Ship Detection in Satellite Imagery Using Machine Learning Techniques

Aryan Atre
line 2: University of Florida
line 3: *Artificial Intelligence Systems*
line 4: Gainesville, Usa
line 5: atrearyan@ufl.edu

***Abstract*— This paper presents a comprehensive analysis of ship detection in maritime surveillance imagery using various machine learning approaches combined with dimensionality reduction techniques. We compare the performance of Logistic Regression and Random Forest classifiers with different dimensionality reduction methods including Principal Component Analysis (PCA) and ISOMAP. Our experimental results demonstrate that dimensionality reduction can significantly improve both computational efficiency and classification accuracy. The best-performing model achieved high accuracy while maintaining reasonable inference times, making it suitable for real-world maritime surveillance applications.**

***Keywords*— Ship Detection, Satellite Imagery, Machine Learning, Random Forest, Logistic Regression, Principal Component Analysis (PCA), ISOMAP, Manifold Learning, Dimensionality Reduction, Computer Vision, Maritime Surveillance, Classification, Feature Extraction, Image Processing, Performance Analysis**

Introduction

Maritime surveillance plays a crucial role in national security, trade route monitoring, and environmental protection. The automatic detection of ships in satellite or aerial imagery has become increasingly important due to the growing volume of surveillance data. However, processing high-dimensional image data presents significant computational challenges. This study investigates the effectiveness of different machine learning approaches combined with dimensionality reduction techniques for ship detection.

Data preprocessing

The dataset consists of satellite images pre-processed into NumPy arrays (X) with corresponding binary labels (y) indicating ship presence. The images are flattened from their original dimensions to create feature vectors suitable for machine learning algorithms.

Maintaining the Integrity of the Specifications

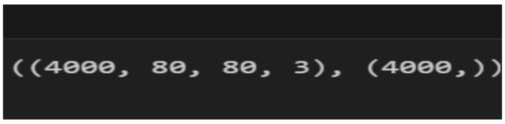
Data Integrity and Preprocessing

Dataset Specifications

Input images are stored in NumPy arrays (ship_data.npy)

Binary labels indicating ship presence (ship_labels.npy)

Original image dimensions: height × width × 3 channels



```
((4000, 80, 80, 3), (4000,))
```

Abbreviations and Acronyms

Technical Terms

PCA: Principal Component Analysis

ISOMAP: Isometric Mapping

RMSE: Root Mean Square Error

LR: Logistic Regression

RF: Random Forest

CV: Cross Validation

Units

- Image Processing Units
- Pixel Values: 0-255 (8-bit colour depth)
- Image Dimensions: pixels (height × width × 3 channels)
- Patch Size: 80 × 80 pixels for sliding window detection
- Performance Measurement Units
- Accuracy: Percentage (0-100%)
- F1-Score: Scale of 0.0 to 1.0
- Training Time: Seconds (s)

Inference Time: Seconds (s)

Equations

Classification Metrics

Accuracy Score Accuracy=TP+TNTP+TN+FP+FN*Accuracy=TP+TN+FP+FNTP+TN***F1-Score** F1=2×Precision×RecallPrecision+Recall*F*1=2×*Precision+RecallPrecision×Recall*

Dimensionality Reduction

PCA Variance Explained Varianceexplained=∑i=1kλi∑i=1nλi*Varianceexplained*

=∑i=1nλi∑i=1kλi
where λᵢ represents eigenvalues

RMSE for Reconstruction RMSE=1n∑i=1n(xi−xi^)2RMSE=n1∑i=1n(xi−xi^)2

Distance Metrics

Euclidean Distance (ISOMAP) d(p,q)=∑i=1n(pi−qi)2d(p,q)=∑i=1n(pi−qi)2

Model Specific

Logistic Regression P(y=1∣x)=11+e−(β0+β1x1+...+βnxn)P(y=1∣x)=1+e−(β0+β1x1+...+βnxn)1

Random Forest Probability P(y∣x)=1T∑t=1Tpt(y∣x)P(y∣x)=T1∑t=1Tpt(y∣x)
where T is the number of trees

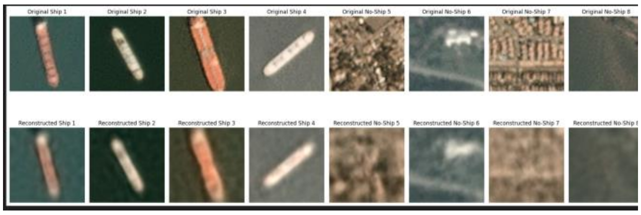
□ x1,x2,…,xnx_1, x_2, \dots, x_nx1,x2,…,xn = feature variables (e.g., Quantity, Total, etc.)
□ β0\beta_0β0 = intercept, representing the baseline value when all features are zero
□ β1,…,βn\beta_1, \dots, \beta_nβ1,…,βn = coefficients for the respective features, indicating the weight or importance of each feature in predicting yyy
□ ϵ\epsilonϵ = error term, representing the noise or deviation from the true values that the model cannot capture

Random Forest

PCA is a widely used technique for linear dimensionality reduction that projects data into a lower-dimensional space while maintaining as much variance as possible. Previous work has shown that PCA can significantly speed up model training without a large loss in accuracy, particularly in problems like face recognition and text classification.

