Night-time cloud classification with machine learning

Akin Kucukkurt

Abstract: Using the decision tree machine learning algorithm provided by Scikit, a module for Python, an algorithm that predicts general cloud coverage and an algorithm that predicts cirrus cloud coverage is trained and tested in both the okta scale and a more simplified scale with the aim of predicting whether the sky is clear , a little cloudy and too cloudy for observations. The code is stored on Github and can be used by anyone for the same application. The images used are from the Bayfordbury Observatory's Allsky camera. An accurate result of 93% is achieved for general cloud coverage that achieves the aims of this project. Whilst a high number of images is required for this project, the results here and in other similar papers show that machine learning has a useful place in identifying and processing images of the sky for various applications.

1. Introduction

The aim of this project is to provide user friendly code which allows anyone to perform decision tree machine learning for cloud coverage as well as explain the code and explain some aspects about decision trees that can affect your results. The code itself is split into 4 parts and is explained through its comments with a short list of variables at the beginning that need to be changed to customise it to your circumstances, an overview of the code is given in the method section and a Github url is provided in Appendix A. The images were given to me and classified in the okta scale, by David Campbell of the Bayfordbury Observatory at the University of Hertfordshire, who also set up the Allsky camera that took those images. The original images are FITS images but their JPEG versions are used here due to their large size. The origins of the okta scale system are unclear, nevertheless this system of 0-8 may be being used as you can draw or imagine a grid or a circle with portions of 1/8 that you can project onto the sky to come up with a simple measure of cloud coverage. Whilst a decision tree is used, other algorithms could have been used like 'random forests' , which is similar, that outperforms the decision tree (Deng 2018). One paper uses this method and only 717 samples to predict cloud types and achieves a good result of around 70% (Huertas-Tato et al. 2017). A general overview covering different methods that measure cloud coverage has also been done (Baran et al. 2020).

2. The Data Set

The camera taking the images is the AllSky-340, which has a Kodak KAI-0340 CCD sensor and a Fujinon FE185C046HA-1 lens and it is located on the roof of the East of England Science Learning Centre which is a few hundred meters away from the Bayfordbury telescopes (Campbell 2010). Information about the images and their classifications are stored inside a CSV file.

filename_i	image_nur	julian_date	humidity	temperatu	pressure	cirrus	cloud	moon_alt	moon_phase	sun_alt	skymag_r	skymag_v	skymag_b	skymag_c	exp_time	pyrg
0	2668441	2457059.5	85	0.9	1023.4	0	0	33	96	-50.24	21.96	21.6	22.25	20.03	7.3318	-23.36
1	2668486	2457059.5	86	0.7	1023.4	0	0	35.4	96	-51.88	21.87	21.51	22.2	19.95	7.3318	-23.64
2	2668514	2457059.5	87	0.5	1023.4	0	0	36.8	96	-52.71	21.9	21.53	22.18	19.98	7.3318	-23.64
3	2668619	2457059.5	89	0	1023.3	0	0	41.1	96	-53.99	21.77	21.47	22.09	19.88	7.997	-23.17

Figure 1: A screen cap of the data-set where the classes of 0-8 in the 'cloud' and 'cirrus' column are used for the algorithm.

The 'filename_index' stores the name/number of each image inside this CSV file, the 'cloud' column has a value from 0-8 representing the classification of general cloud coverage as a fraction of 8 and the 'cirrus' column is the same except for it being only the cirrus cloud coverage. The 'image_number' column is the number of the image inside the Bayfordbury AllSky camera database as it constantly takes pictures of the sky. The 'pyrg' column is the measurement from the pyrgeometer, a device that measures the near-surface infra-red radiation spectrum, for the image where lower numbers represent a clearer sky as -16 is around the too cloudy

to observe mark and 0 is overcast, however this column is not used as it is not a very reliable measure for cloud coverage. The information in the rest of the columns is taken from FITS header of the image and whilst they are potentially useful, they are not used as they are not the focus of this paper. From the 'filename_index' column, 0-4500 are images that are all rated 0 for the 'cloud column' and rated 0-8 in the 'cirrus' column. The general pattern is that about half of this range is rated 0 in the 'cirrus' column with packets of classifications above 0 distributed randomly. The range from 4500-13000 are images that are all rated 0 in the 'cirrus' column and rated 0-8 in the 'cloud' column. The pattern here is about an even number of images for each class with the eighth class having about twice as many images. As such 0-4500 can be referred to as the Cirrus-only subset and 4500-13000 can be referred to as the Cloud only subset.

3. The Method

For my method I used python with the 'Pandas' (McKinney et al. 2010) module for easy image and table manipulation, the 'Scikit' module(Pedregosa et al. 2011) for the machine learning algorithm and the 'Squircle' (Verkhovskiy 2019) module to turn a circular image into a square. I have split my method into four separate python files,'SquircleApplied.py' which deals with image formatting, 'Samples and Features.py', which formats and stores the pixel value into an array to be used by 'Scikit', 'Machine Algorithm Trainer(modified).py' which trains the algorithm using a specific image range and 'Predictor.py' which predicts classifications, compares them and outputs the results in a CSV file.

3.1. Cutting and Transforming the image

The initial image from the sky camera looks like this.

