Lab 4 - Cloud Prediction
Stat 215A, Fall 2014

Fanny Perraudeau, Rafael Valle and Sören Künzel

Abstract

In this project, we have explored statistical models for daytime cloud detection in the polar regions using radiances obtained by the MISR sensor aboard the NASA satellite Terra. We propose a supervised binary model that classifies an observation as cloud or clean, and provides a confidence score. The features in the data set include Normalized Difference Angular Index (NDAI), Correlation between radiances, Standard Deviation of nadir camera pixel values across a scene, and radiances from 5 different angles [2]. During the exploratory data analysis, we investigate the correlation and distribution of these features and their relationship to two classes (cloud and clean). After drawing conclusions from visual and numerical cues, we submit all features to different feature selection methods. Several linear and non-linear statistical models and algorithms are explored and validated against one another. They include Enhanced Linear Correlation Matching, Logistic Regression and Classification and Regression Trees.

1 Introduction

1.1 Binary classification of a semi-discrete response variable

The task of building a supervised model to classify some data as cloud or clean raises questions related to possible gradations of cloudiness and its relationship to the features space. There are many ways to address this relationship. For example, given some balanced cloud and clean data for some location, computer vision transformations can be used to interpolate between the images at a desired step. This interpolation could be used to transform the binary variable into a semi-discrete space. Although this is related to models that provide a confidence score of classification, the interpolation provides an explicit relationship between the cloudiness predicted and the data. In adddition, the distributions of the features are dependent on when and where the data is collected. Naturally, although interesting as a project, addressing all these issues goes beyond the scope of this project. However, we'll interpolate between the likelihooods of each model given data to provide a score of cloudiness. Details will be covered further on this paper.

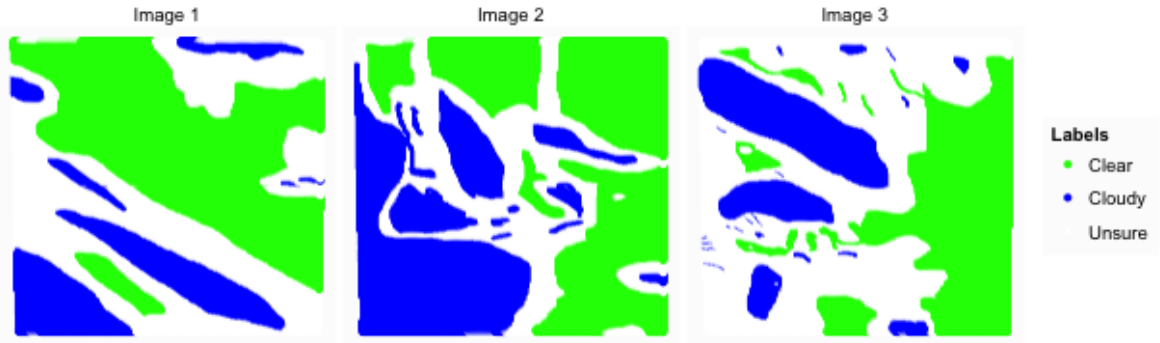

Figure 1.1: Cloud distribution from expert labels

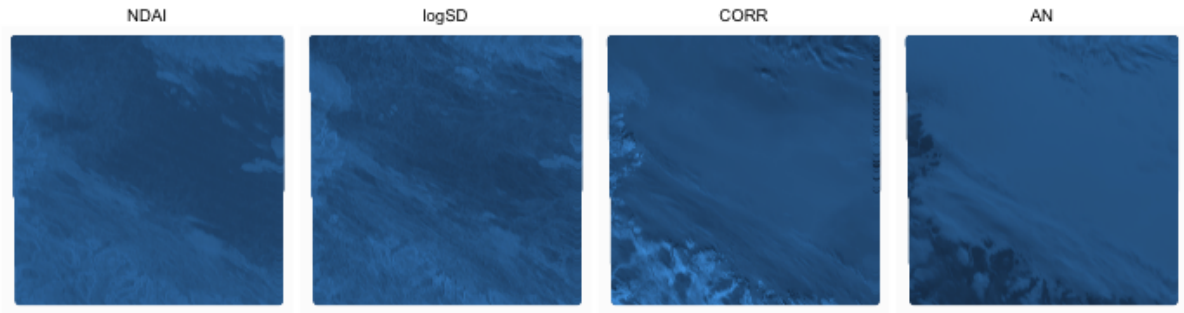

Figure 2.1: Distribution of selected features for Image 1.

2 EDA

2.1 The Features

The aim of the study is to detect clouds on images from the Arctic. It is a challenging problem because clouds and snow-covered surfaces have almost the same proprieties. Unlike traditional image studies that would use only one picture, we used here five cameras with each camera viewing Earth scenes at a different angle. The pictures were taken by the Multiangle Imaging SpectroRadiometer (MISR) on board the National Aeronautics and Space Administration (NASA) Terra satellite launched in 1999. The five view zenith angles of the cameras were 70.5°(DF), 60.0°(CF), 45.6°(BF), and 26.1°(AF) in the forward direction and 0.0°(AN) in the nadir direction. Note that in the paper[2] they used nine cameras instead of five. The data produced by the cameras DF, CF, BF, AF and AN were taken as features in our study. We also used three other physically useful features characterized in the paper: CORR the correlation of MISR images of the same scene from different MISR viewing directions, SD the standard deviation of MISR nadir camera pixel values across a scene and NDAI a normalized difference angular index that characterizes the changes in a scene with changes in the MISR view direction. Finally, the last feature we used was the logarithm of SD (logSD) in order to correct the long tail of this feature. All in all, we studied nine features: DF, CF, BF, AF, AN, CORR, NDAI, SD and logSD.

