Classifying Pyrocumulonimbus Cloud formations

Math 448: Categorical Data Analysis Project

Sequoia Andrade

May 2023

Contents

1 Introduction **4**

2 Dataset **4**

2.1 Exploratory Analysis 5

2.2 Data Pipeline 6

3 Model Selection **7**

3.1 Logistic Regression 8

3.2 KNN 9

3.3 SVM 9

3.4 LDA 10

3.5 QDA 10

3.6 Random Forest 10

3.7 Convolutional Neural Network 11

4 Model Comparison and Discussion **12**

5 Conclusion **13**

6 Appendix: Code **13**

Abstract

Pyrocumulonimbus (pyroCB) clouds are fire-generated convective clouds that pose a safety risk to aerial wildfire suppression operations, as well as local communities via air pollution. Hence, detecting these clouds effectively can improve the safety of aerial wildfire operations by alerting operators to changing airspace conditions. Satellite imagery is used as an input to detect pyroCB events in this project by treating the task as a binary classification problem. Seven models total are tested: logistic regression, KNN, SVM, LDA, QDA, Random Forest, and a convolutional nerual network. After balancing the data using under-sampling, logistic regression performs the best while a convolutional neural network performs the worst. The highest scores obtained were by the logistic regression model, with an accuracy of 0.659, precision of 0.674, recall of 0.659, f-1 of 0.651, and AUC of 0.659. Overall, the finsl model has low predictive power and unsatisfactory performance, despite performing better than other models, thus additional data may be needed.

1 Introduction

This project focuses on detecting pyrocumulonimbus (pyroCB) clouds, which are fire-generated convective clouds (https://en.wikipedia.org/wiki/Cumulonimbus_flammagenitus). These clouds pose a safety risk to aerial wildfire suppression missions due to the physical impact they have on the airspace. Hence, detecting these clouds effectively can improve the safety of aerial wildfire operations by alerting operators to changing airspace conditions. Satellite imagery is used as an input to detect pyroCB events in this project. Existing research and atmospheric experts note that detecting pyroCBs from true color images alone is likely not possible due to the interaction between other variables, such as heat and air currents; however, this project uses true color images as inputs.

In this report, the dataset is described, including exploratory analysis, followed by model selection. The models tested are logistic regression, KNN, LDA, QDA, Random Forest, SVM, and a convolutional neural network. A comparison between the models is provided, followed by a brief discussion and conclusion.

2 Dataset

The dataset was manually created by downloading raw netcdf files of archieved satellite data and reconstructing them into jpg images. In total, this process took about two weeks of constant downloading and converting of files. True color images are constructed from individual data bands according to the recipe provided here: https://unidata.github.io/python-gallery/examples/mapping_GOES16_TrueColor.html#sphx-glr-download-examples-mapping-goes16-truecolor-py. GOES-east and GOES-west satellite data is used to create the dataset, with incidents in the U.S. using the continental U.S. data product and events in Canada using the full disk data product.

To create a single image, the following steps are completed:

1. Query the selected data point by satellite region, year, day of year, and hour in the GOES AWS S3 bucket.
2. Download the netcdf file.
3. Open the netcdf file.
4. Clip the netcdf file to the desired longitude and latitude.
5. Extract the red band.
6. Extract the blue band.
7. Extract the green band.
8. Calculate the true green.
9. Stack the bands into a single array.
10. Save array as jpg image.
11. Delete netcdf file for memory optimization.

Because the dataset was manually created in a short amount of time, there are some concerns about data quality. Specifically, some data points may overlap due to the nature of geospatial imagery. Since images are generated from a lat-long origin with a 5 degree margin in either direction, it is possible for an image to contain area from another image. Due to this, data leakage is possible in model training. However, for the scope of this initial analysis this is ignored.

PyroCB events were identified using the pyrocast dataset provided by SpaceML (https://spaceml.org/repo/project/63691212f97150000d504d4d). The full pyrocast dataset is too large for me to download on my computer, so instead I created a subset of the dataset using only events detected by GOES. PyroCB event information, including latitude, longitude, date, and time, are available in the pyrocb events csv provided in pyrocast. This information was used to access the archived netcdf files in the GOES Amazon Web Service

S3 buckets (ex: https://noaa-goes17.s3.amazonaws.com/index.html). This resulted in a total of 63 PyroCb events (note that some of these contain overlapping images). There are multiple images for each event. Control events were identified using the NASA worldview application (https://worldview.earthdata.nasa.gov/). The criteria for a control event are:

1. fire event with smoke
2. not a pyroCB event

Locations and dates were checked for each proposed control event and confirmed to not overlap with the pyroCB events. If a proposed control event met the criteria, the date, location, and approximate latitude and longitude were recorded. A csv of control events was created to automatically download and construct the control event images, similar to the PyroCB Events. A total of 64 control events were created, with multiple images for each event.

