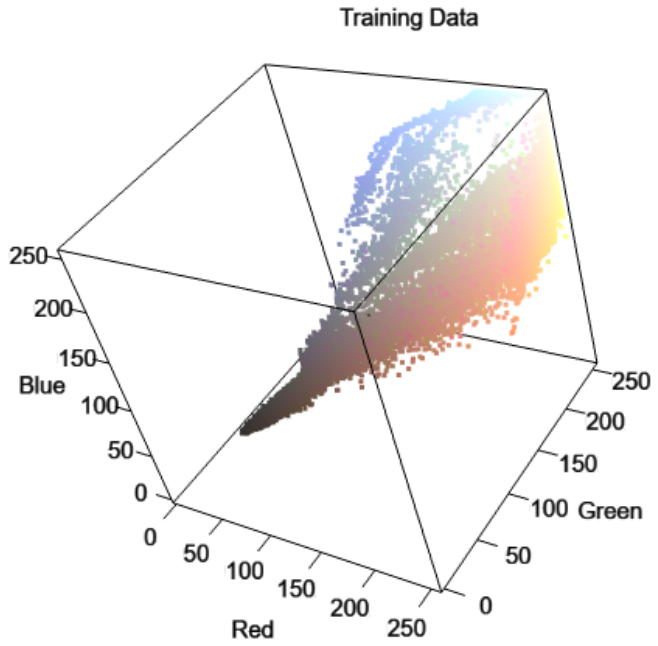
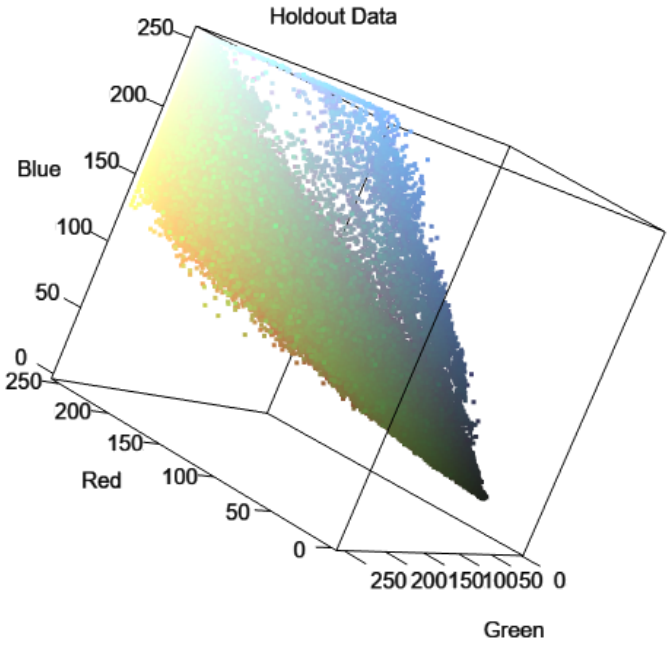




Haiti Earthquake Relief Effort: Finding Blue Tarps II

DS 6030 | Fall 2021 | University of Virginia

Brooks Anderson | email@virginia.edu

December 5, 2021

Introduction

- In the aftermath of the 2010 Haiti earthquake, rescue workers needed a way to find displaced people from aerial photographs.
  - Specifically, rescue workers were looking for blue tarps on the landscape which might represent a temporary shelter.
- While at Rochester Institute of Technology (RIT), Dr. Bill Basener developed a statistical model which took pixel RGB values as input and returned the probability that a given pixel was a blue tarp [1].
- In this project, I expand upon this work and attempt to develop a new model that achieves higher classification accuracy.
- I compared seven model types to the original Mahalanobis Distance Classifier.
  - k-Nearest Neighbors, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Logistic Regression, Random Forest, Support Vector Machine (polynomial kernel), Support Vector Machine (RBF kernel)
- The data for this analysis was provided by Dr. Basener. The original images were collected by airplane and transferred to RIT.
- Although the main objective of this project is to demonstrate my knowledge of various classification methods, it is my hope that this analysis could improve image detection methods in the event of future disasters.

Methods

- This analysis was conducted exclusively in R
- I tested eight models
  1. k-Nearest Neighbors (kNN)
  2. Linear Discriminant Analysis (LDA)
  3. Quadratic Discriminant Analysis (QDA)
  4. Logistic Regression
  5. Random Forest (RF)
  6. Support-vector Machine with Polynomial Kernel (SVM-P)
  7. Support-vector Machine with RBF Kernel (SVM-R)
  8. Mahalanobis Distance Classifier

- The following packages were used,

Package	Purpose
`tidyverse`	Data manipulation
`ggplot2`	General Plotting
`viridis`	Plot color scales
`caret`	Model tuning & validation
`ROCR`	ROC curve tools
`e1071`	SVM tools
`kernlab`	SVM tools

- Confusion matrices were generated with the `confusionMatrix()` function from `caret`

Methods

Training & Testing

- The dataset from part 1 of the project was used to tune the models. The class labels were suppressed into a binary response variable with labels `other` and `tarp`.
- I took a 50/50 split of this dataset; allocating one half for training and the other half for testing. Initially, I attempted to use an 80/20 split. However, the training time was prohibitive. My experiments revealed that reducing the training data from 80% to 50% did not lead to a significant drop in accuracy. More importantly, the parameters selected through cross-validation were unchanged at each proportion.
- For each model, I performed 10-fold cross-validation five times on the training set. Once the parameters were selected, the models were fit on the entire training set.
  - 10-fold cross-validation was performed on methods without tuning parameters, such as LDA. This was done for the sake of experimental consistency and allowed me to make intermediary comparisons between models.
- Following training, predictions were made on the testing subset. Confusion matrices were constructed for each model and the Accuracy, true positive rate (TPR), false positive rate (FPR), and positive predictive value (PPV) were recorded

Methods

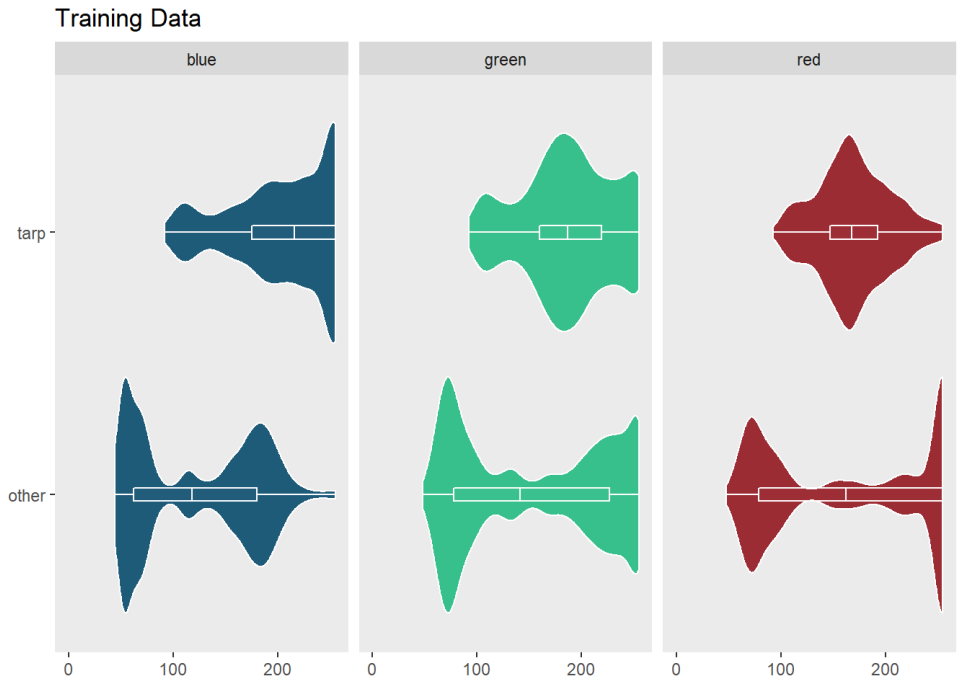
Holdout

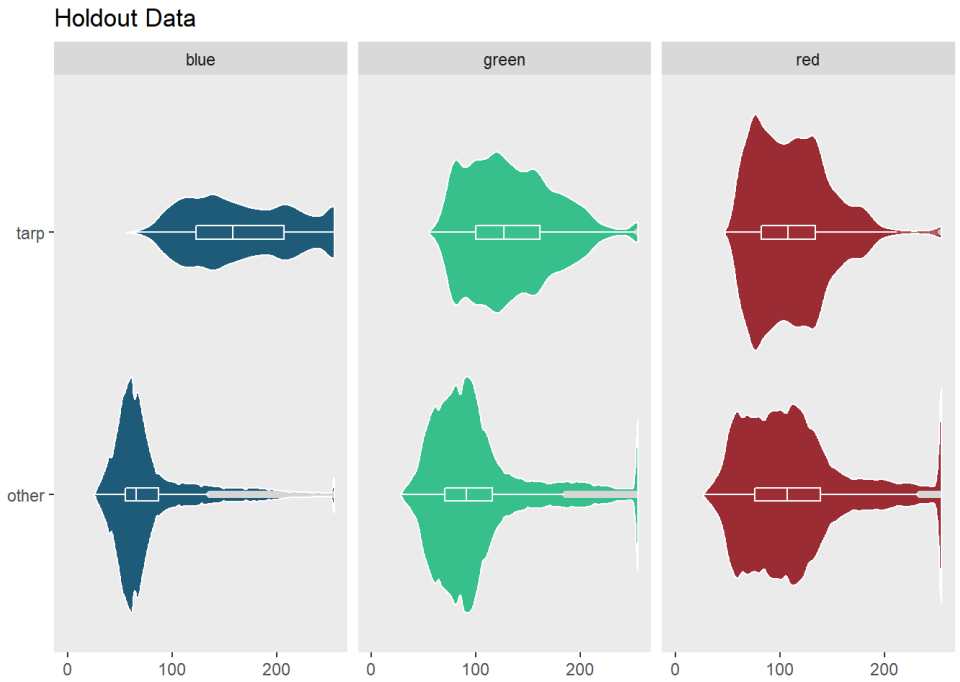
- The raw holdout data was provided in seven .txt files. Unnecessary columns were removed, leaving only the red, green, and blue predictor columns. Binary labels for `tarp` and `other` were applied. All seven files were combined into a single dataframe in R. There are approximately 2 million observations in the holdout set.
- Each model was used to make predictions on the holdout set. Both, class labels and probabilities were predicted. Confusion matrices were generated from the predicted labels. Accuracy, TPR, FPR, and PPV were recorded. The probabilities were used to plot ROC curves and calculate AUC.
- Lastly, I created decision boundary plots for each model. These plots are not useful for comparing model accuracy. However, they illustrate each method's flexibility and give insight into which class of methods might be best.

