Project 2 - https://github.com/nickglai/stat154-project2

Andrew Rall, Nicholas Lai

April 24, 2019

1. Data Collection and Exploration

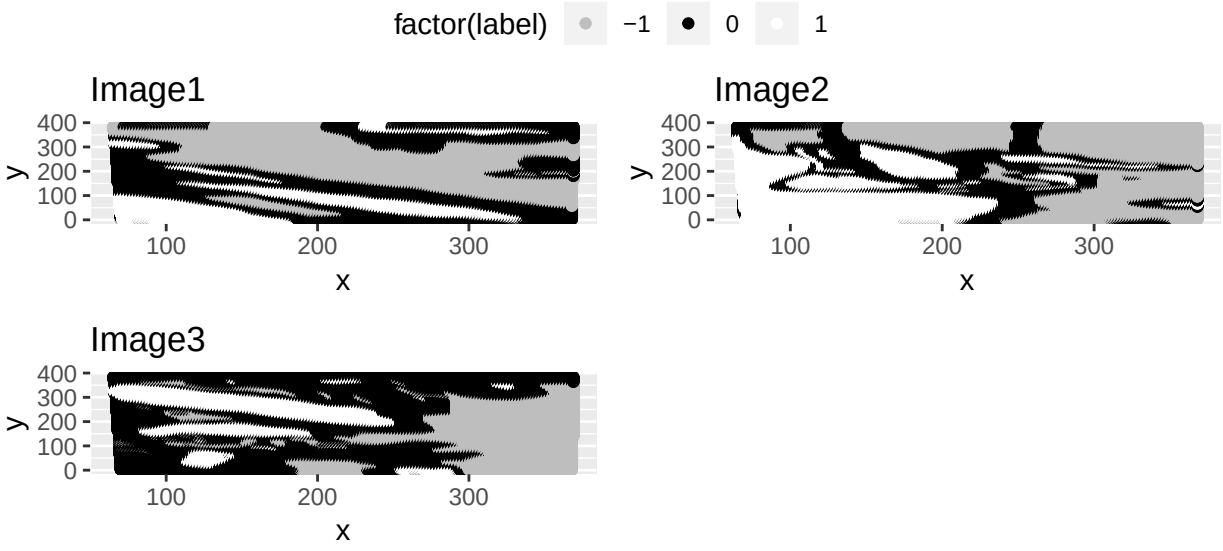
1a.

The aim of this study was to create a sufficiently accurate classification algorithm to distinguish between clouds and arctic ice in image data from the Terra satellite's sophisticated sensor array without the need for constant expert labeling. The image data was collected by the MISR onboard Terra, which through observations of regions of interest from multiple angles is able to generate highly detailed and accurate reflected sunlight readings from the earth's surface. The data was collected in data units containing 7,114,248 1.1 kilometer pixels, and the data considered contains 57 such units. Each pixel of the images contains additional information about relative information about location, relevant engineered features, and the radiance angles of the MISR cameras while the image was taken, and was given true labels in post by expert opinion in order to perform validation. The study concluded that three engineered features were sufficient to distinguish between arctic ice and clouds in image data more accurately than the previous standard method using even very simple classifiers, simplifying the data processing workflow for the MISR output. data This study demonstrates the potential that statistical solutions have for complex scientific problems, and also provides more accurate cloud detection data that can be used by scientists tracking hurricanes, climate patterns, and anything else that depends on the pattern of clouds over the earth.

1b.

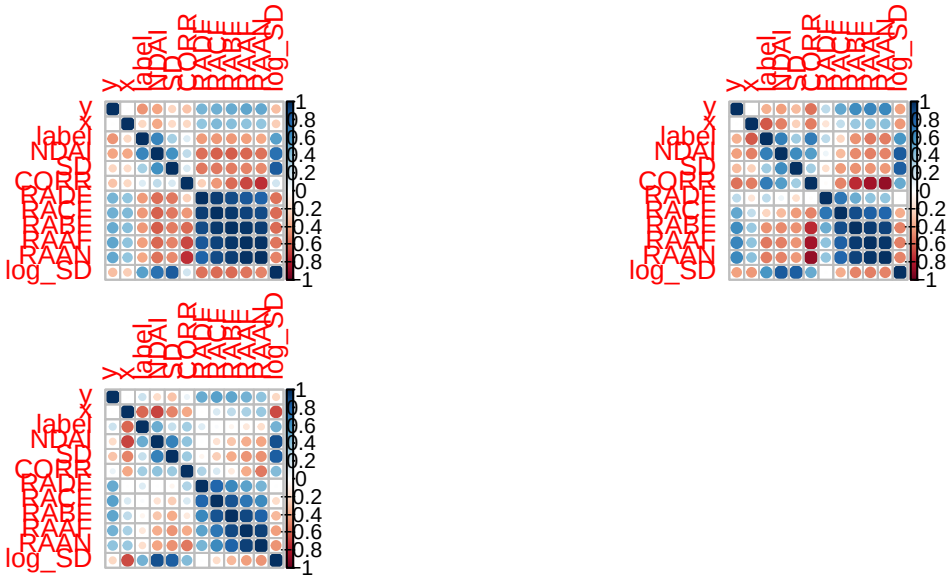
Proportion of pixels for each class all images

```
##
##        -1         0         1
## 0.3677552 0.3978950 0.2343499
```



