Machine Learning for Natural Disaster Relief

Cameron Preasmyer

UNIVERSITY of VIRGINIA | SCHOOL of DATA SCIENCE

Table of Contents

01	Introduction	Project Context and Objective
02	Data	EDA and Data Augmentation
03	Description of Methodology	Models Analyzed, Threshold Selection, Hyperparameter Optimization, Target Metrics
04	Results	Hyperparameter Tuning Results, Threshold Decision, Model Metrics Comparison
05	Final Conclusions	Model Selection and Relevant Findings

01 Introduction

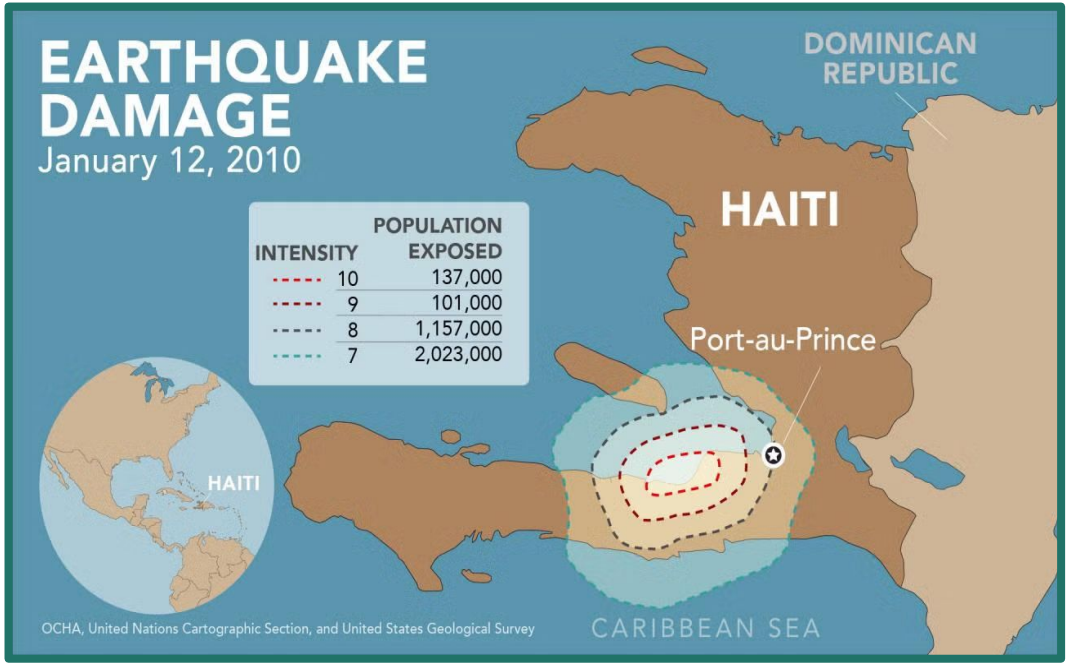
Magnitude 7.0 Earthquake Strikes near Port Au Prince, Haiti

Infrastructural Impact

- **250,000** Residential Buildings Destroyed
- **30,000** Commercial Buildings Destroyed

Human Impact

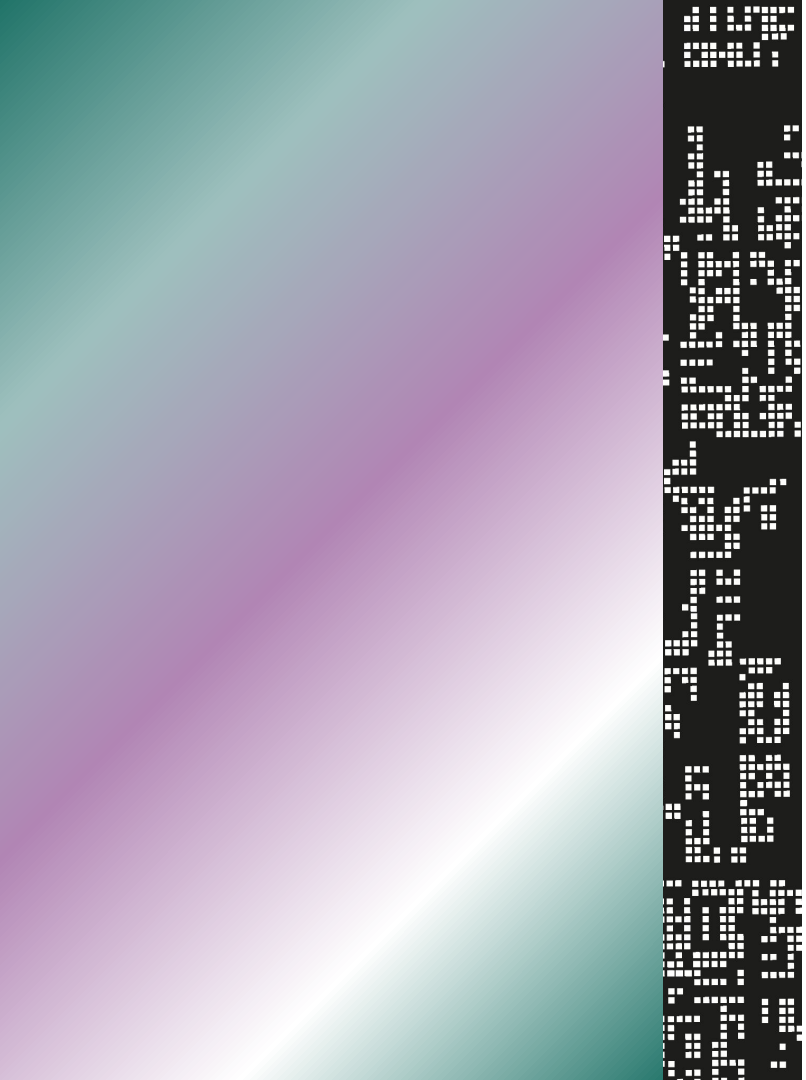
- **222,570** Casualties
- **300,000** Injured
- **1.3 Million** Displaced