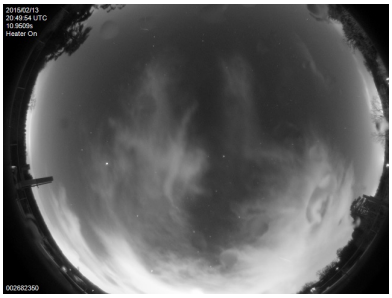


Figure 2: Initial image taken by the Bayfordbury All Sky camera.

It is important to crop this image to remove features that are not necessary for classification as they might interfere with the algorithm and it will lower computation time for this whole method. The land features around the edge fit this description. There is also light pollution around the edges which may look like a cloud to the algorithm. The telescope domes at Bayfordbury cannot depress enough to view the sky towards the outer part of the images anyway so it can be cropped out. To do this I apply a circular mask to crop out the unwanted features and then crop the rectangular image into a square that is still masked to look like a circle.

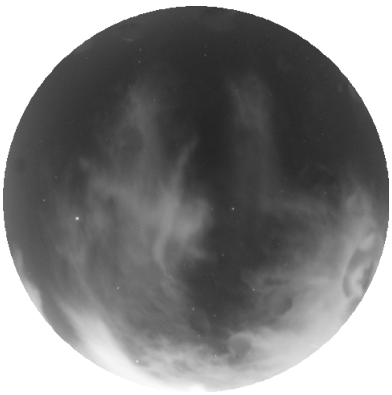
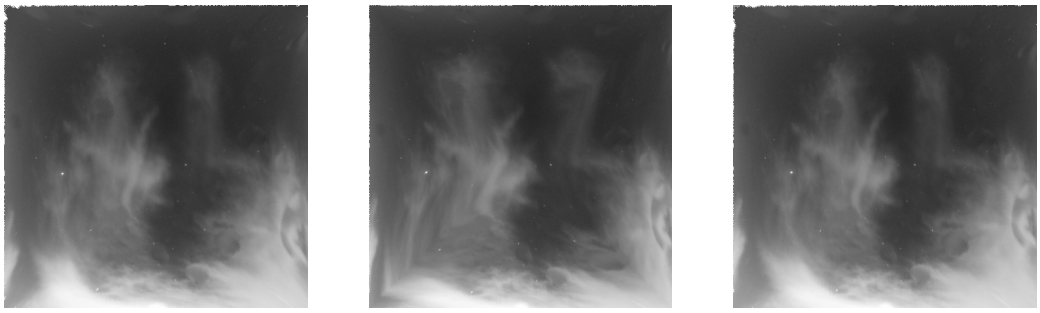


Figure 3: Image with the circular mask applied.

Whilst I could put this straight into the algorithm, the masked area is empty space and the algorithms' decision tree will be simpler without having to take that and the circular nature of the actual data into account. Ideally I want to turn the actual circular data into a square. In order to do this I use the Squircle function. This function only works when the circular looking masked image is a square.



(a) FGS (b) Stretch (c) Elliptical

Figure 4: Image outputs for all three 'squircle' methods.

It is quite obvious that the stretch method is unusable as it deforms the image, so the choice was left between the 'FGS' method which is the Fernández-Guasti squircle method and Nowell's Elliptical Grid mapping. Manuel Fernandez Guasti created a formula that allows you to smoothly blend a square into a circle with an increase in size as the circle becomes more square like (Fong 2015) and so I will refer to this as the smooth square. The 'FGS' method linearly maps concentric circles in a circular image with the smooth square to create a square image, this is shown in Figure B.19 in Appendix B. Philip Nowell created a mapping that converts horizontal and vertical lines in a square to elliptical arcs inside a circular region (Fong 2015), this is shown in Figure B.18 in Appendix B. Neither of these methods conserve area or angle but I decided to use the 'FGS' method as it is linear and so I thought it would be more consistent between images and that it might preserve the area better. Area is measured by the 0-8 okta scale, and the 'Elliptical' mapping is also usable. There is an obvious issue with all three of these images, namely the white pixels around the sides. They are there because the masked image wasn't a perfect circle since it is made up of square pixels. The last part of my code crops it out and makes the whole process iterable.

3.2. Creating the Learning Array

The next piece of code loops through the numpy array of each image and takes out the brightness value for each pixel, and then stores it in a two dimensional array where the parent array is the image number of each image as corresponding to the "filename_index" column and the child array is the brightness for each pixel of the parent image. Then this array is stored as a compressed .npz file that takes up around 1.85 GB of space for 13000 images.

3.3. Training the Algorithm

Lastly to train the algorithm, an array is created through my 'modfunc' function which is an array of image numbers that is decided by user input. Then the CSV file containing the classification for each image is loaded followed by the two dimensional array storing pixel brightness. The next step is the indexing of this array such that it only contains images with the image numbers from the 'modfunc' array. Finally, the algorithm then fits the CSV file classifications to the data array, this is done by scikit, which trains the algorithm that is then stored in a .joblib file. This whole process is put in an 'if' statement which depends on whether you have chosen to use the 'cloud' column or the 'cirrus' column. Also in my case I used 'modfunc' such that it stored 100 numbers for every step of 200, which means that each algorithm I used was trained on half the images of the whole chosen image range.