Methods

After both the training and holdout datasets were constructed and processed, I performed preliminary exploration and analysis

The violin plots below were created to view the distribution of each predictor and to determine if normalization or scaling was required





Methods

Preprocessing & Tuning Parameters

Method	PreProcessing	Tuning Parameters
kNN	center and scale	k – number of neighbors
LDA	*None*	*None*
QDA	*None*	*None*
Logistic Regression	*None*	*None*
RF	*None*	mtry – number of predictors sampled at each step
SVM-Polynomial	center and scale	degree – degree of kernel function C – cost of violation of the margin
SVM-RBF	center and scale	gamma – positive constant in Radial Basis Function C – cost of violation of the margin
Mahalanobis Distance	center and scale	*None*

Methods

Additional Notes

- SVM Polynomial – The out-of-the-box SVM Polynomial Kernel method offered by `caret` does not match the method demonstrated in *An Introduction to Statistical Learning* [2]. I created a custom SVM model that better matched the book and could be trained with `caret`.
- Random Forest – The random forest method offered by `caret` only has one tuning parameter, `mtry`. We used the default values for the number decision trees (`ntrees = 500`) and minimum node size (`nodesize = 5`).
- Mahalanobis Classifier – `caret` does not have a Mahalanobis distance classifier method. Since there are no tuning parameters for this method, I simply used a 50/50 train/test split of the dataset from part I. Then, the means and covariances of the training data were used to calculate distances for the observations in the holdout set. Mahalanobis Distance follows a X^2 distribution. The distances were passed to the `pchisq()` function to calculate the probability that a given pixel represents a tarp.

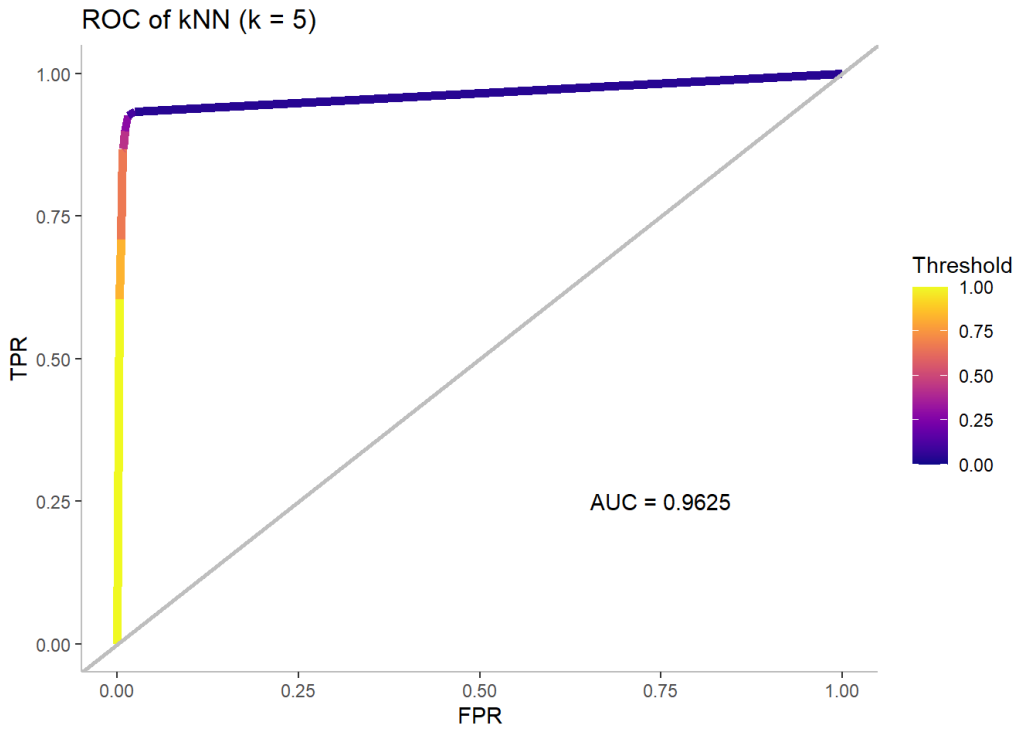
Accuracy Results: Train Data Set

Method	Parameters	Accuracy	TPR	FPR	PPV
kNN	`k = 5`	0.9971	0.9713	0.0020	0.9406
LDA		0.9844	0.8111	0.0099	0.7308
QDA		0.9950	0.8566	**0.0004**	**0.9852**
Logistic Regression		0.9959	0.9021	0.0010	0.9682
Random Forest	`mtry = 2, ntree = 500`	0.9969	0.9595	0.0018	0.9454
SVM-P	`degree = 3, C = 10`	0.9960	0.9179	0.0014	0.9547
SVM-R	`gamma = 5, C= 1000`	**0.9975**	**0.9723**	0.0017	0.9507
Mahalanobis Classifier		0.9883	0.7369	0.0034	0.8785

Accuracy Results: Holdout Data Set

Method	Parameters	Accuracy	TPR	FPR	PPV	AUC
kNN	`k = 5`	0.9916	0.8675	0.0075	0.4570	0.9625
LDA		0.9818	0.8394	0.0172	0.2620	0.9922
QDA		**0.9961**	0.7142	0.0019	**0.7365**	0.9927
Logistic Regression		0.9898	0.9890	0.0102	0.4140	**0.9995**
Random Forest	`mtry = 2, ntree = 500`	0.9913	0.8387	0.0076	0.4458	0.9799
SVM-P	`degree = 3, C = 10`	0.9942	**0.9908**	0.0057	0.5566	0.9991
SVM-R	`gamma = 5, C= 1000`	0.9907	0.7517	0.0076	0.4195	0.9869
Mahalanobis Classifier		0.9932	0.0965	**0.0003**	0.7064	0.7624

