STAT 154 Project 2

Liang Huang, Mian Wei

Data Collection and Exploration

Overview

Given the issue of global warming getting viral, climate changes in polar regions becomes alarming to human future, making the study of the cloud level (which is highly correlated with level of carbon dioxide below) in the region crucial. The goal of the study is hence to build the operational cloud detection algorithm and identify cloud-free pixels from disturbances such as snow and ice-covered surfaces in satellite photos.

The data are collected from the Multiangle Imaging SpectroRadiometer (MISR) which combines data from nine view zenith angles around the globe every 16 days along the 10 MISR orbits of path 26 over the Arctic, northern Greenland, and Baffin Bay. Three main physical features are derived for the algorithm designed by the team, enhanced linear correlation matching (ELCM): the correlation (CORR) of MISR images of the same scene from different MISR viewing directions, the standard deviation (SD_{An}) of MISR nadir camera pixel values across a scene, and a normalized difference angular index (NDAI) that characterizes the changes in a scene with changes in the MISR view direction. Also, to assess the accuracies of models, the expert labels are provided for 10 orbits of MISR data collected over the Arctic during the daylight season of 2002.

The conclusion of the paper is that the ELCM algorithm based on the three features is accurate and provides better spatial coverage than the existing algorithms for cloud detection in the Arctic by combining classification (including QDA) and clustering framework. This is also suitable for massive real-time, sufficiently fast operational MISR data processing. The research pioneers in implementing statistical methods in current scientific problems, motivating prediction of incidence with data. It has demonstrated the power of statistical thinking and the ability of statistics to contribute solutions to modern scientific problems.

Exploratory Data Analysis

From the three pictures data, we could see a brief summary of the data based on the classes they belong to and their distributions. We summarize the data by the weightage of each class as shown in the table on the right. We also plot the data according to their x- and y-coordinates with expert labels. This will tell whether the data are independent and identically distributed (i.i.d.) if there is no clear pattern of the data along both x- and y-axis.

	-1	0	+1
Img 1	43.78%	38.46%	17.77%
Img 2	37.25%	28.63%	34.1%
Img 3	29.29%	52.27%	18.44

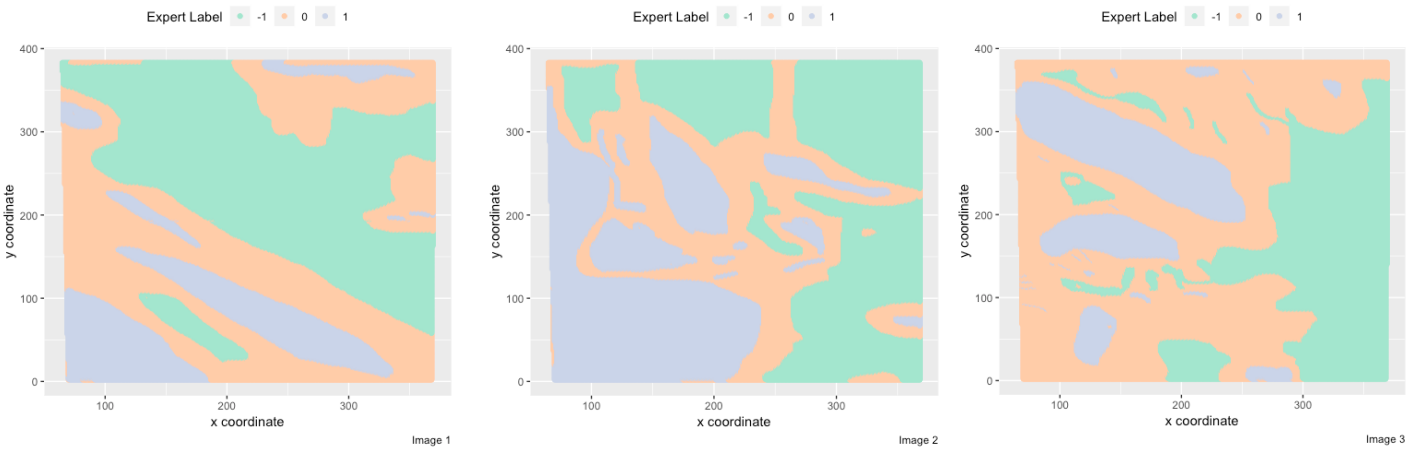


Figure 1 Distribution of data according their expert labels (classes) - Table 1 indicates the data distribution according to their expert labels, the following three graphs are the distribution of data points according to their expert labels.

We could observe that higher x and y-coordinates values tend to be classified under Group -1 (Cloud) while lower values of x and y-coordinates tend to be under Group 1 (no Cloud). This might mean that data points could be grouped together for the same object (with cloud or no cloud). Given that we could see the quite clear pattern, it might not be justifiable to assume the data are i.i.d..