3.4. Predicting values

Once again 'modfunc' is used to create an array of image numbers but in this case it is the images where the algorithm will predict its classification. In my case I used it as shown above but I changed the starting number by 100 which then creates an array of the half of the image range that the algorithm was not trained on. The two dimensional array storing pixel brightness is loaded and is once again filtered to only include image numbers

from the 'modfunc' array. I then used scikit to predict the images whilst also keeping a count of wrong images as well as the difference between the target answer and the prediction. Finally this is all stored as a CSV file with the name derived from 'modfunc' as well as what column you are using. This whole process is put in an 'if' statement which depends on whether you have chosen to use the 'cloud' column or the 'cirrus' column.

4. Analysis of Results

The initial and simplest results are as follows in figures 5 and 6.

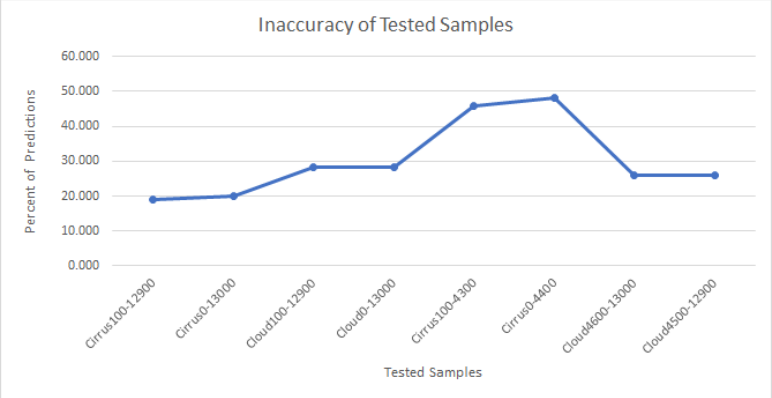


Figure 5: Graph of inaccuracy for each test sample.

Test Sample	Inaccuracy Percent
Cirrus100-12900	18.9
Cirrus0-13000	19.9
Cloud100-12900	28.2
Cloud0-13000	28.3
Cirrus100-4300	45.8
Cirrus0-4400	48.3
Cloud4600-13000	25.8
Cloud4500-12900	25.9

Figure 6: Table of inaccuracy for each test sample.

The test samples describe which algorithm was used, corresponding to the two columns for the classified values in the data-set. It also describes the range of images that were tested which are a series of 100 images for every interval of 200. So the images that each one was trained on is the opposite series e.g., tested on 100-12900 means it was trained on 0-13000. The Inaccuracy Percent column describes the percentage of images the algorithm got wrong which is to say the predicted classification was not equal to David's classification. This isn't a complete measure as there is no value for the accuracy of the algorithm per image, this is simply the overall accuracy. It is rather misleading and doesn't tell you how much or what the algorithm has learned. However I can use the information about the different subsets that the algorithms have performed on, to make some meaningful predictions from this graph.

The Cirrus algorithms have a low accuracy when only the images that were in the Cirrus subset are used. Using what I know of this subset I can say that there isn't an obvious systematic issue with it and so the error is mostly due to measurement error. This presents itself as over-fitting as the algorithms learn the differences in each image instead of getting the general relation. Over-fitting essentially means the signal to noise ratio is low. The signal is the brightness of a cloud with total cloud/brightness coverage which means both subsets, Cloud and Cirrus, are looking for the same signal. The noise is the variation of the images in the subset which should be lower for the Cirrus subset as it only has images of one type of cloud.

Using what I know of the subset and these results I can conclude that the low signal to noise ratio is due to lower number of images. There is a significant difference between the Cirrus100-4300 and the Cirrus0-4400

results, this could be due to the differences in the subset they were trained and tested on, as well as the difference in the number of images they were trained on, as Cirrus100-4300 is trained on 100 more images. The fact that Cloud4600-13000 and Cloud4500-12900 are around twice as accurate and have around twice as many images suggest that the number of images is the dominant factor.

Although Cirrus100-12900 and Cirrus0-13000 have the best results, it is misleading. The Cirrus subset has mostly zeros and in the Cirrus column the Cloud subset has zero for every image, which means that these two algorithms have been trained to label zero so they don't have any practical use.

For the Cloud algorithms those two that were trained only on the Cloud subset have the best result. No doubt that adding the Cirrus subset confused the algorithms as they are all rated zero for the Cloud result column. Though the difference between the two is small considering that Cloud0-13000 and Cloud100-12900 have 50% more images that are all labelled incorrectly as zero. Around half the Cirrus subset is classified as zero so this helps mitigate the effect of adding the incorrectly labelled subset.

4.1. Analysis of difference between prediction and actual classification.

A simple representation of the difference between the algorithm predictions and the actual classifications is shown below in figure 7 which is a meaningful representation of how much each algorithm has learnt.