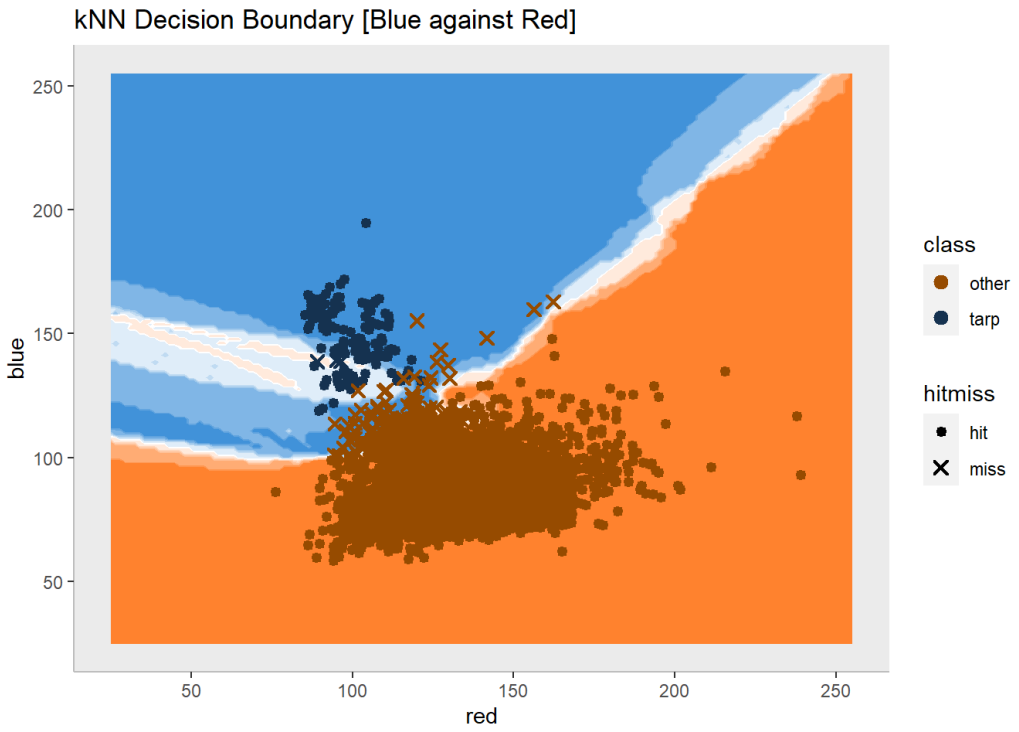
kNN Performance on Holdout Data

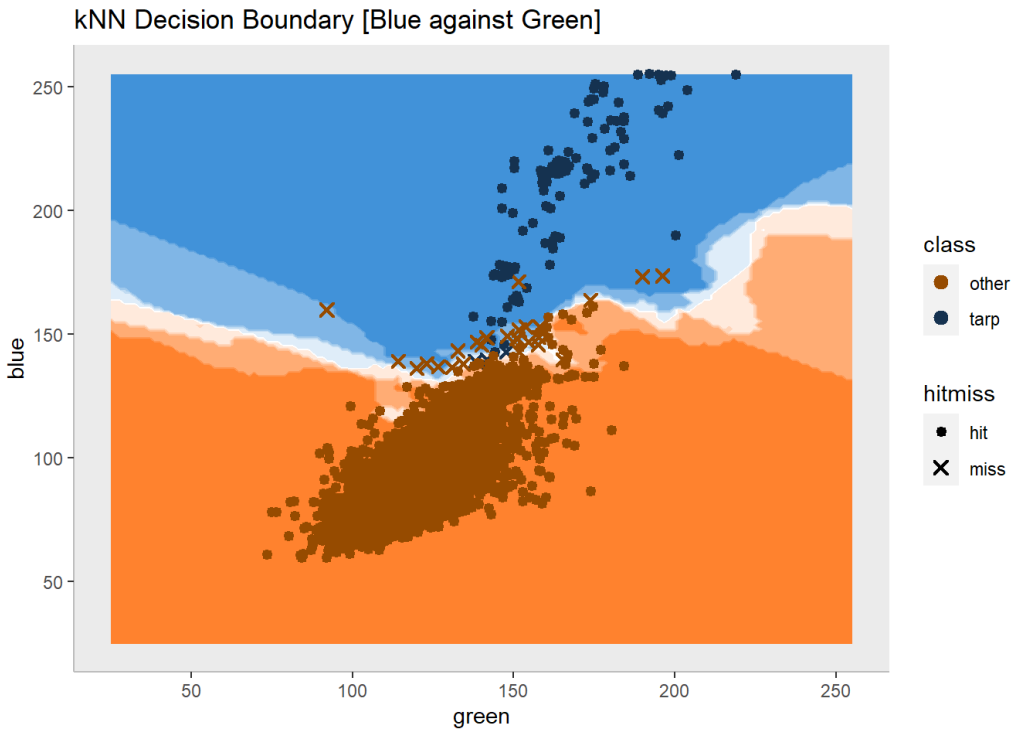


	Reference	
Prediction	Tarp	Other
Tarp	12562	14929
Other	1918	1974768

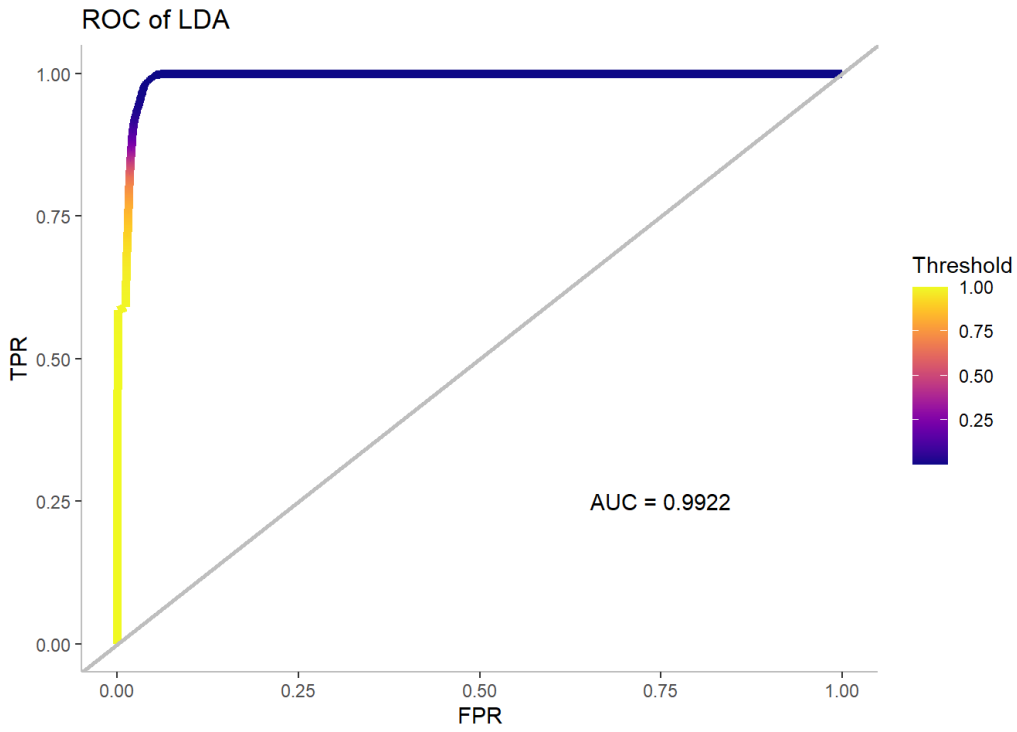
Statistic	Training	Holdout
Accuracy	0.9971	0.9916
TPR	0.9713	0.8675
FPR	0.0020	0.0075
PPV	0.9406	0.4570

kNN Decision Boundary





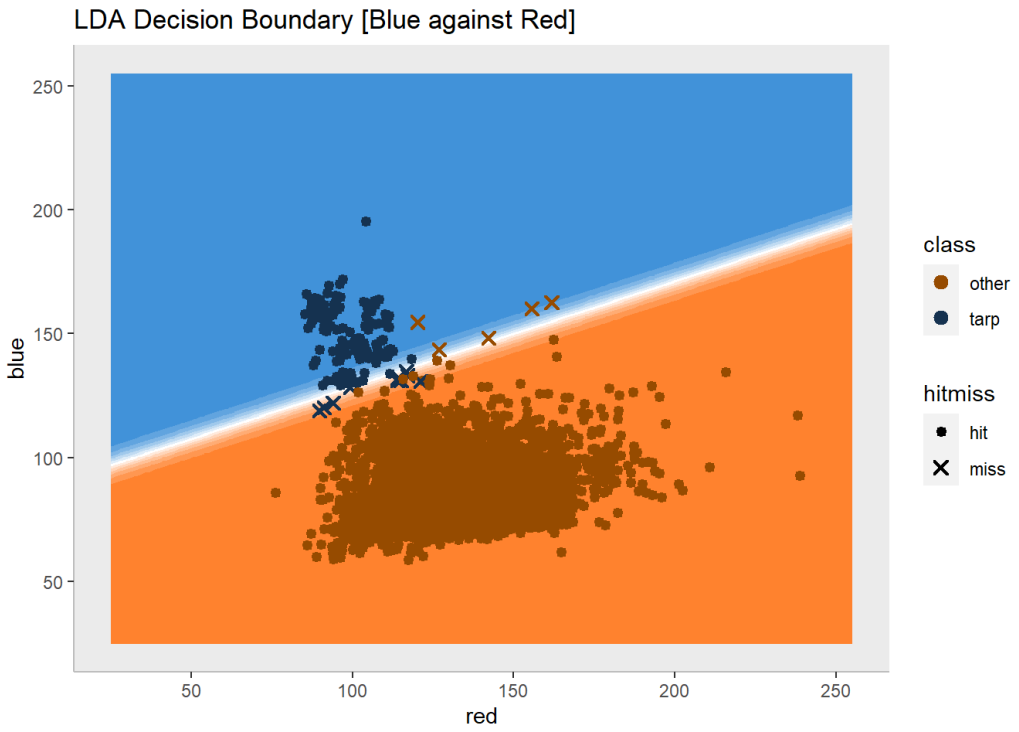
LDA Performance on Holdout Data

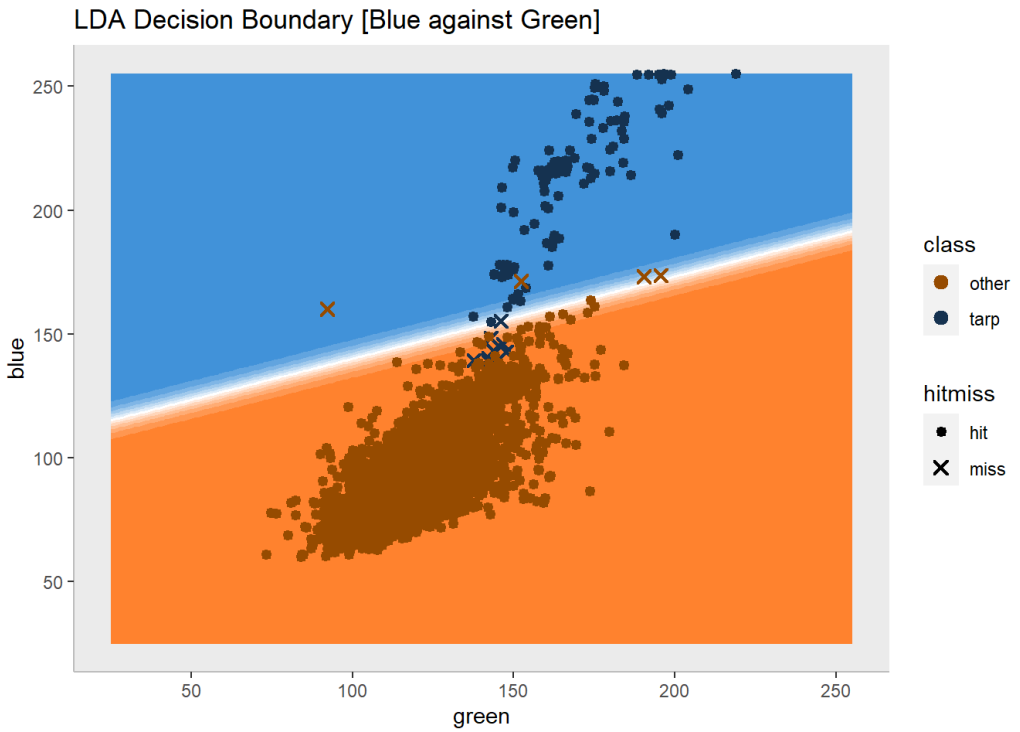


	Reference	
Prediction	Tarp	Other
Tarp	12154	34237
Other	2326	1955460

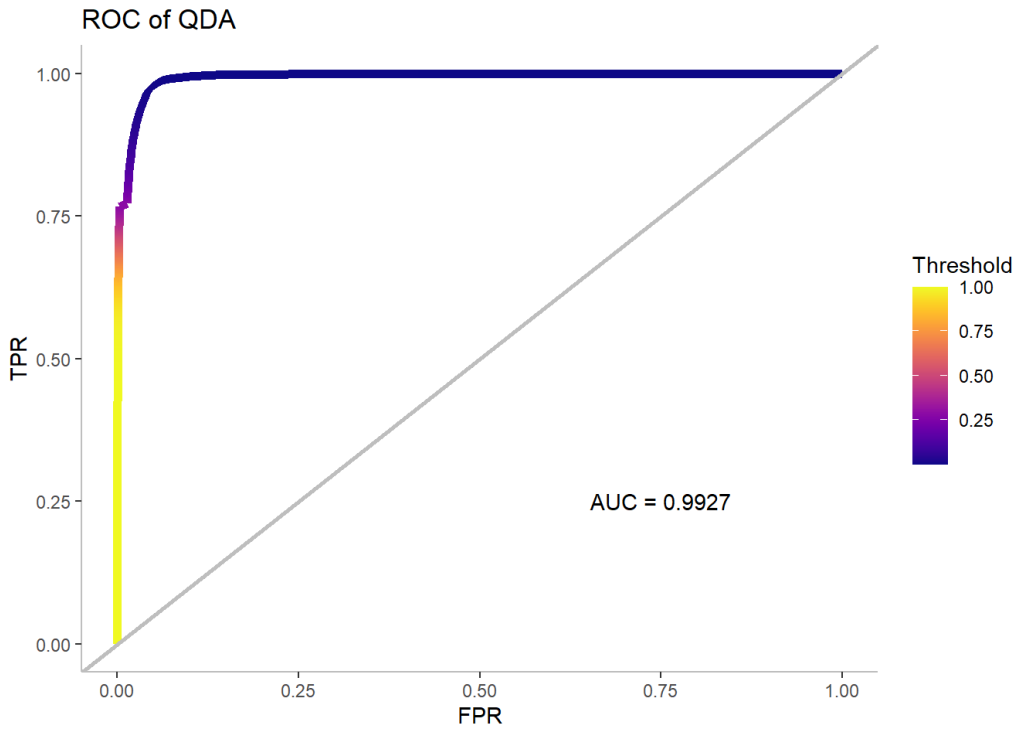
Statistic	Training	Holdout
Accuracy	0.9844	0.9818
TPR	0.8111	0.8394
FPR	0.0099	0.0172
PPV	0.7308	0.2620

