Project 2 - XGBooster Shot

Johnny Antoun (0679537) Jose Pliego (2716768)

November 2021

Contents

1 Data Collection and Exploration

1.1 Paper summary

The impact of increasing amounts of atmospheric carbon dioxide on Earth's climate is an important issue today. The prevalence of carbon dioxide is a major topic due to its association with increasing surface air temperatures, with the strongest dependencies being in the Arctic region.

In the polar regions, cloud coverage is important in modulating the sensitivity of the Arctic to increasing surface air temperatures. Thus, it is useful to be able to detect clouds it that Arctic region. Unfortunately, existing algorithms to detect clouds do not perform well in these regions due to the similarity between visible and infrared electromagnetic radiation emitted from clouds and snow/ice-covered surfaces. For example, the MISR level 2 top-of atmosphere algorithm (L2TC), an existing algorithm, does not perform well in the Arctic because cloud heights are lower than in other regions. In terms of methodology, L2TC produces two cloud masks: the stereo-derived cloud mask (SDCM) and the angular signature cloud mask (ASCM).

This study describes NASA scientists and statisticians attempt to devise improved a cloud detection algorithm that works well in polar regions. Instead of searching for cloud pixels algorithmically, the NASA scientists and statisticians found a better approach is to model the surface because it doesn't change materially from different views. They rely on measurements from the Multiangle Imaging SpectroRadiometer (MISR) that differs from traditional multispectral sensors that take measurements in a single view. The MISR sensor comprises nine cameras at different angles (4 forward, 4 backward and one nadir) in four spectral bands (blue, red, green and near-infrared). The MISC cameras cover a 360-km-width swath of Earth's surface that extend across daylight side of the Earth from Arctic down to Antarctica in approximately 45 minutes with a total 233 distinct, but overlapping, such swaths. MISR completes accumulation of data from all 233 paths in around 15 days with each path subdivided into 180 blocks (block number increasing from the North Pole to the South Pole).

It is clear from the MISR data collection process that the resulting dataset is massive, posing computational constraints. Standard classification frameworks are not readily applicable given the size of the data and thus the difficulty of obtaining expert labels for training. Clustering is not ideal either because data units (three consecutive blocks) could be entirely cloud-covered or cloud-free. Consequently, the challenge is to combine clustering and classification in a computationally efficient manner.

The data collected in this study consists of 6 data units from consecutive 10 MISR orbits of path 26 (rich in surface features). Out of the total of 60 data units, three are excluded because the surfaces were open water, with the total included corresponding to around 7.1 million 1.1-kn resolution pixels. Around 71.5% of valid pixels in the data is labeled by experts.

The proposed algorithm, enhanced linear correlation matching (ELCM), is based on thresholding three features with values that are either fixed or data-adaptive: correlation of MISR images of the same scene from a different angle (CORR), standard deviation of MISR nadir camera pixel values across a scene (SD_{an}) and normalized difference angular index (NDAI) which relates to changes in a scene with changes in view direction. The CORR and SD_{an} cutoff values are set for fixed values during operational processing whereas the NDAI threshold is either kept the same or updated at a new data unit based on a data-adaptive algorithm. Pixels are then labelled as cloudy in two scenarios: $SD_{an} < threshold_{SD}$ or CORR $> threshold_{CORR}$ and NDAI $< threshold_{NDAI}$. At a high level, this labeling relies on the fact that clouds are rarely (if even possible) both extremely smooth and relatively weakly forward scattering. Next, Fisher's QDA (models each class density as a multivariate Gaussian distribution) trained from the ELCM labels is used to provide an estimate of confidence of cloudiness. The probability measure is valued because it is not always possible to accurately classify a pixel as clear or cloudy in the case of partial cloudiness which is especially common at cloud boundaries.