2.2 Comparison of radiances given cloud or clean

Knowing that the MISR satellite crosses the angles Df, Cf, Bf, Af and An within a short-time span and knowing that these angles correspond to angles 70.5, 60, 45.6, 26.1, 0.0 respectively, we compare the intra and inter-class correlation of the radiances under two assumptions:

1. The correlation between radiances is inversely proportional to the distance between angles
2. The correlation between radiances of clear images is higher than cloud images

With the purpose of analyzing these assumptions, we plot a correlation matrix of the radiances of all three images (figure 2.2). The overall correlation of radiances in clean obversations is higher than cloud observations. As the angle distances increase, the correlation decreases, which confirms the first assumption. In addition, the smallest correlation in cloud data is .36 less than the smallest correlation in clean data, which confirms the second assumption. This is in consonance with Yu's paper [2] and is the reason, why we consider variables like the inter Correlation between features and the NDAI to distinguish between the two populations. We will investigate this matter in the next paragraphs.

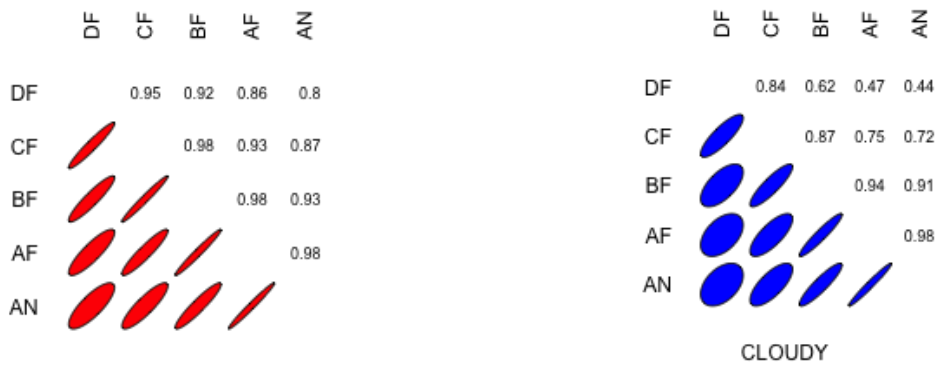

Figure 2.2: Correlation between radiances of cloud and clean observations

2.3 Principal Component Analysis

We performed Principal Component Analysis (PCA) on the five radiances for each of the images. Averaging over the three images, the first two principal components (PC) captured nearly 98% of the variance. The first principal component (PC1) was close to the average over all the radiances. It is interesting that the second principal component gave positive loading to the DF and decreased monotnically over CF, BF, AF to AN. So that AN had a negative loading. switching the sign in each loading would still give us the same component, but this second component seems to somewhat characterize how much the Radiances of the same pixel differ from each other. This has somewhat the flavor of a correlation and might be explained by the fact that pixels for which the second component is very high, are these for which the correlation between the features is also high. This motivates the definition of NDAI. [1]

We explored two ways of calculating principal components: We calculate for each image its own PC or we calculate globally PC and we realized that both ways are very similar, so that it is possible to use the rotation matrix found for one image on other images to get reasonable good Components. However, for images 1 and 2, we received a high correlation between the expert labels and PC2, but, for image 3, the correlation did not seem high enough to predict the labels. Keeping in mind that we we want to build predictors for other pictures, we decided not to take into account the principal components, because they just don't seem to be more useful than for example NDAI.

2.4 Distribution of the features

For each image, we plotted the distribution of the features NDAI, log(SD), CORR and AN. Furthermore, we seperated the distribution between clear data (-1) and cloudy data (1). We realized that CORR was not an efficient predictor as expected : for each image, the CORR data seem to behave very differently. For example, the first peak in the second and third images belongs to clear areas and in the first image it belongs to cloudy areas. This clearly cinforms that CORR alone is not a good predictor. AN did a slightly better job, but its entire distribution on the third image seems to be shifted to the left (Figure 2.3).

We speculate that some other effect, such as the sun, has influenced the image. Under the assumption that there are effects which mainly shift or linearly rescale the entire distribtuion of an image, we concluded that for some features it makes more sense to scale by image seperately. Thus, we are going to consider these feature scaling approaches:

- Scaling image by image seperately. We will refer to this by "standard scaled" and
- Scaling over the entire data set. We will refer to this by "image scaled".

If we compare, for example, the distribution of AN in Figure 2.3 and 2.4, we can see that AN seems to have more similar distributions in the case, where the features were scaled image by image.