ISOMAP, a nonlinear manifold learning technique, is particularly useful in cases where the data lies on a curved manifold. ISOMAP has been applied in various domains, including image recognition and speech analysis, where linear methods like PCA fail to capture complex relationships between features.

Logistic Regression and Random Forest have been extensively used in classification problems. Logistic Regression, being a linear model, tends to perform well when the data is linearly separable, while Random Forests, which are based on an ensemble of decision trees, are more robust in handling complex, non-linear relationships.



- 4.1 Dataset
- The dataset used in this study consists of images or high-dimensional features representing various types of ships. The dataset is split into two parts:
- Training Set (80%): Used to train the models.
- Testing Set (20%): Used for evaluating the models' generalization ability.
- 4.2 Data Preprocessing
- Feature Reshaping: If the dataset consists of image data, the images are flattened into 1D arrays to be used as features for the classifiers. This transformation converts the image pixels into feature vectors.

  Feature Scaling: To ensure that each feature contributes equally to the model's performance, the features are standardized using standard scaling, ensuring a mean of 0 and a standard deviation of 1.In For classification models (Random Forest, Gradient Boosting, Logistic Regression), accuracy is the primary performance metric. The formula for accuracy is:
- 4.3 Dimensionality Reduction
- Principal Component Analysis (PCA): PCA is applied to the feature matrix to reduce the number of dimensions while retaining 90% of the variance. This helps to reduce the noise in the data and increase computational efficiency.

  Isometric Mapping (ISOMAP): ISOMAP is used as a nonlinear dimensionality reduction technique. It preserves the global geometric structure of the data by constructing a neighbourhood graph and performing multidimensional scaling to find a lower-dimensional representation.

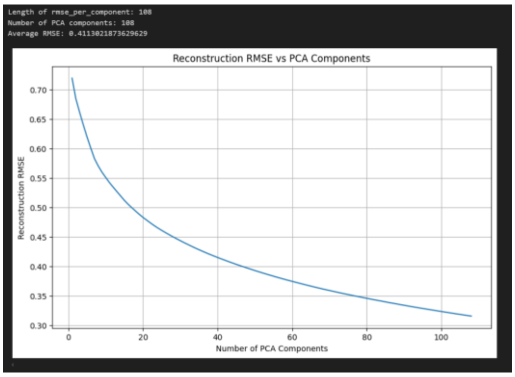
Models and Training strategy

A 4.4 Classification Models

Two classifiers are used in the experiments:

Logistic Regression (LR): A linear classifier that models the probability of class membership based on a linear combination of input features. GridSearchCV is used to optimize the regularization parameter (C).

Random Forest (RF): An ensemble learning method based on constructing multiple decision trees. The Random Forest classifier is tuned using GridSearchCV to optimize the number of trees (n_estimators) and the depth of each tree (max_depth).

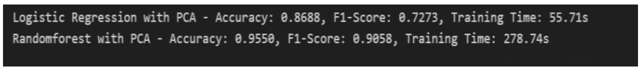


```
Length of rmse_per_component: 108
Number of PCA components: 108
Average RMSE: 0.4113021873629629
```

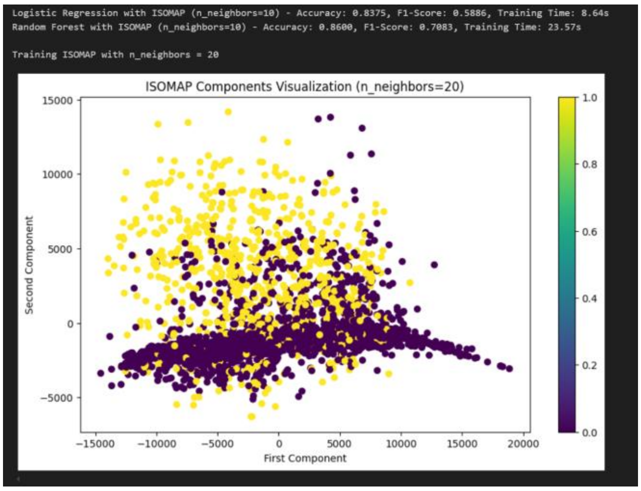
4.5 Evaluation Metrics

The models are evaluated using the following metrics:

