033272595

1.Data Collection and Exploration

(a). The purpose of this study is to detect the cloud distribution in the polar regions based on radiance recorded automatically by the multiangle Imaging SpectroRadiometer (MISR) sensor abroad the NASA satellite Terra. The sensor produces a vast amount of data due to its global coverage at a high spatial resolution and its cameras cover a wide swath on the Earth's surface. The data from path 26 is used in the study of this paper which is collected from 10 MISR orbits of path 26 over the Arctic, northern Greenland, and Baffin Bay on a repeat cycle of 16 days in the period from April 28 through Sept 19, 2002. The algorithm the researchers used based on three physically useful features: CORR, SD, and NDAI based on EDA and domain knowledge. In order to handle the computations of massive MISR data stream in an online fashion, the paper also illustrates the use of ELCM algorithm which combines classification and clustering frameworks in a way that makes it suitable for real-time, operational MISR data processing. And then finally make the probability of cloudiness prediction based on the data units by training Fisher's QDA on the labels produced by the ELCM.
The data showed the ELCM algorithm based on the three features is more accurate and provides better spatial coverage than the existing MISR operational algorithms for cloud detection in the Arctic. The paper shows the feasibility of using innovative solutions to apparently complex scientific problems. The impact of this study showed in the power of statistical thinking and also the ability of statistics to contribute solutions to modern scientific problems. By understanding the flow of visible and infrared radiation through the atmosphere, the future study can translate the cloud properties into more accurate global climate model and furthermore enable the scientific community to have a better understanding of the climate changes in the Arctic from the changes of cloud properties.

(b). In image1, % of pixels for the cloud class is 17.76549%, for the unlabeled class is 38.4556%, for the not cloud class is 43.77891%. In image 2, % of pixels for the cloud class is 34.11172%, for the unlabeled class is 28.63522%, for the not cloud class is 37.25306%. In mage 3, % of pixels for the cloud class is 18.43825%, for the unlabeled class is 52.26746%, for the not cloud class is 29.29429%.

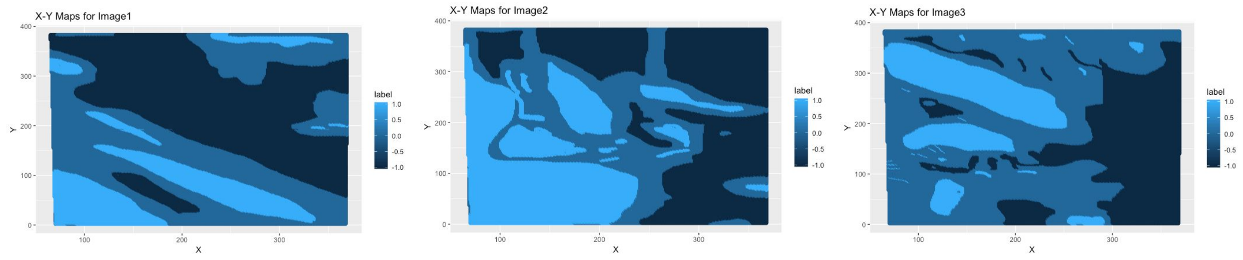


Figure 1

Looking at the above scatter plots for each image, cloud (light blue) and not-cloud (dark blue) classes assemble on different regions, i.e. if a cloud point is more likely to be surrounded by cloud points rather than not-cloud. Thus, we can say there exist obvious clusters for each class. And the unlabeled usually happens in the region between cloud and not-cloud class. Therefore, i.i.d assumption doesn't work for this dataset.

(c). Looking at the pairwise relationship between features themselves for each image below, the relationships between each feature among these three datasets resemble each other, so we decided to focus on one image dataset, which is image 1 in our report. The correlation between NDAI and CORR is 0.251, and between SD and CORR is 0.165, which is very small. The correlation between NDAI and SD is 0.601. And other feature pairs, such as AF-BF and CF-DF have a high correlation. The correlation between AF and BF is 0.979. Based on the scatterplot and correlation value, we picked some feature pairs and made more detailed plots.