For both the control and PyroCB events, each image covers a boxed region with a range of 10 degrees on both latitude and longitude. Since the images are in true color, each image can be interpreted as a 3-dimensional array with 3 color channels - one for red, one for blue, and one for green. Example images are shown in Figure 1

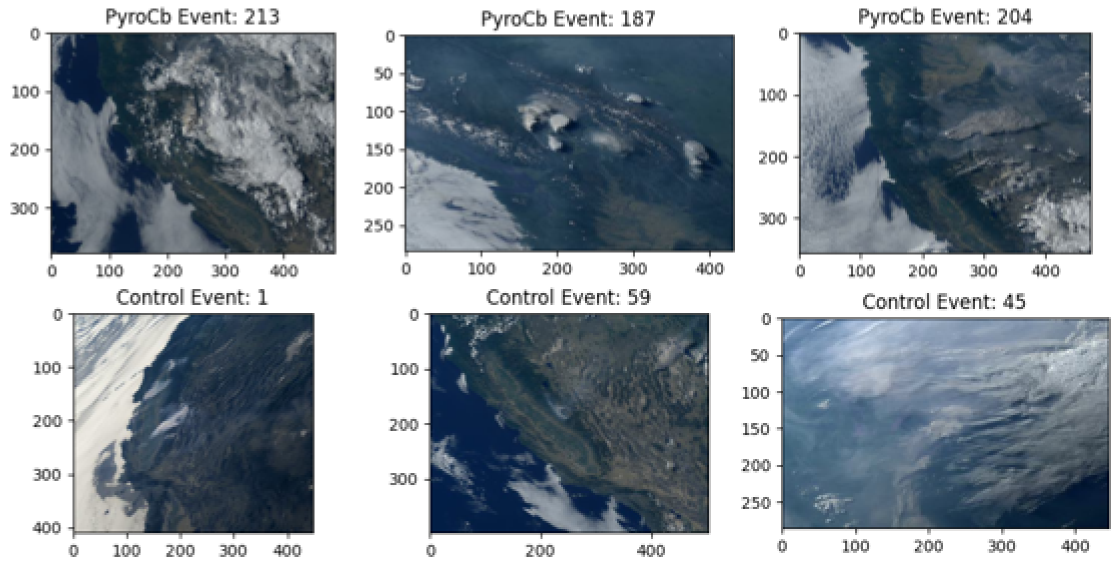
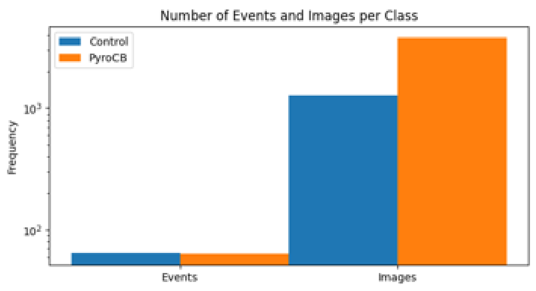


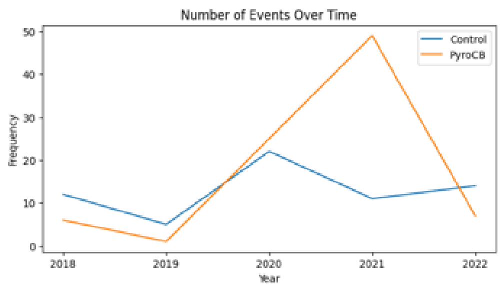
Figure 1: Example pyroCB and control images from the dataset

2.1 Exploratory Analysis

The dataset consists of true color images from wildfire events with 64 control events and 63 pyroCB events. There are multiple images for each event, resulting in a total of 10997 pyrocb images and 2056 control images. From the graphic analysis, we can see that the number of control and pyroCB events are almost the same in Figure 2a; however, there are more pyroCB images than control images. While the number of pyroCB events peaks in 2021, the control dataset contains events more evenly distributed throughout the years, seen in Figure 2b. Both datasets have around the same number of events in the US as in Canada, with the control dataset having slightly more events in the US, as evident in Figures 3 and 4. From the average and standard deviation images in Figure 5, there is no apparent difference between the pyrcoCB and control images. However, these methods may not be appropriate for summarizing the two data classes. The lack of difference between the two datasets does signify that conventional classification models which base their classifications on averages and standard deviations may struggle with detecting pyroCBs. Advanced methods specialized for image analysis, such as convolutional neural networks, may be more appropriate instead.