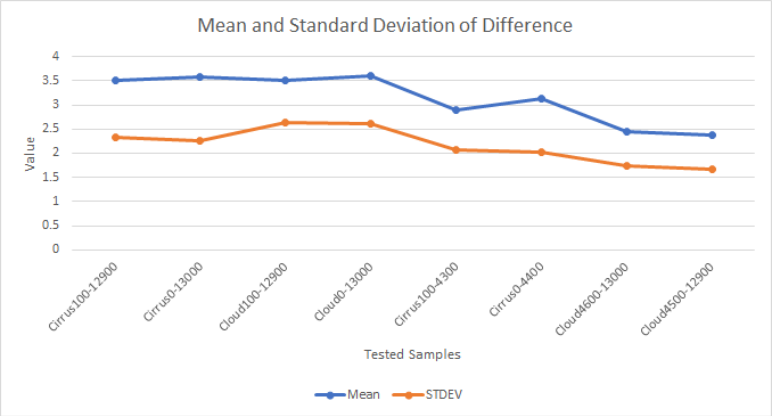


Figure 7: Graph of mean and standard deviation of difference between predicted classes and Davids' classes

These values were calculated by looking at all the wrong predictions and finding the difference between the prediction and the actual classified value for each. Both the Cloud algorithms and the Cirrus algorithms had around the same mean difference when performing over the whole data-set though the Cirrus algorithms have a slightly lower standard deviation which means they were a bit more consistent but not by much. Even though the Cirrus algorithms have the most initial accuracy, we can see from here the effect of the issue that I already mentioned, they are good at classifying zeros but when the image has a value above zero they struggle the most with giving a value near the classified value. The fact that both the Cloud algorithms and the Cirrus algorithms have the same mean and a similar standard deviation is interesting as it would suggest that they both performed similarly even though they dealt with significantly different classifications and data. When I look at a more detailed analysis of the difference later on, there is a significant difference visible between them.

Cirrus100-4300 and Cirrus0-4400 actually perform better even though they have the lowest initial accuracy. They suffer from over-fitting which means that they have a hard time getting the exact value but the algorithms seems to have learned better as they get closer to the real value. This makes sense as they performed with properly classified values instead of just zeros. There is slight difference between them which as discussed above, I concluded to be from the fact that Cirrus100-4300 was trained on 100 more images which makes it a bit better.

Cloud4600-13000 and Cloud4500-12900 have the best results. They have the lowest mean but their standard deviation is slightly closer to the mean so they are a bit less consistent which might be due to the increased variation in the Cloud subset. The low mean is consistent with the fact that they perform on a properly classified subset (0-8) and they have the most number of images. The difference between these two and the Cirrus-only algorithms before them, isn't as much as the difference in initial accuracy. Even though the Cloud algorithms are twice as accurate overall they haven't learnt twice as much which means that the Cirrus subset has a systematic issue that is causing its algorithms to be less accurate overall, which isn't something I was able to predict in the first part of this section. I will analyse where this comes from later on.

A more detailed representation is shown next in figure 8 and 9 which compares high and low differences between predicted and actual values.

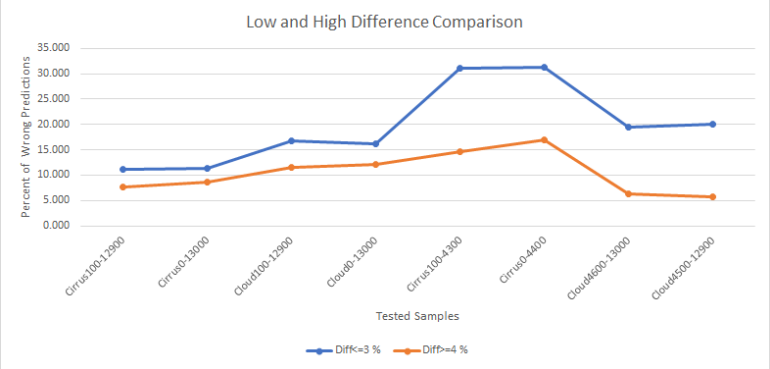


Figure 8: Graph of high and low difference between predicted classes and David's classes.

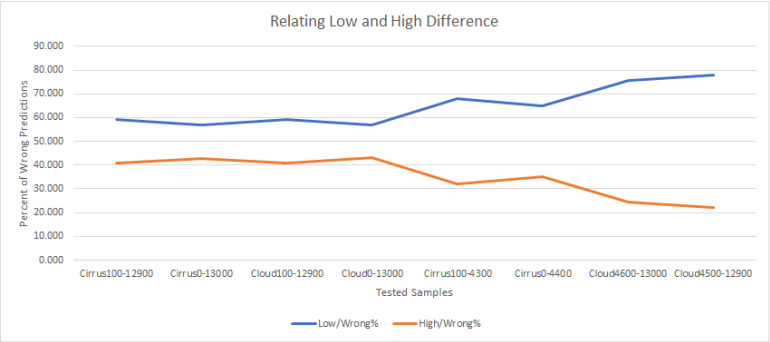


Figure 9: Graph showing what percentage of wrong predictions were high difference(4-7) and low difference(0-3).

Figure 8 represents the percentage of predictions that had a difference of 0-3 from David's classification as the top line and the bottom represents the predictions that had a difference of 4-8. Over-fitting will cause the algorithm to make small mistakes so the low difference line can be approximated as the size of the over-fit error. The high differences are due to a systematic error as it approximately only occurs when the classification has a large error which suggests a mistake. These two can have some overlap but the boundaries I have chosen are approximately where each is respectively dominant. In the case of the four algorithms working on the entire data-set, the systematic error effects the low difference range as well which means the first four points on this graph are misleading, as it suggests the dominant issue was over-fit when it really was not. For that reason I won't be analysing them. An easy way to relate the two lines is to look at what percentage of the total error they make up which leads to figure 9.