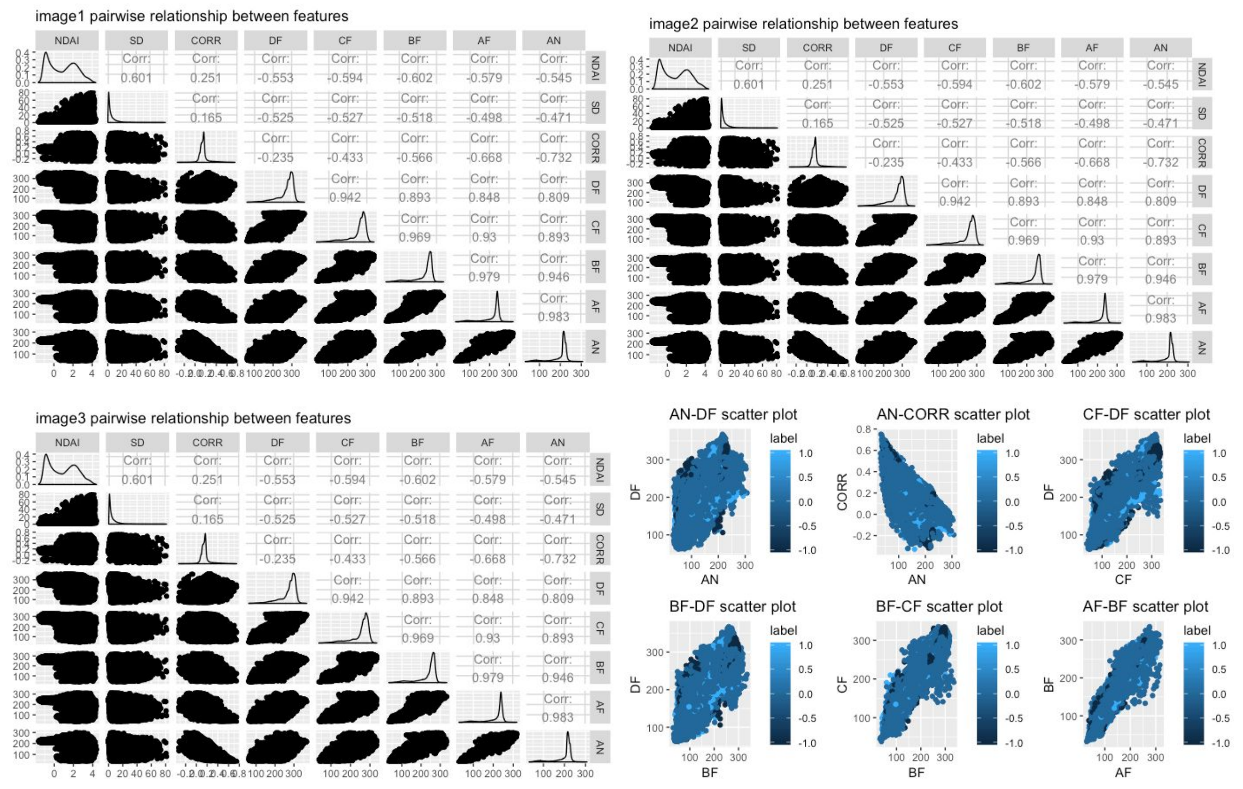


Figure 2

After filtering out the unlabeled data, we made side-by-side boxplot for radiance AF and other features like CORR, NDAI and SD to see the relationship between the expert labels and each feature. By comparing three images, CORR and SD distribute similarly for both cloud label and not-cloud label. However, cloud class distributed with a higher NDAI and not-cloud class has a slightly higher AF than cloud class.

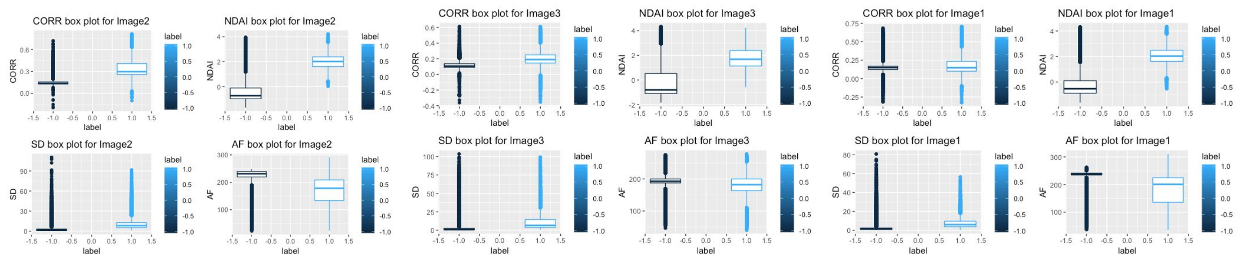


Figure 3

2. Preparation

(a). Since we have already known that the data here does not meet the i.i.d assumption. We intuitively think that we need to randomize the data based on different conditions. We try to isolate each data point from the other so when we split the whole dataset into different small sets, we can cover a large variety of the information and try to minimize the correlation of the data points which come from the same cluster.
First method: we first filter each image into cloud set and cloud free set filter by the label and then randomly pick 80% of the data from each of the set as our training data, 10% as validation and then 10% as test. The final train set would have randomized cloud data and cloud free data based on portion which ensure that we have both conditions as our train and at the same time minimize the dependency between the data points.
Second method: In this method, we are still just dealing with the data with label. We start with splitting the data into 5*5 blocks based on the x-y coordinates. And then randomly picked 15 blocks as our train, 5 for validation and 5 for test. From this method we still keep some of the dependent relationships of the data point within each block but on the whole block splitting level. We treat each of the block as an independent data point. We chose 5*5 dimension because this is big enough to have both cloud and cloud free condition within each block and the number is not too small so that can be computed efficiently.