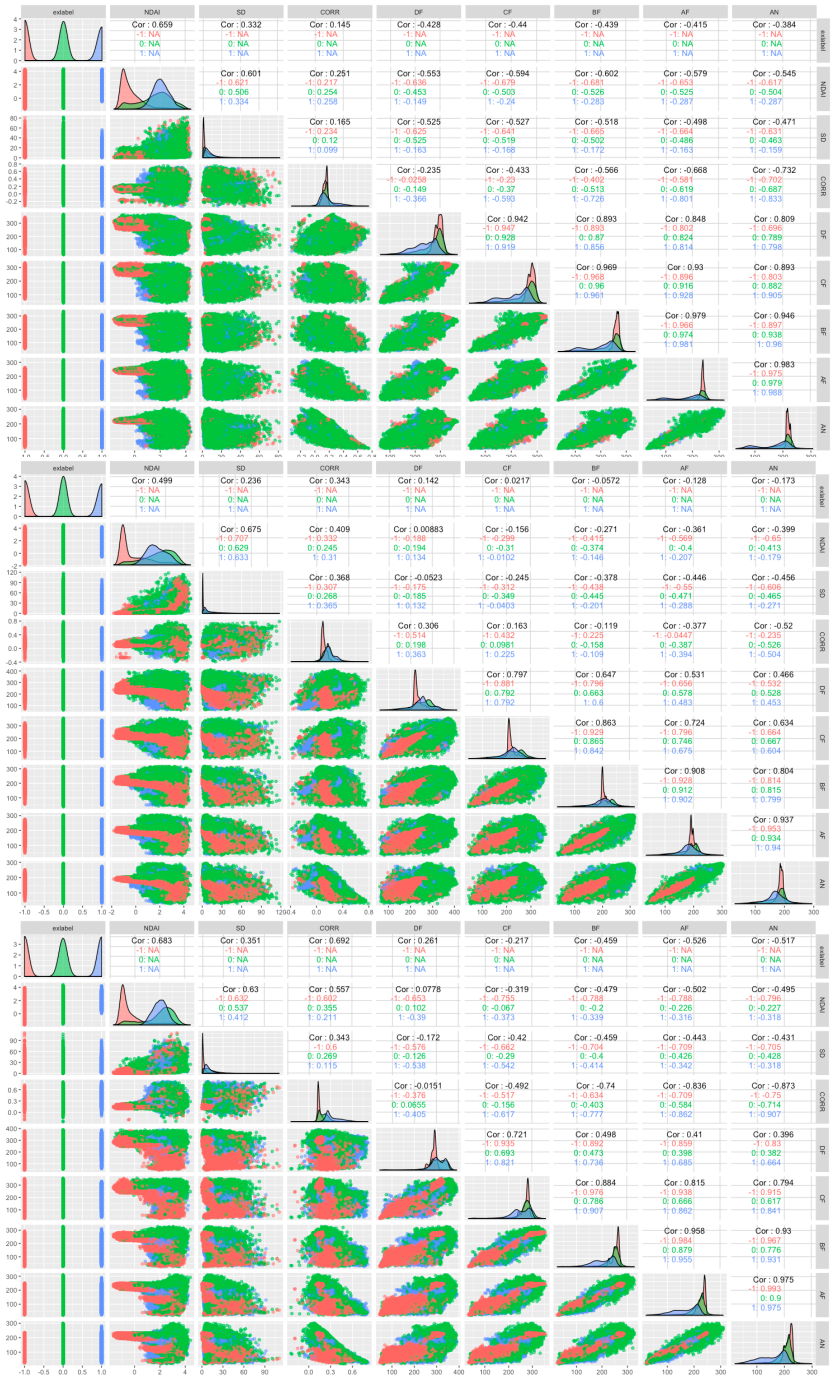
We could further examine the relationship between features. We will be mainly using the scatter plots and density plots to analyse the relationship between individual features and the expert label. It would assist our selection of important features to include in our model and analysis of the importance of these features.

First we study the pairwise plots of the feature data. For the scatterplots, it is not hard to observe that across all three images, there exists linear relationship between every pair of Radiance angle DF, CF, BF, AF and AN while the linear relationship diminishes in image 3. However, for features NDAI, SD and CORR, there exist very little difference when pairing them with different features across three images. This could be because NDAI, SD and CORR are calculated from the other five features by the same formulae.

For the correlation values, there exists a very strong positive correlation within the angle values (DF, CF, BF, AF, AN). This could be because they are taking pictures of the same spot from different angles at the same time. Furthermore, there is no high positive or negative correlation between any of NDAI, SD and CORR with any one of the other angle features. There is also no indicative correlation in the situation of different expert labels for these three features.

For the density plots, we could see the clear two tips of values for two different classes in NDAI while it is difficult to distinguish the clear group of expert labels in SD and CORR plots. This could be due to the ranges of the values are not zoomed in yet since values for SD and CORR might not vary much and the distinction for expert label needs to be selected carefully. Furthermore, the density parts are overlapped for half of the values in the angle plots (DF, CF, BF, AF, AN) and the range of the tip decreases as the angle goes to the center (AF and AN share the same angle in symmetry).

Looking into details, we could see the difference in the tip value of SD and CORR with different expert labels. This could indicate them, together with NDAI as the important features to be included in the model (which will be explained further later). However, we also observe differences in distribution of CORR in different images. This could make us pay attention when dealing with the combined data and test errors.



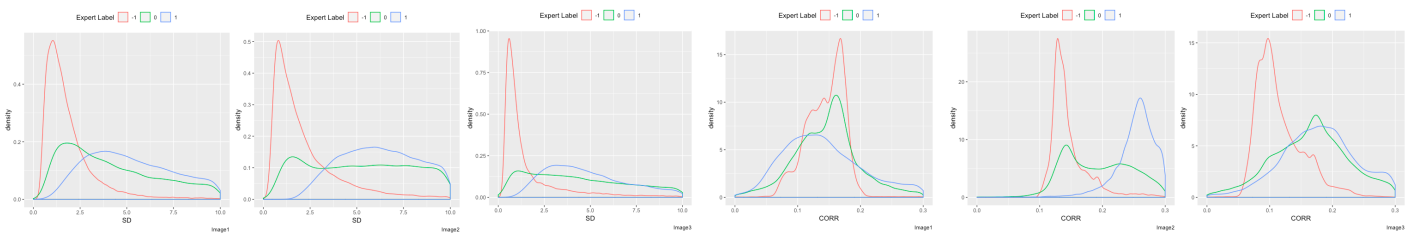


Figure 2 Pairwise relationship for image 1, 2 and 3 (pairwise scatterplot, density plot with expert label and correlations) and detailed density graphs for SD and CORR with expert labels

Preparation

Data Split

To split the data for training, validation and testing, we could use two methods to ensure the data does not follow i.i.d.. The first method is to split the image into 1600 pieces according to their location shown in the x- and y-coordinates and extract randomly from these pieces. This will ensure the togetherness of points in one whole geographic location. We extract 25% of the data for testing and 20% of the training data for validation. The second method is that we use Image 1 as the training set, Image 2 as the test set and Image 3 as the validation set.

Baseline

By setting all the expert labels to -1 in all validation set and test set, we obtain the accuracy as the table below and conclude that 29%-30% would be our baseline.

Set	Accuracy
Test Set 1	60.46%
Validation 1	63.25%
Test Set 2	52.20%
Validation 2	61.37%

First Order Importance

Combining all the images together, we could check the density plots of different features compared to expert label as followed from the analysis of . The most desirable features (first important orders) should be in different values for different expert labels.