[1] Refer to Bin et al.[2] for a definition of NDAI

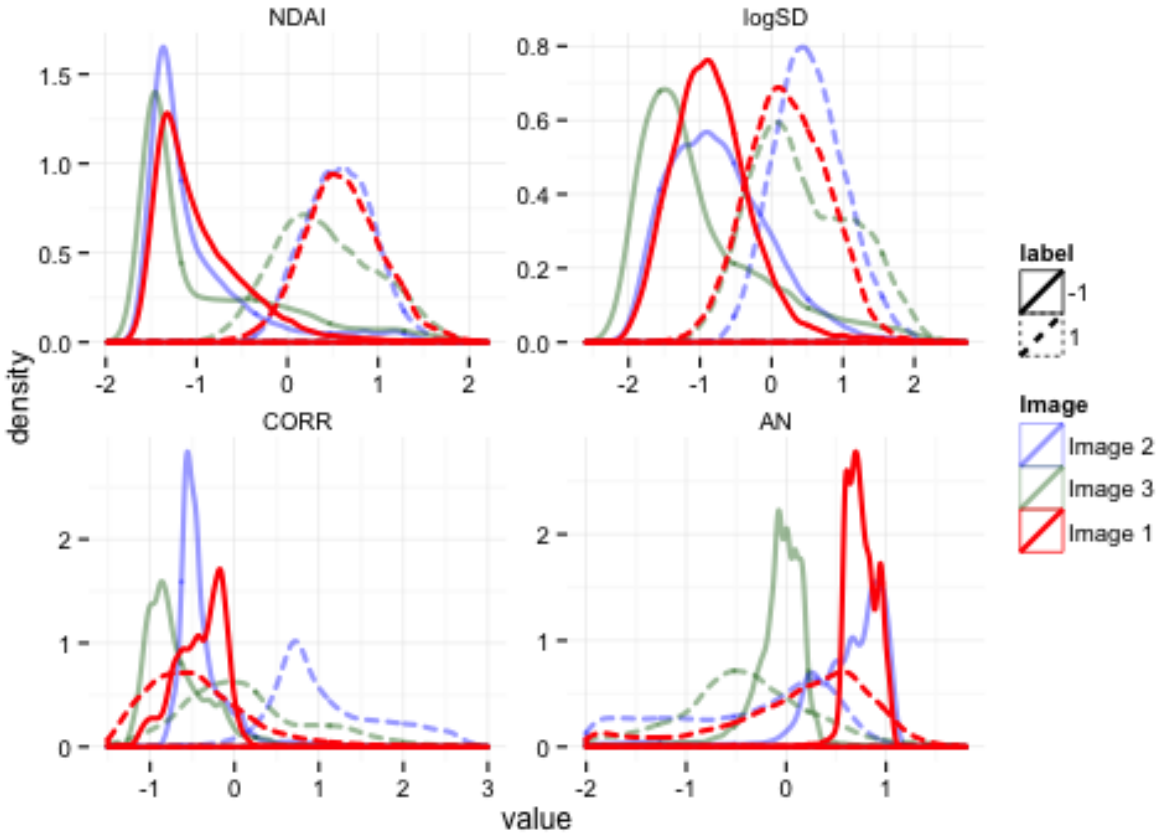

Figure 2.3: The graphs shows the distribution of four selected features scaled by the entire dataset. I1: Image 1, label -1 means that the expert labled it as clear, label 1 means the expert labeled it as cloudy.

Scaling the first image seperately from the other images might lead to better results, because we could probably correct for bias due to some effect that influences the entire picture (for example different sun positions). However, if the distribution in two pictures is very different, a seperate scaling might make things worse. For example, if we scale a completely cloudy and a completely clear picture, we would get distributions for both images distributions with mean 0 and we might not be able to tell that the first image was completly cloudy.

In addition, it is important to mention that our goal is to come up with a classifier that performs well in any image. This classifier will be trained on these three images. Once it is set it does not make much sense to rescale the entire distribution with every new image we see. Instead we were thinking about scaling by the mean and standard deviation of the distributions of these three images. But we realized that for none of our classifiers it really makes a different whether the entire data is scaled by the same mean and the same variance. Furthermore, scaling, for example, the variable SD, destroys some basic relationships between SD, CORR and NDAI and it would give us negative values, which we don't want to have for a variable, which represents a standard deviation. Finally, we scaled for some classifiers image by image and for some others, we did not scale at all.

In our analysis, NDAI is by far the best predictor. The authors in Yu's [2] paper affirm that NDAI varies too much for each image and thus is not a good predictor. However, we cannot confirm this due to the discrepancies between our and their sample size. We assume that we are looking at a subset of images for which the NDAI does not vary a lot and the variable CORR however does. Before applying our proposed methods to other images, we should first check whether or not the NDAI can be assumed to have a similar distribution. Otherwise, scaling the NDAI distribution image by image, might be a good solution.

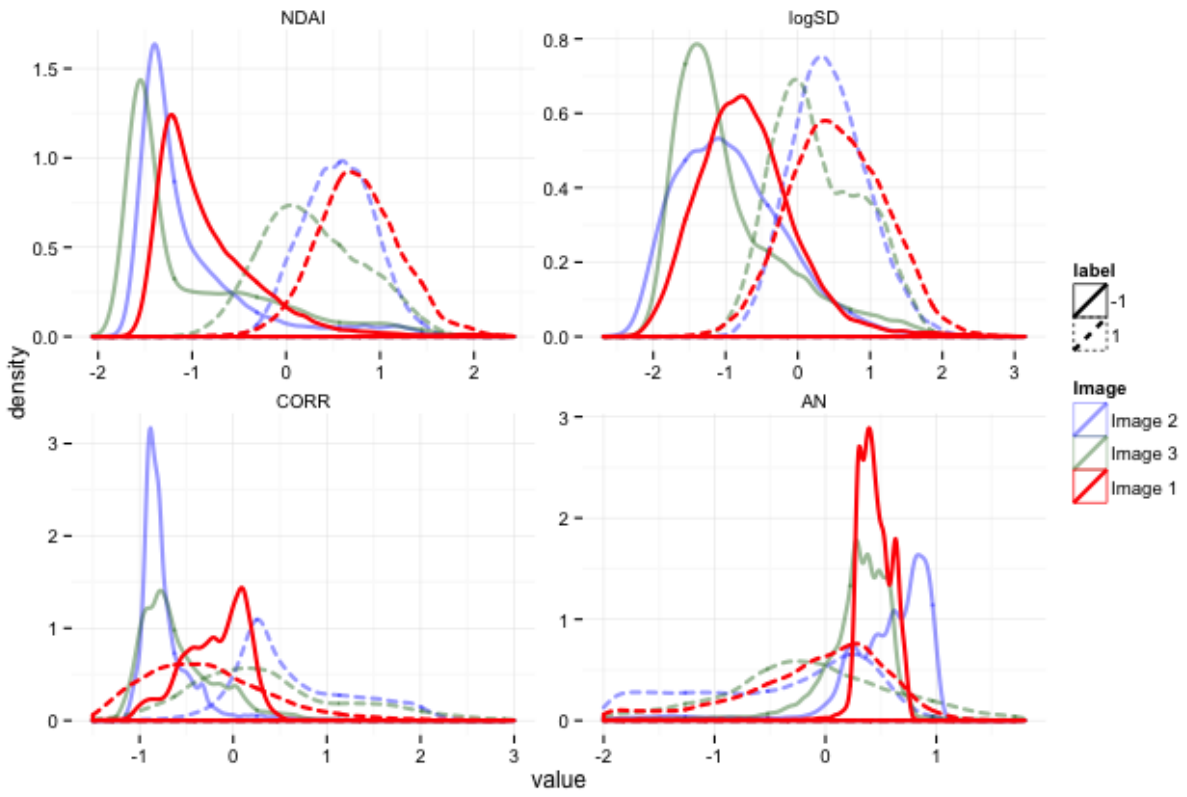

Figure 2.4: The graphs shows the distribution of four selected features scaled by each image seperately. I1: Image 1, label -1 means that the expert labled it as clear, label 1 means the expert labeled it as cloudy.