(b). Under the setting of our first split method, the accuracy on both validation and test sets are equal to 61.0756% since our validation and test sets were selected by a certain portion, i.e. both contain the same amount of cloud free data and the total amounts of data are equal as well. In order to improve the accuracy, we should use nontrivial classifier since int the real-life scenarios, cloud also exists. We cannot simplify our predictions to only one possible situation. If the accuracy of the validation set and test set are different, it means the classifier is nontrivial, since the amount of matched cloud prediction and matched non cloud prediction vary set by set. It provides a baseline to ensure non trivial classification.

(c). To have the best features, we checked both correlation scores between each variables and also fitted PCA algorithm to see the contributions from each variable.

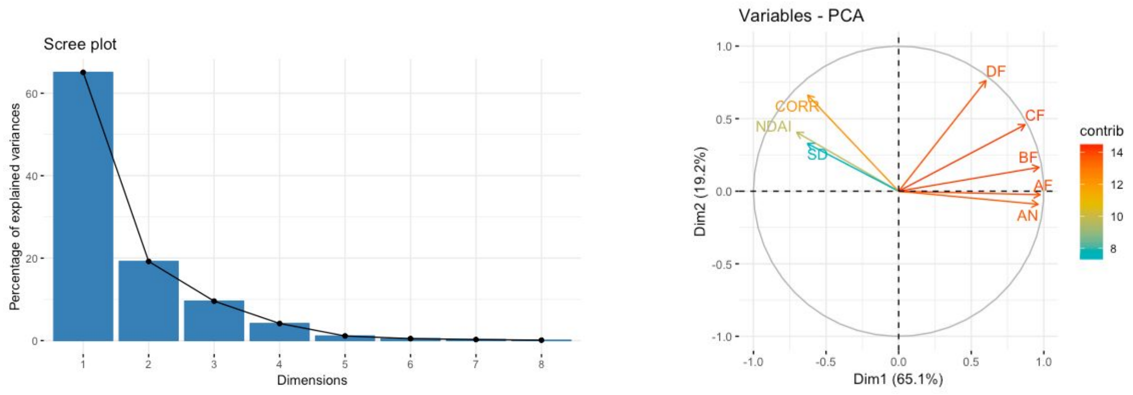


Figure 4

From the PCA analysis, we see that the first three component covered about 93.6% variance. It illustrates the feasibility of reduced 9 features to 3 without compromising on explained variance.

Now we calculated the correlation scores with each variables and we found the 4 features that most correlated with label are NDAI 0.76, CORR 0.55, AF -0.51 and AN -0.5. Since the goal of the study is to make prediction on whether a specific region is cloud or cloud free so we want to define the most related features that can be used in the model to predict cloud conditions with higher accuracy. Combined with the PCA variable contribution plot from figure 4, we decided to choose NDAI, CORR and AF as the best features to use which are the most correlated features with the label and we can see the AF contribute the most on the second component from the PCA analysis since its loading vector gets close to the second dimension the most.

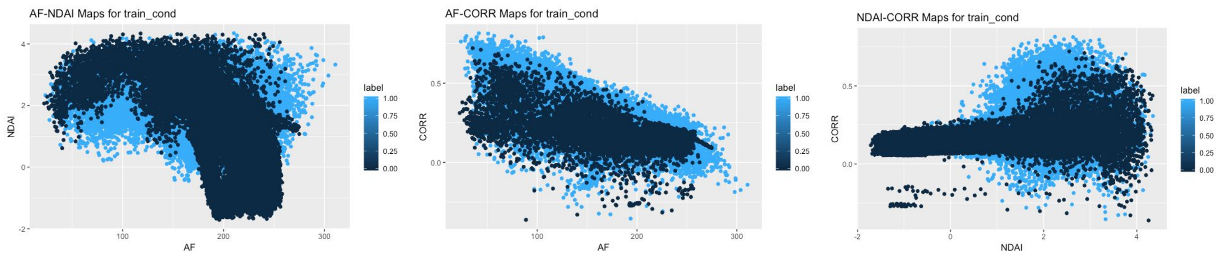


Figure 5

By plotting the scatter plots between the three features colored by label, we see the three features are not highly correlated with each other which is good for our analysis because this three features would catch a larger various information. We can see clearly the cloud free regions are bigger than the region covered by cloud, and the data points with different labels roughly followed the same trend as we move the coordinate in each of the scatter plot.

3. Modeling

(a). As the preparation for modeling, we assigned all the labels as -1 to 0 (filtered out no expert label points prior to the data splitting step) to create a binary variable so we can easily fit binary classification.

