Classifying cargo ships from satellite data using Machine Learning

1st Shaun Heffernan
Student
University of Florida
Gainesville, Florida
shaunheffernan@ufl.edu

***Abstract*—This document shows a analysis satellite imagery from bays in California. The goal is to train classifiers to distinguish if images have ships in them. It includes exploratory data analysis and model creation and performance evaluation.**

I. Introduction

Our data set includes satellite images of ships through San Francisco Bay and San Pedro Bay areas of California. There are 4000 80 x 80 pixel images that are labeled as ship or no_ship. The images were taken from PlanetScope full-frame visual scene products, which are orthorectified to a 3-meter pixel size.

Our goal is to find the best classification model to distinguish whether new satellite pictures are a ship or not a ship. This is important for many users such as shipyard managers who need to keep track of the large amount of ships moving throughout their port. We also want to find the best ways to reduce the dimensionality of the dataset and still have accurate predictions. Reducing the dimensionality would allow us to work with much smaller vectors instead of using a vector of size 80 * 80 * 3, which is the dimension of the images multiplied by the RGB values for each pixel.

The classification models we will use are random forest and logistic regression. We will also implement PCA for dimension reduction and also use Locally linear embedding (LLE) and ISOMAP for manifold learning. To compare the performance between the models with no dimensionality reduction, PCA, and manifold learning, we will look at the training time, inference time, F1 score, and accuracy scores for the validation set.

II. Ship dataset

To make our data usable for our classification models, we first had to flatten every image from an 80 * 80 * 3 array to a single vector of length 19200. Then a standard scaler was used for each image to scale each image for training and predicting. There is also a class imbalance in the data set with 3000 pictures of no ship and 1000 images of a ship. When we create our training and testing split using 80% of the data for training, we make sure to stratify by the labels so the training and test set are balanced.

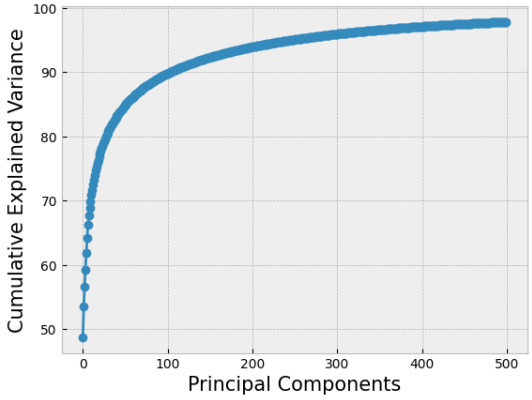

Fig. 1. Variance vs number of principal components

III. Experiments and results

A. No dim reduction

Our first experiment was training the random forest and logistic regression with no dimensionality reduction. For logistic regression we ran a grid search which took 126 seconds and found the best parameter was a C value of 0.001. It's accuracy score was 0.92 and it's F1 score was 0.84. For random forest the grid search took 902 seconds which is much slower then the logistic regression. It found the best parameters were a max depth of 20 and 200 estimators. It's accuracy score was 0.95 and it's F1 score was 0.90. For the models with no dimensionality reduction, random forest performed the best, but it had a much slower training time. This could be caused by the random forest having a larger parameter space to grid search over.

B. PCA 90% variance, RMSE curve, reconstructed visuals

In this graph we plotted the variance vs the amount of principal components in the PCA algorithm. We wanted to find the smallest amount of components we need that can still explain 90% of the datasets variance. Using this graph we found that to get at least 90% of the variance you need 103 components which get the variance of 0.90. This means we can shrink our dimension from 19200 to only 103 while keeping 90% of the variance.

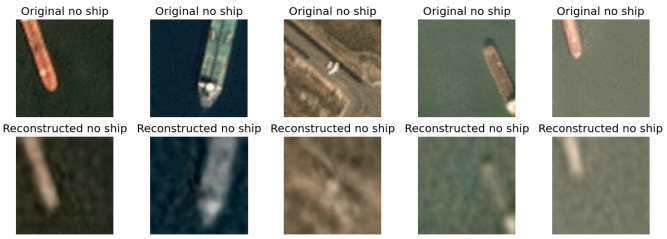

Fig. 2. Original vs Reconstructed images with 103 PCA components

To show what the images would look like if they were reconstructed with only 103 components, we use inverse transform on the vectors that are dimensionally reduced by PCA with 103 components. You can see that with only 103 components, the images still look very similar to their original image which is a 19200 dimension vector. A major size reduction only caused a small drop in the quality of the images because we selected a specific amount of PCA components because we wanted 90% variance.