- **Accuracy**: The proportion of correctly classified instances out of the total instances.
- **F1 Score**: The harmonic mean of precision and recall, which is useful for imbalanced classification tasks.
- **Training Time**: The time taken to train the model is measured and compared across different configurations.



```
Logistic Regression with PCA - Accuracy: 0.8688, F1-Score: 0.7273, Training Time: 55.71s
Randomforest with PCA - Accuracy: 0.9550, F1-Score: 0.9058, Training Time: 278.74s
```



```
Logistic Regression with ISOMAP (n_neighbors=10) - Accuracy: 0.8375, F1-Score: 0.5886, Training Time: 8.64s
Random Forest with ISOMAP (n_neighbors=10) - Accuracy: 0.8600, F1-Score: 0.7083, Training Time: 23.57s

Training ISOMAP with n_neighbors = 20
```

5.1 Model Performance without Dimensionality Reduction

Before applying any dimensionality reduction, both Logistic Regression and Random Forest classifiers are trained using the original feature set. The accuracy and F1 scores are recorded for both models. These serve as baseline metrics for comparison with models using dimensionality reduction.

5.2 Performance with PCA

PCA is applied to reduce the feature space, retaining 90% of the variance. The models are retrained on the reduced feature set, and the accuracy, F1 score, and training time are recorded. The impact of PCA on model performance is discussed, particularly how it affects classification accuracy and training efficiency.

5.3 Performance with ISOMAP

ISOMAP is applied as a nonlinear dimensionality reduction technique. The number of neighbors (n_neighbors) and components (n_components) are varied to assess their effect on model performance. As with PCA, the accuracy, F1 score, and training time are compared with the baseline results.

5.4 Comparative Analysis

A comparative table summarizes the performance of the Logistic Regression and Random Forest classifiers across the different configurations (no dimensionality reduction, with PCA, and with ISOMAP). Graphs or bar charts are used to visually represent the differences in accuracy, F1 score, and training time.

6.1 Effectiveness of Dimensionality Reduction

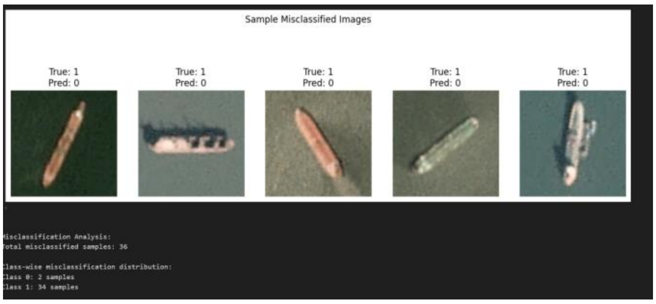
Both PCA and ISOMAP are shown to reduce the dimensionality of the feature space. For linear models like Logistic Regression, PCA often results in improved generalization without significant accuracy loss, especially when the original dataset is highly dimensional. However, for Random Forest, ISOMAP may provide better results by preserving complex, nonlinear relationships in the data.

6.2 Comparison of Logistic Regression and Random Forest

Random Forest generally performs better with higher-dimensional data, benefiting from its ability to capture non-linear patterns. However, PCA can still be effective in reducing overfitting by removing noise. Logistic Regression, though simpler, may see improved efficiency and faster training times when combined with PCA.

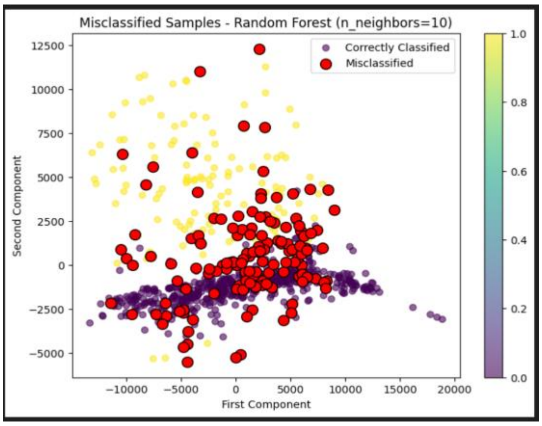
6.3 Misclassification Analysis

By analysing the confusion matrix, we gain insights into which classes the models struggle with the most. Misclassifications are visualized to reveal patterns or areas where additional feature engineering or tuning could improve performance.



```
Misclassification Analysis:
Total misclassified samples: 36

Class-wise misclassification distribution:
Class 0: 2 samples
Class 1: 34 samples
```



Measuring Inference Time One critical aspect of evaluating a machine learning model is measuring its inference time, especially in real-time applications like ship detection. The code implements a function, measure_inference_time(), which measures the average time taken by the model to make predictions on the test data over multiple iterations (set to 10 by default). This helps assess the latency or speed of the model when making predictions, which is particularly important for large datasets or real-time systems.The time is measured by invoking model.predict(X_test) multiple times, where the time taken for each prediction is recorded. The average time across these iterations is returned to give a more stable measure of inference speed, reducing the effect of any outliers or system fluctuations during one-off predictions.