We started off the model fitting with logistic regression where we have the following reasons to believe that logistic regression might have the best fit on our data. First, Logistic regression does not require a linear relationship between the dependent and independent variables like general linear regression and therefore there's no i.i.d assumption required from the data which perfectly matched our dataset since we see clustering patterns from three images which indicated a violation of i.i.d condition. Second, the error terms (residuals) do not need to be normally distributed. Third, homoscedasticity is not required. Finally, by calculating the correlation score between each features, we didn't see high intercorrelations(multicollinearity) among the predictors which also meet the assumption for logistic regression.

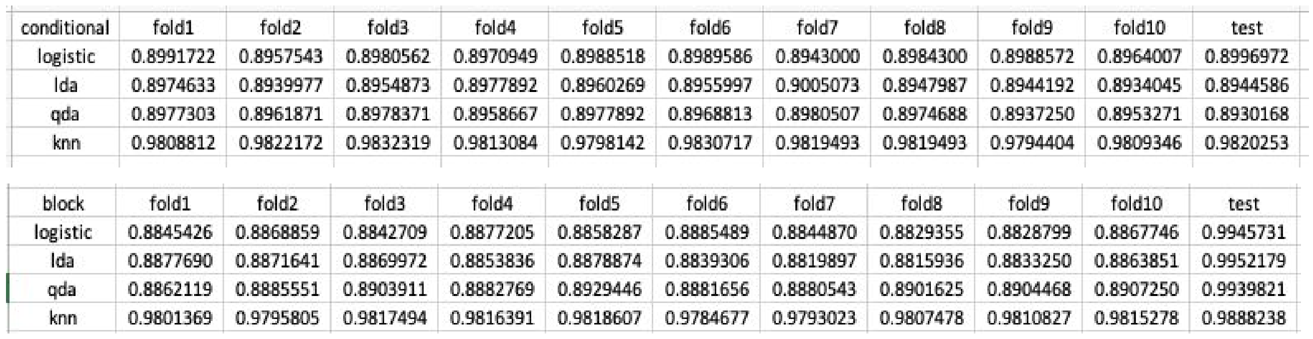
Next we tried LDA classification method on our data. We learned that LDA models the distribution of the predictors X separately in each of the response classes (i.e. given Y). LDA assumes that the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector and a covariance matrix that is common to all K classes. It's not really appropriate in this case since it requires data to have normal distribution. Since our data has spatial dependency, the normal distribution assumption is not met.
If n is small and the distribution of the predictors X is approximately normal in each of the classes, the linear discriminant model is again more stable than the logistic regression model.

The third method we tried is QDA classification method. Compared with LDA, QDA is much more flexible classifier thant LDA, and so has substantially higher variance. This can potentially lead to decreased prediction performance. But there is a trade-off; if LDA's assumption that the K classes share a common covariance matrix is badly off, then LDA can suffer from high bias. Roughly speaking, LDA tends to be a better bet than QDA if there are relatively few training observations and so reducing variance is crucial. The QDA method works well with large dataset so that the variance of the classifier is not a major concern. Our dataset is large enough but it fails QDA's normal distribution assumption as the same as LDA.

The last method we tried is KNN which assumes that the data is in a feature space. More exactly, the data points are in a metric space. In such a space, a notion of distance should be well defined. In our case, the distance is Euclidean distance.
Each of the training data consists of a set of vectors and class label associated with each other. In our case, we have a binary classification problem. We label the class with 1 and 0.
We are also given a single number k, which decides how many neighbors (where neighbors are defined based on the distance metric) influence the classification. We use k = 100 in our case.

KNN also has its inherent disadvantages. It cost us a lot of time to run the whole algorithm because of its high computational complexity. Sample imbalance problem affects the prediction of KNN and can not be ignored in our data -- the sizes of the data of two classes differ seriously. The biggest drawback is that the intrinsic meaning of the data cannot be given. With all reasons I've mentioned above, we decided not to adopt KNN.

Overall, We tried four classifiers, logistic regression, LDA, QDA and KNN and calculated each corresponding accuracy across folds and on the test set. The upper table shows the accuracy for the first split method. And the bottom is the accuracy for the block split. By comparing these two tables, the accuracy across the folds calculated from these two methods is a nuance. Both have around 89-90% accuracy rate. The accuracy in the test set by method 2 is slightly higher than method 1. Moreover, KNN has the highest accuracy among these four classifiers, and KNN assumes the data point in a metric space, and it works well with a small number of inputs.