The graph helps confirm what I predicted about over-fit being the main issue. Cirrus100-4300 has a bit more over-fit error than Cirrus0-4400 which is good as it confirms the effect of having more images as I predicted in the first part of this section. These two have more systematic error than Cloud4600-13000 and Cloud4500-12900 which confirms what I concluded above, which is that the Cirrus subset has a higher systematic error than the Cloud subset.

I can get some more information by looking at the specifics of the low difference from figure 10.

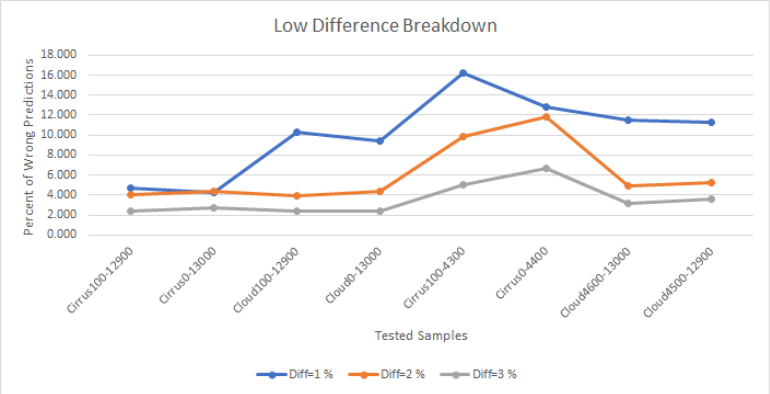


Figure 10: Graph showing differences of 1-3 between predicted classes and David's classes.

Cirrus100-12900 and Cirrus0-13000 stand out with an abnormal distribution due to their flawed nature and there is finally a difference between these two, Cloud100-12900 and Cloud0-13000 which have a normal looking distribution which shows that they aren't as flawed.

Cirrus100-4300 is significantly better than Cirrus0-4400 which is a repeat result. These two results however seem to be inflated in comparison to Cloud4600-13000 and Cloud4500-12900, which is due to having a lower number of images leading more over-fit which is represented in this graph.

In conclusion the best algorithm seems to be Cloud4500-12900, which has the most number of images and even though it suffers from increased variety, the large amount of images seems to make up for it. Training over the whole data-set when it isn't defined for both parameters predictably leads to failure. The cirrus subset seems to suffer from increased systematic error as well over-fit error.

4.2. Image Analysis

I can find an explanation for the systematic error by analysing the image sets. First of all though, it's important to mention the algorithms had no problem dealing with the moon, when looking through the list of images that the algorithms got wrong, virtually none of them were of the moon.

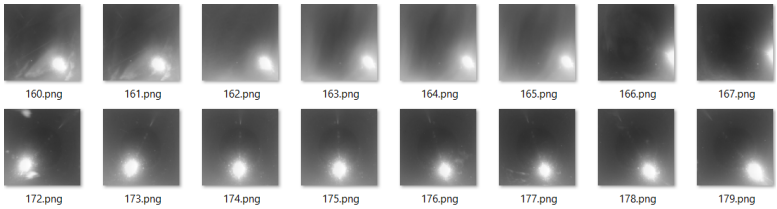


Figure 11: Screen cap showing images with a bright moon

The explanation for this is that the moon is much brighter than any cloud so the algorithms learn to ignore it.

The algorithms, especially the Cirrus-only algorithms can't tell the difference between light pollution and clouds.

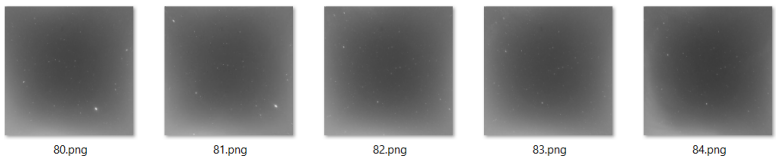


Figure 12: Screen cap of some images showing light pollution in the Cirrus subset.

The information the algorithms have is the gamma value of each pixel and so light pollution has the same effect as clouds, Cirrus-only algorithms especially struggle because Cirrus clouds tend to have low brightness

and are more likely to send the same signal as light pollution. The algorithms did properly predict some of these images but they were very inconsistent.

There is another problem with the Cirrus set, the Cirrus-only algorithms struggled to identify thin Cirrus clouds.

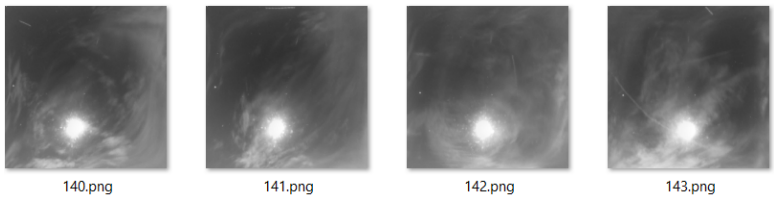


Figure 13: Screen cap of some images showing some thin Cirrus clouds in the Cirrus subset.