2.5 Feature Selection

After deciding that NDAI is one of the most important features to use for prediction, we tried to find another feature, which could be used for classification). We tried to predict cloudy data only based on NDAI and analysed the misclassified reagions. Visually, it was not easy to infer which other variable was the second best predictor. We considered log(SD) to be a strong candidate, because it seemed contain a lot of information NDAI does not contain.

Looking at figure 2.5, we can confirm these findings. Most of the seperation can be done with NDAI. Other features are less efficient to seperate the two populations. However, according to the plots, log(SD) seems to be the second most efficient feature. We confirmed that this is also possible for image one and two in figure 2.6.

On the contrary, this high efficiency of NDAI and logSD could be a byproduct of the high correlation, meaning that using both features yields small increase in information and thus in possibilities to train estimators. In fact, the correlation between NDAI and logSD is very high, .82, while other features have an absolute correlation of at most .5. In other words, we might conclude that logSD is also an important variable, but we do not really get better predicting proberties by using both logSD and NDAI over using just NDAI.

Another strategy is to use the feature which has the lowest mutual information with NDAI. Although this would work very well for categorial features, it is general very difficult to estimate the mutual information of continuous features and we expect such an estimate not to be precise. But in the continuous variables case, an alternative strategy is to choose the feature with the lowest absolute correlation with NDAI. This feature is DF, which has a correlation of about $-.16$.

This result is confirmed by using backwards selection based on AIC for logistic regression models: The feature logSD is deleted very early and the the best features are NDAI, DF and BF. However, the features selected by a CART predictor after tree-pruning using the one SD rule do not include DF or BF. The features selected by CART are NDAI and AN.

3 Prediction methods

In this section, we will present several viable prediction methods for cloud detection. In all these models the response variable Y is binary. 1 means cloud and -1 means clear. The ground truth data is not necessarily binary, as it also provides observations in which the expert was unsure about the observation.

We contemplated two approaches:

- We could set the response variable for missing data to 0. Then we have a model which has three outputs $-1, 0$ and 1.
- Or we could just ignore such datapoints. Then we just have the data which is 1 or -1.

We decided to ignore such data points, because some models we choose are not able to handle "unsure" results. We assumed that these pixels, labled with 0, still are cloudy or clear. This assumption is of course very strict, because the clouds situation can range from clear over slighly foggy to cloudy. As mentioned, this is not discrete and one can implement a measure to estimate "how cloudy" a pixel is, but this is not in the scope of the paper and probably not useful as the expert provides only cloudy clear and unsure data. And he did not asign numbers of cloudyness to the data.

3.1 Logistic Regression

The first estimator we suggest to predict cloudy and clear points is the usual Logistic regression:

$$\ln \frac{\pi}{1-\pi} = X * \beta + \epsilon$$

with X the variable matrix, π the likelihood of Y beeing cloudy and β the unknown parameter. This equation is solved by assuming that $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$ and then using the Newton-Raphson method to find the MLE for