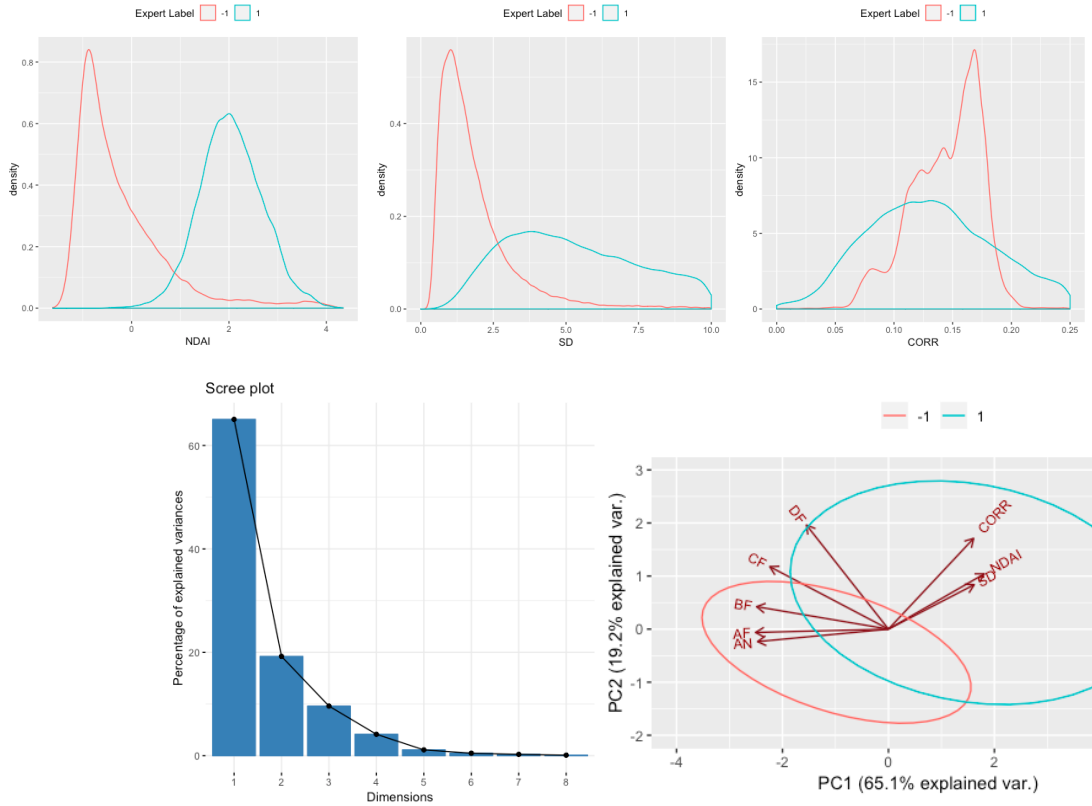


Figure 3 Baseline accuracy as shown above. Density plots of the combined data as shown on the left. PCA Graphs shown below.

From the Principal Component Analysis graphs, we could see that the first two PCs could contain more than 80% of the variation in the data and label -1 points are more tightly clustered compared to label 1 points. It is quite obvious that label 1 points have higher values of DF, CORR, NDAI and SD but lower BF, AF and AN compared to label -1 points. Both of the groups have similar CF values. However, the difference in BF, AF and AN and the difference is more separable in three computed features, CORR, NDAI and SD, which we would choose to be our first order important features for modeling. Lastly, we prepare the **`CVgeneric`** as follow to function to help us test our models.

```
classification_acc = function(fitted, actual) return(mean(fitted!=actual))
genericCV = function(classifier, features, labels, K, loss_func=classification_acc) {
  set.seed(1)
  folds <- createFolds(labels, k = K)
  CVerrors = data.frame(matrix(ncol = 0, nrow = 1))
  for (i in 1:K) {
    fold_index = folds[[paste("Fold", i, sep="")]]
    cv = features[fold_index, ]
    train_set = features[-fold_index, ]
    predicted = predict(classifier, newdata=cv)
    pe_index = paste("Fold", i, sep="")
    CVerrors[pe_index] = loss_func(predicted, labels)
  }
  return(CVerrors)
```

Modeling

We use a five-fold cross validation method to examine different methods. Split 1 and Split 2 refers to the methods of splitting data. The Error table is for Cross Validation error and test error of different folds and Class Table is confusion matrix for correct and wrong classification where rows represents actual and column is for prediction. We also calculate the error for weightage of false classification over the right ones. We will also compare the performance of different split methods and make a conclusion on their stability in the end.

We will examine the assumption of different methods together with its performance using test error and confusion matrices. This will help us find the most suitable model for further diagnosis.

K-Nearest Neighbour (KNN) Method

The basic idea of KNN is to compute distances from new data point to exist clusters and classify the new one to the nearest cluster. From which, we can see that KNN needs data to train models in advance. The advantages of this method are it doesn't need extra information of data, and needs relatively small calculation. In our case, clouds tend to stay together and it is the same for the clear space hence KNN could potentially work well. The constraint is that it doesn't work well when boundaries are blur, which could limit its performance in our case where boundaries could be important.

Split 1

CV Error 1	8.64%
CV Error 2	8.70%
CV Error 3	8.60%
CV Error 4	8.57%
CV Error 5	8.51%
Test Error	9.00%

Class 1

	-1	1	Error
-1	31438	3183	10.12%
1	1751	18470	9.48%

Split 2

CV Error 1	13.82%
CV Error 2	13.89%
CV Error 3	14.36%
CV Error 4	13.69%
CV Error 5	13.74%
Test Error	10.33%

Class 2

	-1	1	Error
-1	38303	4579	11.95%
1	3909	35357	11.06%

Figure 4 KNN method results

Our test error for Split 1 is 9.00% and for Split 2 is 10.33%. We have observed Split 1 has lower CV error (by around 5%) than Split 2, which is higher than its test error. This might be due to smaller validation size compared to the our large test set (one full image). In overall, KNN method has pretty good performance due to the low test errors. Split 1 performs better compared to Split 2.

For the confusion matrices, we could see most are classified correctly but the model classifies -1 (no cloud) as 1 (cloud) more than 1 as -1. This might attribute to the overlapping area between label -1 and label 1 as shown in the Figure 3 PCA graph. Split 1 performs better compared to Split 2 in terms of the weightage of false classification over the right one.

The advantage of using KNN method is that we do not need to make any typical theoretical assumptions on it and this is very useful in the real world cases, such as the prediction of existence of cloud in our case.

Logistic Regression

The basic idea of logistic regression is linear regression but it does not expect linear relationship between the dependent and independent variables. It does not make many of the key assumptions of linear regression and general linear models in ordinary least squares algorithms (OLS) such as linearity, normality, homoscedasticity, and measurement level. By setting a threshold manually, we transform regression into a classifier. This could work well in our case since the existence of cloud has reflected difference values in our first order important features hence they could exist some relationship between the expert label and these features.