(a) Total number of events for each class



(b) Time series of the frequency of control and pyroCB events in the dataset

Figure 2: Control and pyroCB event counts

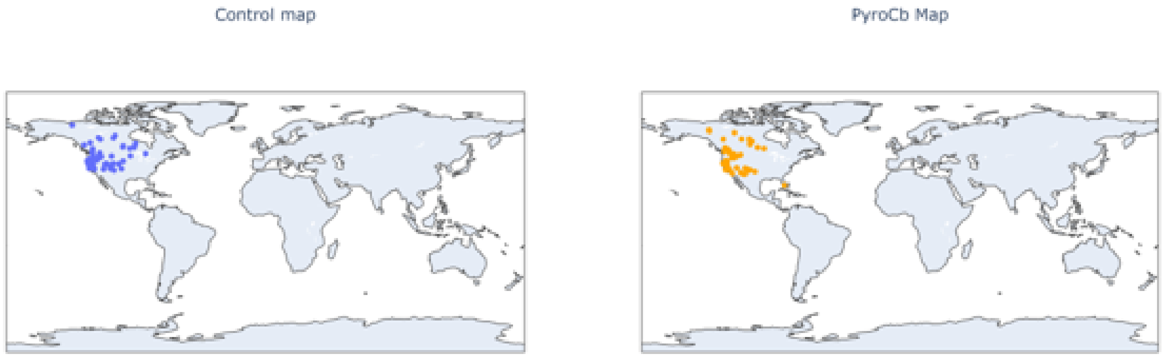


Figure 3: Geographic locations of pyroCB and Control Events

2.2 Data Pipeline

Since most conventional models do not perform well with the high-dimensional data inherent in images, additional data processing is performed. Each image is resized, converted to greyscale, and has the histogram of oriented gradients (HOG) calculated prior to being input into the model. Images are resized from any intial shape to a (256,256) shape array, which creates a uniform input dimension. The reshaped images still have three color channels, which are combined into one channel by converting the image to greyscale. The HOG is a well-established method to extract features from an image, in turn simplifying the input for a statistical model. Intuitively, HOG identifies features through gradients by assuming that meaningful shapes in images are captured by changes in color and intensity, which can be captured through oriented gradients. The steps to calculate the HOG in this implementation can be found in the sklearn image documentation (https://scikit-image.org/docs/stable/api/skimage.feature.html#skimage.feature.hog) and include:

1. Normalizing the image - normalize each pixel value by scaling according to a L2 norm by default
2. Calculating the gradient for each pixel
3. Generating a histogram of the gradients
4. Normalizing the histogram
5. Flattening the histogram into a 1-dimensional feature vector

Example images and their HOGs are provided in Figure 6. The general idea is that pyroCBs may have a signature HOG that can differentiate them from non-pyroCBs.

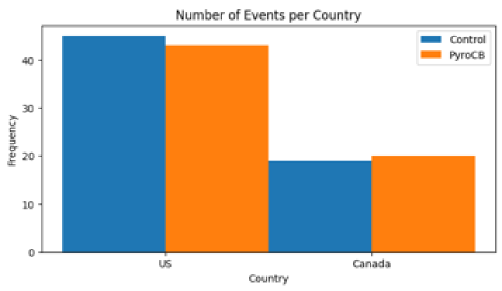
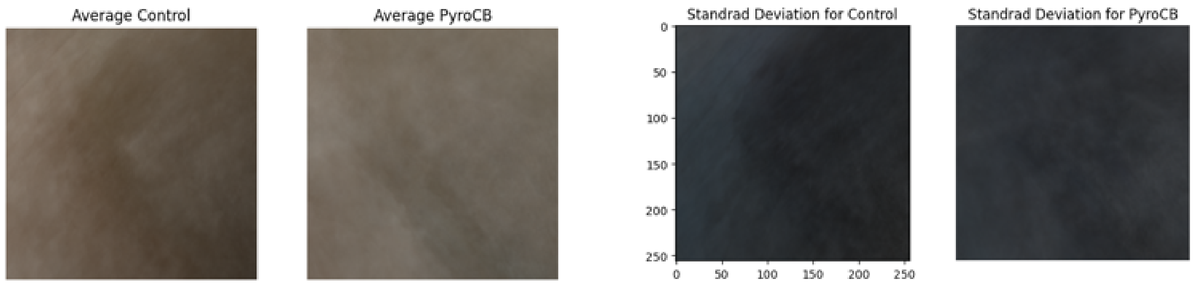