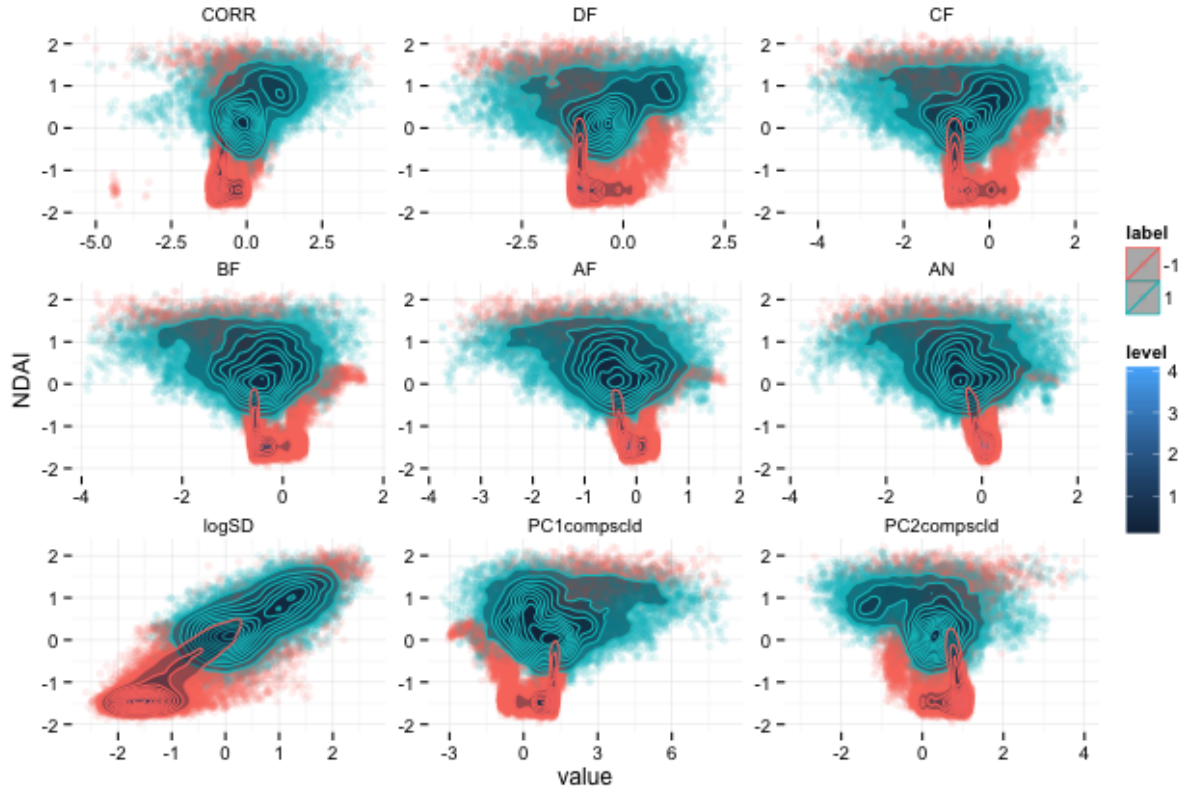

Figure 2.5: Two dimensional density plot: y-axis is the distribution of NDAI and the x-axis is the other variable. This figure was created with data from image 3.

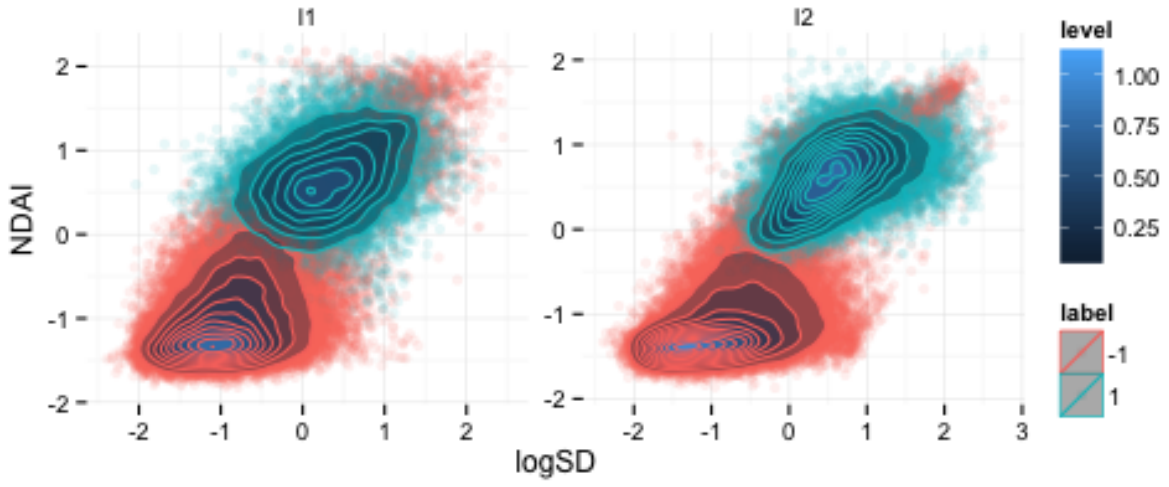

Figure 2.6: NDAI logSD distribution for Image one and two.

Table 1: Thresholds and Accuracy resulting from ELCM Algorithm assuming thresholds are fixed.

Data Set		**LogSD**		**NDAI**	
Learning Set	Testing Set	Threshold	Accuracy	Threshold	Accuracy
I2 and I3	I1	1.3	0.85	1	0.82
I1 and I3	I2	1.3	0.86	1	0.93
I1 and I2	I3	1.4	0.79	1	0.82

β. The standard R package gives by default probabiliteies for features to be 0 (p values). These p values are all very small ($< 10^{-16}$) for all variables. This is an effect, we can expect due to the high correlation between the different features and the big dataset.

That is why we do not really trust these results and we want to tryout two different predictors, which are based on the idea of Logistic regression: One uses all variables (NDAI, log(SD), CORR, DF, CF, BF, AF and AN) and the other uses three variables only (NDAI, CORR and DF). We chose these variables by using backwards selection using AIC. It is interesting to notice that we received the same result using forwards selection.

As we pointed out earlier, there might be reasons to scale the data in different ways. Eventually, we proposed four estimators, which are based on logistic regression with different scaling schema and different variables :

1. LSA: **L**ogistic Regression on the **S**tandard scaled data based on **A**ll features,
2. LIA: **L**ogistic Regression on the **I**mage scaled data based on **A**ll features,
3. LS3: **L**ogistic Regression on the **S**tandard scaled data based on **3** features,
4. LI3: **L**ogistic Regression on the **I**mage scaled data based on **3** features.

3.2 ELCM

We developed a model based on the same idea as the enhanced linear correlation matching (ELCM) algorithm of the paper[2]. Using our own implementation of the algorithm, we wanted to find thresholds for selected features to classify pixels for the presence of clouds. According to our exploratory data analysis (Part 1), NDAI and logSD seemed to be the best predictors and are the features used here. We developed two different models with cutoff values being either fixed or data-adaptive.

3.2.1 Grid search assuming a fixed threshold exists

In this part, we assume that the thresholds for NDAI and logSD are stable and robust for our three images. This means that we assume that the distributions of NDAI and logSD for cloud and cloud-free pixels are well separated by the sample thresholds for our tree images, and for the images we have not seen yet.

Method We performed a grid search of the thresholds for NDAI and logSD from [-3, 6] in steps of 0.1 to identify the value leading to the smallest classification error relative to the expert labels. For each step, the number of accurately labelled pixels over the total number of pixels is determined. Our best threshold is the threshold with the highest accuracy rate for the learning set. The grid search was performed on unscaled, scaled by image and scaled over the three images features.

Result Table 1 shows the results for unscaled variables. For logSD and NDAI, the thresholds were found using only the learning set. Then we evaluated accuracy of the thresholds using the testing set. The learning set is two of the three images while the testing set is the remaining image. For unscaled data, we found that

Table 2: Thresholds and Accuracy resulting from ELCM Algorithm assuming thresholds are data adaptive.