Logistic regression requires data to be independent of one another, this might limit its usage in our case where our data are not independent of one another. We obtain the following results.

Split 1

CV Error 1	10.82%
CV Error 2	10.83%
CV Error 3	10.47%
CV Error 4	10.91%
CV Error 5	10.87%
Test Error	11.24%

Class 1

	-1	1	Error
-1	27110	3276	12.08%
1	2594	19267	13.46%

Split 2

CV Error 1	13.87%
CV Error 2	13.56%
CV Error 3	14.13%
CV Error 4	13.56%
CV Error 5	13.85%
Test Error	7.06%

Class 2

	-1	1	Error
-1	38991	1906	4.49%
1	3891	37360	10.41%

Figure 5 Logistic regression method results

The test error is 11.24% for Split 1 and 7.06% for Split 2. Split 1 has lower CV error (by around 3%) than that of Split 2 where its test error is much smaller compared to its CV error. It might not be credible to use the test error here to comment on performance of Split 2. The error rate is higher compared to that of the KNN method.

Also we could see from the confusion matrix that for Split 1, more points are classified wrongly from 1 (cloud) to -1 (0 in the model for no cloud) which the error for wrong classification is all higher compared to the KNN method (Split 1). In overall, logistic regression doesn't perform as good as KNN method. It is hard to tell which split method has done better than the other from the matrices.

It is noteworthy that logistic regression require the assumption on independent dataset which limits its usefulness in the real world cases. The model also shows the limits in our case since our data are not i.i.d..

Linear Discriminant Analysis (LDA)

The basic idea of LDA is project clusters down to one dimension, and then imagine it create seperate probability density function for each cluster, what we need to do is to maximize the differences between them. It could be used for multi-class classification but it could be unstable with non-linear separable samples.

The assumption of LDA is that data are drawn from multivariate Gaussian model with same covariance matrix, so we first check whether the three variables we have chosen satisfy the assumption.

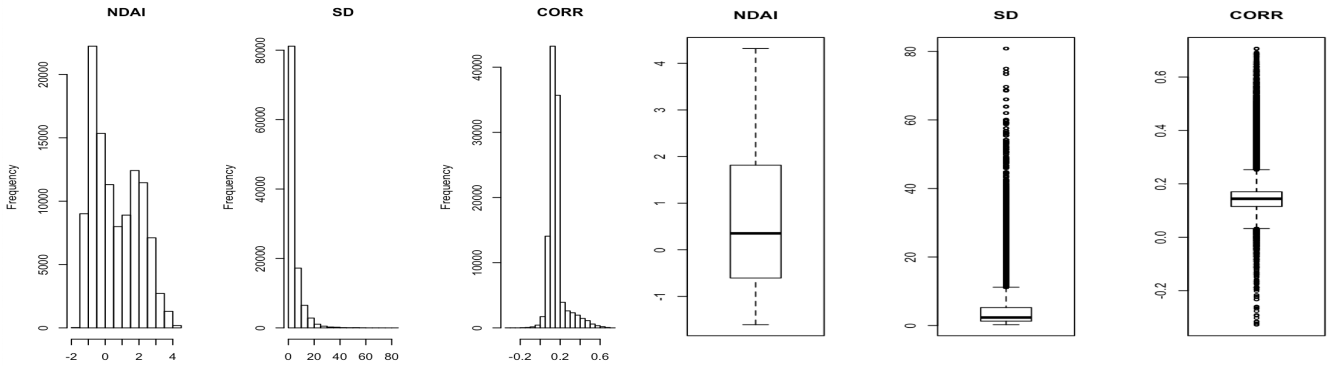


Figure 6 Checking for Gaussian distribution of data

We can see the three variables roughly show Gaussian distribution. We then train it using logistic regression and obtain the following results.

Split 1	
CV Error 1	10.27%
CV Error 2	10.29%
CV Error 3	10.02%
CV Error 4	10.29%
CV Error 5	10.29%
Test Error	10.59%

Class 1

	-1	1	Error
-1	26908	2737	10.17%
1	2796	19806	14.12%

Split 2	
CV Error 1	13.19%
CV Error 2	12.86%
CV Error 3	13.35%
CV Error 4	12.89%
CV Error 5	13.00%
Test Error	6.41%

Class 2

	-1	1	Error
-1	38807	1189	3.06%
1	4075	38077	10.70%

Figure 7 LDA method results

The test error for Split 1 is 10.59% and that for Split 2 is 6.41%. Split 1 has lower CV error (by around 3%) than that of Split 2 where its test error is much smaller compared to its CV error. It might not be credible to use the test error here to comment on performance of Split 2. The error rate is higher compared to that of the KNN method and logistic regression.

From the confusion matrices, more points are classified wrongly from 1 (cloud) to -1 (0 in the model for no cloud) which the error for wrong classification is all higher compared to the KNN method (Split 1). In overall, LDA doesn't perform as good as KNN method.

Still, given the assumption of data following Gaussian distribution of the same covariance matrix, this model might not work well with our dataset and many real world cases.

Quadratic Discriminant Analysis (QDA)

QDA method has almost the same intuition as LDA but it assumes data are drawn from multivariate Gaussian distribution of different covariance matrices, giving more freedom to the data used. We obtain the following results.

Split 1	
CV Error 1	10.33%
CV Error 2	10.11%
CV Error 3	10.08%
CV Error 4	10.32%
CV Error 5	10.54%
Test Error	11.23%

Class 1			
	-1	1	Error
-1	27305	3469	12.70%
1	2399	19074	12.58%

Split 2	
CV Error 1	14.10%
CV Error 2	13.72%
CV Error 3	14.33%
CV Error 4	14.00%
CV Error 5	13.98%
Test Error	5.01%

Class 2			
	-1	1	Error
-1	39263	495	1.26%
1	3619	38771	9.33%

Figure 8 QDA method results

The test error is 11.23% for Split 1 and 5.01% for Split 2. Split 1 has lower CV error (by around 4%) than that of Split 2 where its test error is much smaller compared to its CV error. It might not be credible to use the test error here to comment on performance of Split 2. The error rate is higher compared to that of the KNN method, logistic regression and LDA (but very close to LDA). This might be due to very similar assumption in these two methods.

From the confusion matrices, we could see that there is almost an equal number for both wrong classification rate.