Figure 4: Distribution of events over the US and Canada



(a) Average of the images

(b) Standard deviation of the images

Figure 5

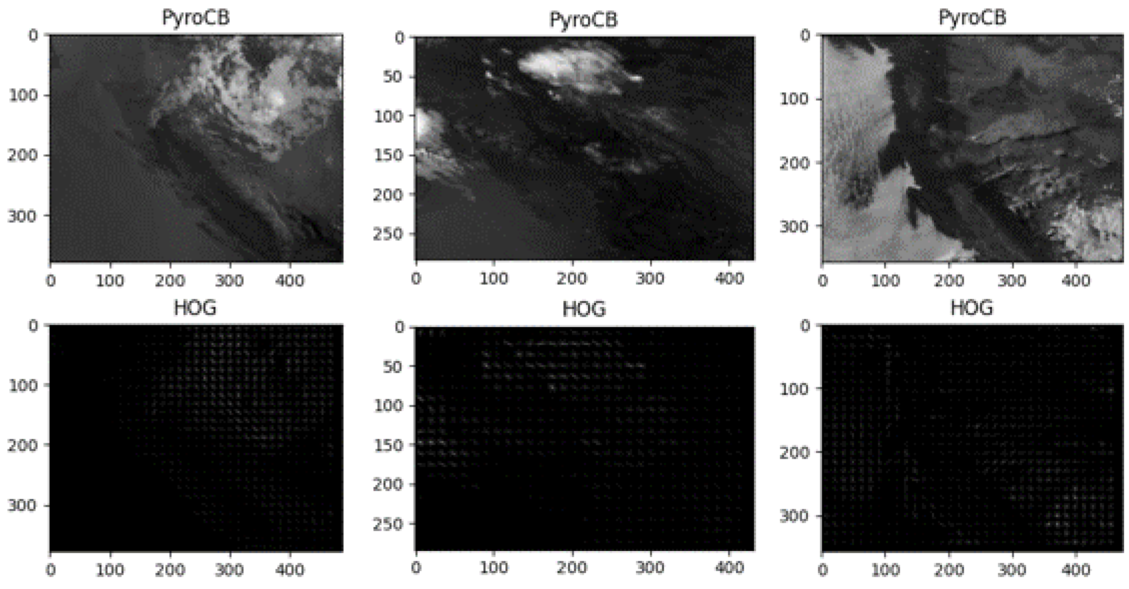


Figure 6: Example histogram of oriented gradients for sample images from the pyroCB class

3 Model Selection

Each model is trained and tested using stratified 5-fold cross-validation to estimate true model performance. The cross-validation folds are defined prior to training so each model trains/tests on the same 5-fold set. As seen in the exploratory analysis, the number of images is imbalanced. Majority class undersampling is performed to balance the number of each image in each fold of the 5-fold cross-validation. Models are evaluated on the test set of each fold using accuracy, precision, recall, f-1, and ROC curves. The accuracy score provides the proportion of total predictions that are correct, given by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Where TP is true positive, TN is true negative, FP is false positive, and FN is false negative. Precision provides a measure of the proportion of true positives in relation to total positives, given by:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Hence a model with a precision score close to one will have very few false positive classifications. Recall provides a measure of the proportion of true positives that are identified by the model, given by:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Thus a model with a recall score close to one will correctly classify almost all of the true positive cases. F-1 score provides a harmonic mean of recall and precision, resulting in a balance between false positive and true positive rate"

$$F - 1 = 2\frac{precision * recall}{precision + recall} \tag{4}$$

Receiver operating characteristics (ROC) curves are generated for each fold, then averaged to get each model's average ROC curve. All fold ROC curves and the average curve are displayed in a plot after training each model. The ROC curve provides a measure of the predictive power of the model, where curves close to the upper left corner with an area under the curve close to one are ideal.