Including thick cirrus clouds and David labelling very light clouds as zero seems to have confused the algorithms as they are essentially getting two separate signals and so it fails to recognise thin cirrus clouds. Couple that with the issue discussed before and the fact that it has less images, I can see why the Cirrus algorithms had a large systematic error and the lowest overall accuracy. The entire reason of having a cirrus category was to identify a type of cloud that doesn't interfere much with telescopic observations but thick cirrus clouds do have a large interference so they should actually not be included in this category.

Earlier I predicted that the Cloud subset should have more variety and therefore should suffers from its effects. It has any type of cloud inside it, including cirrus clouds.

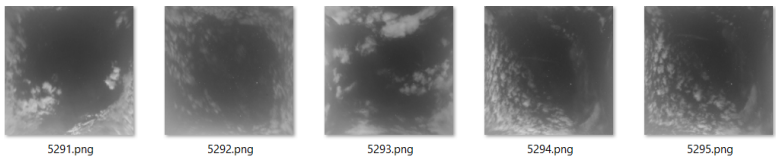


Figure 14: Screen cap of images showing cirrus clouds in the Cloud subset.

Every Cloud image is classified as zero in the cirrus column and with this type of variety that looks likes the Cirrus set, I can put another reason on why Cirrus0-13000 and Cirrus100-12900 were a failure. However the images of cirrus clouds in this subset are fairly thick and bright, despite the variety in cloud types and shapes, the Cloud subset has clouds that are of similar brightness which means this subset doesn't suffer from the issue mentioned above despite seemingly having more variety. You can see more evidence for these explanations with figure 15. Also, whilst about half of the Cirrus subset is classified as zero, the Cloud subset has about equal numbers of images for each classification with the eighth class being about twice as big as the rest.

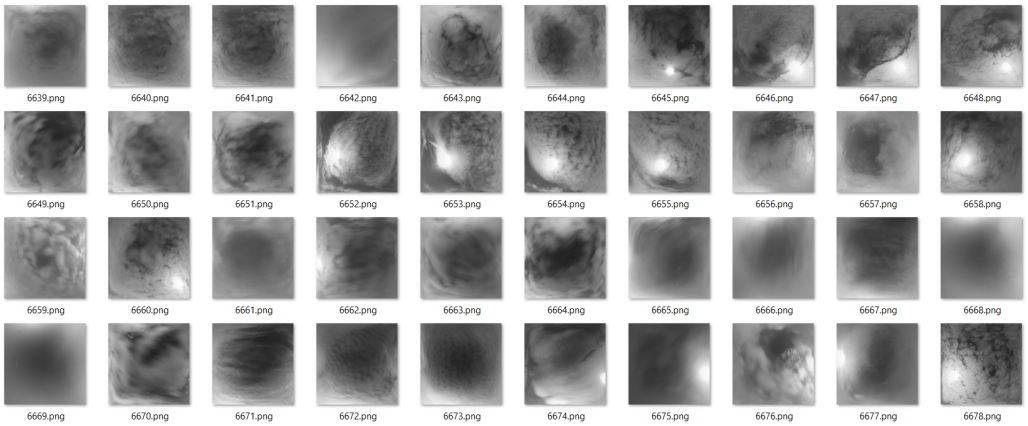


Figure 15: Screen cap of images showing varied cloud cover in the Cloud subset.

Whilst I can now tell why the Cirrus subset was more flawed than the Cloud subset, this image analysis allows me to see what was wrong with the classification system overall and how I can improve the algorithms.

The classification system needs to become more simple. Having a specific measure of 0-8 for cloud coverage complicates the signal and introduces more measurement error. Most images won't exactly fit into each class and classification by human eye introduces an error, the best way to reduce this is to simplify the classification system by reducing the number of classes. For example a 0-1-2 system can be used, as the aim of these algorithms is to tell whether the sky is clear, whether there is a small amount of cloud that is still suitable for telescopic observations or whether it is too cloudy to observe. The issues with the Cirrus subset can be resolved by only including thin low brightness cirrus clouds as well including more images that don't have the classification of '0'. Of course increasing the overall number of images will also improve the algorithm.

4.3. Application of Improvements

To apply some of the improvements mentioned above, the first step I took was to simplify the classification system. I tested two types of classes, type-2 in which the classifications are reduced to 0 and 1 where 0 is equal to 0 and 1 is equal to the other previous classes (1,2,3,4,5,6,7,8) and type-3 in which the classifications are reduced to 0,1 and 2 where 0 is equal to 0, 1 is equal to previous classes (1,2) and 2 is equal to the rest of the previous classes (3,4,5,6,7,8). The type-2 classification isn't practical for the aim of this algorithm as it just detects any presence of cloud whereas the aim is to be able to distinguish between no cloud where the sky is suitable for observations, small amounts of cloud where the sky is still suitable for some observations and large amounts of cloud where the sky is not observable. This system is represented in the type-3 classification so it is what I'll be using from now on.