Random Forest

The basic idea of random forest is decision tree. By growing multiply trees instead of a single tree, random forest guarantees the accuracy of classification. Comparing to bagging, random forest only use subset of features, which reduce the calculation enormously. No extra information or assumption of data is needed. The parameters used here is all default values of R function. We trained 500 trees, use 1 variable at each split

Split 1	
CV Error 1	8.21%
CV Error 2	8.25%
CV Error 3	8.13%
CV Error 4	8.32%
CV Error 5	8.34%
Test Error	8.18%

Class 1			
	-1	1	Error
-1	27360	1928	7.05%
1	2344	20615	11.37%

Split 2	
CV Error 1	9.74%
CV Error 2	9.80%
CV Error 3	10.22%
CV Error 4	9.80%
CV Error 5	9.69%
Test Error	6.77%

Class 2			
	-1	1	Error
-1	75997	8201	10.79%
1	4263	37452	11.38%

Figure 9 Random Forest results

The test error is 8.18% for Split 1 and 6.77% for Split 2. Split 1 has lower CV error (by around 1%) than that of Split 2 where its test error is much smaller compared to its CV error. It might not be credible to use the test error here to comment on performance of Split 2. The error rate is lowest among all methods based on its credible Split 1 method.

From the confusion matrices, we could see that more points are classified wrongly from 1 (cloud) to -1 (0 in the model for no cloud) which the error for wrong classification is all lower compared to the KNN method (Split 1).

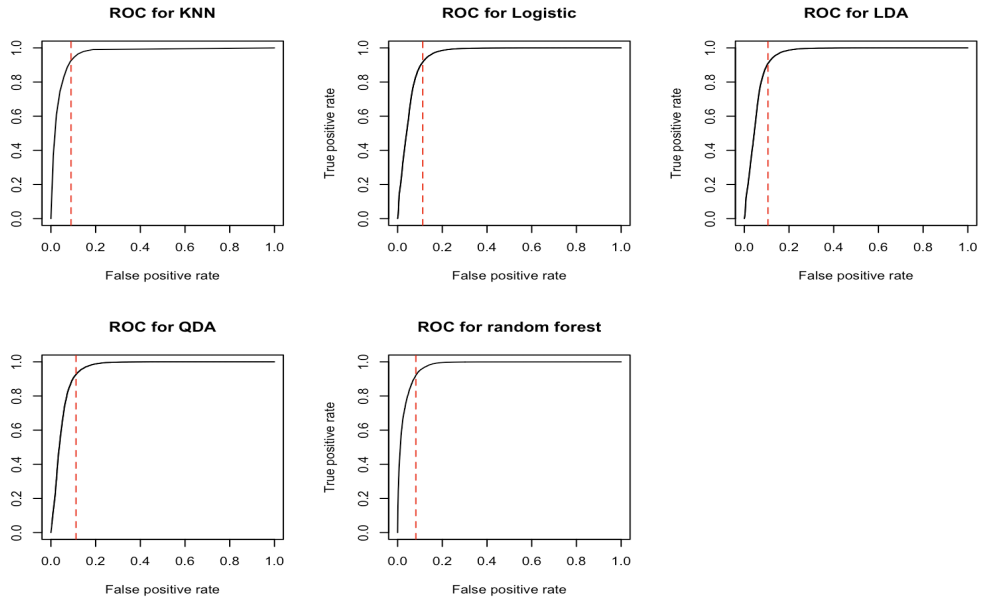
Other Ways to Evaluate the Classification Models

Receiver Operating Characteristic Curves

We choose Split 2 to generate ROC curves, the cut-off values are test error of different methods which is also the point that is nearest to the point (0, 1). This is the best point to be selected in the graph since it achieves the possibly achievable lowest False Positive (FP) and highest True Positive (TP) at the same time, minimising the effect of the trade-off between the two. Plot all curves in one graph to select best classifiers.

About cut-off points, we write a function to find the point which is closed to the (0,1) point on ROC curve. The function works good on logistic, LDA, and QDA. When it comes to KNN method, the function output negative value which can't be cut-off point, so we just use the test error as cut points in KNN. We mark the cut-off points with dot red lines. You can see the intersections are the points we selected.

We also put all curves in one plot to help choose the best classifiers. It's clear that KNN and random forest had the best performance.



	AUC
KNN	0.9647
Logistic	0.9502
LDA	0.9482
QDA	0.9517
RF	0.9734

Figure 10 Individual ROC curve of different methods and their Area Under the Curve