conditional	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9	fold10	test
logistic	0.8991722	0.8957543	0.8980562	0.8970949	0.8988518	0.8989586	0.8943000	0.8984300	0.8988572	0.8964007	0.8996972
lda	0.8974633	0.8939977	0.8954873	0.8977892	0.8960269	0.8955997	0.9005073	0.8947987	0.8944192	0.8934045	0.8944586
qda	0.8977303	0.8961871	0.8978371	0.8958667	0.8977892	0.8968813	0.8980507	0.8974688	0.8937250	0.8953271	0.8930168
knn	0.9808812	0.9822172	0.9832319	0.9813084	0.9798142	0.9830717	0.9819493	0.9819493	0.9794404	0.9809346	0.9820253

block	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9	fold10	test
logistic	0.8845426	0.8868859	0.8842709	0.8877205	0.8858287	0.8885489	0.8844870	0.8829355	0.8828799	0.8867746	0.9945731
lda	0.8877690	0.8871641	0.8869972	0.8853836	0.8878874	0.8839306	0.8819897	0.8815936	0.8833250	0.8863851	0.9952179
qda	0.8862119	0.8885551	0.8903911	0.8882769	0.8929446	0.8881656	0.8880543	0.8901625	0.8904468	0.8907250	0.9939821
knn	0.9801369	0.9795805	0.9817494	0.9816391	0.9818607	0.9784677	0.9793023	0.9807478	0.9810827	0.9815278	0.9888238

Figure 6

(b) The figure 7 below showed the ROC plots of four classification algorithms in two splitting methods, each with a cutoff line marked with the specific optimal cutoff value.
The cut off value was calculated by the distance function with the sensitivity and specificity as our variables.

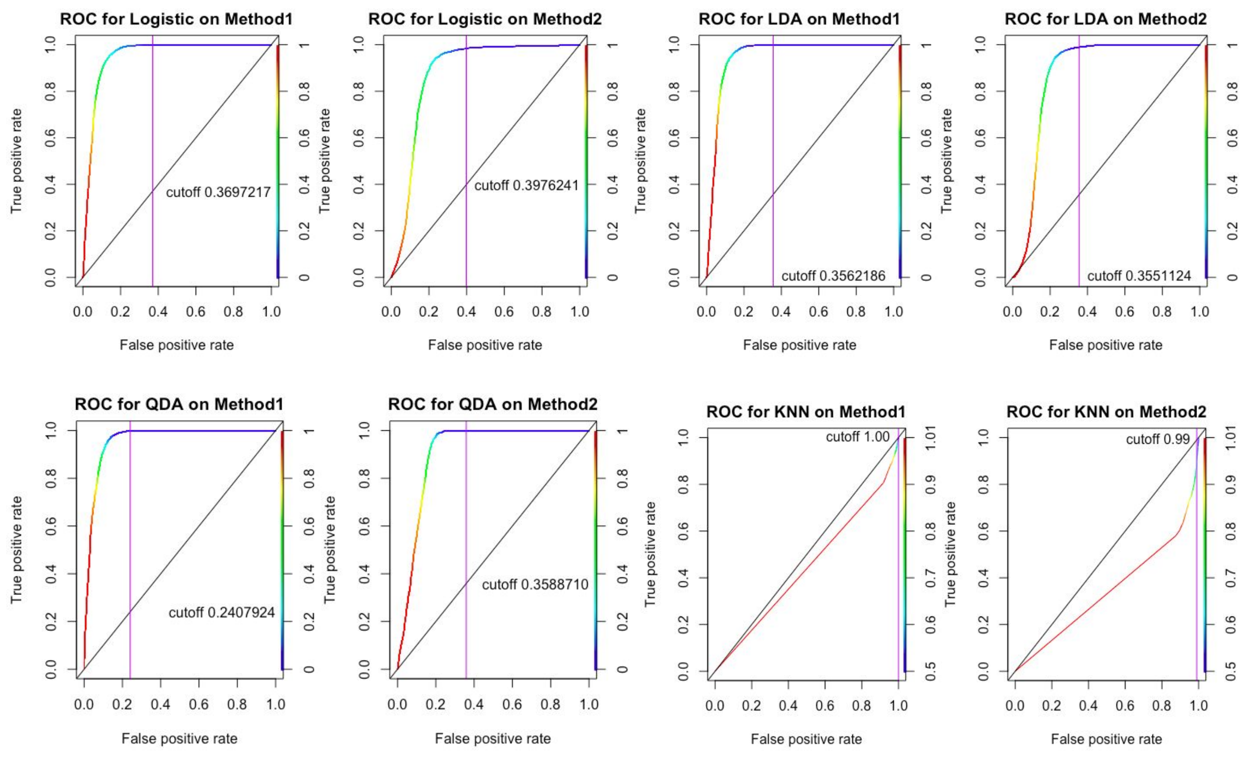


Figure 7

$$d = \sqrt{(1 - sensitivity)^2 + (1 - specificity)^2}$$

The above formula gives the distance to the top-left corner of the ROC curve for each cutoff value. We would like to have lower distances to the corner which gives us the highest true positive rate. Next we look at the area under the curve. We found the first splitting method has a greater region of AUC over all four classifications. We can interpret the AUC as the probability that the criterion value of an individual data point drawn at random from the whole dataset of those with a positive condition (here it means region with cloud) is larger than the criterion value of another individual point drawn at random from the whole dataset of those where the condition is negative (cloud free). From the visualization from figure 7 we can tell that method 1 with QDA provides the lowest cut off value and its AUC is quite large compared to others which is in a good AUC range; in contrast, the KNN classification algorithm performed poorly with both methods on the ROC analysis with AUC less than 0.5 and a really high cut off value about 1 which indicates no diagnostic ability.