Data Set		**LogSD**		**NDAI**	
Learning Set	Testing Set	Threshold	Accuracy	Threshold	Accuracy
80% of I1	20% of I1	0.92	0.81	0.39	0.87
80% of I2	20% of I2	0.92	0.83	0.56	0.93
80% of I3	20% of I3	0.64	0.81	0.16	0.81

the thresholds for NDAI and logSD were stable and robust over the three images with a threshold of about 1.3 for logSD and 1 for NDAI. Looking at the distribution of the two variables logSD and NDAI for cloud and cloud-free pixels, these values seemed reasonable (see blue line for image 2 and feature NDAI on Figure 3.1).

3.2.2 EM algorithm assuming the threshold is data adaptive

Method As in the paper[2], we fitted a one-dimensional mixture model of two Gaussian distributions to observed NDAI and logSD by the EM algorithm initialized by the k-means algorithm. The R function kmean was used to initialize the EM algorithm with each pixel being assigned to the cloud or cloud-free cluster. Then, the R function normalmixEM from package mixtools was used to estimate the distribution of NDAI and logSD, and the R function turnpoints was used to find the minimum of the distribution between the fitted means of the two clusters. When no turnpoint exists between the two means, the average of the two means was taken. Finally, we considered the possibility to get images with either almost only no-cloud pixels or almost only cloudy pixels. To prevent from predicting an unrealistic threshold for such images, we defined boundaries for logSD and NDAI thresholds. It means that if the threshold predicted by our algorithm was out of our boundaries, we set the threshold at the value of the nearest boundary.

This method is unsupervised, because we never used expert labels to find the threshold. Expert labels were only used to evaluate the accuracy of the model. Unlike the ELCM algorithm assuming thresholds are fixed, if we were given an image without expert labels, we would still be able to predict the threshold for this particular image.

Result The thresholds were found using the EM algorithm on the entire data of each image. For each threshold, accuracy was then calculated using again the entire data set for one image. Here, there is no need to divide the data set into a learning set and a testing set because we are not using the expert labels to find the threshold. As we are already blind of the labels, there is no risk of overfitting for this model. Data shown in table 2 is for unscaled data. Accuracies found for the three images are high. The thresholds are different from one image to another except for images 1 and 2 with feature logSD.

3.2.3 Conclusion ELCM algorithm

The thresholds found with the two different assumptions (thresholds fixed or data adaptive) could seem different. However, looking at the distributions of logSD and NDAI, it does not make a big difference to choose one or the other threshold. See Figure3.1. With the two assumptions, we found high accuracies making it difficult to tell which method is the best. Further investigations were lead using ROC curves and a cross-validation for the ELCM algorithm assuming thresholds are stable to try to determine the best method.

3.3 CART estimator

Classification And Regression Trees are recursive partitioning methods that build decision trees to predict a response on class variable Y from inputs from n-dimensional explanatory features $X_1, X_2, ..., X_n$. This type of model is extremely useful where a single predictive model does not hold over the entire data space

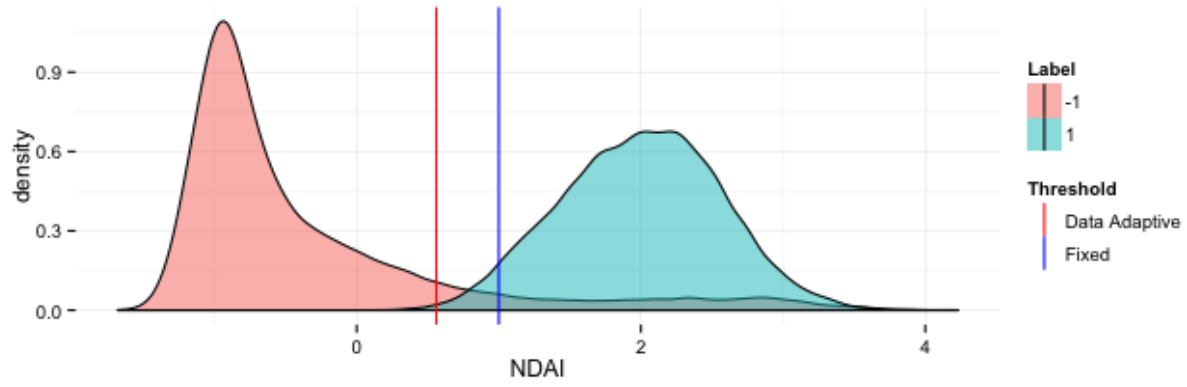

Figure 3.1: Distribution of NDAI for unscaled image 2.

due to non-linearities between the features. The algorithm was introduced in 1984 by statisticians at UC Berkeley and Stanford. In the implementation we used, the full-tree is pruned by using cross-validation and feature selection is a consequence of this pruning method. CART uses a a two-method algorithm called Error-Complexity pruning, which uses the measure $EC_\alpha(T)$, which is defined as:

$$EC_\alpha(T) = Err(T) + \alpha * \mathbf{card}(\widetilde{T})$$

where,

$Err(T)$ is the resubstitution error estimate of tree T;
$\mathbf{card}(\widetilde{T})$ is the cardinality of set $\widetilde{T}$ containing the leaves of tree T;
and α is called the complexity parameter and defines the cost of each leaf.

The sequence of pruned trees is generated by successively pruning the node t to minimize the following function :

$$g(t,T) = \frac{Err(t) - Err(T_t)}{\mathbf{card}(\widetilde{T}) - 1}$$

where,

T_t is the sub-tree of T rooted at node t;
and $\mathbf{card}(\widetilde{T})$ is the number of leaves of this sub-tree.

The package rpart computes the best value for α by using cross-validation. For pruning the tree, we choose the complexity parameter associated with minimum error and taking into account the upper limit of the one standard deviation rule.