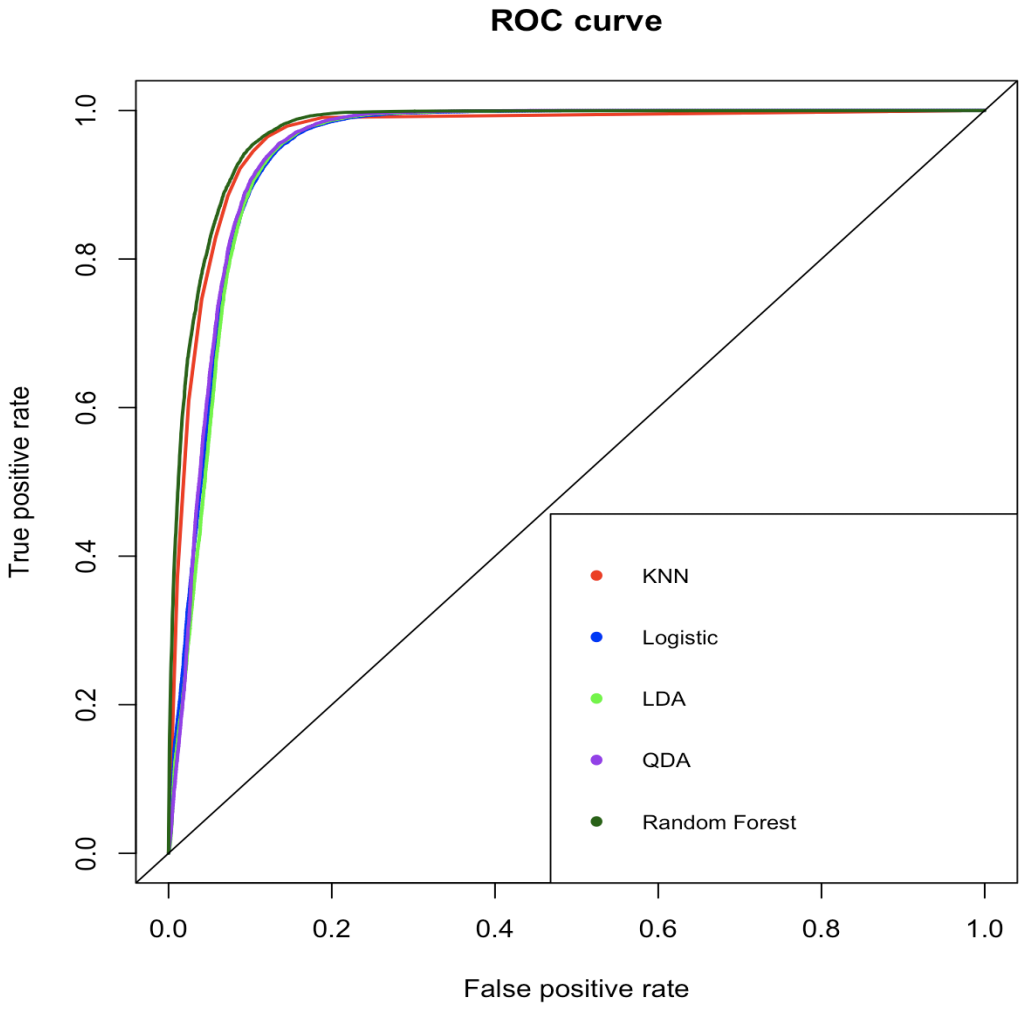


Figure 11 Combined graph and the AUC summary of different methods

Area Under the Curve

We further examine the Area Under the Curve (AUC). AUC measures the area of the ROC plot that is underneath the curve. It is useful as a single summary of the model performance. Our result is as shown in Figure 10.

Considering performances of four models, we select KNN and random forest as our best classifiers. One of the reasons why these two models perform good may be they don't need extra information of data, while the others need

Gaussian data. In diagnostics we will compare the two methods in multiply aspects. We will not the model from random forest since it could be computationally consuming.

Summary

In conclusion, different models have demonstrated different accuracy and they are quite different in the accuracy rates and classification errors in terms of false classifications. This has helped us understand different performance of different methods and the limitations especially those from assumptions. KNN and random forest which do require any assumption on data distribution have hence stood out given our data are obviously not from i.i.d. distribution. Also, we only compare the confusion matrix in magnitude instead of the rate of false classification. We will further examine this in the ROC section with better visualisation and the improvements of the models.

Furthermore, split method 2 (separate by Image 1, 2 and 3) has performed quite unstably among different methods showing much lower test error compared to CV errors compared to split method 1 (splitting into different blocks and select randomly). This might be due to the wholesome of data could provide similar information compared to other images provided while CV divide the image into different unknown blocks. Therefore, we will continue using split method 1 to further our study in evaluating different models we have presented.

Diagnostics

We will adjust our hyperparameter in KNN method to find the convergence and identify the optimal level. We will also find any misclassification pattern in different images and try to find the root causes. Lastly, we will propose one stable and better classifier, Support Vector Machines (SVM) demonstrating using a smaller sample.

In-depth Analysis of KNN

Parameter select

Considering that KNN method only has one parameter k, we did in-depth analysis begin with change the value of k, and try to find some pattern. With k from 1 to 30, the test errors of Split 1 are 9.00%.

It's easy to discern that test error decreases as the value of k increases. We had discussed the core idea of KNN method, the new point is classified to the nearest cluster according to the distance from nearest k labeled points. If value of k is too small, the model is easy to be disturbed by outliers and misclassification; if the value of k is too big, the model will be underfitting and the calculation is a hard work. So we now use 'elbow method' to pick the optimal k value. There is an 'elbow' near 10, so we choose k=10. What's more, it's safe to say that error converge to around 0.94 as the k increases to big value.

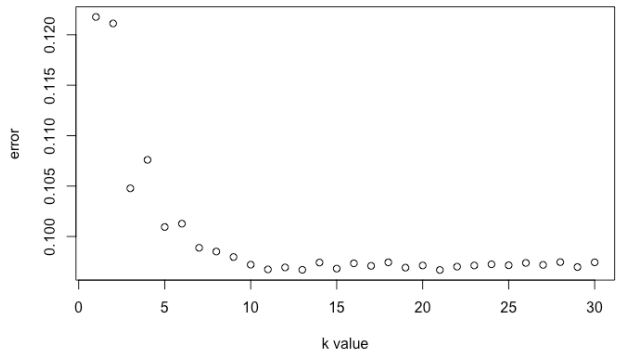


Figure 12 Error reduces as k increases

Pros & Cons

Besides, we try to find why KNN works well here. As the below graph shows, different label points gather together, and have clear boundaries, which perfectly satisfies the preconditions KNN need.

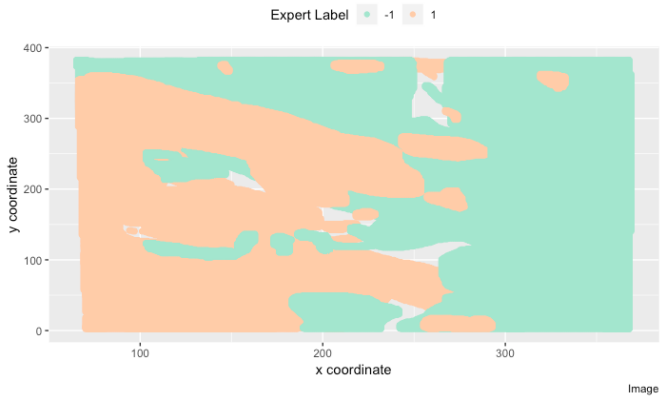


Figure 13 overview

KNN also has some disadvantages. First, it doesn't build a real classifier, in other words, it has no output. That's why KNN often used as the cornerstone of following models. Second, KNN needs larger computation compare to logistic, LDA, and QDA. This would be growingly costly with larger k values (trade-off between accuracy and model complexity).