3.1 Logistic Regression

The logistic regression model provides a classification by modeling the log odds of belonging to the target class, in this case a pyroCB event. The general logistic regression model including p predictors is given by:

$$logit(P(Y = 1)) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p \tag{5}$$

where $logit(P(Y = 1)) = log(\frac{P(Y=1)}{1-P(Y=1)})$. In the context of this image classification task, the input X vector is the array of the histogram of oriented gradients and y is the class label where 1 is a pyroCB event. The logistic regression model was chosen for this application due to its ease of probability interpretation and wide applicability.

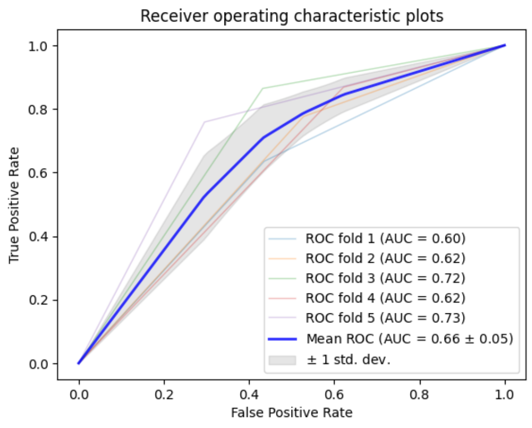


Figure 7: ROC curves over the five cross validation fold for the logistic regression model

Figure 8: Classification metrics on the validation set over the five folds for logistic regression model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.600	0.600	0.600	0.600	0.600	258	177	229	148
2	0.624	0.637	0.624	0.615	0.624	291	198	177	84
3	0.716	0.737	0.716	0.710	0.716	344	172	226	54
4	0.623	0.662	0.623	0.599	0.623	409	293	178	62
5	0.732	0.732	0.732	0.731	0.732	308	120	286	98
avg	0.659	0.674	0.659	0.651	0.659	322	192	219	89

3.2 KNN

The K-nearest neighbors model provides a classification by identifying the k-closest events in the input space, then assigning the class to the new input based on the classes of the closest inputs. The underlying assumption in KNN is that inputs belonging to the same class will be close together in the feature or input space. KNN is chosen for this application due to its non-parametric nature, which tends to be well-suited for high-dimensional inputs such as images.

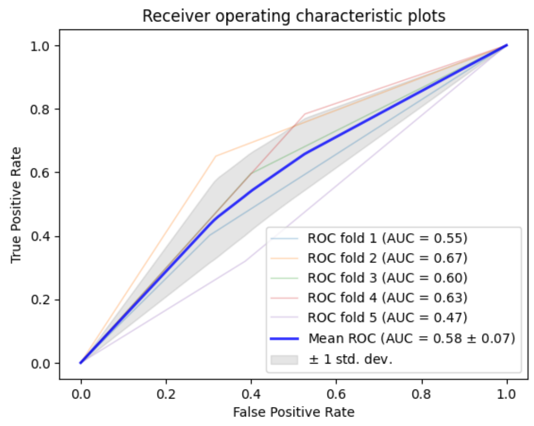


Figure 9: ROC curves over the five cross validation fold for the KNN model

Figure 10: Classification metrics on the validation set over the five folds for KNN model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.549	0.554	0.549	0.539	0.549	163	123	283	243
2	0.667	0.667	0.667	0.667	0.667	244	119	256	131
3	0.580	0.598	0.598	0.598	0.598	237	159	239	161
4	0.628	0.642	0.628	0.619	0.628	369	248	223	102
5	0.467	0.464	0.467	0.455	0.467	130	157	249	276
avg	0.582	0.585	0.582	0.576	0.582	229	161	250	183

3.3 SVM

Support vector machines perform classification by identifying a hyperplane in the feature space that best fits the classes. The hyperplane can be linear or non-linear depending on the model specification. Given a labeled training set of observations, SVM classifiers find an optimized hyperplane boundary in the feature space that separates the classes. SVM classifiers have a history of successful performance for image classification applications and thus have been chosen as a potential model for identifying pyroCB images.