LDA Decision Boundary





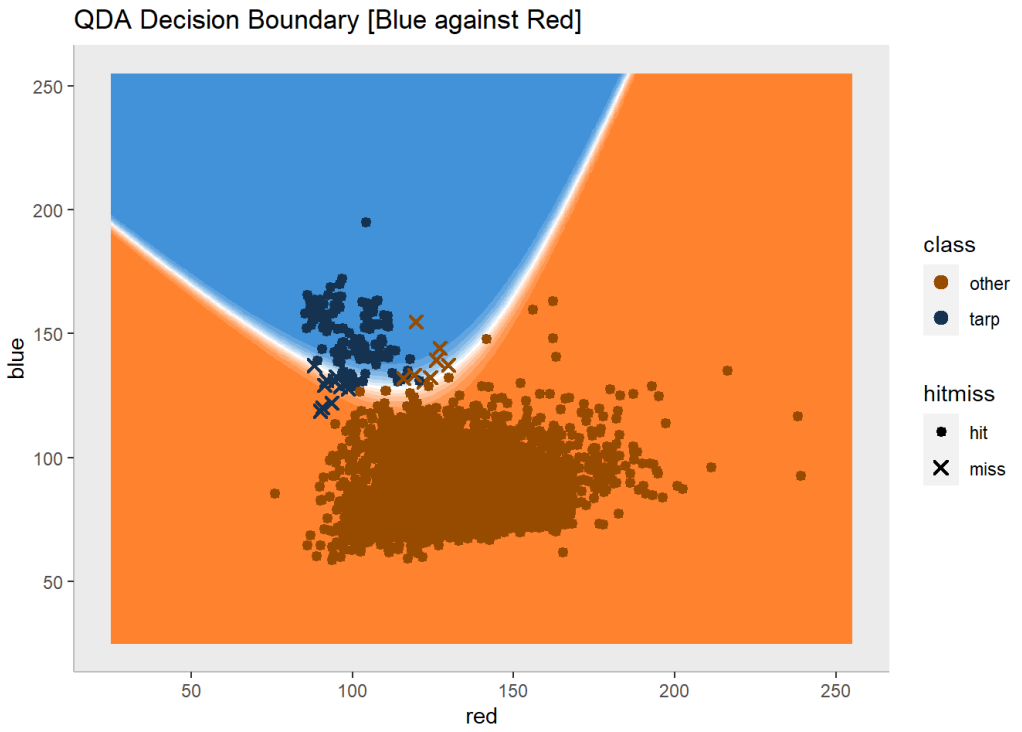
QDA Performance on Holdout Data

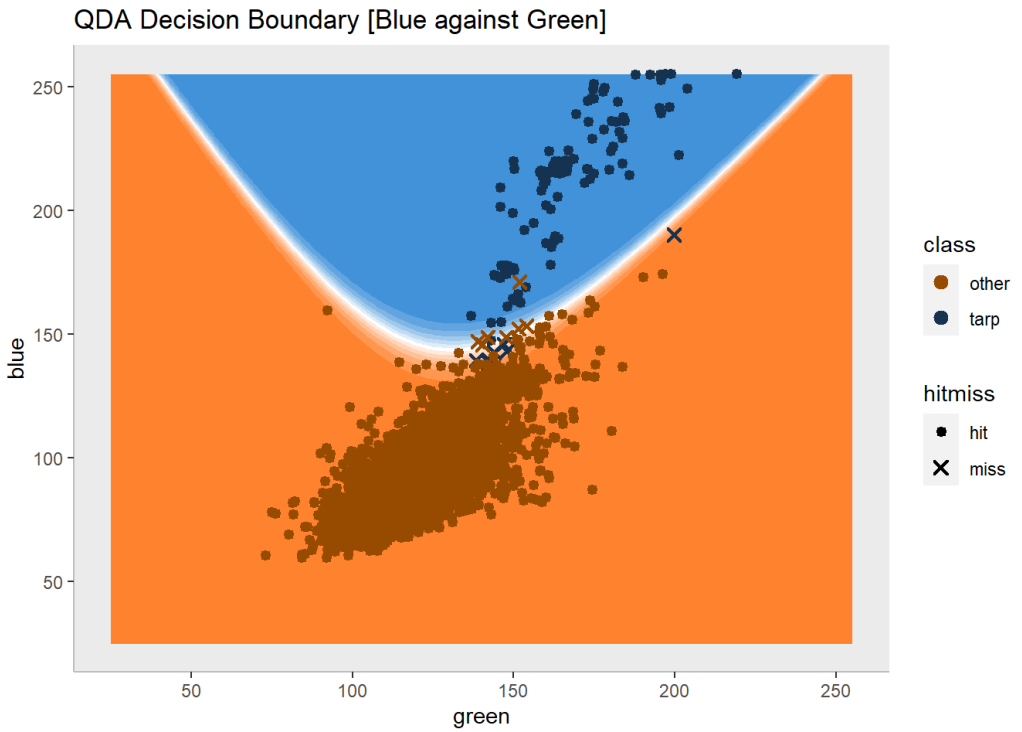


		Reference	
	Prediction	Tarp	Other
	Tarp	10341	3699
	Other	4139	1985998

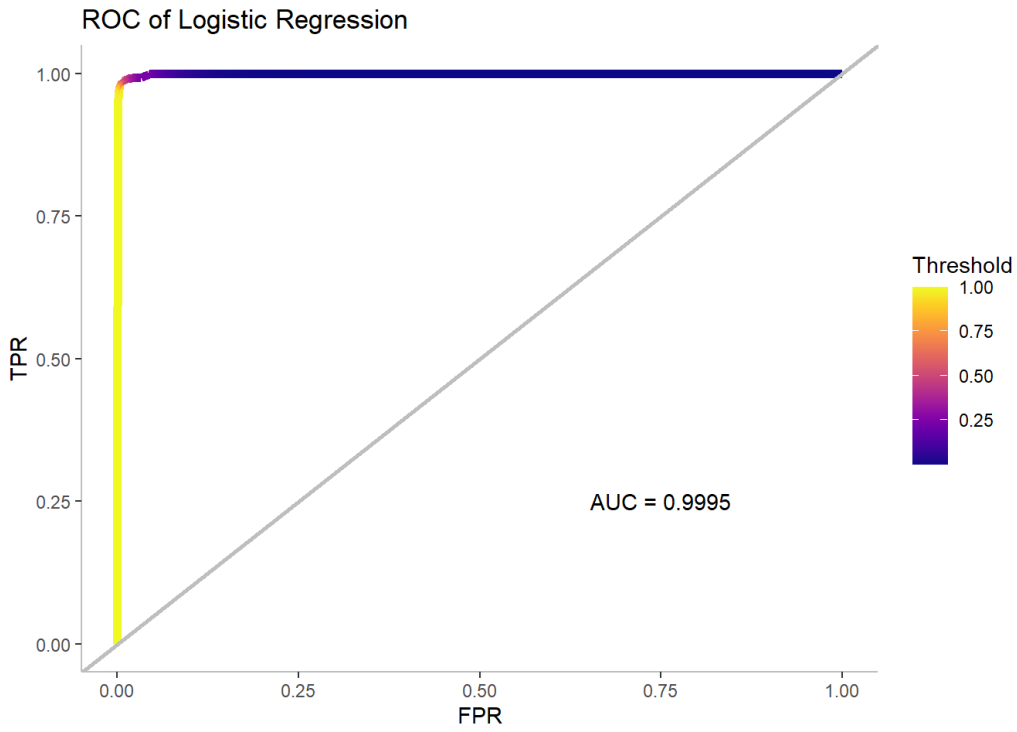
		Training	Holdout
Statistic	Accuracy	0.9950	0.9961
	TPR	0.8566	0.7142
	FPR	0.0004	0.0019
	PPV	0.9852	0.7365