Blue Squares

Rescue Mission

- LiDAR and aerial imagery were implemented to aid relief efforts
- Data Scientists teams saw an abundance of ‘blue squares’ in the aerial imagery
- Displaced citizens used blue tarps to create makeshift villages
- Images converted into tabular data comprising the RGB intensity of each pixel



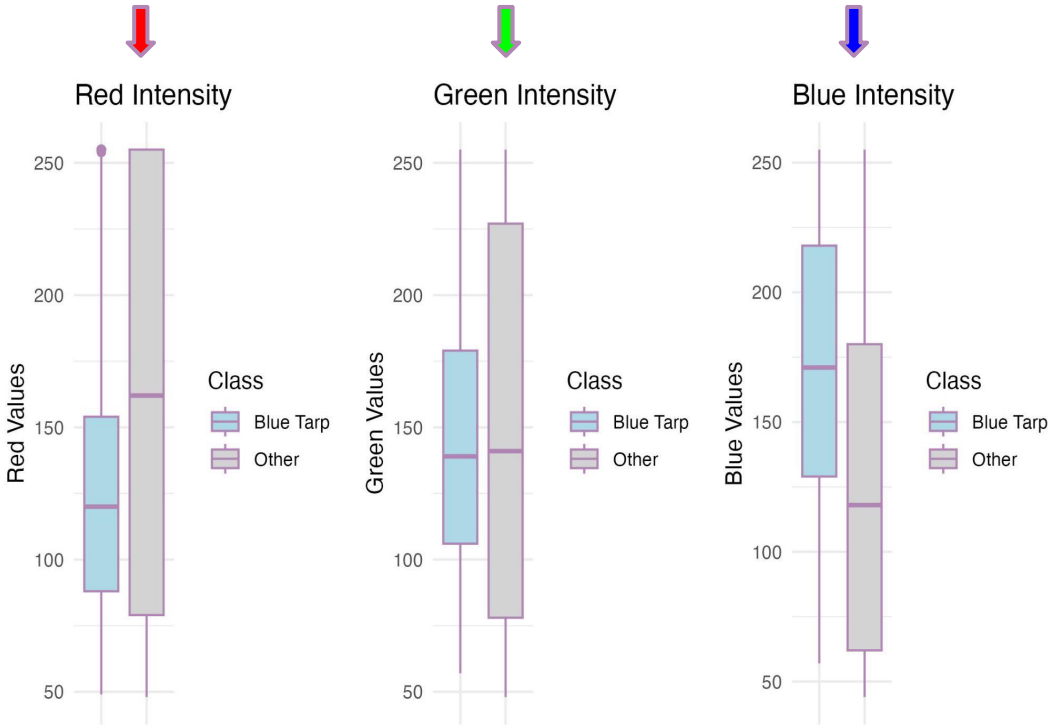
Objective

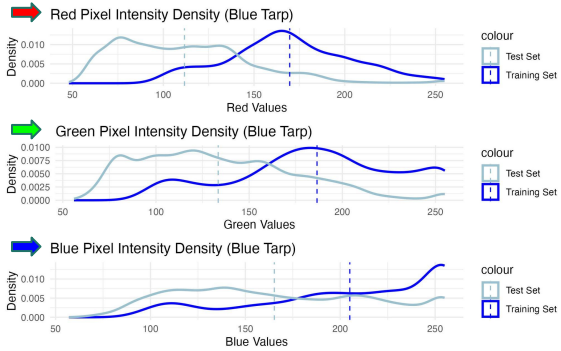
Evaluate statistical learning algorithms to optimize quick and accurate identification of blue tarps to aid relief efforts.

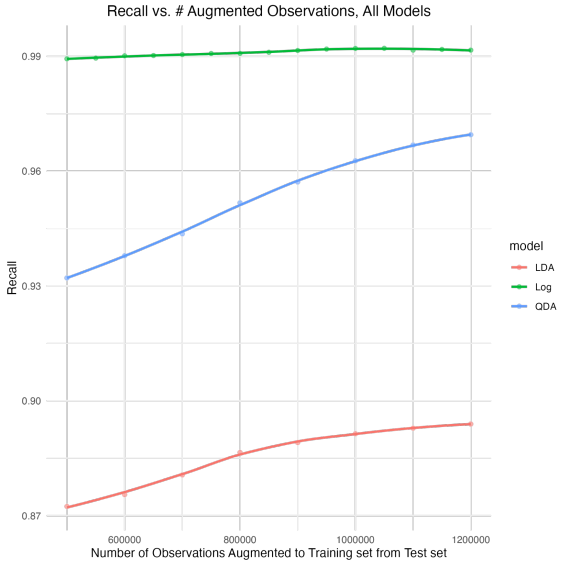
Data 02

The Pixels

- Training and Testing data split into two classes
  - Blue Tarp
  - Other
- ‘Blue Tarp’ pixel color intensities follow a **R**<**G**<**B** trend
- More Variance and less structure in the color intensity of ‘Other’ pixels