For each of the images, each of the pixel classes tend to clump together. Intuitively, a cloud is going to occupy mupliple pixels, so if a pixel is known to be a cloud pixel, then all adjacent pixels are more likely to be cloud pixels as well. This means that an i.i.d assumption is not justified for this dataset.

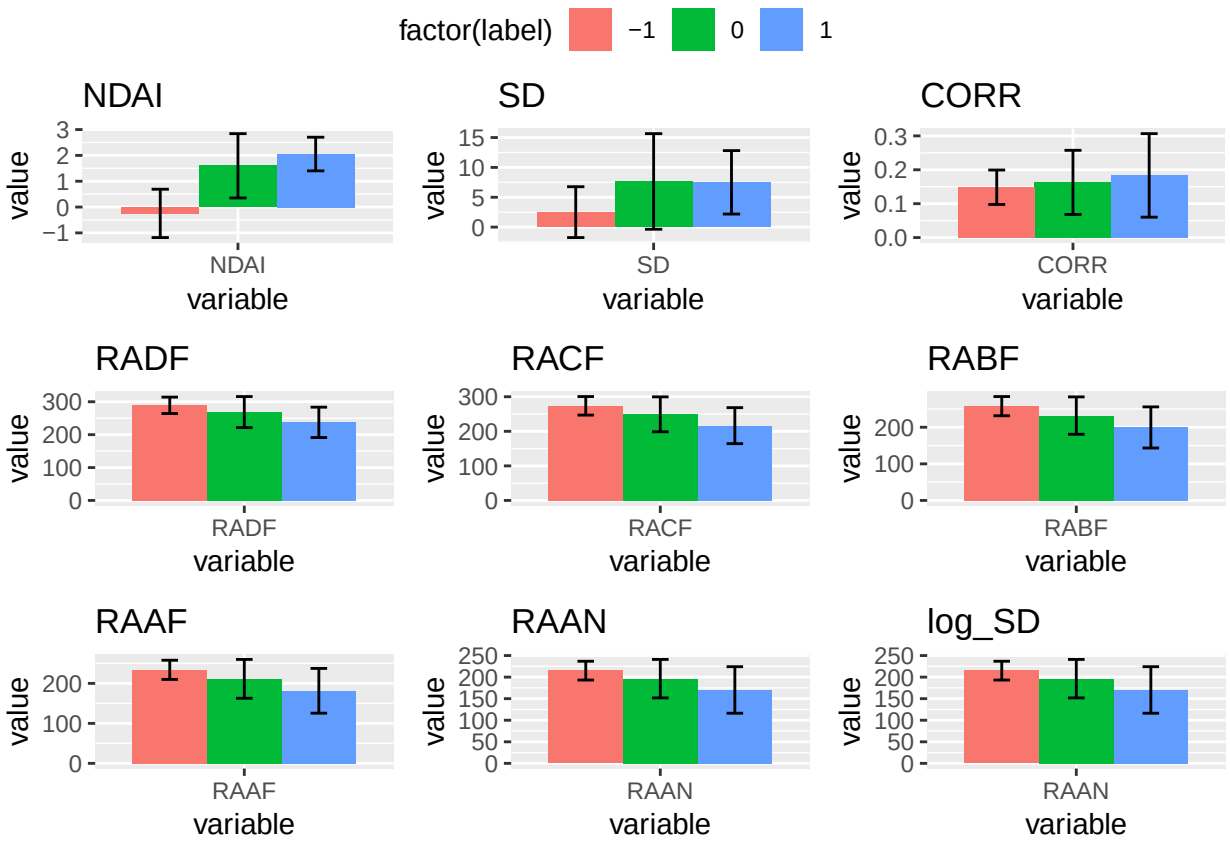
1c.




```
## # A tibble: 3 x 12
##   label     y     x   NDAI    SD  CORR  RADF  RACF  RABF  RAAF  RAAN log_SD
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1    -1  241.  234. -0.245  2.51 0.149  289.  274.  258.  234.  215.  0.509
## 2     0  179.  217.  1.60   7.65 0.163  269.  249.  232.  211.  196.  1.61
## 3     1  107.  182.  2.05   7.50 0.183  237.  216.  199.  181.  170.  1.80
```

```
## # A tibble: 3 x 12
##   label     y     x   NDAI    SD  CORR  RADF  RACF  RABF  RAAF  RAAN log_SD
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1    -1  233.  280. -0.348  3.14 0.149  280.  267.  248.  224.  209.  0.625
## 2     0  206.  216.  1.97  12.7  0.207  307.  273.  243.  216.  201.  2.17
## 3     1  139.  152.  2.00  10.6  0.338  303.  248.  202.  169.  156.  2.15
```

```
## # A tibble: 3 x 12
##   label     y     x   NDAI    SD  CORR  RADF  RACF  RABF  RAAF  RAAN log_SD
##   <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1    -1  162.  297. -0.180  3.48 0.116  234.  219.  205.  192.  184.  0.334
## 2     0  198.  195.  1.90  14.2  0.185  253.  227.  206.  186.  175.  2.09
## 3     1  229.  158.  1.76  10.7  0.201  249.  219.  198.  180.  169.  2.00
```