QDA Decision Boundary





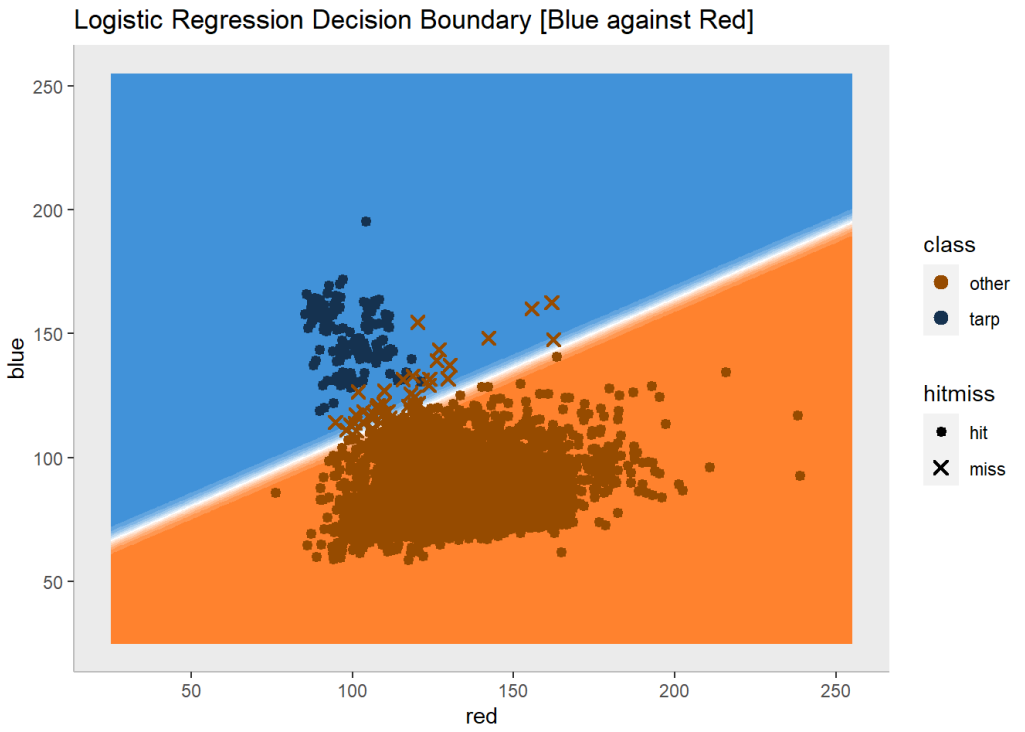
Logistic Regression Performance on Holdout Data

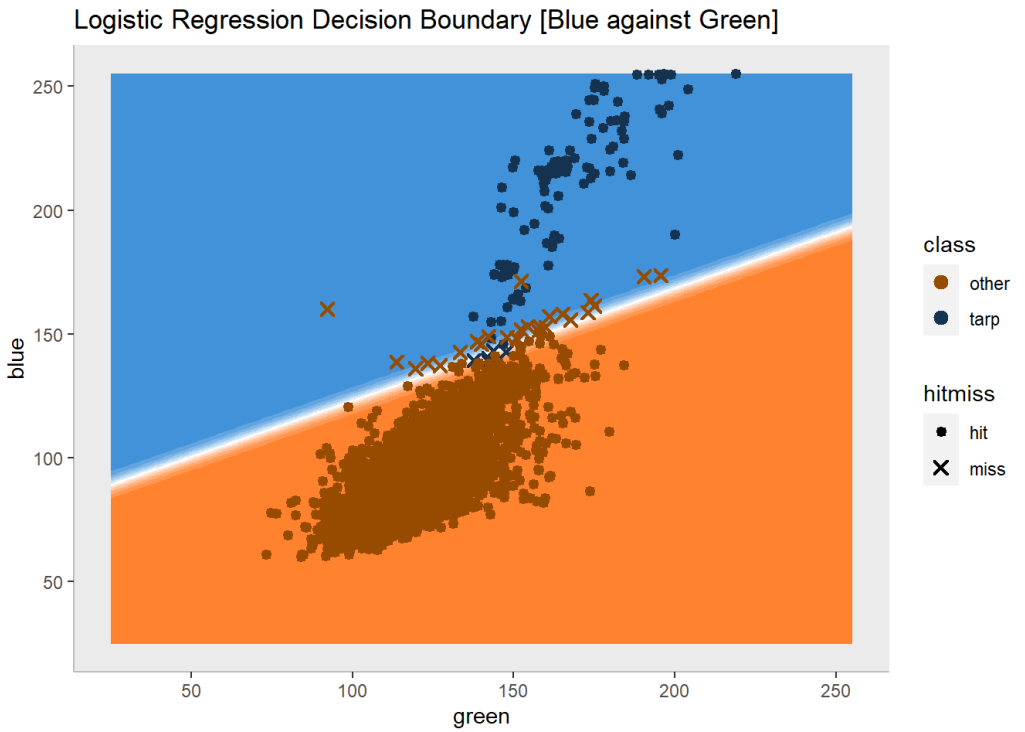


	Reference	
Prediction	Tarp	Other
Tarp	14321	20274
Other	159	1969423

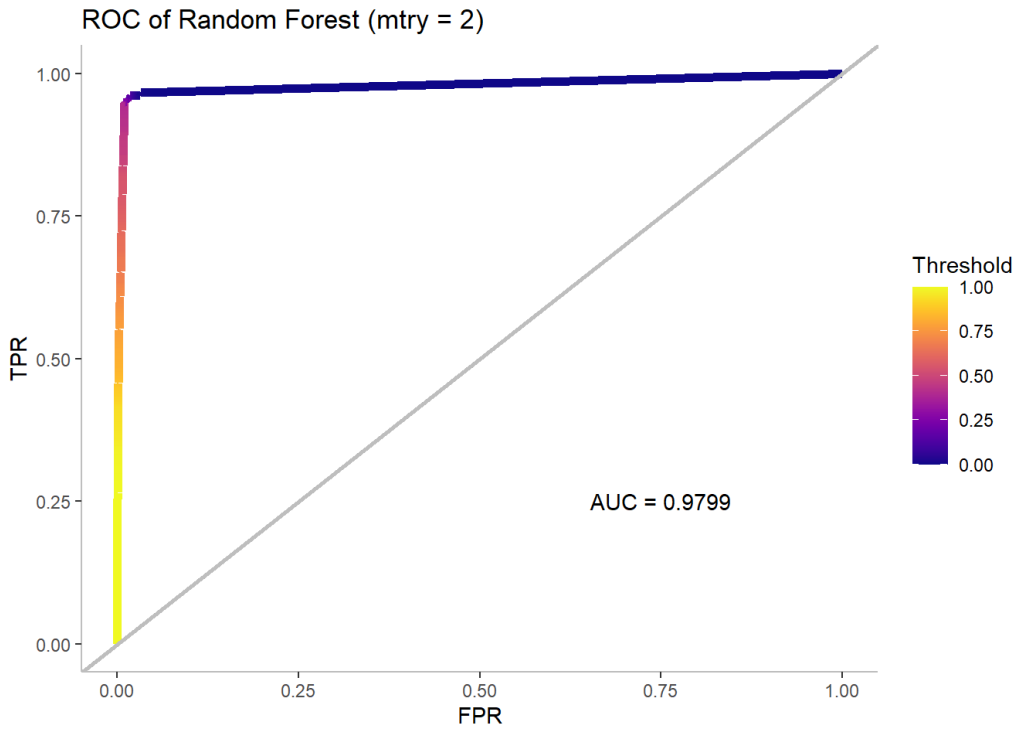
Statistic	Training	Holdout
Accuracy	0.9959	0.9898
TPR	0.9021	0.9890
FPR	0.0010	0.0102
PPV	0.9682	0.4140

Logistic Regression Decision Boundary





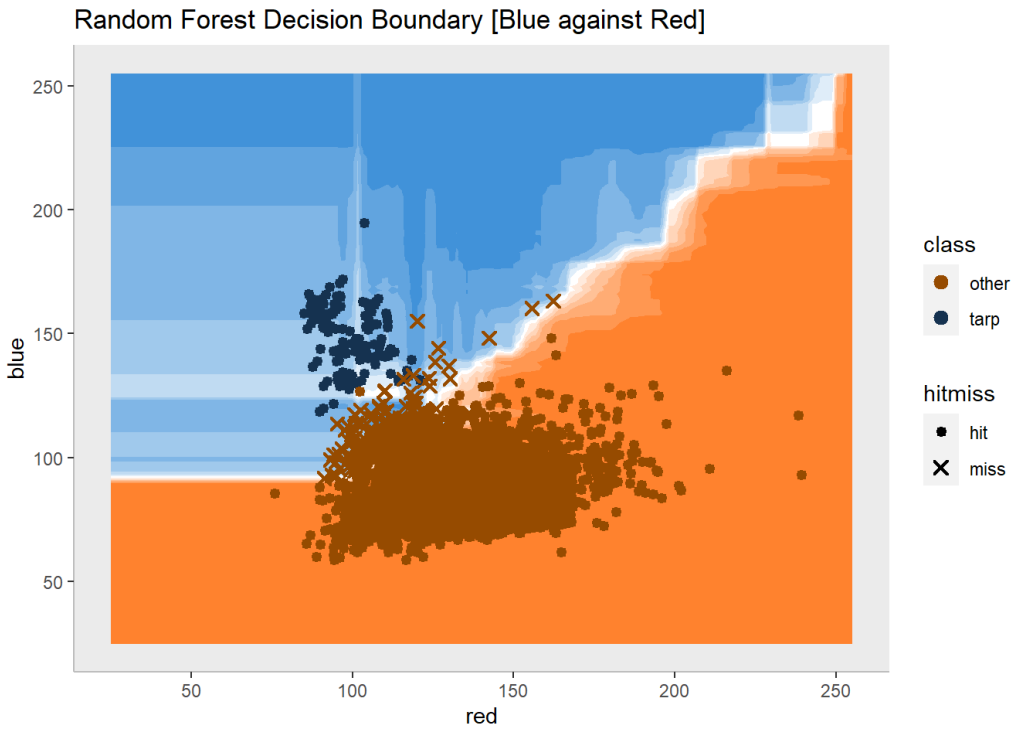
Random Forest Performance on Holdout Data