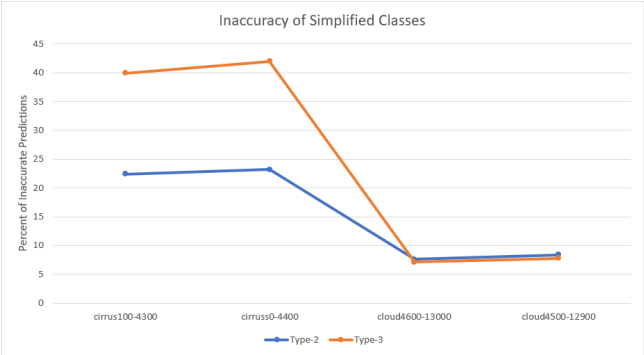


Figure 16: Graph showing the inaccuracy of the new class types (0-1) and (0-1-2).

There is a vastly lower inaccuracy for the simplest system, type-2, for the Cirrus subset. Type-2 is a system which reduces over-fit error as it boosts the signal in the signal to noise ratio which means that this vast difference indicates that the Cirrus subset suffers from large amounts of over-fit error. In comparison to the original classification system there is an improvement of about 5-6% (for type-3) for the Cirrus subset, which is a much smaller difference than type-2. This is accounted for by the fact that the (3,4,5,6,7,8) classes are very small in size in the Cirrus subset. This means that type-3 doesn't boost the signal as much, the addition of the 1,2,3,4,5,6,7,8 classes in type-2 is much bigger so it has a larger effect. As for the Cloud subset there is an improvement of 18% (for type-3) when compared to the original classification system which tells us that a significant portion of the error was due to over-fit, which is to a lesser extent than in the Cirrus subset. Type-3 performs slightly better than type-2 in the cloud subset which indicates that the impractical nature of type-2 leads to a systematic error , perhaps due to the unevenness of the class sizes. It can also be inferred that the remaining error in the Cloud subset is not due to over-fit as type-2 decreases over-fit error , more than type-3, but here it has very little effect in comparison to type-3.

The next step was to test out different depth sizes whilst keeping the type-3 system.

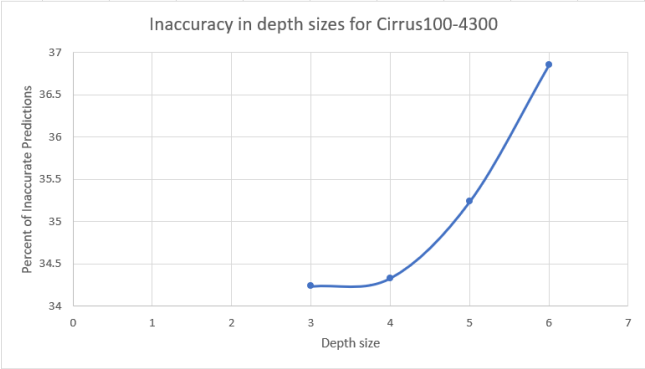


Figure 17: Graph showing the inaccuracy of different depth sizes for Cirrus100-4300.

Figure 17 clearly shows that going above the minimum depth of 3 increases inaccuracy, there is an improvement of about 8% for depth 3, going up to 66% accurate in comparison to the type-3 result. Reducing the depth size reduces over-fit so the Cirrus subset still had some over-fit error in its overall error. The Cloud subset seemed to be unaffected by this process with variations of only 0.2%, going up to 93% accurate, when comparing type-3 to depth 3, but it did present the same relation where the inaccuracy increased slightly as depth increased. This small improvement, contrasting to the large improvement in the Cirrus subset, tells me that the Cloud subset may have no more remaining over-fit error which would therefore mean that the remaining inaccuracy is due to systematic error, which mostly means David's error and so it is reasonable to say David was around 93% accurate in his classifications.

4.4. Conclusion

Using the okta system, an accuracy of 74% is achieved for general cloud coverage and an accuracy of 52-54% is achieved for cirrus cloud only coverage. The deciding factor for these accuracies were that there were more images for general cloud coverage and that it had images that were similar. The algorithms were able to tell the moon apart from cloud coverage due its immense brightness but they struggled with telling apart light pollution from cloud coverage. Using a simplified system of 0-1-2 and a depth size of 3, an accuracy of 93% is achieved for general cloud coverage and an accuracy of 66% is achieved for cirrus cloud only coverage. The general cloud coverage seemed to be unaffected by depth size and so it can be concluded that the remaining error is due to systematic error that is caused by human classification, which is done by David. In other words, I can conclude David was 93% accurate in his classification. This high accuracy achieves the aim of the project to create an algorithm that can tell whether the sky is clear, whether the sky has some cloud but is still suitable for telescopic observations and whether the sky is too cloudy for telescopic observations. The code for this process is provided in the Github url in Appendix A, and it is user-friendly and can be used to reproduce the process shown in this paper. There were some factors not included in the training of the algorithm like the different exposure times of each image, which could be normalised to a certain value for the whole data-set. This result does show that machine learning algorithms can work accurately with images of the sky, which can be used for other applications than cloud coverage.

Acknowledgements

I would like to specially thank David Campbell setting up the All Sky camera as well as classifying and providing the images used in the data-set. I also wish to thank my friend Orkan Y. Mustafa for helping me with some programming issues. Lastly I would like to thank Mohammed A. Khan for helping with the revision of this paper.