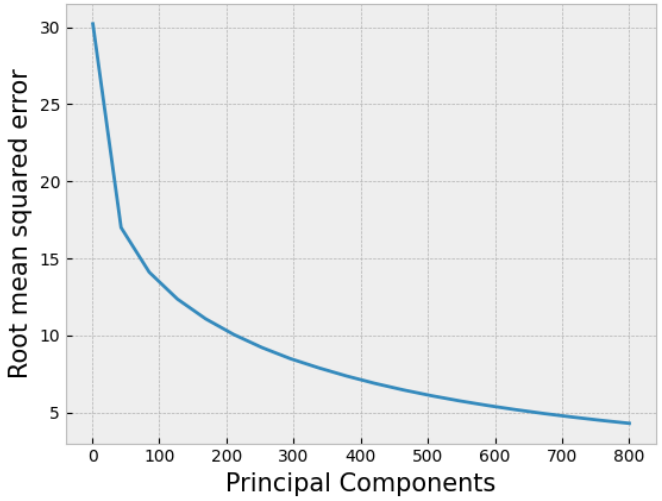

Fig. 3. RMSE vs PCA number of components

In this figure we can see how the amount of principal components effects the root mean squared error of the reconstructed image and the original. It shows how more components makes the error shrink, but the slope begins to shrink and you can see that the error difference between 600 components and 800 components is almost negligible. This further shows the tradeoff between the size of the image vs the quality of the image.

The logistic regression with PCA took 913 seconds compared to the random forest with PCA taking 2568. The random forest did have better F1 and accuracy scores with 0.89 and 0.95 respectively. The logistic regression was much quicker but only had F1 of 0.81 and accuracy of 0.91, so there is a tradeoff between time and model performance. What is interesting is that the PCA n_components were vastly different for each of the best models. The random forest used only 20 components where logistic regression used 200 components.

For logistic regression, it performed worse with PCA and had an F1 score of 0.83 and accuracy of 0.92. This is interesting because if it performed better without PCA, why was the max n_components not used in the best model from the grid search CV. It was also much faster without PCA, which is surprising because it is faster and has better performance. For random forest, the non PCA had F1 of 0.90 and accuracy of 0.95 which is very similar to the model with PCA. And without PCA it was much faster taking only 903 seconds to complete the entire grid search training. Using PCA had much longer training times, but the grid params used where much larger because we were searching through the optimal number of n_components for PCA. So the longer time wouldn't have been an issue if we already knew what n_components we wanted to use and it would most likely be faster then training a model using all of the components. The PCA models had slightly worse performance then without PCA, but they are much smaller dimensional models which are better for most use cases and likely have faster inference time.

C. Manifold learning, plot components

For the manifold learning section we used two algorithms for dimensionality reduction. We used Locally linear embedding (LLE) and ISOMAP for both of our models.

For logistic regression with LLE, the best parameterss were a C of 10, 10 LLE components, and 20 LLE neighbors. It took 1437 to train the grid search and the best model had an accuracy of 0.82 and a F1 score of 0.48.

For logistic regression with ISOMAP, the best parameters were a C of 0.0001, 10 ISOMAP components, and 5 ISOMAP estimators. It took 1543 seconds to train the grid search and the best model had an accuracy of 0.89 and a F1 score of 0.74.

For random forest with LLE, the best parameters were a max depth of 20, 300 estimators, 10 LLE components, and 20 neighbors. It took 2822 seconds to train the grid search and the best model had an accuracy of 0.91 and a F1 score of 0.82.

For random forest with ISOMAP, the best parameters were a max depth of 20, 100 estimators, 10 ISOMAP components, and 20 neighbors. It took 3045 seconds to train the grid search and the best model had an accuracy of 0.93 and a F1 score of 0.85.