To assess the different algorithms, results are compared against expert labelled pixels (71.5% of data). In addition to the methods discussed previously, a simple-minded offline SVM classifier is trained on randomly sampled expert labels (standard 10-fold cross-validation). After checking the results, the scientists and statisticians confirmed ELCM (91.8%) performed better than both ASCM (83.23%) and SDCM (80%), a

significant improvement. The offline SVM performed worse than the ELCM but was in similar range as ASCM and SDCM. In addition, the ELCM-QDA did not improve agreement rate with expert labels relative to ELCM algorithm but does provide additional color represented by the probability labels. While the ELCM algorithm has above a 90% agreement for most cases, some data units have as low as 70%. The reason behind this drawback is lower CORR when there are no clouds, a phenomenon that occurs due to poor terrain data registration (generally at sharp elevation changes). There is no solution to this problem because it requests updating MISR terrain height database.

In conclusion, the study shows that the three physical features (CORR, SD_{an}, NDAI) contain enough information to separate clouds from ice- and snow-covered surfaces. The ELCM algorithm combines classification and clustering in a way that makes it suitable for real-time, operational MISR data processing and is more accurate than existing MISR operations algorithms. Statisticians are involving in all the steps of data processing unlike other projects where they come in after the fact to develop methodologies. This study showed that by careful combination and application-specific modification of existing statistical methods, innovative solutions to difficult problems can be crafted.

1.2 Data Summary

From this section and throughout our work, we "trimmed" the images so that they are perfectly rectangular. To do this, we filtered each image to have x-coordinate values between 70 and 368. By doing this cleaning step, we lost 2902 pixels in total (0.84%), with each of the three images loosing a similar proportion of pixels. We decided to do this trimming because some features like Gabor filters require a rectangular matrix as an input (Chen (2006), Haghighat (2015), Mouselimis (2021)).

As a first step in the exploratory data analysis, let's take a look at the distribution of the expert labels in each image and in aggregate. The distribution is shown as a percentage frequency table (table 1). We can see that the distribution varies greatly from one image to another, with image 1 having a similar amount of cloud and non-cloud pixels, image 2 having a majority of non-cloud pixels, and image 3 having most of its pixels unlabeled.

Label	Img1	Img2	Img3	Total
No cloud	37.54%	43.98%	29.35%	36.96%
Unlabeled	28.68%	38.24%	52.17%	39.7%
Cloud	33.77%	17.78%	18.48%	23.34%

Table 1: Label distribution in the data.

To get a sense of how labels look like in space, we can plot them in the (x, y) plane. In figure 1 we can see that cloudy regions and surface regions are separated by unlabeled regions, where the expert could not say for sure whether the pixels correspond to clouds or ice/snow.

(a) Labels for image 1. (b) Labels for image 2. (c) Labels for image 3.

Figure 1: Image labels plotted in the (x, y) plane. White pixels correspond to clouds, gray pixels to land surface, and black pixels are unlabeled.

Figure 1 is very interesting because it shows the strong spatial dependence present in the images. Cloudy pixels are generally surrounded by more cloudy pixels, and the same pattern repeats for surface (non-cloudy) pixels. This phenomenon presents a significant challenge for pixel classification because we cannot treat the different pixels as independent and identically distributed. We must account for the spatial dependence in our classification models so they can generalize well to future unobserved images.

1.3 Exploratory Data Analysis

To get a broad sense of how the predictors are related between them and with the labels, we include a correlation plot of the relevant variables in figure 2. The label column is encoded as 1 if the pixel is a cloud and -1 if it is not, so a positive correlation between label and a predictor can be roughly interpreted as an increase in the predictor being associated with an increase in "cloudiness." For this figure, we remove the pixels that are unlabeled to retain the numerical interpretation of the labels and the correlation with the predictors.

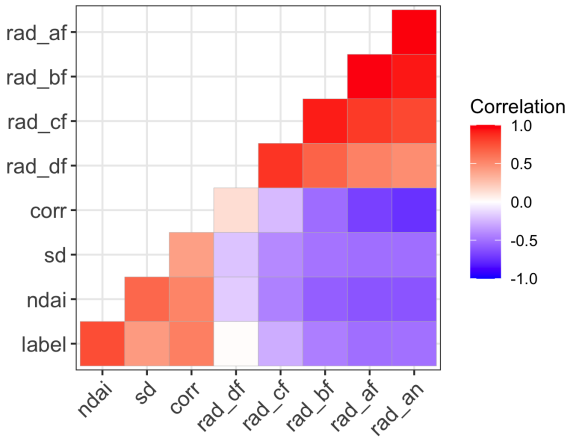


Figure 2: Visual representation of Pearson's correlation matrix.