Model Evaluation

The evaluate_model() function is responsible for evaluating the performance of each model. This function takes the following steps: Inference Time Measurement: It calls measure_inference_time() to determine how quickly the model makes predictions on the test data. Prediction: The model makes predictions on the X_test dataset (the features of the test data). Accuracy and F1-Score Calculation:

Accuracy is calculated using accuracy_score(), which measures the proportion of correctly classified instances. It is a standard metric for classification tasks but may not be sufficient for imbalanced classes.

F1-Score is calculated using f1_score(). The F1-score is the harmonic mean of precision and recall and provides a better

understanding of model performance when dealing with imbalanced datasets (e.g., a situation where one class is significantly more prevalent than another).

Evaluating Multiple Models In the main() function, several trained models are loaded from disk using joblib.load() (e.g., 'Q1_LR', 'Q1_RF', 'Q3_LR_PCA', etc.). These models are then evaluated using the evaluate_model() function. The models tested include both:

Original models (e.g., linear regression and random forests)

Dimensionality-reduced models (e.g., models trained after applying PCA or other techniques like ISOMAP). Each model is tested using the same X_test and y_test datasets. Results for each model (accuracy, F1-score, inference time) are collected in a list of dictionaries and later converted into a Pandas DataFrame for easy comparison.

4. Results Summary and Detailed Analysis

Once all models have been evaluated, the results are printed in a table format. The table shows: The model name

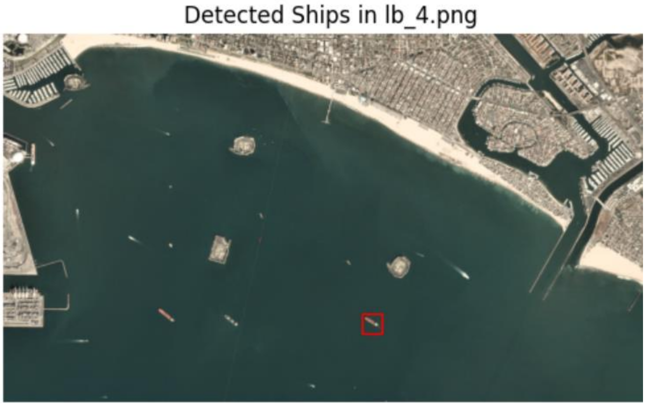
Accuracy F1-Score Inference Time for each model

Ship Detection Visualization

In addition to the evaluation of model performance on the test set, there is also a focus on practical **ship detection** using the trained models. The `detect_ships_in_scene()` function processes an image and detects ships within it using a sliding window approach:

- **Sliding Window**: The image is divided into smaller patches (of size `80x80` by default). Each patch is processed by the model to predict whether a ship is present.
- **Prediction**: If the model predicts a ship in a given patch (using a classifier such as RandomForest), the location of the patch is recorded.
- **Visualization**: The detected ship locations are then visualized by drawing bounding boxes around the detected ships on the image, which helps assess how well the model is able to detect ships in various parts of the scene.

This visual feedback allows for a quick assessment of the model's effectiveness in real-world applications, where detecting and localizing ships in images or video streams is critical.



V. Conclusion

This paper demonstrates an end-to-end machine learning pipeline for supermarket sales data. Through careful preprocessing, feature engineering, and hyperparameter tuning, we optimized multiple models for both classification and regression tasks. Future work may explore additional regularization techniques and deep learning approaches for more complex predictions.

In preprocessing, the numerical units were standardized using StandardScaler to have zero mean and unit variance. This transformation was important to ensure that all features are on the same scale, especially when fitting regression models like Linear and Lasso Regression.

For categorical variables (e.g., Branch, Gender, Product Line), the values were converted into binary (one-hot) encoded features.

[1] [1] I. T. Jolliffe, "Principal Component Analysis," Springer Series in Statistics, 2nd ed., Springer, 2002.

[2] L. Tenenbaum, V. Silva, and J. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," Science, vol. 290, no. 5500, pp. 2319–2323, 2000.