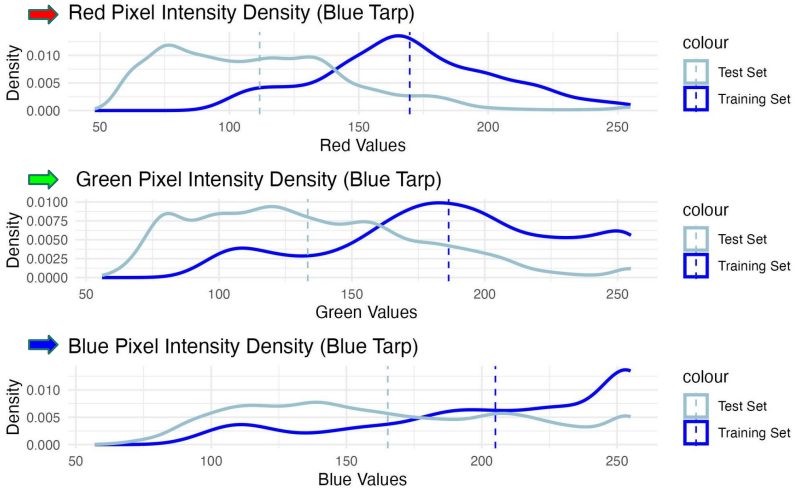
Data Augmentation

- Differences in the means of color intensity in ‘Blue Tarp’ observations between the training and test sets
- Created a function to see the effects on recall after augmenting data points from the test set (~2 Million observations) to the training set (~60,000 observations)
- Randomly sampled 1,000,000 data points from test set, discarded data points from the ‘Other’ class, and augmented the remaining ‘Blue Tarp’ observations into the training set
- Ratio of ‘Blue Tarp’ to ‘Other’ remained the same in the test set due to random sampling

Data Augmentation Results

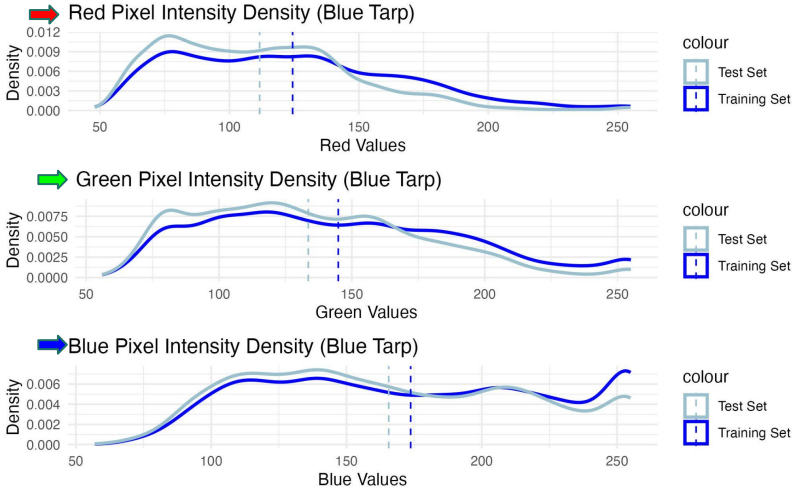
- Substantial differences in mean values (dotted vertical lines) of color columns between training and testing sets for ‘Blue Tarp’ observations
- Post-Augment test set mean values and distribution remained largely unchanged after augmentation

Pre Augmentation



- Augmentation led to a substantial increase in both AUC scores and Precision-Recall curves for LDA, QDA and Logistic Regression
- Training set is much more representative of real world data (test set) with mean values much closer together