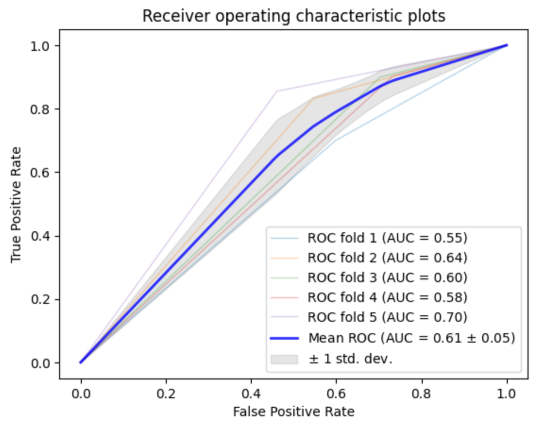


Figure 11: ROC curves over the five cross validation fold for the SVM model

Figure 12: Classification metrics on the validation set over the five folds for SVM model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.550	0.555	0.550	0.540	0.550	284	243	163	122
2	0.643	0.667	0.643	0.629	0.643	312	205	170	63
3	0.580	0.654	0.598	0.558	0.598	358	280	118	40
4	0.585	0.642	0.585	0.538	0.585	425	345	126	46
5	0.697	0.719	0.697	0.689	0.697	347	187	219	59
avg	0.615	0.647	0.615	0.591	0.615	345	252	159	66

3.4 LDA

The Linear Discriminant Analysis model performs classification by estimating probabilities in Bayes' theorem. In particular, LDA models the distribution of observations given each class, $P(X = x|Y = k)$, assuming the distribution is normal with equal variances for each class. The model of the probability of a given input, x, for each class, k, results in a linear decision boundary on the feature space that is used to assign a class. In this application, the LDA model will identify the probability of each image HOG given each class. This model was chosen due to its usefulness for high-dimensional data.

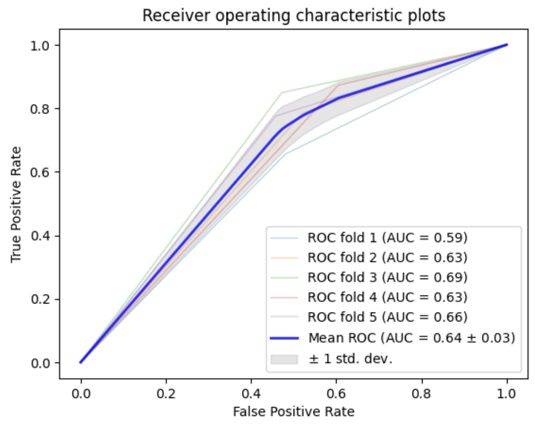


Figure 13: ROC curves over the five cross validation fold for the LDA model

Figure 14: Classification metrics on the validation set over the five folds for LDA model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.587	0.589	0.587	0.586	0.587	266	195	211	140
2	0.632	0.646	0.632	0.623	0.632	295	196	179	80
3	0.688	0.710	0.688	0.680	0.688	338	188	210	60
4	0.634	0.673	0.634	0.612	0.634	411	285	186	60
5	0.659	0.668	0.659	0.654	0.659	315	186	220	91
avg	0.640	0.657	0.640	0.631	0.640	325	210	201	86

3.5 QDA

The Quadratic Discriminant Analysis model is similar to the LDA model, except that QDA has a quadratic decision boundary. A quadratic decision boundary originates from QDA not assuming equal variances in the feature space across the classes. QDA was chosen for this application as a comparison to LDA.

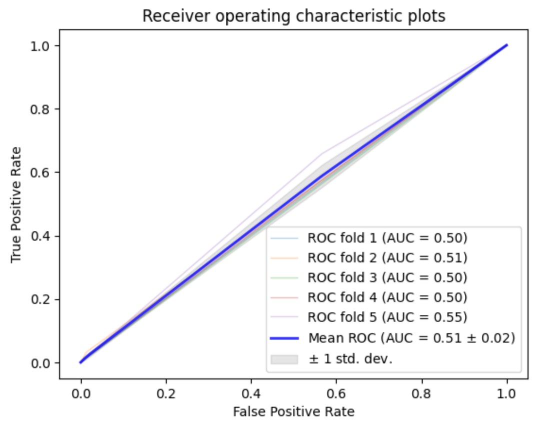


Figure 15: ROC curves over the five cross validation fold for the QDA model

Figure 16: Classification metrics on the validation set over the five folds for QDA model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.501	0.501	0.501	0.499	0.501	176	175	231	230
2	0.509	0.605	0.509	0.365	0.509	12	5	370	363
3	0.497	0.495	0.497	0.423	0.497	341	343	55	57
4	0.504	0.504	0.504	0.502	0.504	267	263	208	204
5	0.546	0.548	0.546	0.540	0.546	267	230	176	139
avg	0.512	0.531	0.512	0.466	0.512	213	203	208	199

3.6 Random Forest

The random forest classifier is a bagging algorithm that creates a random group of decision trees with varying depths and features. A decision tree is another type of non-parametric model which builds hierarchical tree based on the features to support classification. At each decision node in the tree, a binary split-point is created based on a selected feature. Then, paths from the root node through decision nodes are built to reach terminal nodes, which provide a classification for each input. Decision tree split-points are optimized according to the labeled training data. Random forests are composed of a collection of decision trees, where a classification can be made for a new input by passing the input through each decision tree in the forest and assigning the most probable class. This model was chosen for this application due to their high performance across a variety of applications; however, they are known to struggle with image data.