Figure 2 shows a positive correlation between label and the author-defined variables `ndai`, `sd`, and `corr`, and also shows a negative correlation between label and most radiance measurements. It is also interesting to note that the author-defined variables are negatively correlated with the radiance measurements while positively correlated among themselves. Similarly, radiance measurements are positively correlated. This was expected since these variables are all measuring light reflection at different angles.

To take a closer look at some of these relationships, figure 3 shows the distribution of the author-defined variables and `rad_an` for the two pixel classes. The figure includes density estimates and the median for each class. All variables were centered and scaled. For the variable `sd`, we use a logarithmic transformation because the original variable is heavily right-skewed.

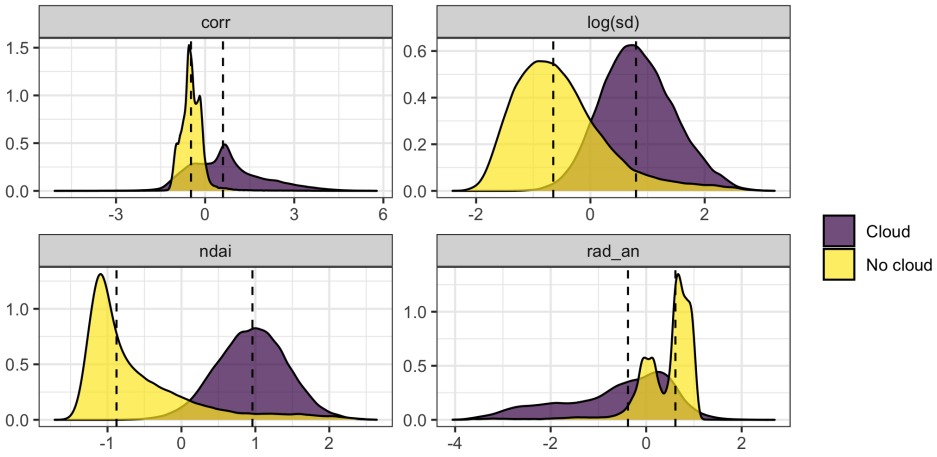


Figure 3: Distribution of author-defined variables and nadir radiance (rad_an) across pixel labels (median values highlighted).

In figure 3 we can see that the distributions are visibly different for both classes. The differences and the distance between the medians are greater in the plots of `log(sd)` and `ndai`. It is interesting to note that Shi et al. (2008) use fixed cutoff thresholds for `sd` and `corr` but not for `ndai` since the "appropriate thresholds vary from one data unit to another." In our case, it seems like `ndai` is the most discriminating feature. We formalize this notion of "best features" on section 2.