Post Augmentation



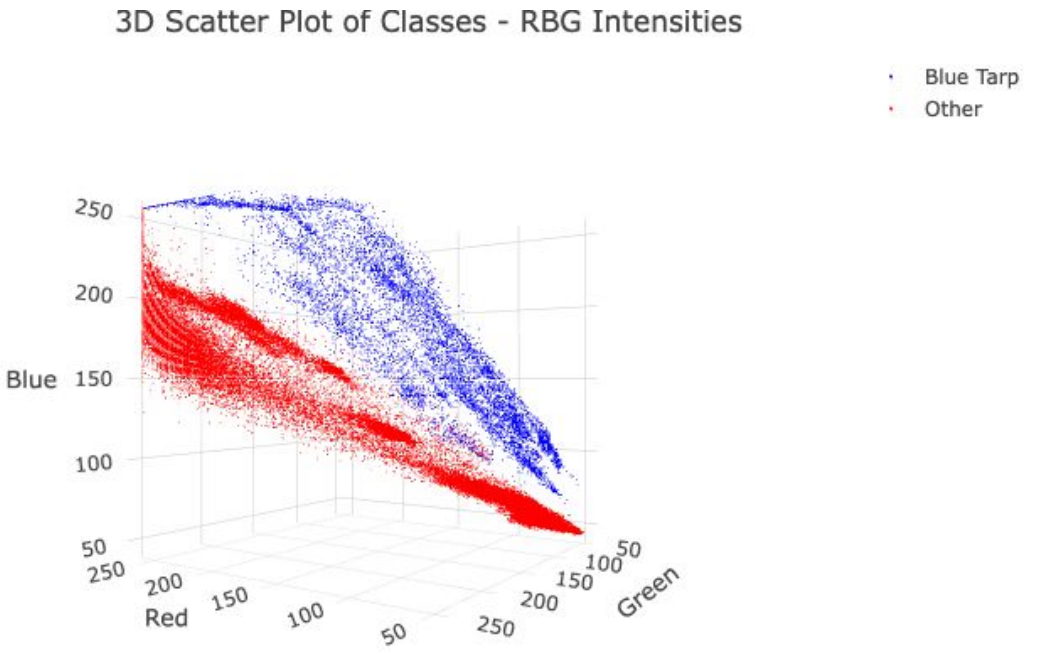
Class Imbalance

- Classes are highly imbalanced in **both** the test and training sets
- Augmenting data from test set increased Blue Tarp representation in the training set from **3.2%** to **13.08%**
- Class imbalance was not altered in the test set to emulate real-world data and retain the integrity of model results

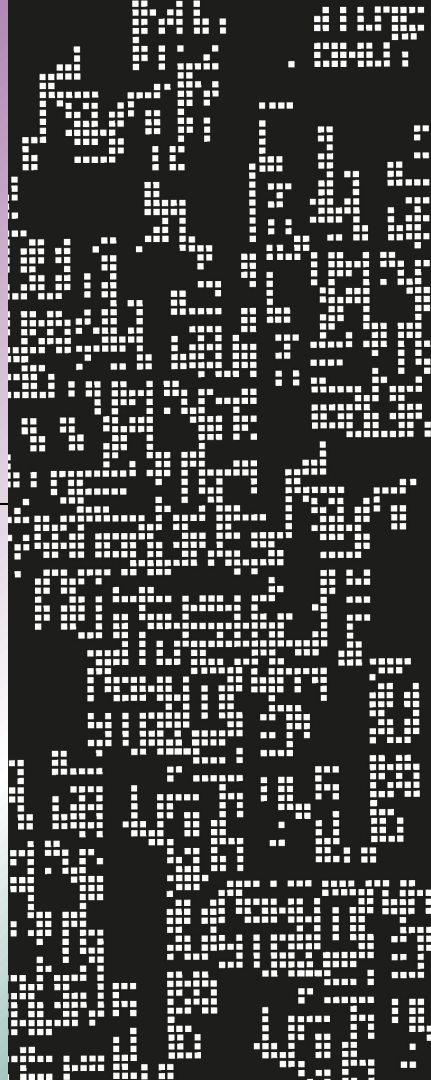
Training Set: Pre Aug.		Training Set: Post Aug.		Test Set: Pre Aug.		Test Set: Post Aug.	
Data Points:	**63,241**	Data Points:	**70,428**	Data Points:	**2,004,177**	Data Points:	**1,004,177**
Blue Tarp:	**3.2%**	Blue Tarp:	**13.08%**	Blue Tarp:	**0.72%**	Blue Tarp:	**0.73%**
Other:	**96.8%**	Other:	**86.92%**	Other:	**99.28%**	Other:	**99.27%**

Class Separation in Training Set

- Although there are roughly 7.5 times more ‘Other’ observations in the training set, class separation is visually noticeable
- Small amounts of ‘Other’ points like within the ‘Blue Tarp’ euclidean space
- Unlikely to perfectly classify everything correctly, but the classes are likely to be majorly separable by the models



03 Description of Methodology



01

Create Workflows for the different models

Each model starts with a basic workflow used to evaluate CV and Test Results

02

Cross Validate and Tune Hyperparameters

10-Fold Cross Validation is used to detect over or under fitting, and to find optimal tuning values

03

Select Prediction Thresholds

Scanning functions are used to find a prediction threshold which optimize chosen metrics

04

Compare Model Metrics

Model metrics are compared to identify the best tool to use on future data

Models & Hyperparameters

Non-Tunable Models Evaluated

Models

- Logistic Regression (MASS engine)
- Linear Discriminant Analysis (LDA)
- Quadratic Discriminant Analysis (QDA)

Tunable Models Evaluated

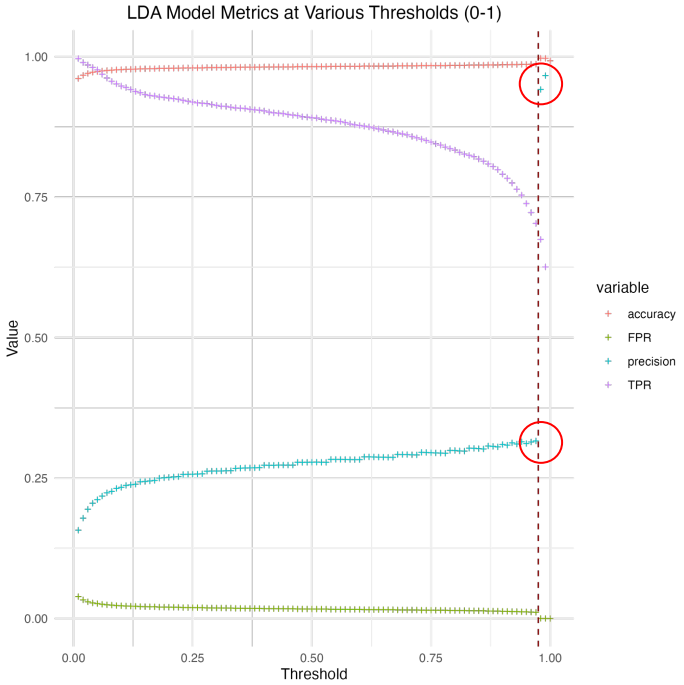
Models

K-Nearest Neighbor (KNN)	Logistic Regression (glmnet engine)	Random Forest	Support Vector Machines (SVM)