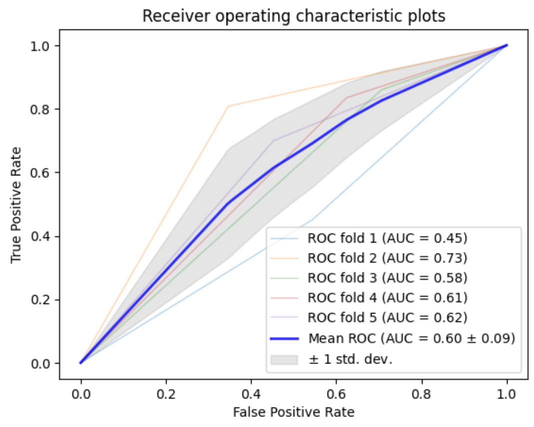


Figure 17: ROC curves over the five cross validation fold for the random forest model

Figure 18: Classification metrics on the validation set over the five folds for random forest model

Fold	Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
1	0.453	0.453	0.4532	0.453	0.453	183	221	185	223
2	0.731	0.736	0.731	0.729	0.731	303	130	245	72
3	0.577	0.613	0.577	0.540	0.577	342	281	117	56
4	0.605	0.633	0.605	0.583	0.605	393	294	177	78
5	0.623	0.626	0.623	0.621	0.623	284	184	222	122
avg	0.598	0.612	0.598	0.585	0.598	301	222	189	110

3.7 Convolutional Neural Network

Convolutional neural networks are known for their high performance in image analysis tasks, including classification. The core feature of a convolutional neural network is the presence of one or more convolutional layers. A convolutional layer applies a convolution to the output of the previous layer, where a convolution applies a filter to the input layer, producing a new output. A 3 layer convolutional neural network is applied to the pyroCB classification problem.

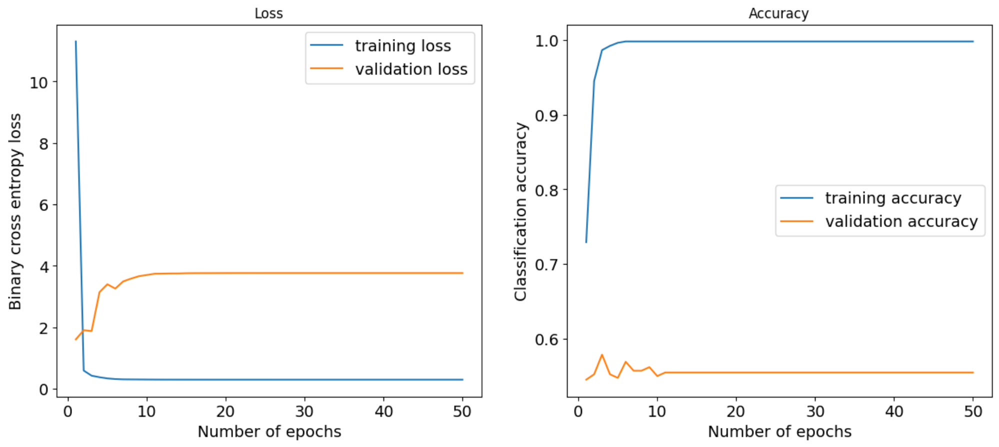


Figure 19: Accuracy and loss during training for the convolutional neural network.