The calculated model in figure 3.2 is probably theoretically the best, but the further distinction based on AN, does not make a big difference, as less than .5 % of all data are have a feature NDAI of less than $-.2$ and an AN of less then $-.4$. So basically this model just makes a distinction based on NDAI. That is why in the following we will mainly anlyse four types of models:

1. CARTS : **CART** Estimator based on all variables which are **S**tandard scaled,
2. CARTI : **CART** Estimator based on all variables which are **I**mage scaled,
3. CART1S: **CART** Estimator based on NDAI which are **S**tandard scaled,
4. CART1I: **CART** Estimator based on NDAI which are **I**mage scaled.

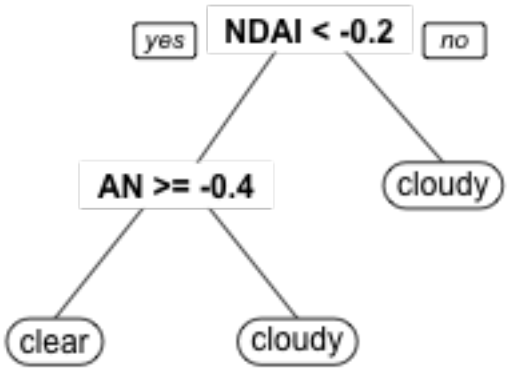

Figure 3.2: CART trained on all predictors

4 Evaluation of the predictors

4.1 ROC Curve

We plotted the ROC curve (figure 4.1) with false positive rate on vertical axis and true positive rate on horizontal axis. Each point on the curve is set for a particular threshold. We can see that none of our estimators is inadmissible, because no technique proposes an predictor which is performing always worse than other predictors. This is a sign that all estimators might be useful in our case and we need better criterions to decide which estimator is the best.

4.2 AIC and BIC

The Akaike and Bayesian Information Criterion area criteria for model selection based on the number of parameters and the maximized value of the likelihood function of the model. The main difference between AIC and BIC lies on the penalty term related to the number of observations. Whereas both criteria have a penalty term related to K, the nuber of parameters, they differ on the weight applied to it. AIC penalizes K by some constant (usually 2), while BIC penalizes the the criteria by the natural log of the number of observations.

$$AIC = K * 2 - 2\ln(\hat{L})$$

$$BIC = K * \ln(n) - 2\ln(\hat{L})$$

It is interesting to notice that both the AIC and BIC penalize lack of fit more than model complexity. These criteria are useful because our problem is well conditioned (k « n). and the samples are large, most likely producing asymptotic results.

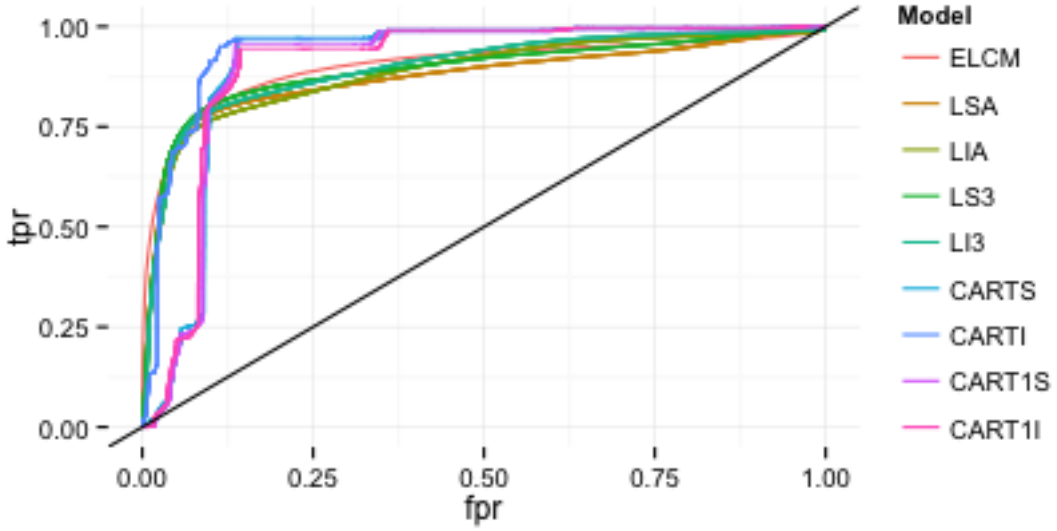

Figure 4.1: Here we can see the ROC curves for different estimators. Note, that the CART estimators has a discrete flavor, because the scores they provide are based on fixed theshold.

One possible reason to use AIC over Cross-Validation is to avoid long computation times. This only holds asymptotically because minimizing the AIC is similar to minimizing the CV score [3]. On the same token, asymptotically and for linear models, minimizing BIC is similar to leave-k-out cross validation [1].

4.3 Cross-validation

We implemented two types of cross-validation and we want to start with the canoncial, but for this case not optimal one:

4.3.1 Standard Cross Validation

We implemented the following cross validation algorithm:

1.) Delete all pixels which are not labeled as cloud or clear
2.) Randomly seperate the entire data set D into k equally sized data sets such that

$$\cup_{i=1}^{k} U_i = D \quad \text{and} \quad \text{for all } i \neq j \;\; U_i \cap U_j = \emptyset$$

for *i = 1, 2, ..., k* **do**
  3.) Train Model on $D \backslash U_i$
  4.) Predict the labels of U_i
  5.) Calculate the propotion of wrongly predicted points
end

In this method, we discarded the data which the expert was unsure about. This is about 40% of the entire data set. This is of course a problem. When we say in the following that we predicted 90% of the data correctly, we really mean that we predicted 90% of the data, for which the expert was able to identify the pixel as cloudy or clear, correctly. Of course we are also able to predict the data which was not classified by the expert, but we should not assume that we will perform here as good as in the case, which was classified by the expert.