Hyperparameters

K-Nearest Neighbor (KNN)	Logistic Regression (glmnet engine)	Random Forest	Support Vector Machines (SVM)
➢ K Neighbors	➢ Penalty	➢ # of Trees ➢ # of Selected Variables ➢ # of Data points per node	➢ Degree (poly kernel) ➢ RBF_Sigma(rbf kernel) ➢ Cost ➢ Margin

Threshold Selection



- Threshold scanning function evaluates model metrics from end-to-end
- Smaller decreases in FPR create huge improvements in precision due to class imbalance.
- Looking at the shifts in classic metrics provides a window to look deeper into performance around those threshold values

Thresholds are finally selected by the following metric:

$$1 + \frac{TP}{FP}$$

While also manually accounting for small TPR and FPR fluctuations and trends

Model Metrics & Performance

Model performance is compared by the output of the following expression

$$1 + \frac{TP}{FP}$$

The models can be employed on future data using the following inequality:

$$\sum \hat{y}_{BlueTarp} - \lambda \cdot E(FP) > 0$$

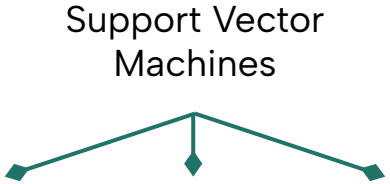
where $E(FP) = \frac{\sum \hat{y}_{Other}}{1-FPR} \cdot FPR$ **and** $\lambda \in \mathbb{R}, \lambda > 1$

For the model to flag as intended, the value of lambda must be chosen carefully after deriving model metrics such that

$$\lambda \in (1, \frac{\sum \hat{y}_{BlueTarp}}{E(FP)})$$

Results 04

Model Tuning Results

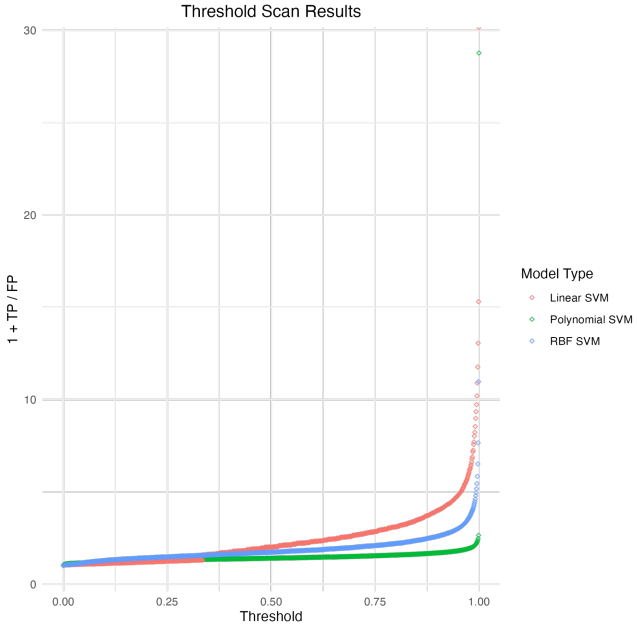


Poly Kernel	RBF Kernel	Linear Kernel	Random Forest	K-Nearest Neighbors	Logistic Regression
Cost: **1.34**	Cost: **2.97**	Cost: **0.064**	mtry: **2**	Neighbors: **11**	Penalty: **1.26e-05**
Margin: **0.249**	Margin: **0.181**	Margin: **0.064**	Trees: **1607**		
Degree: **5**	Sigma: **0.942**		min_n: **10**		

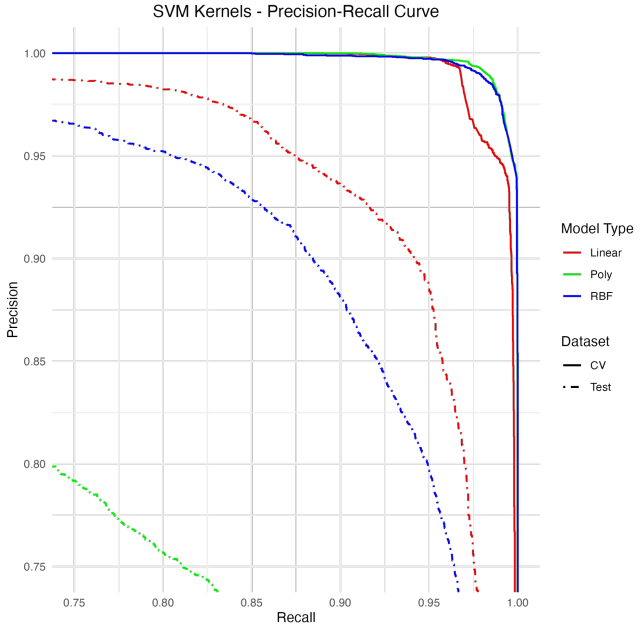
All hyperparameters are tuned using 10-fold cross validation with folds from the training set