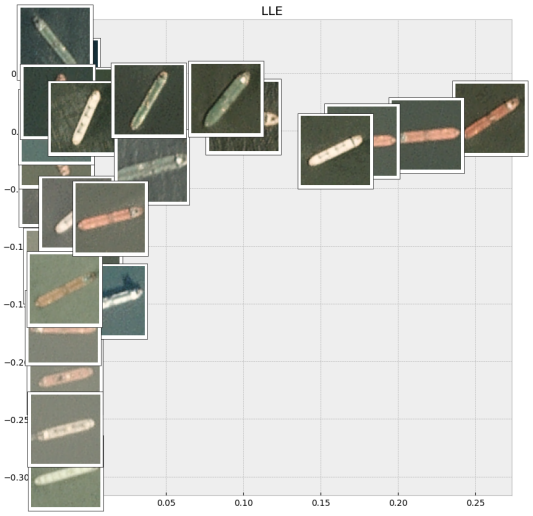

Fig. 4. LLE first 2 dimensions plot components

This graph plots the first 2 dimensions from the manifold learning algorithm LLE. In the top left of the graph the ships are vertical and have much darker colors and backgrounds. On the bottom left of the graph the ships and their background are much more washed out and similar in color while the top right has very contrasting and bright colors between the ship and the water. So the Y axis might be quality where the ships higher up on the graph have better quality and the ones lower down are much worse in quality. The X axis could be contrast because the ships that are farther right on the x axis contrast much better with their background and are easier to see.

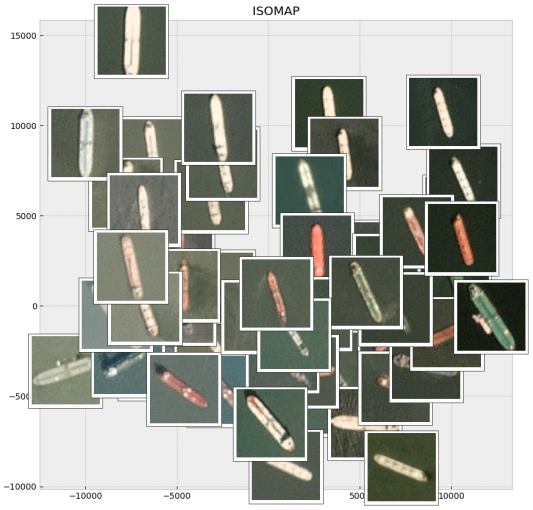

Fig. 5. ISOMAP first 2 dimensions plot components

You can see a pattern that as its going negative X and positive Y the ships are going from pointing left to pointing vertical. It looks as if moving positive in the Y axis represents the ship pointing up because the ones at the bottom are facing left or right and the ones at the top are vertical. The X axis represents color where it seems the ships on the left are more dull and blend in with their background and the ships in the positive X axis are much more vibrant and contrast with their background

IV. Best pipeline and error analysis

A. Model comparison

Below is every models accuracy score, F1 score, and inference time on the test set (20% of original data set). The inference scores slightly vary each time the model makes predictions because the CPU speeds vary.

TABLE I
Model Comparison on Test Set

Model	Accuracy	F1 Score	Inference Time (s)
Logistic Regression (No DR)	0.921	0.836	0.055
Random Forest (No DR)	0.954	0.901	0.093
Logistic Regression + PCA	0.910	0.818	0.209
Random Forest + PCA	0.953	0.898	0.100
Logistic Regression + LLE	0.821	0.487	3.606
Random Forest + LLE	0.911	0.817	3.661
Logistic Regression + ISO	0.885	0.744	2.618
Random Forest + ISO	0.930	0.851	2.729

Random forest without dimensionality reduction had the best performance of all the models and had the second fasted inference time of 0.098 seconds, so I will choose that model as the best and build the confusion matrix.