NDAI is significantlly smaller for not clouds. SD is reasonably smaller for not clouds. CORR is slightly smaller for not clouds. The radiances appear to be slightly larger for not clouds.

---

2. Preparation

2a.

To split the data into train, validation, and test sets we will construct boundaries (essentially forming a checkerboard) for each of the three images. We will then randomly assign squares of our checkerboard such that one section has 60% of the data as our train set, 20% of the data as our validation set, and the remaining 20% will be our test set. The thinking behind this strategy is that the points are not i.i.d. so randomly assigning a point to the train, val, or test set would cause a problem. If it is known that a point is a cloud, then all of the surrounding points are more likely to be cloud points as well (this also holds for the other two classes). Because of this phenomenon, splitting the images such that most points retain their spatial neighbors helps address the issue of spatial dependence. An alternative method to construct boundaries would be to simply introduce two lines to each image, segmenting it into three parts with a 60/20/20 ratio. It should be noted that the left border of each image is not a vertical line so some of the squares will not actually be squares. I will now construct a 5x5 checkerboard for each image.

Split Method 1: Checkerboard

Construct 25 Image1 squares:

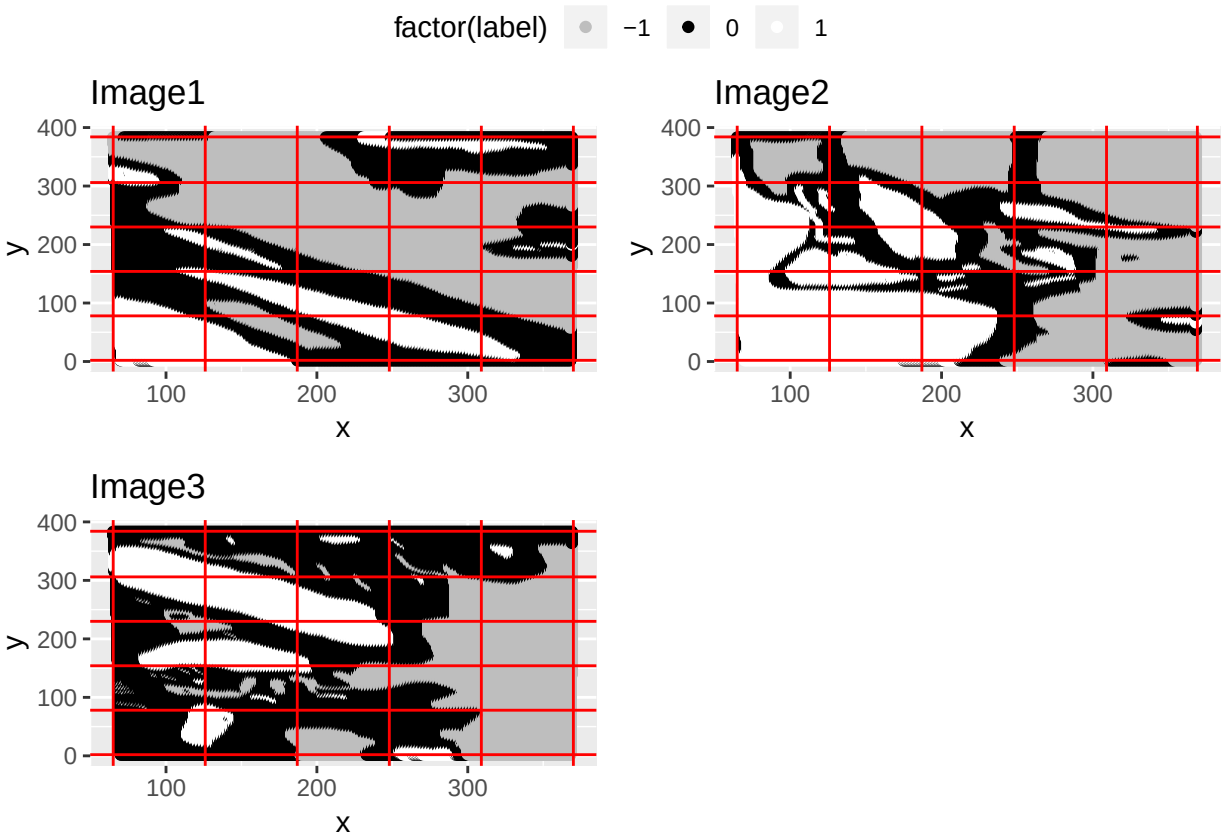
Construct 25 Image2 squares:

Construct 25 Image3 squares:

Put all squares into list:

Randomly select 45 for Train, 15 for Valid, 15 for test:

Partition of Three Images:



Split Method 2: Two Lines of Separation

We will partition the image with two lines, splitting it into 60/20/20 rectangles. The logic for this is on previously discussed lines.

2b.

Trivial Classifier that classifies all points as -1 (not cloud).

Valid Accuracy:

```
## [1] 0.408727
```

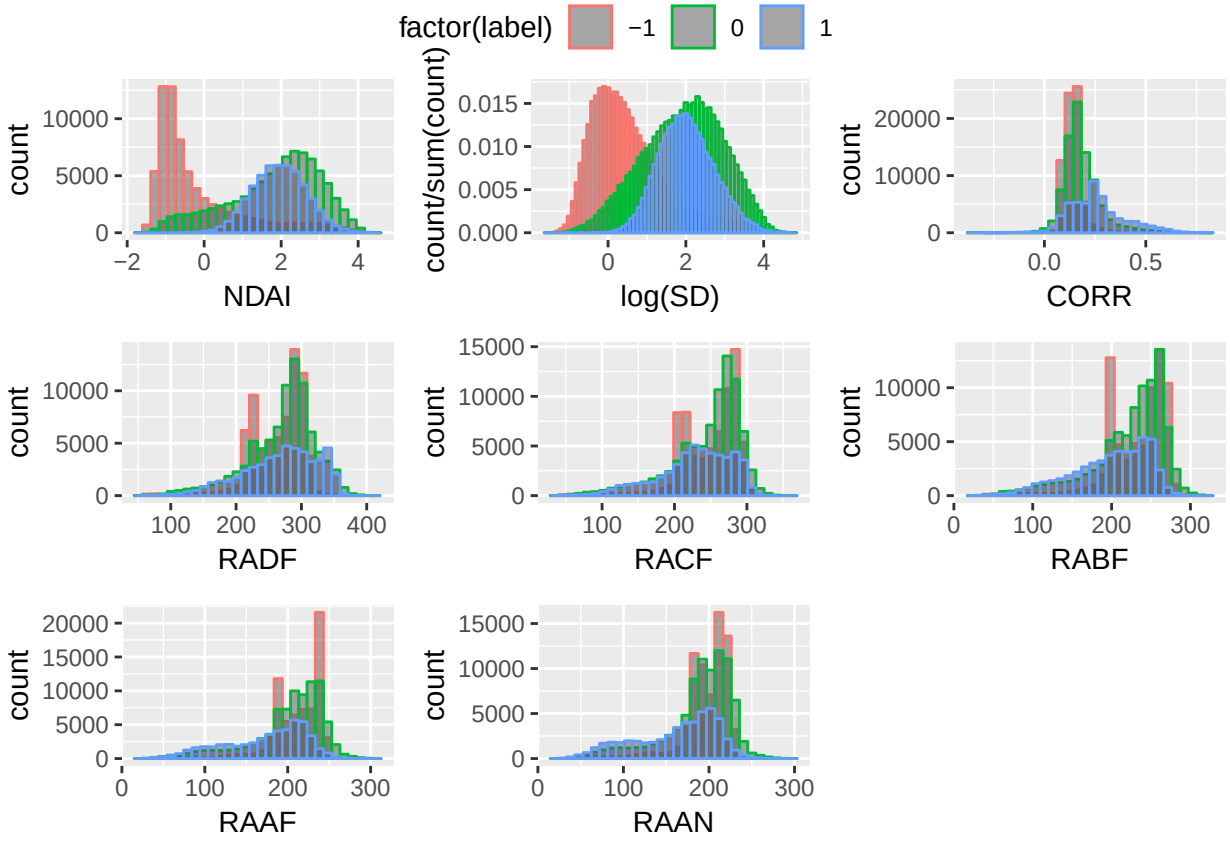
Test Accuracy:

```
## [1] 0.3930983
```

The above classifier would have a very high accuracy if a very high proportion of the points were -1 (not cloud), but that was not the case with this data.

2c.

NDAI looks like fairly separate distributions in our train set.