SVM Kernel Selection

- Linear Kernel outperformed the other kernels on test data
- Other models performed better on CV folds – signs of overfitting

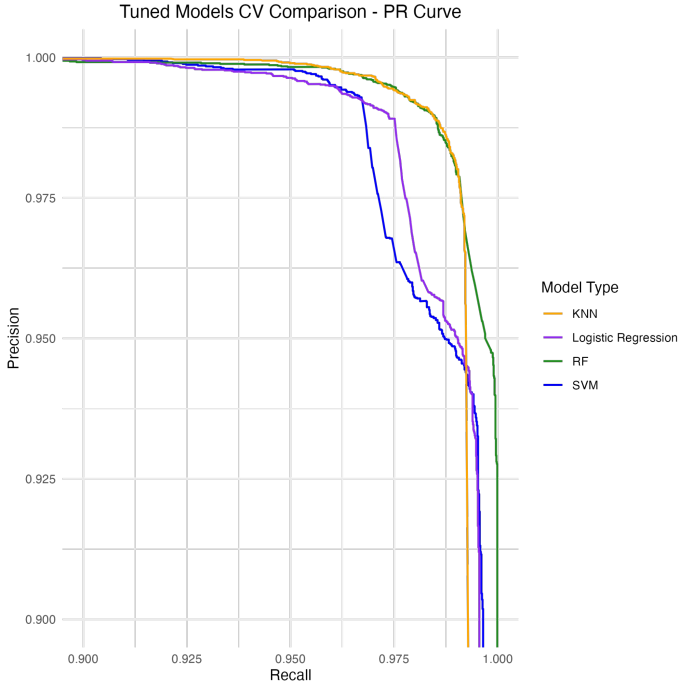


- No Surprise – EDA Showed data was highly linearly separable
- We will now use the **Linear Kernel** when comparing models