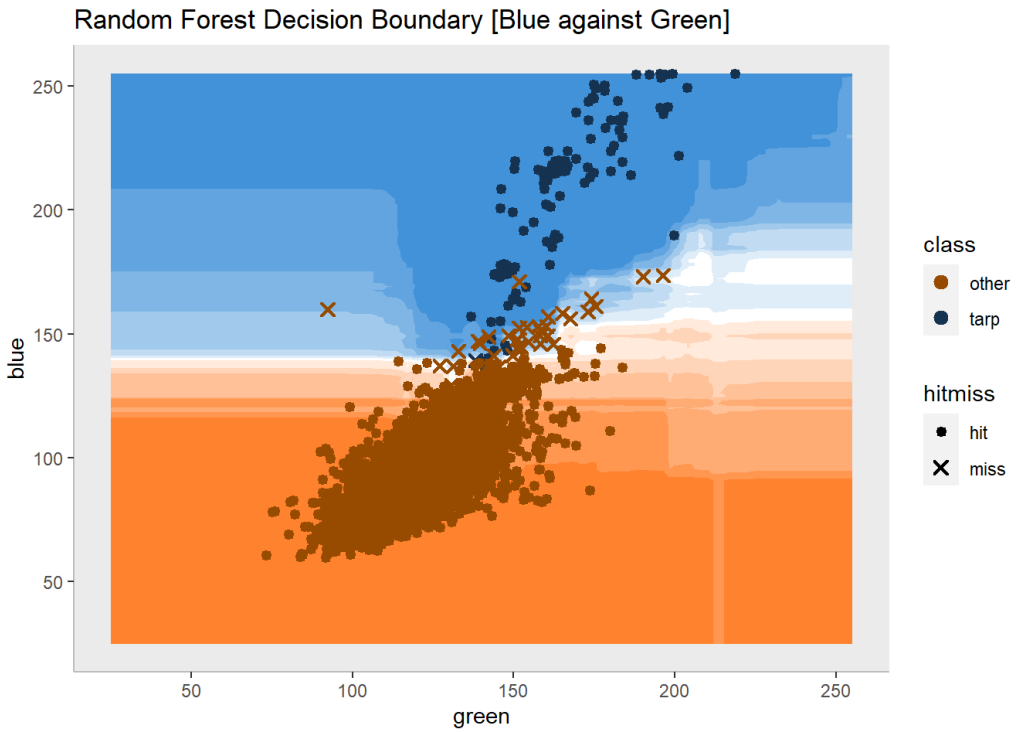


	Reference	
Prediction	Tarp	Other
Tarp	12144	15094
Other	2336	1974603

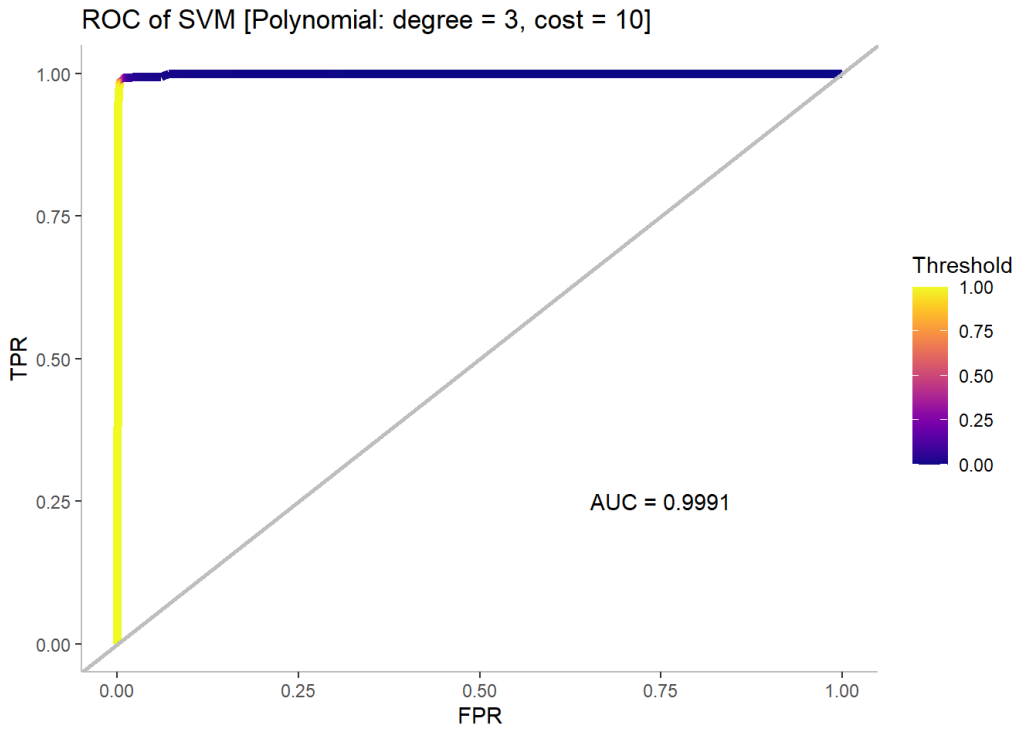
Statistic	Training	Holdout
Accuracy	0.9969	0.9913
TPR	0.9595	0.8387
FPR	0.0018	0.0076
PPV	0.9454	0.4458

Random Forest Decision Boundary





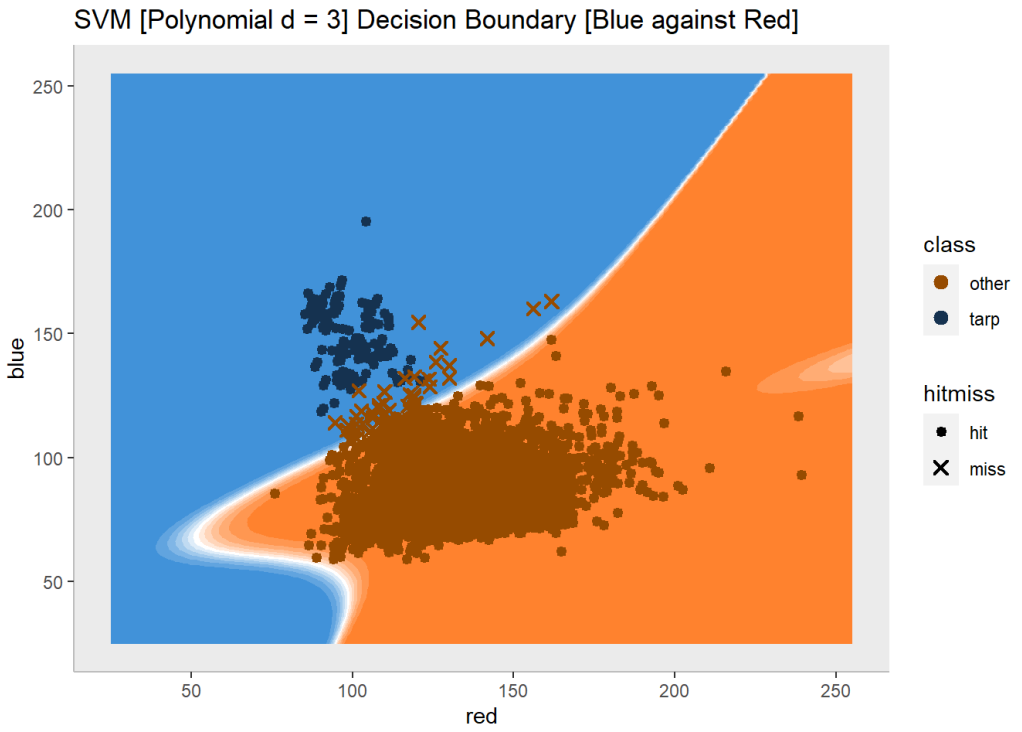
SVM Polynomial Performance on Holdout Data