4. Diagnostics

(a). By doing the first three parts of analysis, we feel that logistic regression is the best classification among all classifier we tried. Here are more detailed reasons. First, LDA assumes equality of covariance among the predictor variables across each all level of labels, which doesn't work in our image dataset. Moreover, both LDA and QDA ask the data to be drawn from a multivariate Gaussian distribution. By figure 8 the distribution plot for label is not normal obviously, so LDA and QDA are not appropriate in this image dataset. Although KNN has a higher accuracy by part 3(a), it's computationally expensive and has a high memory requirement, and it is sensitive to irrelevant features and the scale of the data. In addition, the ROC curve for KNN has a bad performance. Overall, logistic regression is a good choice. In our logistic regression, the parameter we use is AF, CORR and NDAI, and their corresponding coefficients are 0.01028837, 10.23332969 and 1.74273803 with the intercept of -5.94332208. And the number of iteration is six.

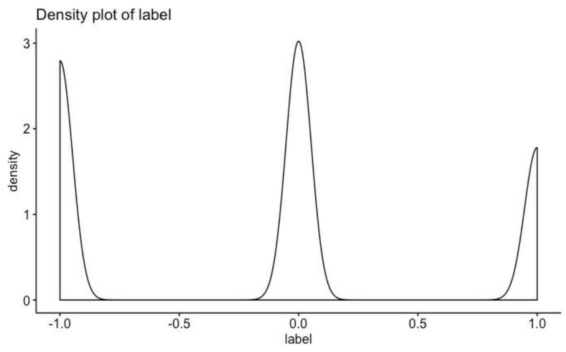


Figure 8

(b). We made two scatter plots for the misclassified points from the total image (image1 + image2 + image3) for each split method. In our test set, we filtered out the label 0 data and marked label -1 as 0 as mentioned above at part 3(a). The dark blue represents the points that are not cloud but wrongly marked as cloud. And the
light blue means the points that are cloud but marked as not cloud. From the left plot of figure 9 which is for the first split method, the dark blue points follow the same trend as the light blue points, and both of them mainly distributed at the upper left corner and lower right side. We can find the same result in figure 10. In the meanwhile, the right graph of figure 9 shows the misclassification for the block split method. And the two test blocks located the lower left and upper right corners. by looking at each block, the light blue and dark blue points distributed separately. For example, in the lower left block, the dark blue points and light blue points are at two diagonal corners. Similarly, in the upper right block, the light blue points assemble along the negative diagonals, but the dark blue points only cluster at one corner. This result also applies in figure 11. On the other side, there exist some similar patterns in the misclassification error for two split methods. No matter where the data exactly locates, both light blue points and dark blue data appeared in clusters. Additionally, the misclassification rate for the first split method is 10.26097% and the rate for the second split method is 4.783764%.
For the first split method, the misclassification error happens more frequently in two regions: low x and high y region and high x and low y. As for the second split method, there is no particular

region for misclassification since the block was generated by different ranges of x and y values, and the misclassification clusters appear in various places for each block.