NDAI and log(SD) are the only two features that appear to have very different distributions. I will use the correlation matrix to determine the third feature to use.

```
## [1] "NDAI"   "log_SD" "CORR"
```

Our 3 best features are NDAI, log_SD, and CORR.

2d.

For the purposes of this problem, the 0 class (unlabeled) is not relevant so I will drop all entries of this class and then encode the -1 class (not cloud) as 0 to make the next steps easier. Thus, **0 is our class for not cloud and 1 is our class for cloud**. The `CVGeneric` function is in the code portion of our repository.

---

3. Modeling

3a.

Model 1: Logistic Regression

The assumptions of logistic regression are as follows: Independent Observations, No Multicolinearity of features, Large Sample Size,Linearity of features and log-odds

Our observations are not independent. Knowledge of relative position will give information about the pixels within a vicinity of each other, as clouds tend to be continuous in space. Our features are not multicolinear. The paper describes how the three features were motivated and calculated, and they do not remotely encode the same information. An examination of the above plots of the distributions of the variables confirm this. The sample is the number of pixels in the training subset, which are of a considerable number (n=122124). The linearity of features and log-odds is hard to verify, but we can do so indirectly by seeing the performance of the model.

```
## [1] "Accuracy for each of K Folds:"
```

```
## [1] 0.7146625 0.9256008 0.8909660 0.8313033 0.6984351
```

We can see from the reported confusion matrix that the model has a 91% accuracy in prediction, 84% precision, and 95% recall on the test data. This standard of performance is very high, far better than the benchmark trivial classifier, and is a promising sign that the paper's conclusion that the three engineered features are sufficient for the classification problem.

Model 2: LDA

The assumptions of LDA are as follows:

- Normality of features (by group), No multicolinearity, Independent observations, Homoskedasticity (equal covariances), large sample size

Of the assumptions not discussed previously: The features, as seen on the data exploration plots, are not heavily skewed, even when broken into seperate labels. We transformed the `sd` variables with the logarithm in order to meet this assumption. Homoskedasticity is reasonable but not quite true from the same plots, but LDA should not be overly sensitive to slight violations of this assumption.

```
## [1] "Accuracy for each of K Folds:"
```

```
## [1] 0.7667774 0.9239319 0.9010559 0.8243019 0.7172687
```

We can see from the reported confusion matrix that the model has a 92% accuracy in prediction, 88% precision, and 94.5% recall on the test data. This standard of performance is very high, far better than the benchmark trivial classifier, and is a promising sign that the paper's conclusion that the three engineered features are sufficient for the classification problem.

Model 3: QDA

The only difference in LDA and QDA assumption is on the variances, where QDA has a slightly relaxed assumption on the covariances. The discussion of the validity of these assumptions is in the LDA section.

```
## [1] "Accuracy for each of K Folds:"
```

```
## [1] 0.7935430 0.9357143 0.9012906 0.8273676 0.6951142
```

We can see from the reported confusion matrix that the model has a 93.5% accuracy in prediction, 91% precision, and 95% recall on the test data. This standard of performance is very high, far better than the benchmark trivial classifier, and is a promising sign that the paper's conclusion that the three engineered features are sufficient for the classification problem.

Model 4: XGBoost

Here we are using a boosting algorithm with a logistic loss function. This means that the assumptions of this model are shared with logistic regression, with the added caveat that the initial weak learner not be overfit, to produce meaningful improvement over iterations. This is taken care of in the XGBoost algorithm.

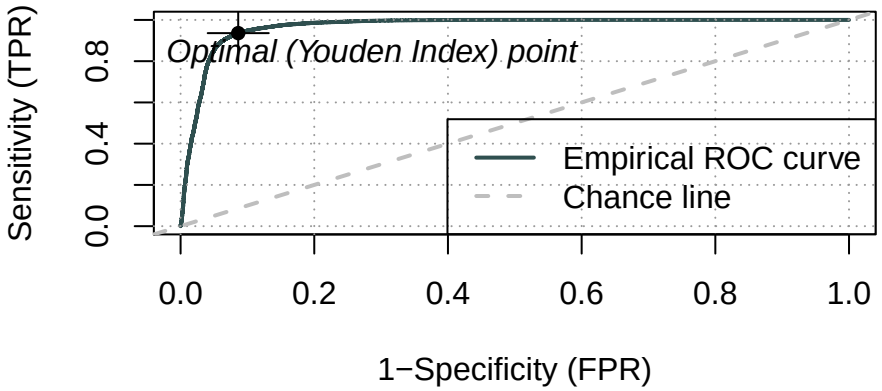
```
## [1]  train-error:0.098770
## [2]  train-error:0.097019
## [1]  train-error:0.110173
## [2]  train-error:0.109826
## [1]  train-error:0.108602
## [2]  train-error:0.105961
## [1]  train-error:0.089890
## [2]  train-error:0.086114
## [1]  train-error:0.085618
## [2]  train-error:0.081974
## [1] "Accuracy for each of K Folds:"

## [1] 0.8751587 0.9167223 0.9205319 0.7933549 0.7668547
```