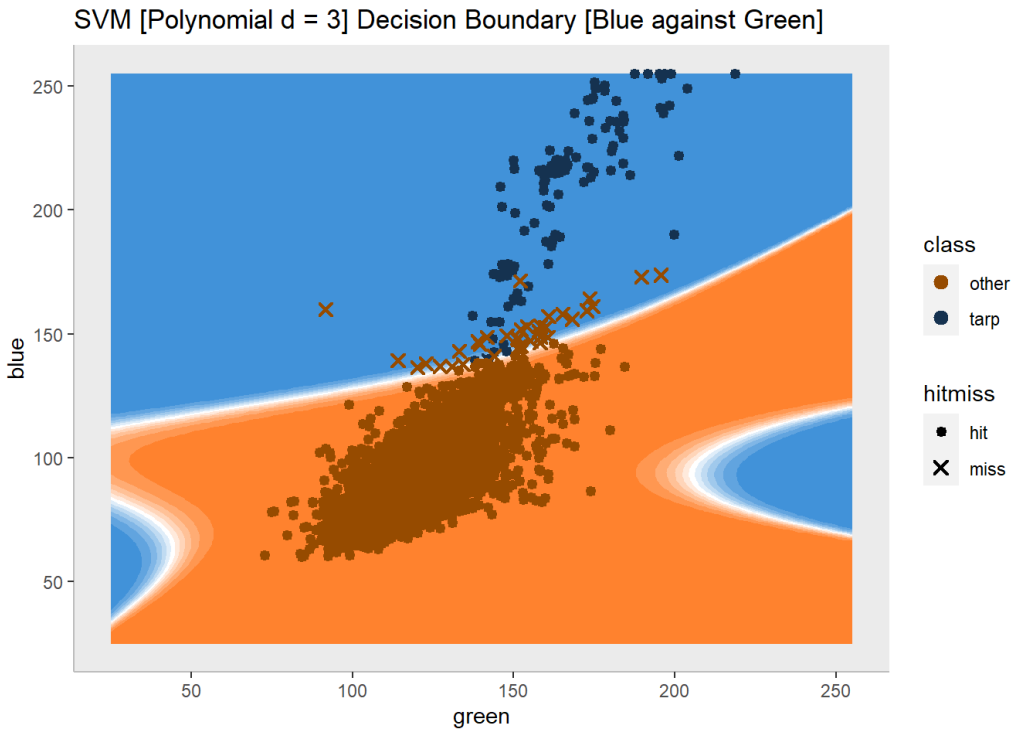


	Reference	
Prediction	Tarp	Other
Tarp	14347	11427
Other	133	1978270

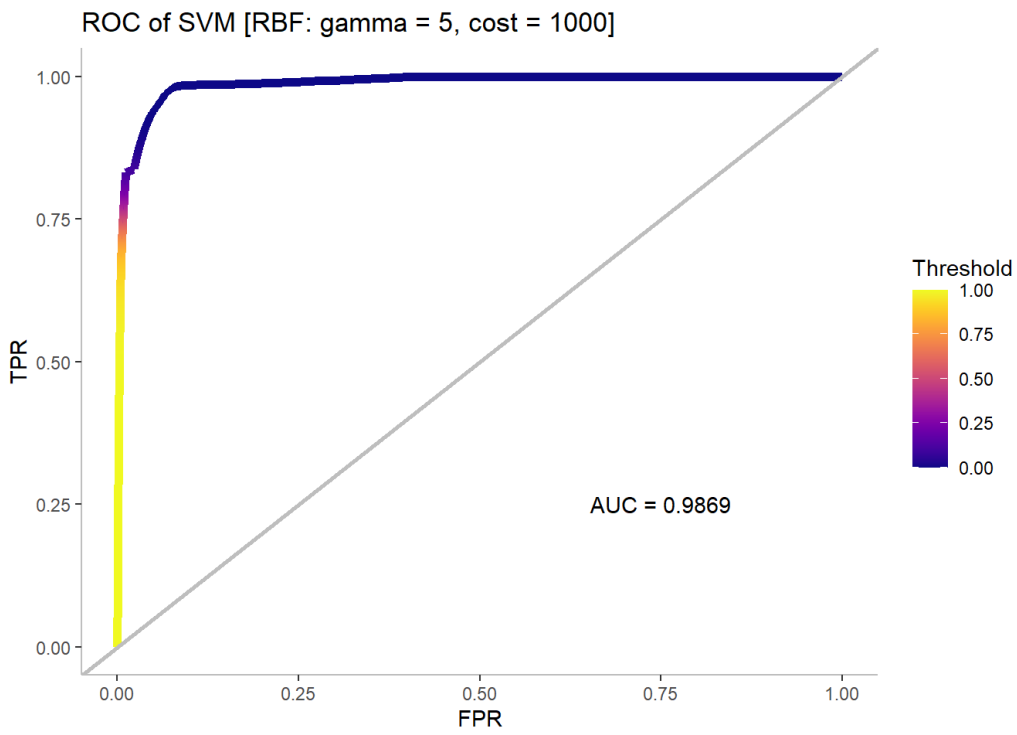
Statistic	Training	Holdout
Accuracy	0.9960	0.9942
TPR	0.9179	0.9908
FPR	0.0014	0.0057
PPV	0.9547	0.5566

SVM Polynomial Decision Boundary





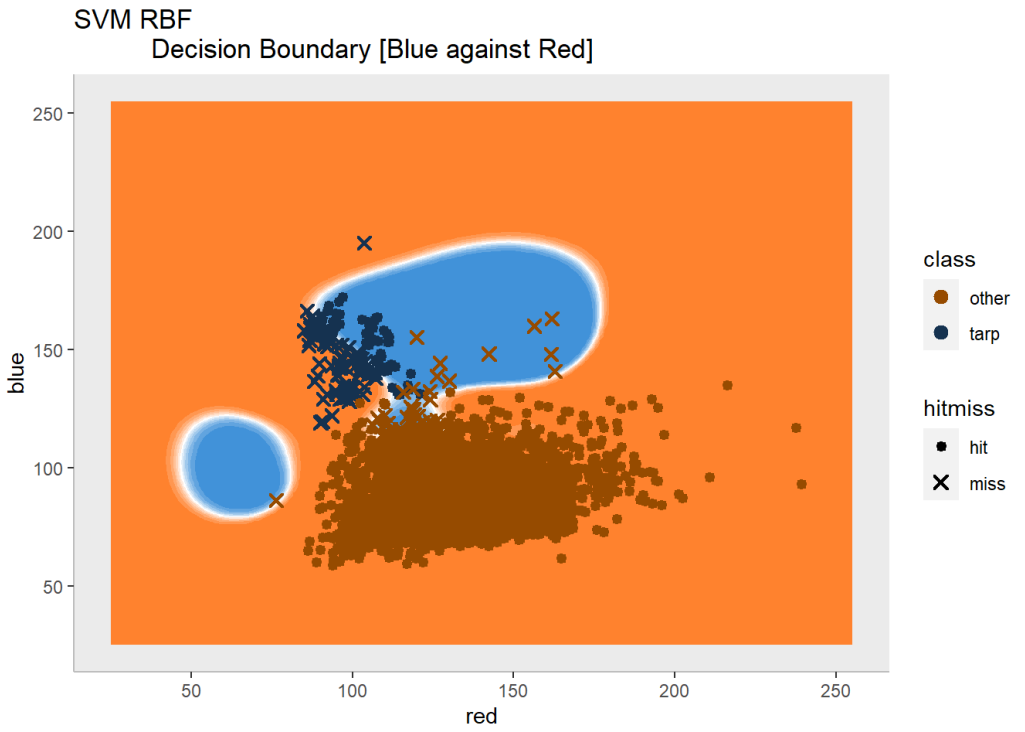
SVM RBF Performance on Holdout Data

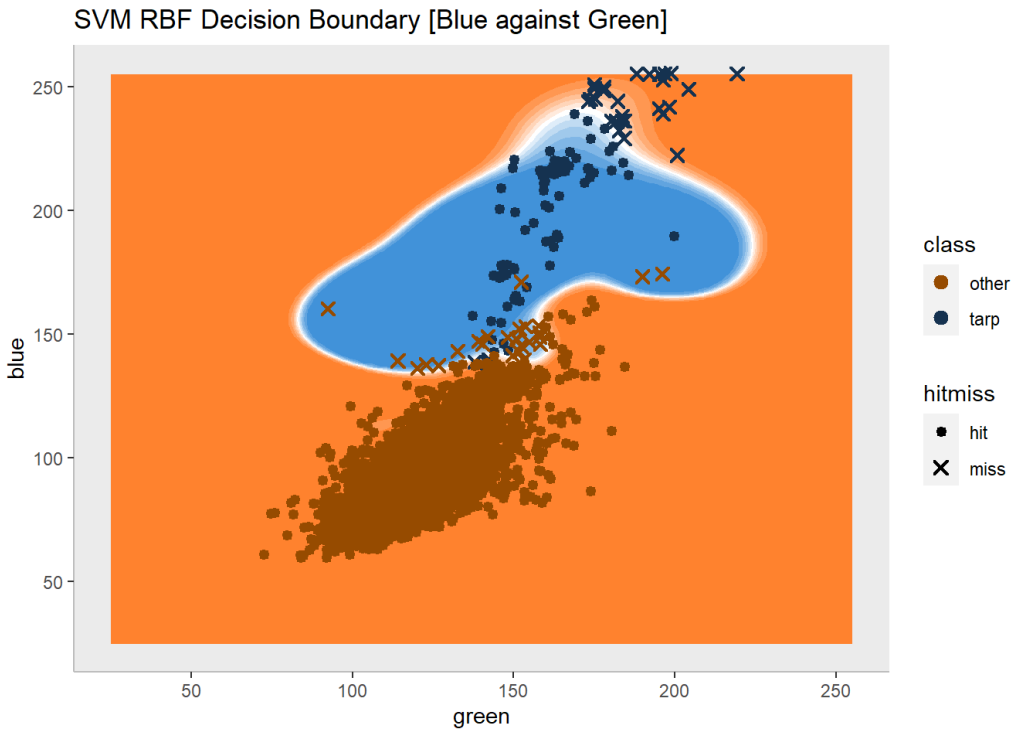


	Reference	
Prediction	Tarp	Other
Tarp	10885	15064
Other	3595	1974633

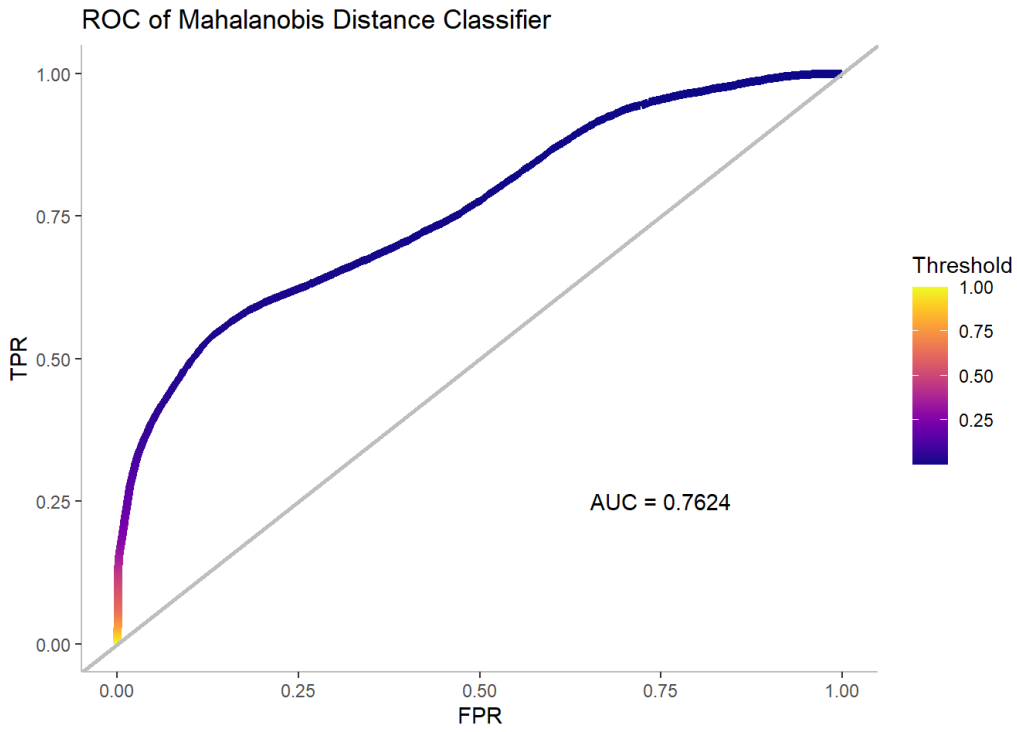
Statistic	Training	Holdout
Accuracy	0.9975	0.9907
TPR	0.9723	0.7517
FPR	0.0017	0.0076
PPV	0.9507	0.4195