Misclassification Pattern

Use PCA to visualize the misclassified points

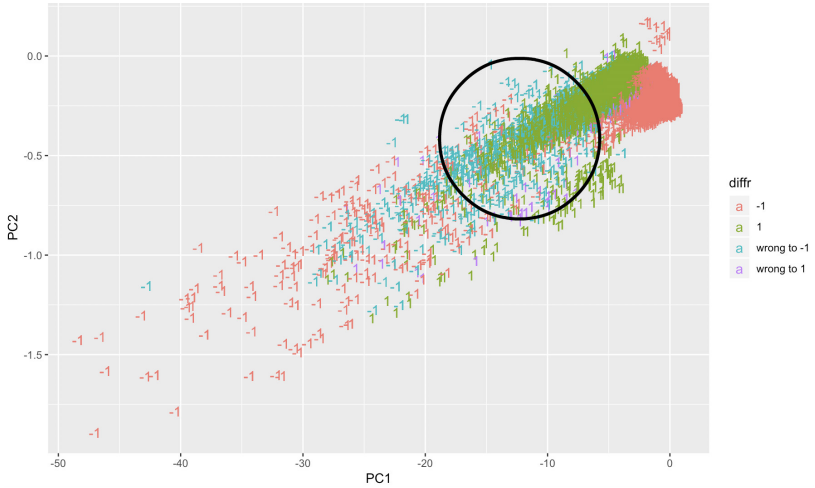


Figure 14 Misclassified point PCA

We can see most misclassified points were classified to -1 label. What's more, the blue points gathered in black circle.If we classify manually, it's natural to classify them to label 1. However, the machine was may influenced by the long tail of label -1 laid on the left bottom of graph.

Besides, we also plot wrong class points according to x- and y- coordinates. The purple points are points misclassify to label -1. They gathered in black circles.

Figure 15 Wrongly classified points distribution in the image

Quantitative characteristics of wrong points

We select wrongly classified points and draw distribution plots and boxplots.

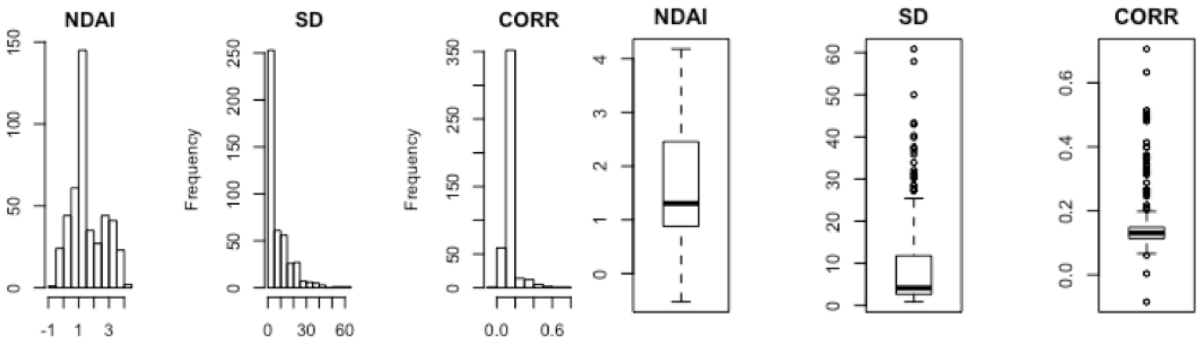


Figure 16 Distribution analysis for wrongly classified points

Compared with Figure 5, we can find some different patterns:

1. NDAI of wrong points gathered in range 1.5 to 2.5.
2. The SD variable doesn't have very big outliers except one.
3. CORR is quite similar to the distribution of whole data.

Better Classifier

Cutting off the Long Tail

Our mainly thought is to cut off the long tail which is mainly -1 label in PCA graph, and implement random forest.

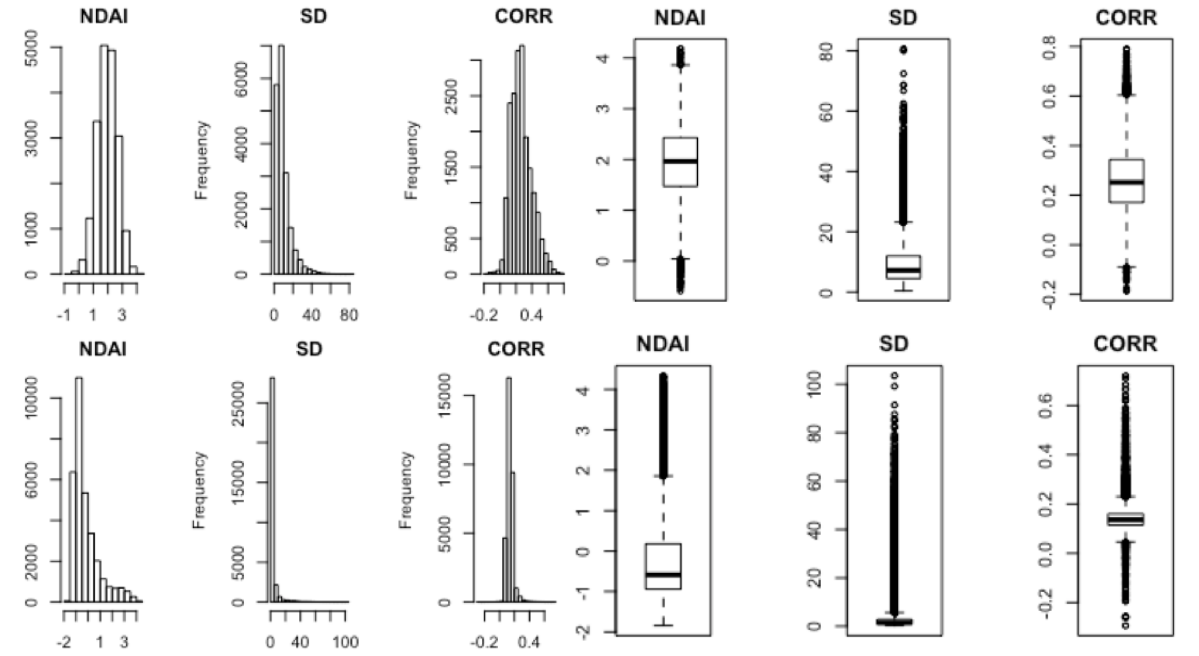


Figure 17 Data Distribution for label 1 (up two) and label -1 (bottom 2) points

For these three variables, we observe the difference:

1. NDAI of -1 label has negative value compare to non-negative NDAI of 1 label.
2. SD of -1 label have outliers bigger than 40 compare to 1 label.
3. For CORR, the labels are similar.

From previous graph, we choose -15 on coordinate PC1 as threshold, and then implement random forest on on-tail data and the test error is around 8.47%.