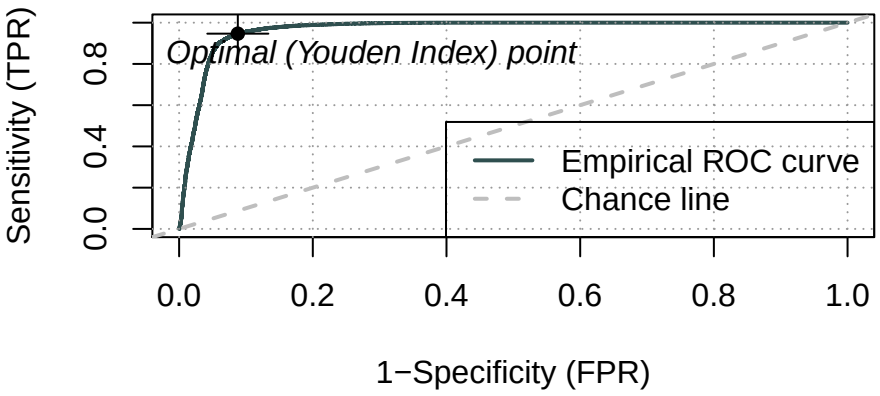
We can see from the reported confusion matrix that the model has a 93% accuracy in prediction, 92% precision, and 96.5% recall on the test data. This standard of performance is very high, far better than the benchmark trivial classifier, and is a promising sign that the paper's conclusion that the three engineered features are sufficient for the classification problem.

3b.

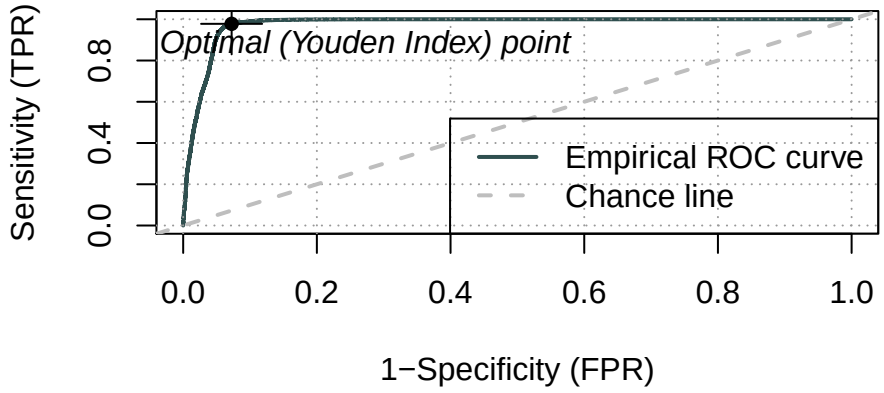
The cutoff values in the following plots were chosen in accordance with Youden's J statistic. This statistic is calculated as $precision + recall - 1$, and it is plotted at the point where the distance from the ROC curve to the chance line is equal to J statistic.



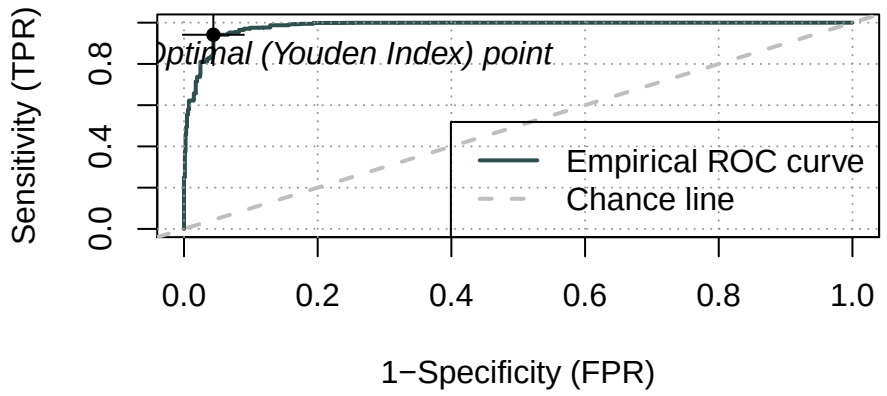
Logistic


Sensitivity (TPR)
0.0
0.4
0.8
Optimal (Youden Index) point
Empirical ROC curve
Chance line
0.0
0.2
0.4
0.6
0.8
1.0
1−Specificity (FPR)

LDA


Sensitivity (TPR)
0.0
0.4
0.8
Optimal (Youden Index) point
Empirical ROC curve
Chance line
0.0
0.2
0.4
0.6
0.8
1.0
1−Specificity (FPR)

QDA



XGBoost

All the ROC curves show that the classifiers perform well, with XGBoost claiming a small edge in performance.