Naturally, cross-validation must be evaluated under the problem that is being adressed and its assumptions. It is well known that the images have a high correlation between neighboring pixels and there seem to be

	AIC	BIC	Std CV	pred 1	pred 2	pred 3
LSA	11.373	11.383	89.805	76.891	92.663	79.642
LIA	11.412	11.422	89.830	76.891	92.663	79.642
LS3	11.697	11.701	89.339	76.520	93.765	83.164
LI3	11.811	11.815	89.337	76.520	93.765	83.164
CARTS	NA	NA	90.453	81.583	94.415	82.357
CARTI	NA	NA	90.547	81.631	89.243	80.440
CART1S	NA	NA	89.948	90.572	93.500	82.048
CART1I	NA	NA	89.052	87.171	93.493	80.440
ELCM fixed	NA	NA	85.976	88.943	91.473	77.464

Figure 4.2: AIC and BIC: in 10^4, Std CV: Ouput of the standard Cross Validation (in %), pred 1: Output of the Cross Validation per image: train the model on image 2 and 3 and then predict image 1 (in %).

effects within images. To overcome this, we designed a cross-validation method that somewhat accounts for this by training on 2 images and then predicting on another one. Another possible approach, not implemented by us, would be to slice the images into smaller patches and then do cross validation on these smaller patches. However this approach would be able to handle issues which arsie due to effects because of the correlation due to a neighborhood. But it would not be able to handle effects, which affect an entire picture. One example would be the sun position, when a particular image was taken.

4.3.2 Cross-Validation per image

So let us now consider a slightly different type of cross-validation. Instead of randomly dividing the data set into parts, we took the data of each image to be a subset. The algorithm then looks like this:

1.) Delete all pixels which are not labeled as cloud or clear
2.) Define U_i := "Data from Image i" **for** *i = 1, 2, 3* **do**
  3.) Train Model on $D \backslash U_i$
  4.) Predict the labels of U_i
  5.) Calculate the propotion of wrongly predicted points
end

This kind of cross validation test a very special case of robusteness. An estimator which is very sensible to effects which are specific for a particular image or neighborhood, will perform very poorly.

We eventually concluded that this type of cross validation is best criterion of model selection for this problem. But as the sample size is so small, we would want to have more pictures to make this meachnism really give good results. That is why we also consider the other model selection methods.

4.3.3 Performance of the ELCM data adaptive model

For the ELCM algorithm assuming thresholds are data adaptive, it makes little sense to compute a cross validation. Indeed, as said in part 3.2 this classifier is unsupervised and specific to each image. The algorithm uses the distribution of the feature to infer the threshold. Thus, we used the entire data set of each image to both predict the thresholds and evaluate the accuracy. See the accuracies in Table3. Here accuracies were calculated using the two features logSD and NDAI. Pixels were first classified using NDAI, then using logSD.

	Accuracy
Image 1	0.91
Image 2	0.94
Image 3	0.83

Table 3: Accuracies of ELCM algorithm assuming thresholds are data adaptive

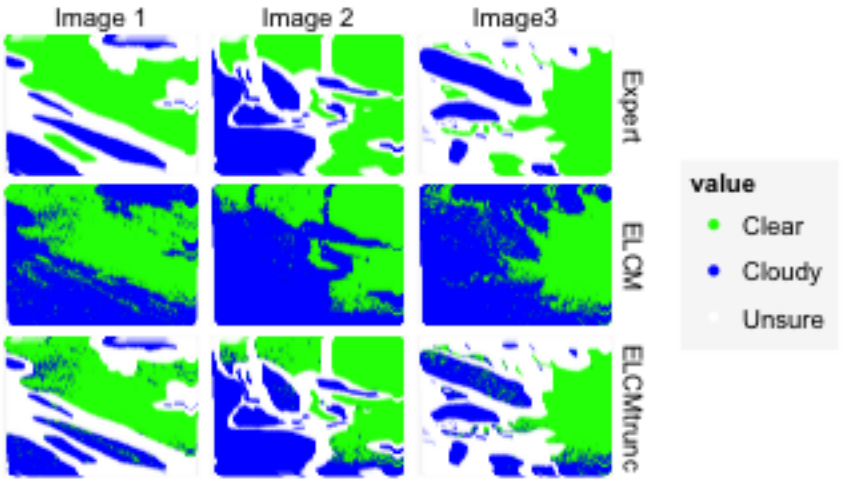

Figure 5.1: The first row shows the expert labels, the second the predicted values with our CART predictor and the third row shows the predicted values only on the set where the expert was sure.

5 Conclusion

We concluded that the best predictor is the ELCM estimator with an adaptive threshold choice using the varivables variables NDAI and log(SD). Figure 5.1 shows that this predictor perfoms well for all images, but there are still a few things we would want to improve.

First of all, our predictor does not make any use of the **location** of a particular pixel. It seems quite unintuitive that a pixel is clear while it is sourrounded by clouds. We would expect such a pixel to be cloudy. For further research it would be a good idea to consider **smoothing** the probabilities, such that a pixel is shrinked towards the average of the pixel in its neighborhood. This might make the predictor more robust against outlier, but it could perform poorly in areas with a lot of clouds and clear points very close to each other, for example at the boundary.

The predictor is working with non fixed thresholds. This makes it possible, that we adapt for every image seperately and thus we can hope that we will even perform very well for images which are looking different from ours. However as we are choosing for each image a different threshold, we might run into trouble, when the image is only cloudy or only clear. We tried to protect us against these cases by always choosing the threshold in a particular interval, but we have not seen data which was only cloudy or only clear. Thus we should investigate what might happen in such a case to find an optimal strategy to choose the threshold in these cases.

References

[1] Jun Shao. "An asymptotic theory for linear model selection". In: *Statistica Sinica* 7.2 (1997), pp. 221–242.

[2] Tao Shi et al. "Daytime arctic cloud detection based on multi-angle satellite data with case studies". In: *Journal of the American Statistical Association* 103.482 (2008), pp. 584–593.

[3] Mervyn Stone. "An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1977), pp. 44–47.