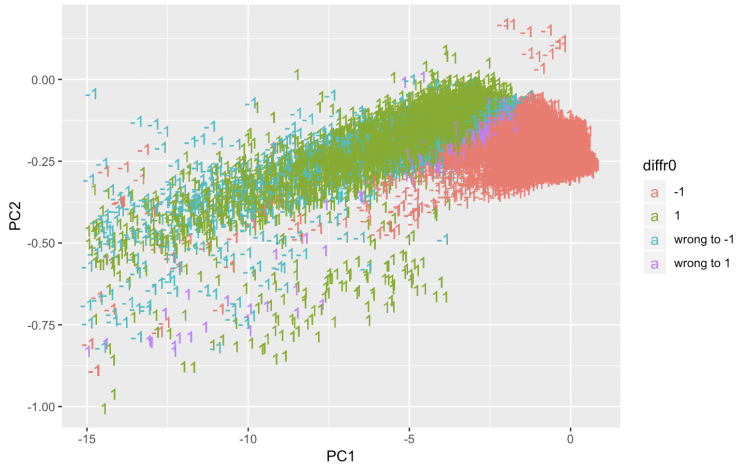


Figure 18 Random Forest after modification in PC

The result is slightly better than previous, but considering that it's not the whole part classification, we don't think the hypothesis we made is extremely valuable. The tail may influence the classifier but not significantly hence the model only improve in a very limited scale. From the visual graph, we can also see that misclassified points lying on boundaries and particular region is not changed.

Support Vector Machines

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane.

Due to the constraints of our machine, we have tried a tiny sample from Image 1 by splitting method 1 (5% of the 1600 blocks created). We only tried to use the radial kernel with cost 5 hence we could see the immediate result and generate the following graph

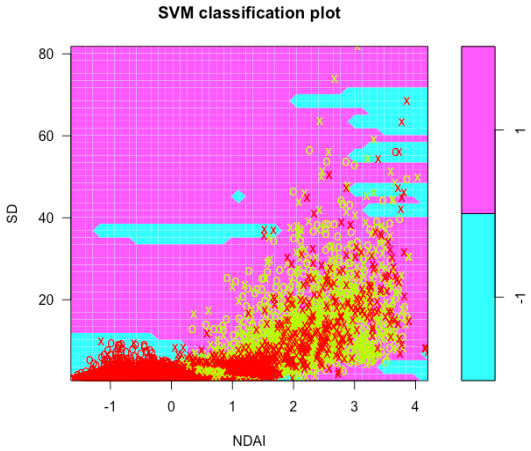


Figure 19 SVM Method graph

We have calculated the test error as 6.81% and this has performed relatively better compared to KNN, logistic regression, LDA and QDA while it is quite comparable to random forest. As a result, we propose this being a better classifier for the dataset if powerful machines could carry on for the large data set.

Different Split

We already seen in modeling part that different split have influence on the final results. In general, split 2 has larger cv errors than split 1, but because split 2 use the whole file 2 as test set, which leads to smaller test error. In this part, we try to use file 1 or file 3 as test set and the rest two as train set to see how error will change.

Fit 1 used file 1 as test set, fit 2 used file 3 as test set. RF fitted is confusion matrix of model, RF predicted is table of model fit in test set. Test error are generally higher than error in confusion matrix. Used file 3 as test set has larger error.

	RF fitted 1				RF predicted 1		
	-1	1	Class. Error		-1	1	Class. Error
-1	69476	7158	9.34%	-1	46993	4278	8.34%
1	4674	55836	7.72%	1	3453	16193	17.58%

RF fitted 2				RF predicted 2			
	-1	1	Class. Error		-1	1	Class. Error
-1	87650	5678	6.08%	-1	28986	5940	17.00%
1	1874	57863	3.14%	1	4766	15304	23.75%

Figure 20 Effects of different splits: confusion matrices by random forest

Compared with the previous results using Split 2, there is very little difference where the false classification rate is (9.74%, 11.28%). Therefore we could probably conclude that there might be limited changes if we simply modify the way of splitting, especially given that Split 2 might not be a good method to demonstrate the power of models.

We did in-depth analysis of KNN in split 1, here we redid the same analysis on split 2.

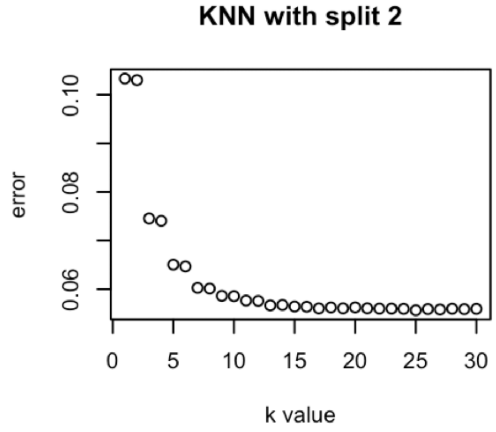


Figure 21 KNN convergence for modified Split 2

There are some different errors with same k value, however, the convergence doesn't change. We can still see that error decrease with the increase of k value.

Conclusion

We tried 5 models: KNN, Logistic Regression , LDA, QDA, and Random Forest. The best two methods with smallest error are KNN and random forest. This could be explained by their relatively less constraints on the assumption of the data (such as independence for logistic regression and Gaussian distribution of data for LDA and QDA). We also have noticed that considering calculation complexity, these two methods are not as easy as the other three. KNN is probably the simplest classify, and it proves work pretty well even for complex question but it does not give a concrete boundary. We also discuss how to choose the appropriate the hyperparameter k in the Diagnosis section, and use PCA to show that relatively clear boundary may serve as a possible reason for why KNN does well here.

Furthermore, we found that misclassified points have some quantitative and visual patterns using distribution plots and boxplots. A hypothesis was made here stated the long tail contribute to the misclassify -1 label. We test the hypothesis and prove it to be rejected, and then come up some idea about how to build a better classify using SVM. However, SVM would require very large amount of computational work and by our small sample, it has led to relatively stable and low test errors.

Lastly, we change our split, separately using Image 2 as test set when the rest two as train set. By implementing random forest with these different splits, we show that way of splitting data can modify our model enormously.

Our project has demonstrated the performance of popular classification method on the expert labels for cloud classification. It challenges the assumptions of some classification methods and those without these constraints stand out. We believe that based on the

Acknowledgement

Huang Liang has done first three parts and formatting. Wei Mian has done the third and fourth parts Modelling and Diagnosis. Our Github link would be: https://github.com/Auroraaah/STAT154-project2 where the readers could access our code the reports.