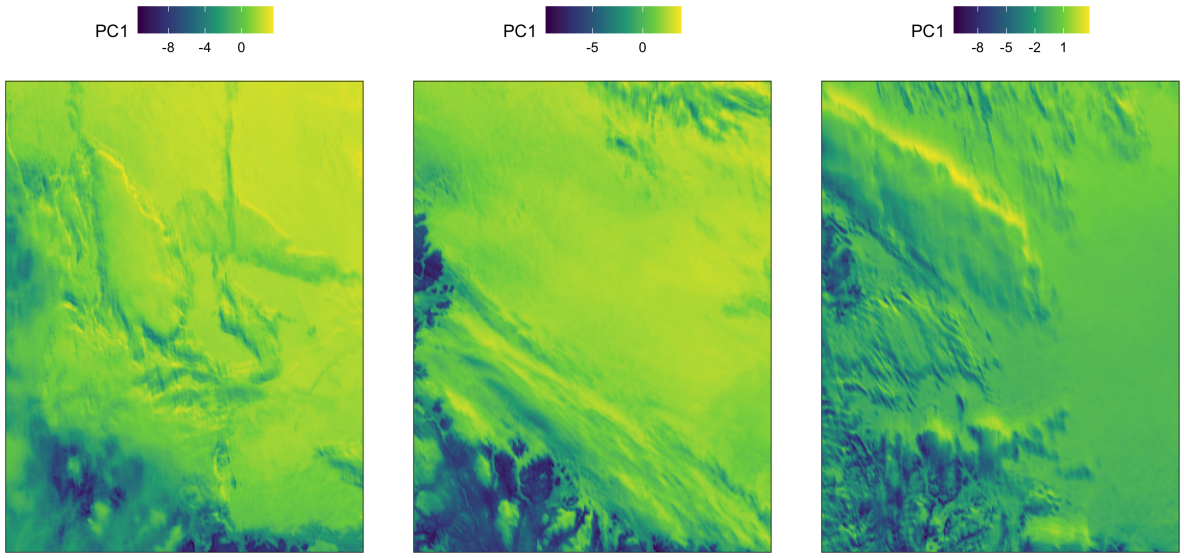
We include two scatterplots between the predictors that convey interesting properties in figure 4. The relationship between `ndai` and `sd` is shown in panel 4a. We see that the two variables are positively correlated (as was shown in figure 2) but also see that the spread of `sd` increases as `ndai` increases. Similarly, panel 4b shows the negative correlation between `corr` and `rad_an`, while also showing that the variance in `rad_an` increases as `corr` decreases. We chose to include these two scatterplots because other correlated variables show a more straightforward linear relationship without the heteroscedasticity property.

(a) Scatterplot between `ndai` and `sd`.

(b) Scatterplot between `corr` and `rad_an`.

Figure 4: Scatterplots for two pairs of highly correlated features.

To conclude the exploratory data analysis, we use Principal Component Analysis to reconstruct the three images using information from all the predictors. Since the principal components do not use label information, we obtain the scores and loadings using all the pixels. In figure 5 we can see the first principal component scores plotted in space. Note that the first principal component shows spatial dependence between neighboring pixels. This detail is interesting because the (x, y) coordinates were not used to calculate the principal components, so it supports the authors' claim that some features convey implicit spatial structure when defined using local patches of pixels.



(a) PC1 representation for image 1. (b) PC1 representation for image 2. (c) PC1 representation for image 3.

Figure 5: First principal component scores plotted in the (x, y) plane.

The first two principal components account for 80.7% of the variance in the dataset. Using the principal components as predictors instead of the original features may be useful in the modeling stage. As we saw in this analysis, some of the original predictors are highly correlated and using a set of orthogonal predictors (principal components) may yield better model results.

Finally, to support the claim that the principal components contain some of the spatial structure present in the data, we calculate the median and interquartile range (IQR) of the first two principal components by label. We choose IQR as a measure of spread and the median as representing the center because the distributions are not symmetric or unimodal.

Label	PC	Median	IQR
No cloud	PC1	1.69	1.94
No cloud	PC2	-0.50	0.88
Cloud	PC1	-0.86	3.06
Cloud	PC2	0.98	2.12

Table 2: Summary statistics for principal components.

In table 2 we see that the first two principal components have more spread in cloudy pixels, and the medians are different for both types of labels. This suggests that principal components may be useful in classification models. In section 3 we formalize these claims and assess model performance.