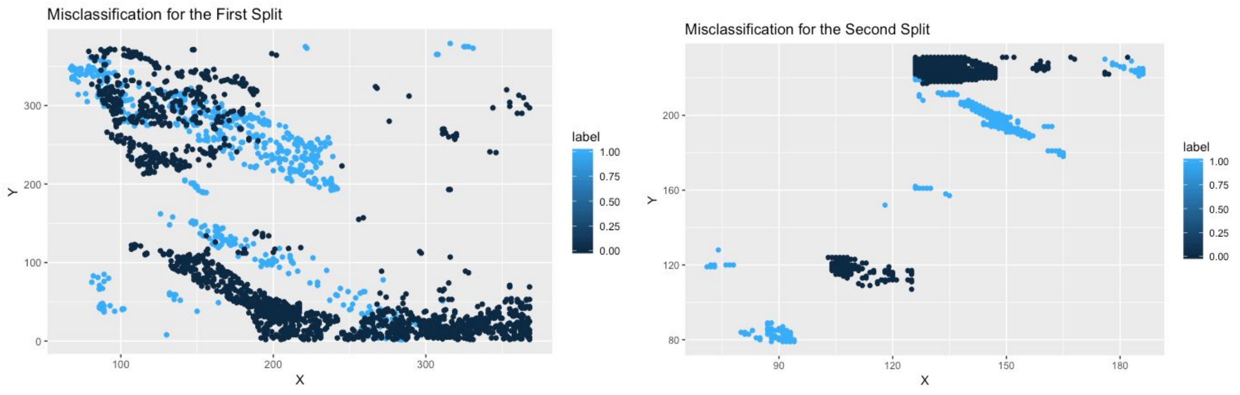


Figure 9. Misclassification for total image data under two split methods

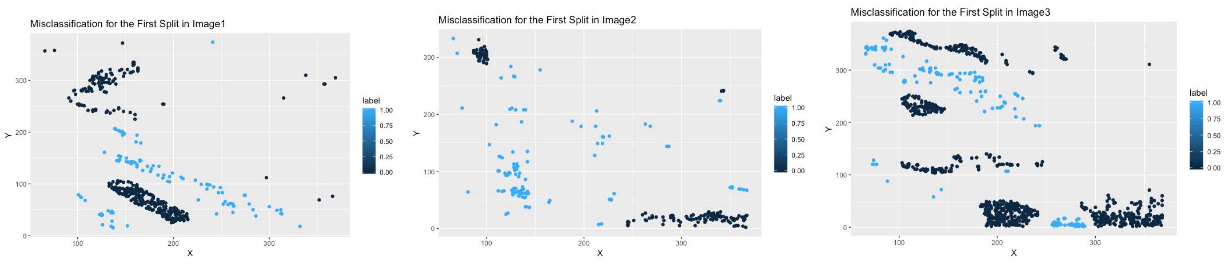


Figure 10. Misclassification for each image data under the first split

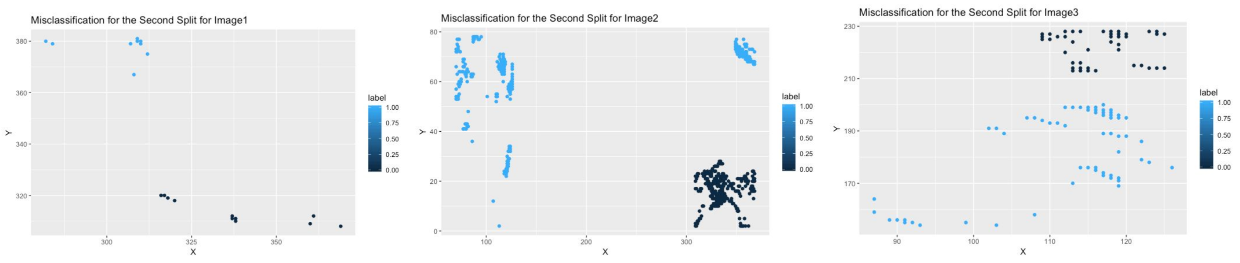
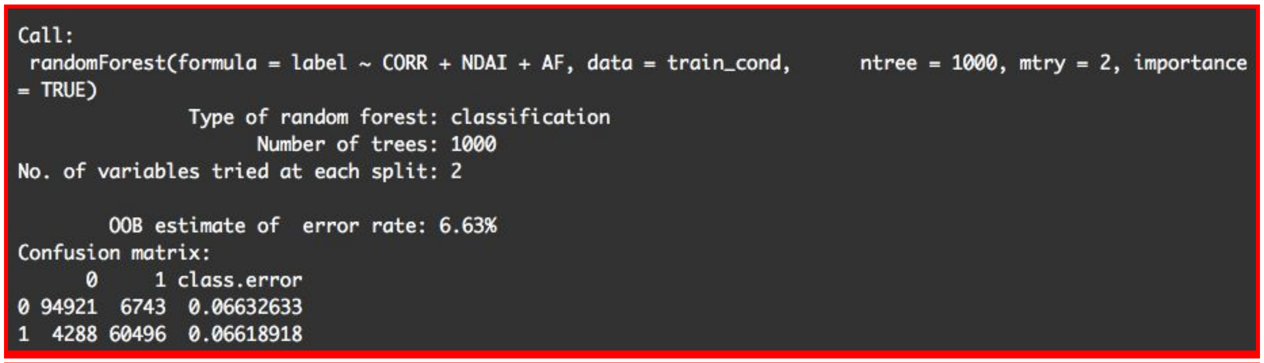