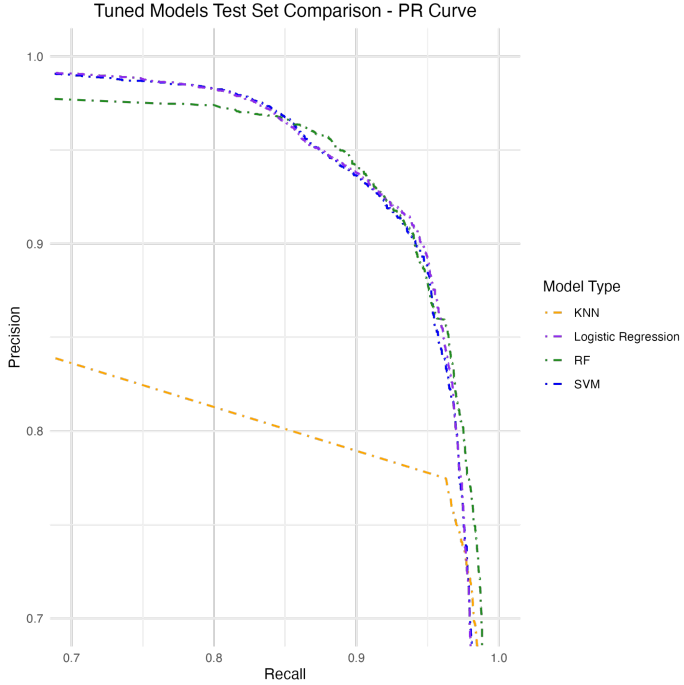


Tuned Model CV & Test Results

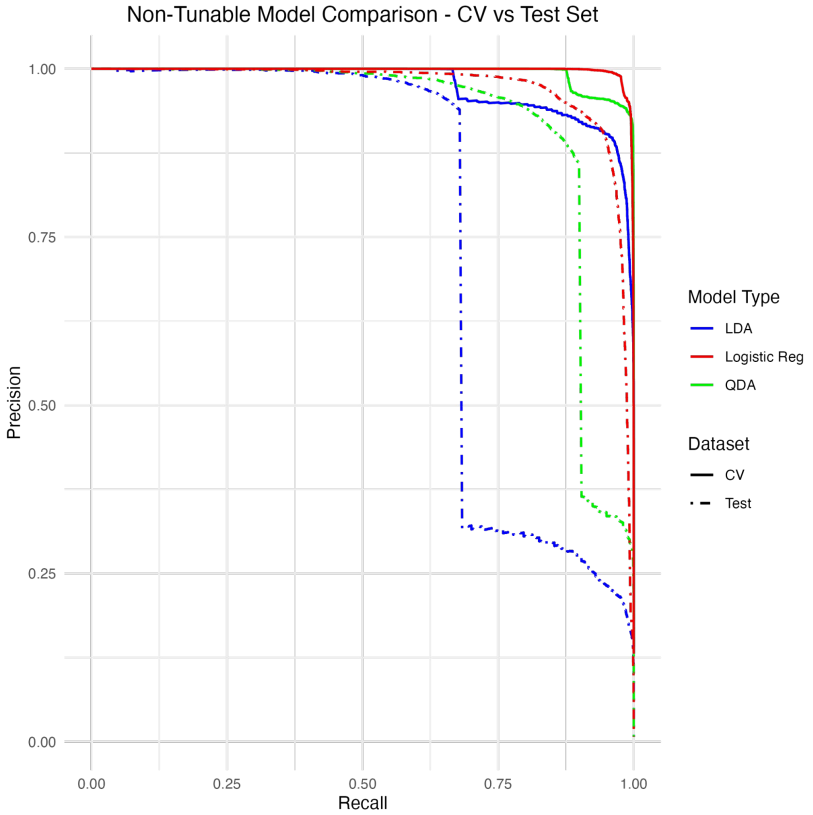
Cross Validation



Test Set



Non-Tunable Model CV & Test Results

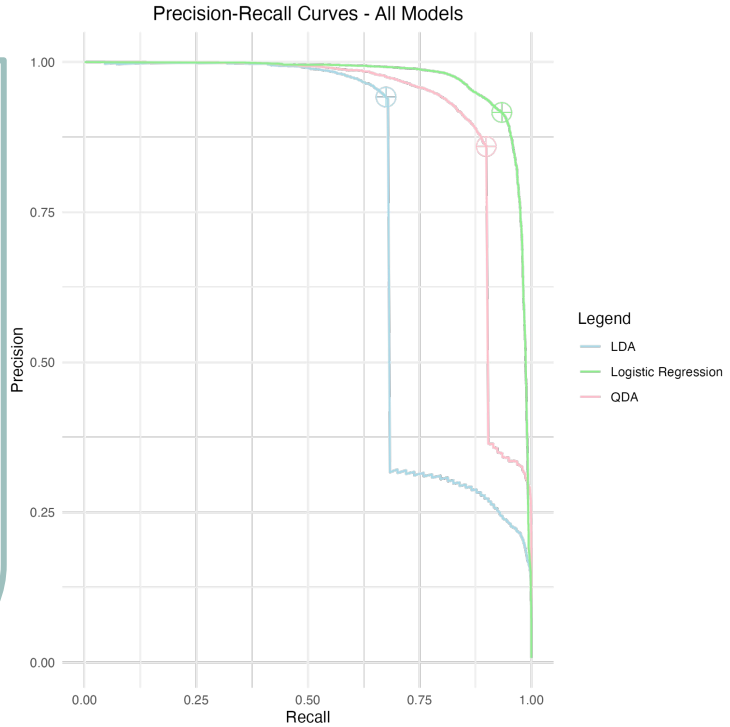


- Logistic Regression is a top performer with both CV and Test Performance
- Logistic Regression and QDA both show low variance with decent bias
- LDA is decent at middle ranges of precision and recall

Non-Tunable Model Metrics

	Threshold	TPR	FPR	Precision	Accuracy
Linear Discriminant Analysis	0.98	0.674	**.0003**	**0.941**	0.997
Quadratic Discriminant Analysis	0.834	0.899	0.001	0.859	0.998
Logistic Regression	0.997	**0.934**	0.001	0.916	**0.999**

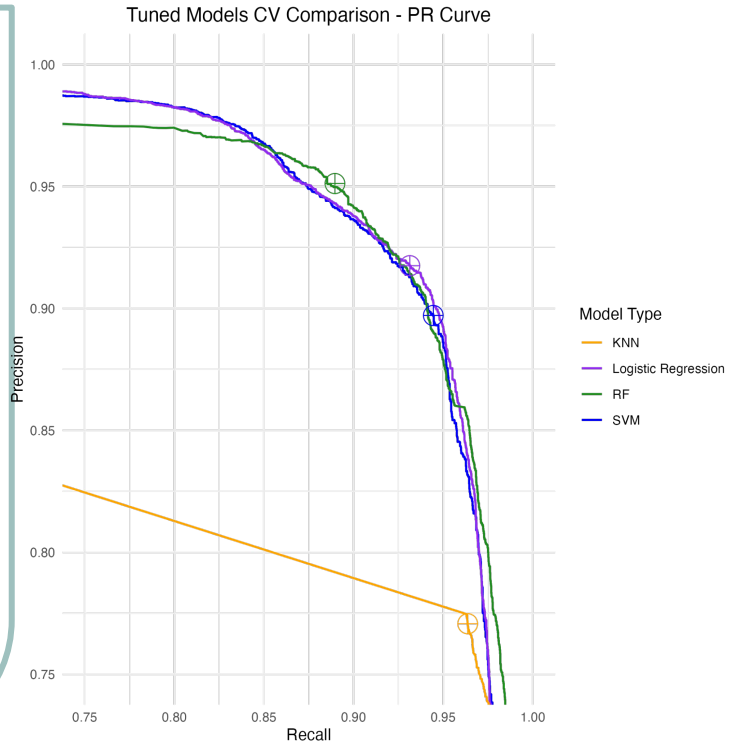
Test data



Tuned Model Metrics

Test data

	Threshold	TPR	FPR	Precision	Accuracy
Random Forest	0.998	0.8896	**0.0003**	**0.9512**	0.9989
Support Vector Machine (SVM) Linear Kernel	0.994	0.9445	.00079	0.8971	0.9988
Logistic Regression	0.9983	0.9314	0.0006	0.9175	**0.9989**
K-Nearest Neighbors	0.997	**0.9637**	0.0021	0.7707	0.9977