2 Preparation

2.1 Data Split

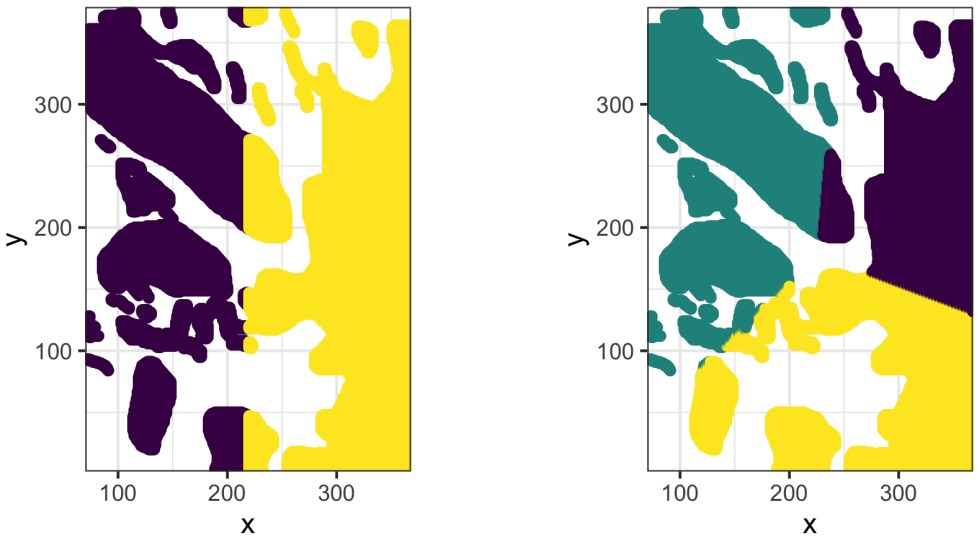
In this section, we propose two methods to split the data into three sets (train, validation, and test). The two methods are block splitting (Valavi et al. (2019)) and clustering (Brenning (2012)). Conventional random splitting is not ideal for spatial data since pixels close to each other tend to be more similar. This means that we could be filtering information from the training to the test set with neighboring pixels, and model assessments in the test set would be overly optimistic.

The idea behind block splitting is dividing each image into similar sized blocks. The blocks should be big enough so that the spatial correlation between blocks is relatively low while simultaneously capturing the spatial correlation structure of the pixels within each block. We also have to be careful and make sure that our test set does not contain an overwhelming majority of cloud or non-cloud labels, since a test set with any of these characteristics would not be a good proxy for how a model will perform with unseen data.

We chose to split each image into two blocks and take one random block as test set, another random block as validation set, and the four remaining blocks as training data. This corresponds to a 66.6%-16.6%-16.6% training-validation-test split. After trying different block sizes, we settled for two blocks in each image so we can capture a diversity of pixel characteristics in each set, while also making sure that there are enough pixels to train classification models. When we use cross-validation to fit the models in section 3, then the train-test split becomes a 83.3%-16.6% split. The blocks were defined including the unlabeled pixels, even though these pixels are not useful for classification. We decided to include these pixels on the block design because creating blocks that account for missing pixel patches is complicated and our blocks are big enough so that none of the sets has an overwhelming majority of unlabeled pixels.

For the clustering split, we use k-means clustering on the (x, y) coordinates for each image (see Brenning (2012)). We identify two main advantages of using k-means over block splitting. The first is that we can remove the unlabeled pixels so that the clustering algorithm only uses coordinate information of the pixels that will actually be used for modeling. The second advantage is that we can specify any number of clusters, while the block splits restrict us to using a number of blocks specified as rows$\times$columns.

Following the same reasoning as in the block splits, we choose to split each image in 3 clusters, taking two of the nine clusters as test set, one as validation set, and the remaining six as training set. This corresponds to a 66.6%-11.1%-22.2% training-validation-test split. We also center and scale the coordinates before clustering so there is not more variation in one of the dimensions.



(a) Block splits for image 3.

(b) K-means splits for image 3.

Figure 6: Block and clustering splits for image 3.

Figure 6 shows a visual representation of the clustering and block splits for image 3. The unlabeled points were removed after doing the block splits. We can see how k-means is not restricted to a square grid structure and can take more flexible shapes. It is important to note that there is still some information leakage between points that are close to the group borders, but it is considerably better than using purely random splitting (see Valavi et al. (2019), Brenning (2012), Chen (2006)).

2.2 Baseline model

We report the accuracy of a trivial classifier that predicts every pixel as non-cloudy on the test and validation sets, using both splitting methods mentioned before. This trivial classifier serves as a baseline comparison for the classification models used in section 3. The accuracies are shown in table 3.

Splitting Method	Set	Accuracy
Block	Validation	78.8%
Block	Test	20.2%
K-means	Validation	81.8%
K-means	Test	86.3%

Table 3: Accuracy of a trivial classifier.