Figure 11. Misclassification for each image data under the second split

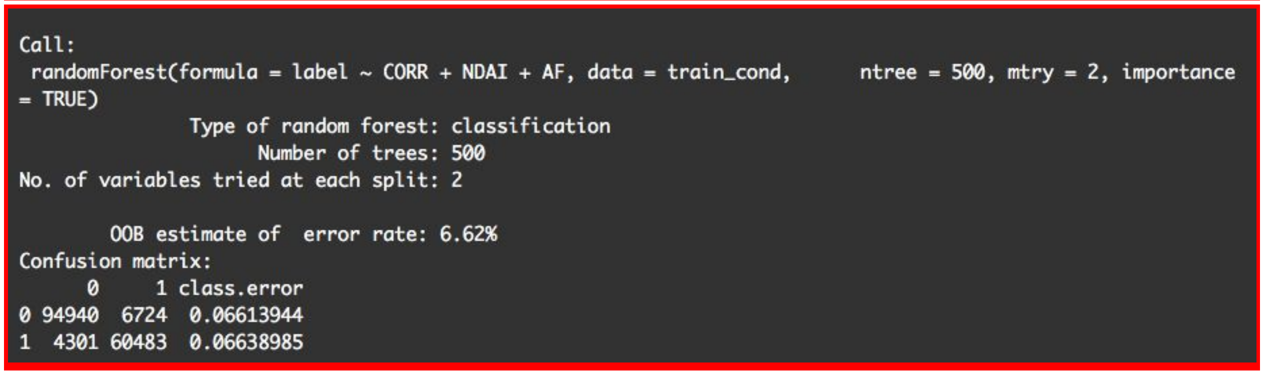
(c). Based on 4(a) and 4(b), we'd like to check if a model with iterations and then eventually converges to a particular points would give us a better fit. Since we noticed from the beginning that our data does not have i.i.d assumption. We'd like to use discriminative modeling that we don't need to make distributional assumptions about the Xi's. Here we tried random forest method as a better classification model works by creating multiple decision trees and then combining the output generated by each of the decision trees. Without i.i.d assumption, the resampled dataset will not have a joint distribution similar to that of the original dataset.
From the data preparation step, we decided to use AF, NDAI, and CORR as our best features to use to predict response on the label. Here we decide to be consistent with the feature choices as before. By the default value of the randomForest function, the hyperparameter ntree=500,

and mtry=#features/3. Since we only have 3 features to fit on the model here we choose mtry=2 because when we turn the value to 1 by the default calculation, the split variable would be completely random, so all variables get a chance but can lead to overly biased results.



```
Call:
 randomForest(formula = label ~ CORR + NDAI + AF, data = train_cond,      ntree = 1000, mtry = 2, importance
= TRUE)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 2

        OOB estimate of  error rate: 6.63%
Confusion matrix:
      0     1 class.error
0 94921  6743  0.06632633
1  4288 60496  0.06618918
```



```
Call:
 randomForest(formula = label ~ CORR + NDAI + AF, data = train_cond,      ntree = 500, mtry = 2, importance
= TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 6.62%
Confusion matrix:
      0     1 class.error
0 94940  6724  0.06613944
1  4301 60483  0.06638985
```

Figure 12

Figure 10 showed the table of model prediction performance based on different hyper parameter choices. Here we can see by setting ntree=1000, we can have a quite similar error rate so we decide to go with the default of 500 which can ensure each of the data points show up in a couple replications and also reduce the computation complexity.
Now we calculate the accuracy of our random forest model which gives us an accuracy score of 0.988 on the first splitting method. The result showed this is a better classification method compared with the previous four method in 3(a) evaluating the model fitting assumptions and the accuracy score which illustrates its prediction feasibility on future data without expert labels.

(d). Yes, our results in parts 4(a) and 4(b) will change as the split method changes. From the previous parts, we get two different results from two split methods, although there exist some similarities. Thus we have reason to believe that the result would change as we change the split method. Data splitting method as the first step on the data analysis before model fitting selections, it can affect our prediction response significantly. We'd like to have a reasonable splitting portion that we have enough data size as our train, validation and test set. Besides of the splitting portion, we also look carefully into whether each set contains all different information that relates to our target variable predictions so we can have a higher test accuracy and less error rates.

(e). In our conclusion, random forest classification method works the best in analyzing cloud region distribution. Besides the reason said above, random forest is one of the most accurate algorithms, and it can handle high-dimensional data without feature selection. Also, it is efficient to implement. Compared to general algorithms, it can use the OOB (out of bag) evaluate the error rate, instead of the need to choose the validation set to measure error. And random forest can give accurate label estimation even under the situation that in our original dataset a large proportion of data has zero label. But it's not as easy to visually interpret, and out data has spatial dependency. So from this perspective, logistics regression is also a good choice.

Github Repo link:
https://github.com/XuYi0317/STAT-154-Project-2-Cloud-Data

Contribution Statement:
We discuss all the questions and worked on the code together. And split the writing part to half and half. The resources we used: google, Data Science 101 and R documentation, etc. Thanks for the help from office hour and piazza.