SVM RBF Decision Boundary





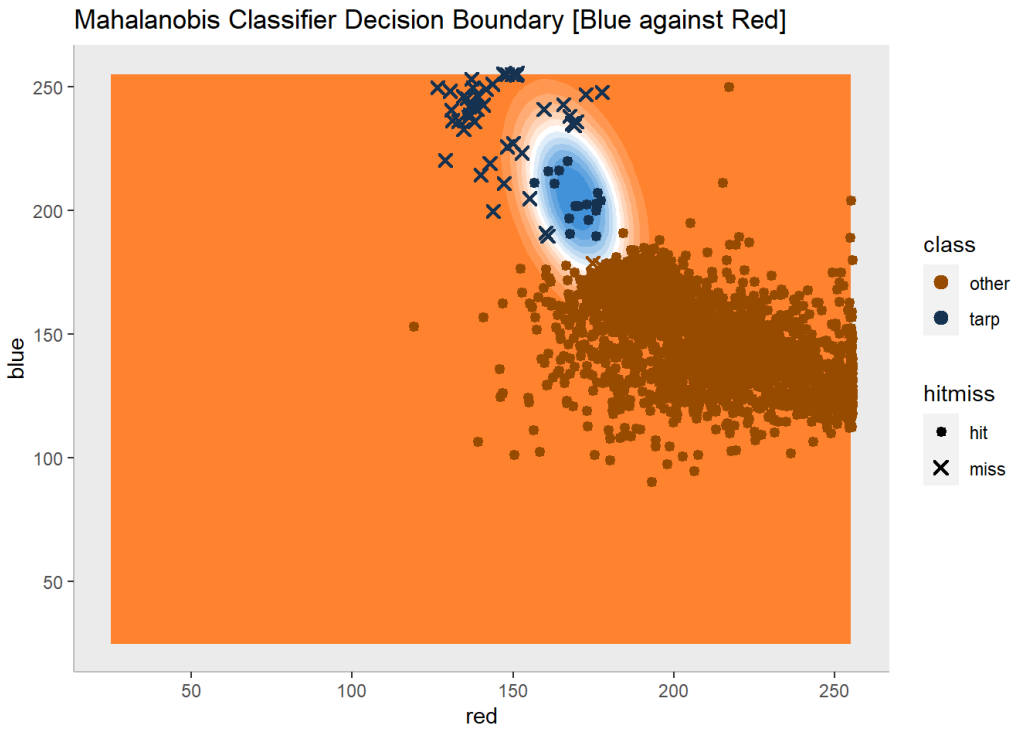
Mahalanobis Performance on Holdout Data

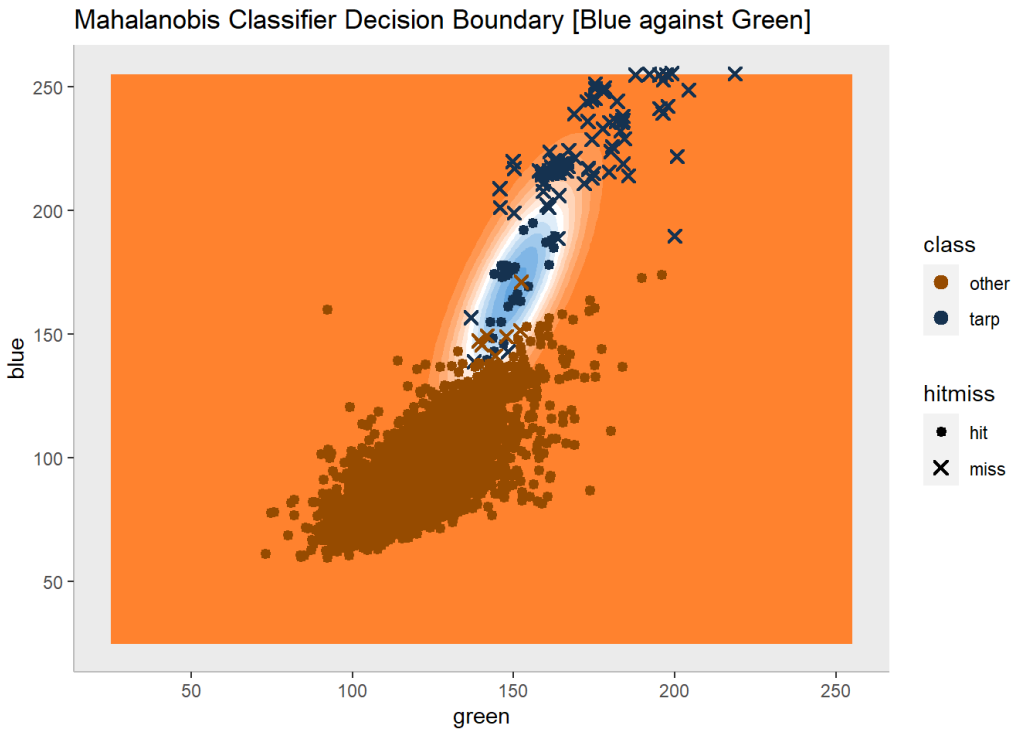


	Reference	
Prediction	Tarp	Other
Tarp	1398	581
Other	13082	1989116

Statistic	Training	Holdout
Accuracy	0.9883	0.9932
TPR	0.7369	0.0965
FPR	0.0034	0.0003
PPV	0.8785	0.7064

Mahalanobis Classifier Decision Boundary





Conclusions

1. The best performing method is Quadratic Discriminant Analysis (QDA).
   - On the holdout set, the QDA model has the highest overall accuracy as well as the highest positive predictive value (PPV).
   - QDA's high PPV comes at the cost of a relatively low True Positive Rate (TPR). Some may favor the SVM model with a polynomial kernel as it achieves the highest TPR.

2. PPV and TPR are the most useful metrics for this problem.
   - PPV tells us what fraction of predicted tarp pixels are actual tarps. The complement to PPV, false discovery rate, tells us how often a non-tarp pixel is classified as a tarp. In this circumstance, PPV is useful because it prevents rescue workers from traveling to misidentified locations.
   - TPR tells us the fraction of actual tarp pixels that were properly classified. If this value is small, it means the model is misclassifying many of the actual tarps and rescue workers may not be sent to those locations.
   - Personally, I favor PPV over TPR since the amount of tarp pixels in each dataset is small. It would be simple to draw a large decision boundary that captures all tarp pixels and achieves a perfect TPR. However, this boundary would also capture a significant number of non-tarp pixels.

Conclusions

3. For all models, the overall accuracy for both the training and holdout sets is high (>98%). However, other key metrics see a significant drop in performance when the models are applied to the holdout set.
   - High variance models (kNN, RF, and SVM-R) saw a significant drop in both TPR and PPV. This suggests that the models are overfit to the training data and do a poor job of identifying tarps on a generalized dataset.
   - Interestingly, high bias models (logistic regression and LDA) saw a modest increase in TPR when applied to the holdout set. This comes at the cost of a 2-fold increase in FPR for LDA and 10-fold increase in FPR for logistic regression, and a substantial decrease in PPV for both. This means that the high bias models are classifying too many pixels as tarps.

4. The training dataset is not an ideal representative sample of the holdout set.
   - The violin plots created in the exploratory phase of the analysis show that the RGB values in the training dataset skew higher on average than the holdout set. This essentially means that the images in the training set are brighter. If images are collected at various points throughout the day, the sun's position could affect the brightness of an image, and even the hue. Hypothetically, a model trained on bright, midday images might have difficulty identifying morning or evening images.
   - While this discrepancy in predictor distribution poses a problem for all models, the highly flexible models and the Mahalanobis classifier were the most affected. The flexible models and Mahalanobis classifier create a tight decision boundary around the cluster of tarp pixels in the training dataset. When generalized, they may miss tarps in images with different lighting conditions or camera settings.

Conclusions

5. Generally, the models with linear decisions boundaries (LDA and logistic regression) are poorly suited to this problem.
   - The linear decision boundary in 3D RGB space can be represented by a hyperplane, where one side of the plane classifies the pixels as tarps, and other side as non-tarps. In actuality, tarp pixels may only exist in a smaller cluster. Suppose, a small body of water is present in an image. To the human eye, water and tarps are visually distinct. However, they may exist on the same side of a linear decision boundary in RGB space. Linear classifiers are likely too rigid.

6. The Mahalanobis distance classifier offers overall accuracy that is comparable to the more sophisticated models. However, it is the worst—by far—at identifying tarps in the holdout set.
   - The TPR of the Mahalanobis classifier on the holdout set was only 0.0965, far below the performance on the training set. This method creates a small sphere (or ellipsoid in Euclidean coordinates) around the mean of the tarps in the training set. Points inside the sphere are classified as tarps. The population mean of tarps is likely different from the training sample mean. Thus, the classifier is likely to miss any tarps that are far from the training sample mean.
   - I conclude that the Mahalanobis classifier is sensitive to training sample. Fortunately, it is a quick method to train and would likely benefit from a larger sample.

Conclusions

7. Except for the Mahalanobis classifier, each model had a high AUC (>0.96) on the holdout set.
   - The high values tell us that the models are generally good at separating the two classes. However, AUC is not the most useful metric for distinguishing between models in this scenario where the number of non-tarp observations vastly exceeds the number of tarp observation. A relatively crude classifier, like logistic regression, can achieve a high AUC by over-classifying observations as tarps.
   - Logistic regression has a high TPR since it correctly classifies most of the true tarps. Unfortunately, it also misclassifies many non-tarps as tarps. The FPR remains low due to the large number of non-tarp observations that are correctly classified. The logistic regression model has the highest AUC, yet it misclassifies far too many non-tarps to be of value.

8. Another advantage of QDA over other models is speed.
   - QDA is substantially quicker to train than kNN, random forest, and SVM. Additionally, predictions can be made faster with QDA than with kNN. In this emergency scenario, where time is a factor, a QDA may be preferred to other candidates.

Improvements

I outline some possible improvements and extensions of this analysis.

- Transform RGB values to HSB (hue, saturation, and brightness) to be included as features
- Test more sophisticated models like gradient boosting and neural networks
- Project feature space onto principal components to decorrelate red, green, and blue predictors

References

[1] Aardt, Jan van et al. “Geospatial Disaster Response during the Haiti Earthquake: A Case Study Spanning Airborne Deployment, Data Collection, Transfer, Processing, and Dissemination.” *Photogrammetric Engineering and Remote Sensing* 77 (2011): 943-952.

[2] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. An Introduction to Statistical Learning : with Applications in R. New York :Springer, 2013.