3c.

See output of confusion matrix on test sets above.

4. Diagnostics

a.

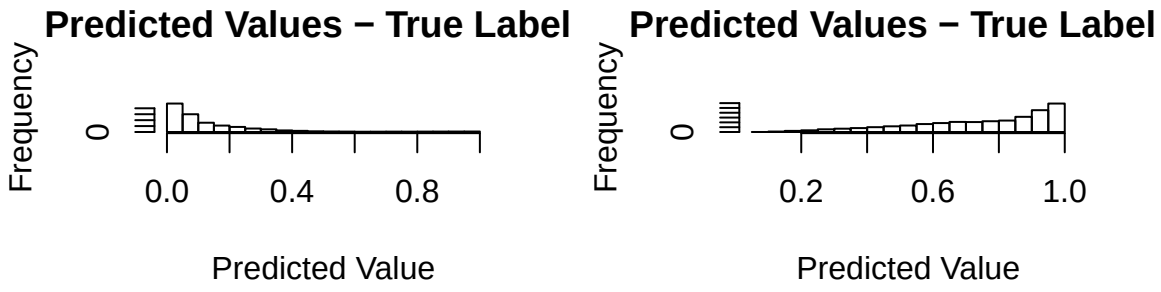
We are going to examine the performance of the logistic regression model in-depth. Many diagnostics are contained within the summary output of our logistic regression model:

```
##
## Call:
## glm(formula = label ~ NDAI + log_SD + CORR, family = binomial(link = "logit"),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6309  -0.3928  -0.2493   0.5311   2.4525
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.01743    0.02219 -135.98   <2e-16 ***
## NDAI         1.22323    0.01179  103.75   <2e-16 ***
## log_SD       0.14403    0.01413   10.19   <2e-16 ***
## CORR         7.07784    0.10535   67.19   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 165491  on 122123  degrees of freedom
## Residual deviance:  85352  on 122120  degrees of freedom
## AIC: 85360
##
## Number of Fisher Scoring iterations: 5
```

Here, we see that the null deviance (the deviance of the model with intercept only) is nearly twice as high as the residual deviance, the analog of RSS in logistic regression. This means that the model is giving us a lot of meaningful information about the class of the data. The Fischer scoring algorithm is a slight modification of Newton's method for calculating the MLE estimates of the logistic regression coefficients, with hessian of the log likelyhood function being replaced by its expectation. Since in the case of logistic regression it never depended on `y` in the first place, the two methods are the same. The algorithm converged quickly, in five iterations. The estimated parameters of our logistic regression model are all very significant, with the probability that the true value of each being zero being less than 0.001 percent under model assumptions by a z-test against that null hypothesis. This is strong evidence that all parameters in our model are important in the classification problem.

Examning plots of true labels versus predicted probability:



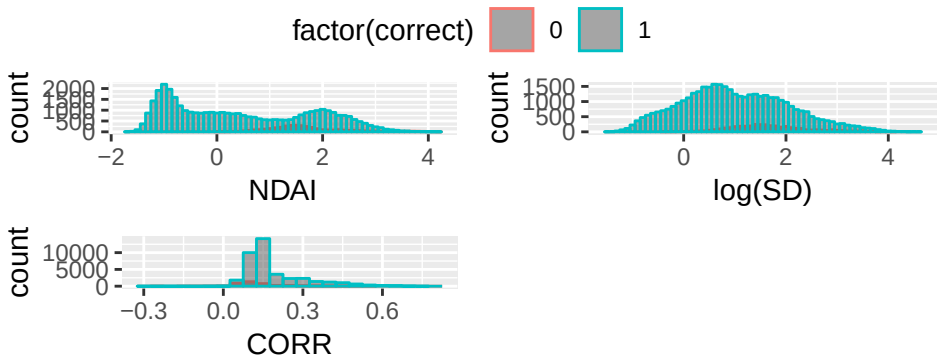
We can see from the above plots that the logistic classifier is giving distributionally meaningfully distinct predicted values by the true class labels, so the classifier is performing well (in agreement with the ROC curve).

4b.

The models tested in Question 3 have about the same predictive power, so for continuity we will examine trends in the misclassification of logistic regression.

```
## [1] 3783
```

In order to see if there are trends in the misclassified data, we plot histograms overlaying the distribution of variables stratified by correct/incorrectly classified. We begin by examining the three variables involved in fitting the model.