Model Performance Comparison

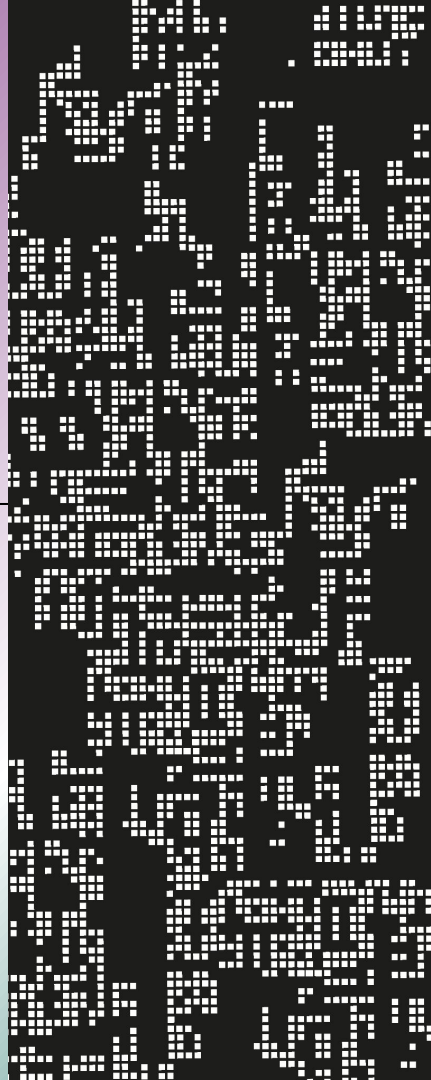
	Model	1+TP/FP
1.	**Random Forest**	**17.418**
2.	LDA	17.134
3.	Logistic Regression (Tuned)	12.118

	Model	TPR
1.	**KNN**	**0.964**
2.	Logistic Regression (Non Tuned)	0.951
3.	SVM – Linear Kernel	0.945

	Model	FPR
1.	**LDA**	**0.0003**
2.	Random Forest	0.0004
3.	Logistic Regression (Tuned)	0.0006

Tunable and Non-Tunable models - All metrics derived from test set

05 Final Conclusions



Algorithm Performance

Cross Validation

★ **Random Forest**

Honorable Mention – KNN

- CV Performed Similar to KNN
- Achieved a lower FPR than KNN
- Didn't show signs of overfitting like KNN

Holdout Data

★ **Random Forest**

Honorable Mention – LDA

- Achieved lower FPR and highest Precision of all models
- Low signs of overfitting from CV to Test Performance

Recommended for use

★ **Random Forest**

Honorable Mention – LDA

- Highest 1+TP/FP Metric which roughly translates to flagging performance
- Low FPR and high TPR will allow for variance in metrics with good flagging performance

Relevant Metrics

Precision

Loosely correlates to the flagging criteria

False Positive Rate

Highly Imbalanced Data means small changes in FPR create big difference in TP/FP

True Positive Rate

Also known as Recall, it is ideal to keep this as high as possible **<u>after</u>** controlling for FPR

Why Random Forest?

Recall the flagging inequality

$$\sum \hat{y}_{BlueTarp} - \lambda \cdot E(FP) > 0$$

Where

$$E(FP) = \frac{\sum \hat{y}_{Other}}{1-FPR} \cdot FPR$$

And

$$\lambda \in (1, \frac{\sum \hat{y}_{BlueTarp}}{E(FP)})$$

$$\frac{\sum \hat{y}_{BlueTarp}}{E(FP)} = \frac{E(TP)+E(FP)}{E(FP)} = 1 + \frac{E(TP)}{E(FP)} \approx \boxed{1 + \frac{TP}{FP}}$$

- Random Forest has the highest value of $1 + \frac{TP}{FP}$
  - ➢ Largest possible range of lambda **(1, 17.418)**
  - ➢ Absorbs the largest amount of variation in FP and TP while flagging images

Random Forest Example

Arbitrarily Set $\lambda = 5$

$$6950 - 5 * 399 = 6950 - 1995 = \boxed{4955 > 0}$$

TP+FP λ FP

Random Forest will flag this image as having a Blue Tarp, but will also withstand images that contain **4.1 times** as many False Positives at the given lambda value per the equation:

$$x = -\frac{TP}{FP(1-\lambda)}$$

Deeper Observations

Using Multiple Models: An End-to-End Solution

1) Use top performing model to flag images
2) (Optional) Implement a second model on flagged images to eliminate as many FP pixels as possible (possibly different from flagging model)
3) Use a DBSCAN model to Identify clusters within each picture and collect geospatial data from midpoints and compile these geospatial data from all images
4) Use a second clustering model to cluster aggregated geospatial data across larger areas (example: x-axis Lat, y-axis Long)

Benefits

- Optimizes relief efforts by identifying blue tarp 'hot zones' where rescue efforts can more accurately estimate resources needed to aid the area
- Flagging images first avoids clusters being in images that have no blue tarp

Drawbacks

- Potentially computationally expensive

Class boundaries are Linearly Separable

1) For this data, high variance models such as QDA, KNN, and SVM's with the Radial Basis Function and Polynomial Kernels performed the worse on test data and tended to overfit
2) High bias models that assume linearity such as SVM with a linear kernel, Logistic Regression, and Random Forest performed the best
3) 3d Plots during the EDA phase of the project show relatively linear separation between classes

Takeaways

- Models that assume linearity can be less computationally expensive than flexible models
- For this data, the use of models that assume linearity, such as Logistic Regression, can solve computational restrictions as they arise with other models, such as Random Forest, without sacrificing model performance