We can see that the trivial classifier performance is greatly dependent on the set used to evaluate. Trivial classifiers perform well when the data is unbalanced, in this context when a set is composed of mostly non-cloudy pixels. In table 3 we see that the validation sets for both splits, as well as the test set obtained with k-means, consist of mostly non-cloudy pixels so the classifier has relatively high accuracy. However, the accuracy is still far from 100% so the hope is that a well designed classification model will outperform the trivial classifier. We also see that our splitting methods work well since we manage to capture different pixel behaviors in each set.

2.3 Feature selection

To formalize the selection of "best features" mentioned in section 1, we fit single-predictor models using each of the predictors and assess this models' performance. We fit a logistic regression on each training set (block

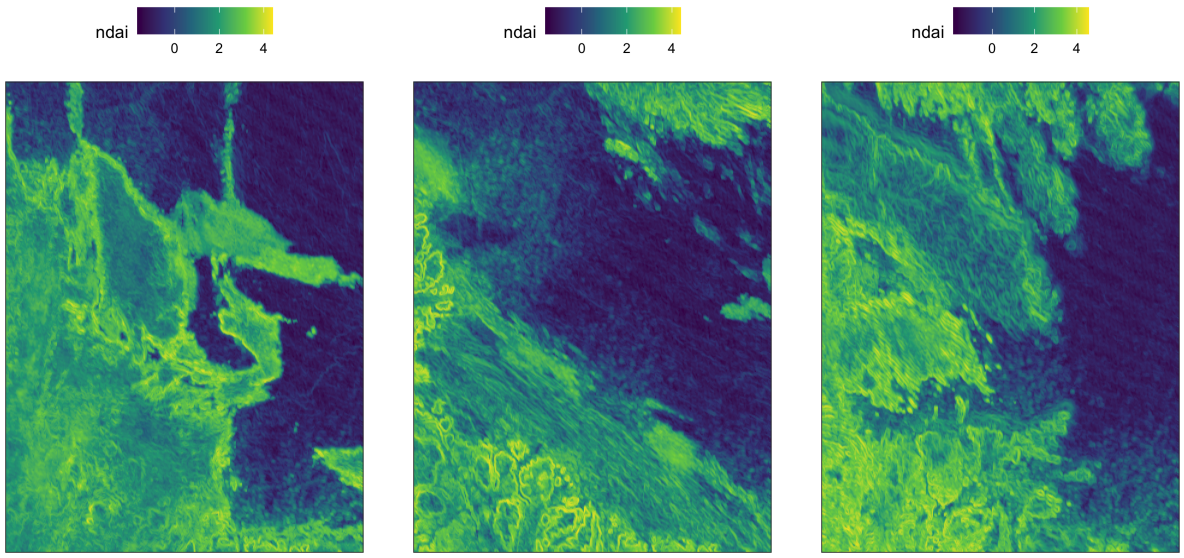
and cluster splits), calculate the area under the curve (AUC) of the ROC Curve in the validation sets, and average this values to rank the predictors. We choose AUC as a selection criterion because this metric takes into account all possible probability threshold cutoffs, and we choose logistic regression as an initial model because this model returns a coefficient for the predictor and we can relate the probability of a pixel being cloudy or not with a higher or lower value of each predictor.

The results are shown in table 4. We can see that `ndai` is the clear winner, followed by `sd`, both of which are author-defined predictors. Going back to figure 3, we expected `ndai` to perform well because the distributions between the two classes were well separated.

var	AUC Block	AUC K-means	Avg AUC
ndai	1.00	0.99	0.99
sd	0.94	0.91	0.92
rad_af	0.82	0.93	0.87
rad_bf	0.87	0.83	0.85
rad_an	0.71	0.88	0.80
corr	0.31	0.99	0.65
rad_cf	0.87	0.20	0.54
rad_df	0.86	0.05	0.45

Table 4: Single-predictor model performance.

As a visual assessment, we show the values of `ndai` plotted in space for the three original images. In figure 7 we see what figure 3 suggested, that higher values of `ndai` are associated with cloudy pixels. This is further supported by the logistic regression model having a positive coefficient $\hat{\beta}_{ndai} = 2.008$.