Graphically, there is at most weak evidence of a trend in misclassification in terms of these three variables. The distributions seem consistent between correctly and incorrectly classified variables. We can perform a t-test on CORR values (The only one that has a bit of non-overlap) to confirm this quantitatively:

```
##
##  Welch Two Sample t-test
##
## data:  logistic_test$CORR[logistic_test$correct == 1] and logistic_test$CORR[logistic_test$correct ==
## t = 46.341, df = 5261.6, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.06619900 0.07204737
## sample estimates:
## mean of x mean of y
## 0.1885550 0.1194318
```

The t-test shows that there is a difference of means, but on the scale of the data this difference is small.

Now we will check if there is misclassification bias in the variables not considered in the regression model. We begin as we did before with similarly constructed plots.

There does not appear to be a trend in misclassification in the location of the pixels on the images when splitting the data by rectangles. The distributions are overlaid on top of one another.

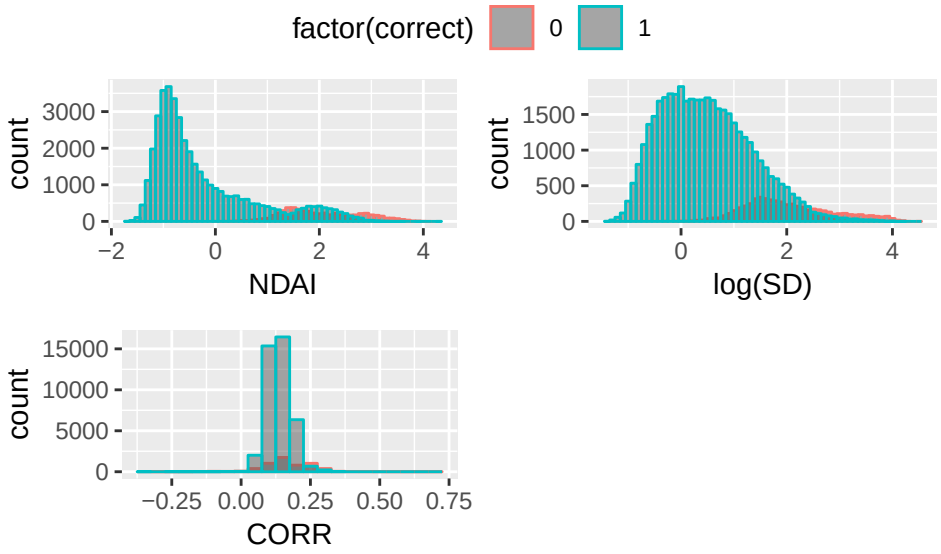
4c.

Based on the above examination's conclusions, there is no evidence to believe that the model will perform poorly on future unlabeled data. There are no notable trends in misclassification error, and the predictive power as discussed in question 3 is high (80-90%+ in both precision and recall, and overall accuracy).

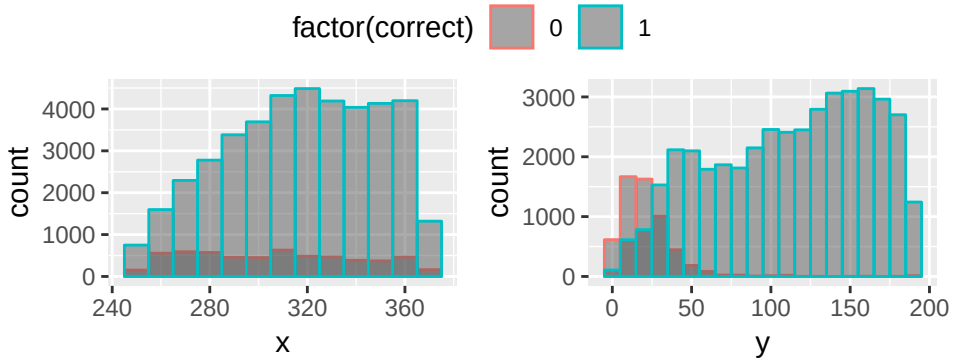
Because of these facts, there is no obvious better classifier.

4d.

We will now evaluate the performance of the model trained on the alternate 60/20/20 split of the data. The Fischer Scoring iterations take one more step to converge, which is not a significant difference. The parameters all remain significant, however.



Higher values of SD and NDAI are associated with misclassification error.



This splitting of the data seems to have a big trend in misclassification error at low values of y. This makes intuitive sense, because the split is less geometrically random than the previous method of splitting.

4e. Conclusions

The biggest takeaway from our analysis has been that, as per the paper's conclusions, the three features engineered, `SD`, `CORR`, and `NDAI`, are sufficient to accurately classify the vast majority of pixels. No matter the method used, the classifiers trained on these three features have above 80% in both precision and recall, as well as accuracy. However, as the above analysis shows, some care must be taken on how the data is split to dodge systematic error. A naive split of the data with two lines produced a trend in miscalssification error towards low y-values, high `SD`, and high `NDAI`. The more robust split into sectors has very little trend in misclassification error. Time to convergence is fast in any case for logistic regression, and as the parameters are all significant to the model.