References

Baran, Ágnes et al. (2020). *Machine learning for total cloud cover prediction*. arXiv: 2001.05948 [stat.ML].

Campbell, D. (2010). *Widefield Imaging at Bayfordbury Observatory*. URL: http://www-cs-faculty.stanford.edu/~uno/abcde.html.

Deng, H. (2018). *Why random forests outperform decision trees*. URL: https://towardsdatascience.com/why-random-forests-outperform-decision-trees-1b0f175a0b5.

Fong, Chamberlain (2015). *Analytical Methods for Squaring the Disc*. arXiv: 1509.06344 [math.HO].

Huertas-Tato, J. et al. (2017). "Automatic Cloud-Type Classification Based On the Combined Use of a Sky Camera and a Ceilometer". In: *Journal of Geophysical Research: Atmospheres* 122.20, pp. 11, 045–11, 061.
McKinney, Wes et al. (2010). "Data structures for statistical computing in python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX, pp. 51–56.
Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python ". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.
Verkhovskiy, B. (2019). *Squircle*. URL: `https://pypi.org/project/squircle/`.

Appendix A. Link to the Github where my code is stored

`https://github.com/AkinGH/Cloud-Classification-Project`

Appendix B. Squircle mapping comparison

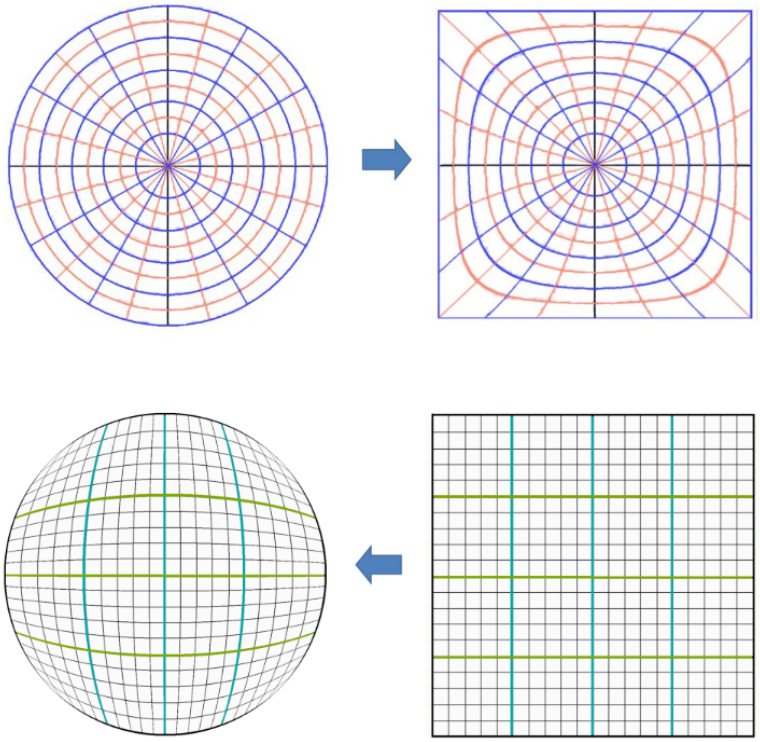


Figure B.18: Grid maps showing the effects of Elliptical squircle mapping.

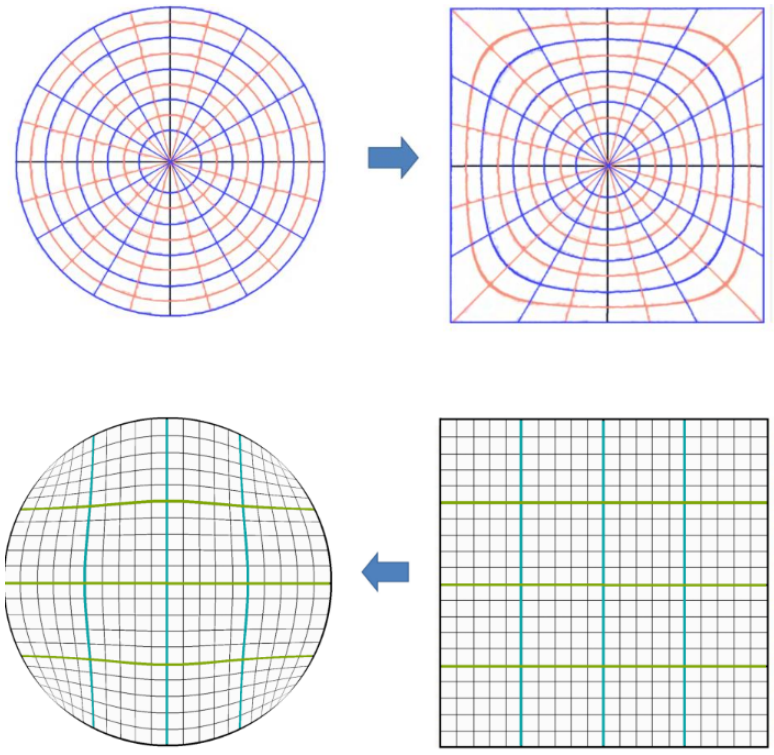


Figure B.19: Grid maps showing the effects of FG squircle mapping.