(a) Image 1. (b) Image 2. (c) Image 3.

Figure 7: `ndai` values plotted in the (x, y) plane.

The visual cues in figures 3 and 7, along with the quantification of AUC in table 4, suggest that `ndai` is the most useful feature for predicting pixel labels. The AUC results suggest `sd` and `rad_af` as the second and third most important features, but this can change based on how the predictors behave when used simultaneously in a classification model.

2.4 Cross Validation

To ease the implementation of different classification algorithms using the splitting methods discussed before (block splits and k-means), we used the R package `tidymodels` (Kuhn and Wickham (2020)). First of all, we created two splitting functions, one for each method, that return a split object as created by the package `rsample` (Silge et al. (2021)). This split object can then be used with the `tidymodels` constructors.

As a brief overview, our function takes as inputs a training set, split method, classification algorithm, a set of metrics, and additional parameters related to these objects, and returns the metrics obtained in each cross validation fold.

3 Modeling

For the modeling, we use the block splits created in section 2 as test and training sets. For the training set, we combine both the original training and validation set. Because we have to split the training set for cross validation, we rearrange the training set by shifting the x coordinates for the different images. This way, whether we use block splits or k-means, blocks and clusters will mainly contain pixels from the same image.

In terms of binary classifiers, we decided to fit logistic regression, LDA, QDA, SVM, XGBoost and Naive Bayes. For each classifier, we do 9-fold cross validation in the two separate scenarios, data split using block or k-means. We choose to use 9 folds as it falls in between the standard 5 and 10. We found it to be rational to go smaller than the initial split (testing/training/validation) of 6 as we are using all 9 of the segments for testing. We did not want to go too small on the blocks as we would get blocks with only cloud or no clouds at all.

Assumptions :

Logistic Regression:

LDA: Estimates a multivariate distribution for the predictors separately for the data in each class (Gaussian with common covariance matrix). Bayes' theorem is used to compute the probability of each class, given the predictor values.

QDA: Estimates a multivariate distribution for the predictors separately for the data in each class (Gaussian with separate covariance matrices). Bayes' theorem is used to compute the probability of each class, given the predictor values.

SVM:

XGBoost:

Naive Bayes: We assume independance of predictors

Bibliography

Brenning, Alexander. 2012. "Spatial Cross-Validation and Bootstrap for the Assessment of Prediction Rules in Remote Sensing: The r Package Sperrorest." In *2012 IEEE International Geoscience and Remote Sensing Symposium*, 5372–75. https://doi.org/10.1109/IGARSS.2012.6352393.

Chen, C. H. 2006. *Signal and Image Processing for Remote Sensing*. 1st ed. CRC Press.

Haghighat, Mohammad. 2015. *gabor: Gabor Feature Extraction*. https://github.com/mhaghighat/gabor.

Kuhn, Max, and Hadley Wickham. 2020. *Tidymodels: A Collection of Packages for Modeling and Machine Learning Using Tidyverse Principles*. https://www.tidymodels.org.

Mouselimis, Lampros. 2021. *OpenImageR: An Image Processing Toolkit*. https://CRAN.R-project.org/package=OpenImageR.

Shi, Tao, Bin Yu, Eugene E Clothiaux, and Amy J Braverman. 2008. "Daytime Arctic Cloud Detection Based on Multi-Angle Satellite Data with Case Studies." *Journal of the American Statistical Association* 103 (482): 584–93. https://doi.org/10.1198/016214507000001283.

Silge, Julia, Fanny Chow, Max Kuhn, and Hadley Wickham. 2021. *Rsample: General Resampling Infrastructure*. https://CRAN.R-project.org/package=rsample.

Valavi, Roozbeh, Jane Elith, José J. Lahoz-Monfort, and Gurutzeta Guillera-Arroita. 2019. "blockCV: An r Package for Generating Spatially or Environmentally Separated Folds for k-Fold Cross-Validation of Species Distribution Models." *Methods in Ecology and Evolution* 10 (2): 225–32.