B. Confusion matrix for best model

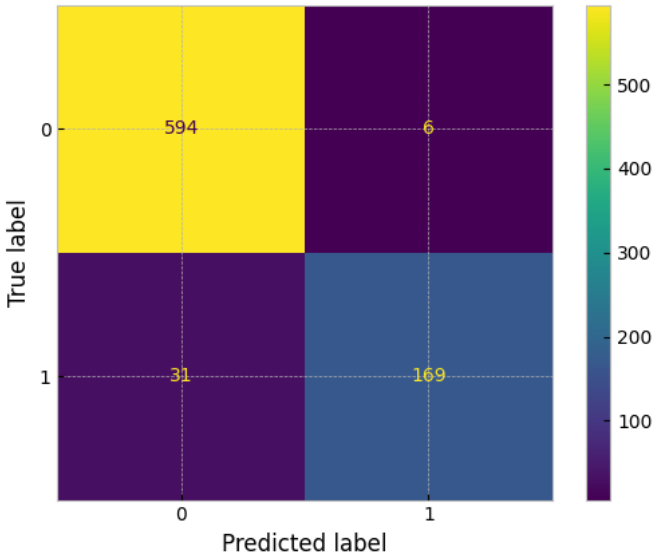

Fig. 6. Confusion matrix for random forest without dimensionality reduction

The model predicted ship when there was actually no ship 6 times and it predicted no ship when there was a ship 31 times. This means the model is struggling to classify the ship images much more then classifying the no ship images. This could be due to some no ship images containing parts of ships which makes it harder to distinguish, also there is more no_ship images in the data set which also explains the

misclassifications imbalance. Going forward to address miss classifying ship as no ship, we can change the scoring metric to recall to punish false negatives more. This would create more false alarms (classifying no_ship as a ship) but it could be more beneficial because a shipyard likely doesn't want false negatives because that would mean some ships could be untracked.

C. Visualization of the misclassifications

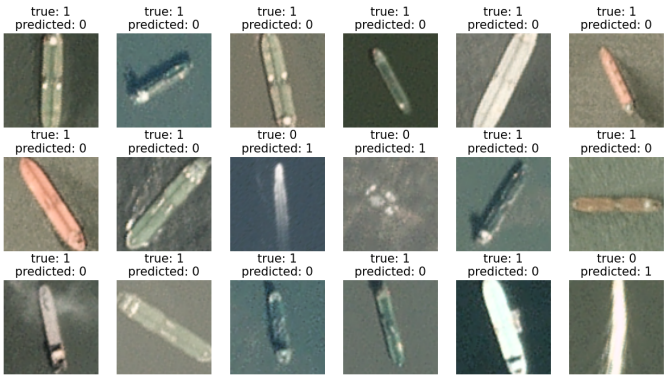

Fig. 7. Misclassified samples

A lot of the false negatives are because the ship is either similar color to the water or its partially cut off and has weird orientation. For the false positives, there are objects or waves in the water which resemble the ships closely. Moving forward using recall as the scoring metric would make sure there are less false negatives which make up a majority of the misclassifications. Also using something like color grading could help the ships become better identifiable compared to the water.

V. Conclusion

When finding out which model is the best deployment it is important to first analyze the use case. The random forest model without dimensionality reduction was the highest performing model, but it also requires full size pictures to make predictions and further train the model. If you have a large system like a shipping yard you might want to deploy that system because you want to optimize entirely for performance. But if the model is deployed on a edge device or a satellite, you might not be able to transmit 19200 dimension images for classification, so you would need a model with dimensionality reduction at the cost of some of the models performance. In the case you care about the size of the images, random forest with PCA would be a very good model to choose. It has very similar performance to random forest without dimensionality reduction and only needs 20 PCA components which is orders of magnitude smaller then the full images size.

For the manifold learning approaches, random forest with ISOMAP performed the best and it's performance wasn't much far behind random forest with PCA. But the inference time is 2.7 seconds which is much longer then all of the other non-manifold learning algorithms. This is because it is a lot of computation to create the non-linear dimensionality reduction. The manifold learning algorithms where useful for visualization, but there performance was not the best and the inference time was much longer then other models trained.

The limitations of this model are that it struggles a lot with is false negatives. This can be bad if the use case is for a shipyard tracking system, you would not want to lose track of ships in your yard because that could lead to crashes. One of the causes for this could be the imbalanced dataset, only 1000 of the 4000 images where a ship. A way to address this issue is to train the model using recall as the scoring metric, which reduces false negatives, or collect more labeled training data that have images of ships.