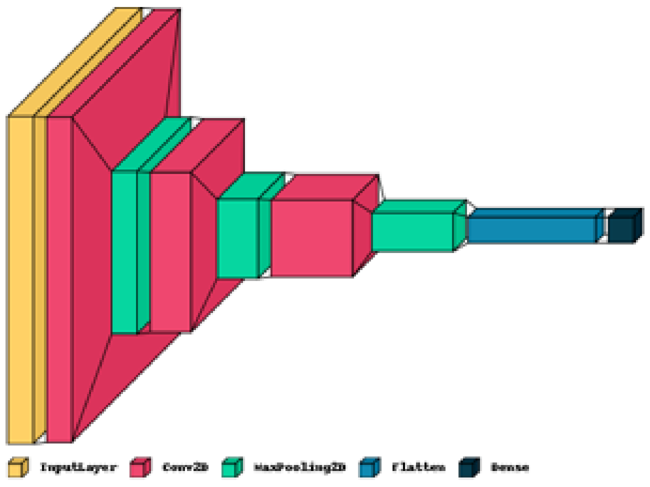


Figure 20: CNN model architecture

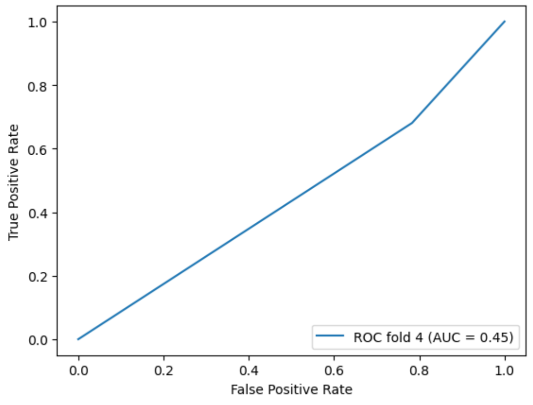


Figure 21: ROC curve on the validation set for the CNN model.

Table 1: Classification metrics for the CNN on the validation set

Accuracy	Precision	Recall	F-1	AUC	TP	FP	TN	FN
0.449	0.435	0.449	0.417	0.449	113	130	36	53

4 Model Comparison and Discussion

Figure 22 summarizes the average performance of each model across the 5 validation folds, except for the CNN which only used one validation set. The logistic regression model performs the best across the board, followed by the LDA and SVM models. The QDA and CNN models perform the worst, the CNN not doing any better than a random guess. However, the best models have accuracy less than 70%, which is a fairly low rate. Since none of the models obtain a high accuracy, it is possible that classifying pyroCbs requires additional data than just true color imagery, such as temperature, pressure, and water vapor.

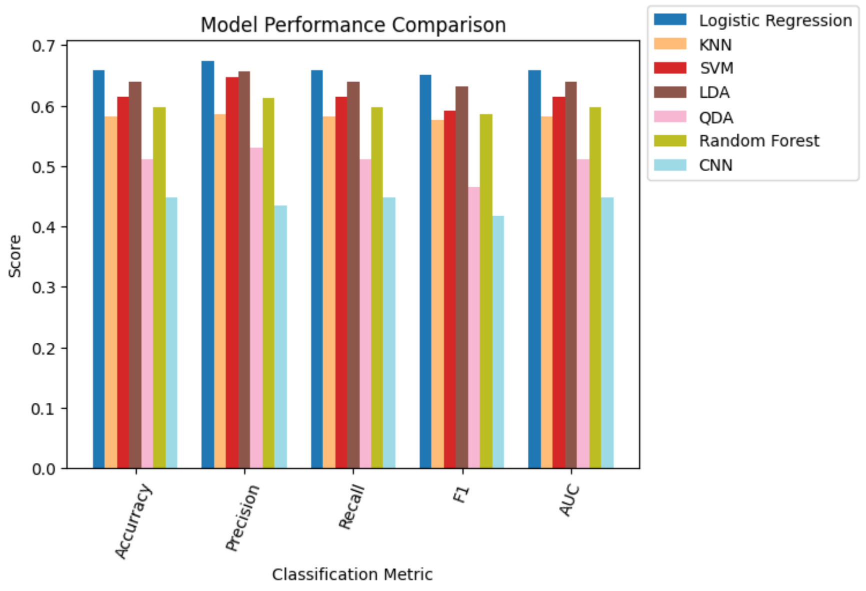


Figure 22: Comparison of the seven models tested, with average metrics on the validation sets shown

5 Conclusion

The goal of this project was to build a pyroCb classifier, with seven models total tested. After balancing the data using under sampling, logistic regression performs the best while a convolutional neural network performs the worst. The highest scores obtained were by the logistic regression model, with an accuracy of 0.659, precision of 0.674, recall of 0.659, f-1 of 0.651, and AUC of 0.659. Overall, the model has low predictive power and unsatisfactory performance. Future work can investigate existing physics-based pyroCb detection models and include relevant additional variables.

6 Appendix: Code