From the Sea to Statistics: Using Machine Learning to Predict Sea Turtle Nesting Patterns

Angelina Scamardo
Advisors: Dr. Samantha Seals & Dr. Phillip Schmutz

August 11, 2025

Outline

Data Background

- Loggerhead turtles nest across large areas on Pensacola beach
- How are non-nested areas evaluated if no physical turtle's nest exists to gather data from?
- Pseudo-Absence (PA) data aids us in evaluating the nesting patterns of the turtles when no physical data is present.
  - PA values represent a **potential** nest observation based on background environmental data.
  - These randomly generated/"fake" values allow us to determine if the non-nested environment is similar to or different from the known nested environment.

Introduction

- We will evaluate 3 different PA observation to nested observation ratios - 2:1, 5:1, and 10:1 (all recorded in 2020)
  - Purpose: develop different data sets to fill out the unknown environment region/area.
  - Higher ratios = more PA nests = more information of area
  - Increasing number of PAs/0s in the data set makes nesting/1s a "rare event"
- Note: Our data sets are relatively small:

Ratio	**Nested**	**PA**	**Total**
2:1	17	34	51
5:1	17	85	102
10:1	17	170	187

Literature Review

- Environmental research has been done regarding PA point ratios and ML modeling - specifically using Random Forest (RF) Modeling
- Culver and colleagues determined as the ratio decreases, the accuracy of predictions for nest presence (sensitivity) increases and the accuracy for nest absence (specificity) decreases. [1]
- Elevation and crawl distance were the strongest predictors of Kemp's ridley sea turtle nest presence [1]
- Turtles avoided nesting on beaches with extreme geomorphic features (i.e. steep slopes, wide/flat areas) [1]

Research Questions

- Are we able to predict whether or not a loggerhead turtle will nest (based on limited data) using ML, as done in previous studies?
- Which ML model can most accurately predict nesting vs. non-nesting locations for loggerhead turtles, RF or SVM?

Variables Considered

- Nested status:

$$y = \begin{cases} 0 & \text{if pseudo-absence nest} \\ 1 & \text{if observed nest} \end{cases}$$

- Nest elevation: Mean seal level (MSL) elevation of the nest.
- Beach slope: Slope angle of the dry beach [nest to high tide]
- Foreshore slope: Slope angle of the wet beach [high (0.25 meters MSL) to low tide (-0.14 m MSL)]
- Nest distance: Crawl distance
- Dune height: MSL elevation of highest the dune featured at the nest location.

Methods

- **What is an SVM?**
  - Support Vector Machine (SVM) is a supervised ML model [2]
  - Finds the best boundary to separate classes using kernel tricks for non-linear data [2]
    - Decision boundary is determined by solving an optimization problem in terms of the kernel [2]
  - Kernels are functions that map data into a higher dimension where linear separation is possible (does not transform data itself) [2]
- **Why use SVM?** Our data has complex, nonlinear relationships - SVMs handle this well.

Methods

SVM Steps:

Use RBF kernel to separate data (most commonly used for non-linear data). RBF Kernel Formula: [2]

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right)$$

- $\|\mathbf{x} - \mathbf{z}\|^2$ = Euclidean distance
- γ = Controls how many data points are influenced by a single data point
  - High γ: Small radius, complex/curvy boundary (risks overfitting).
  - Low γ: Large radius, smoother boundary (risks underfitting).
- $\exp(-\gamma \cdot \text{distance})$ = Similarity score — higher/closer to 1 if x and z are close together, lower/closer to 0 if far apart

Methods

- The SVM prediction function for non-linear data is used to decide what class a new data point belongs to: [2]

$$f(\vec{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b$$

where:

- α_i: Learned weights during optimization problem (non-zero values = support vectors)
- y_i: Class label (± 1)
- $K(\vec{x}_i, \vec{x})$: Kernel between training point $\vec{x}_i$ and test point $\vec{x}$

- **What is a RF?**
  - Random forest (RF) is a supervised ML model
  - Builds decision trees and combines outputs for more accurate/stable predictions. [3]
- **Why use RF?**
  - Can handle imbalanced data better than SVMs
  - Reduces bias [3]
  - Less prone to overfitting [3]
  - Used in previous environmental studies

- **RF Steps:**
  - **Gini Impurity:** Measures how pure a node is/how mixed the classes are. Chooses splits that minimize impurity. [3]

$$G = 1 - \sum_{i=1}^{C} p_i^2$$

    where C = number of classes, and p_i^2 = proportion of samples belonging to class i in the node
  - **Averaging/Voting:** After all trees are built, RF returns majority class for classification [3]
  - **Bagging (Bootstrap Aggregating):** Sampling with replacement from training data. [3]

Methods

ML Approach:
Data Preparation:

- Cleaned data by removing NAs and converting dependent variables to factors. [5][6]
- Normalized numerical features (z-scores) for SVM; handled missing values with mean imputation after splitting. [5][6]
- Set random seed for reproducibility. [5]
- Split data into training/testing sets:
  - 70% training / 30% testing
  - 80% training / 20% testing

Methods

Model Training and Evaluation:

- **SVM:** Used radial basis function (RBF) kernel.
- **Random Forest:** Trained using 5-fold cross-validation.
  - Split the data into 5 equal parts, train the data with 1 part, and test on the remaining parts. [4]
  - Repeat 5 times, then average all test results. [4]
- For both models, evaluated using confusion matrix, accuracy, Kappa statistic, specificity, sensitivity, and balanced accuracy.

Methods

- **Confusion Matrices** provide a detailed breakdown of classification performance. [6][7]

$$\begin{bmatrix} \text{True Negatives (TN)} & \text{False Positives (FP)} \\ \text{False Negatives (FN)} & \text{True Positives (TP)} \end{bmatrix}$$

  - We are looking for ALL observations to be in TN and TP.
  - Observations in FP and FN indicate errors in model's predictions.

Methods

- **Accuracy** tells us how often the model makes correct predictions overall. [6][7]

$$\frac{TP + TN}{TP + TN + FP + FN}$$

- We want our accuracy to be as close to 1 (or 100%) as possible. Accuracies above 80% are considered "good" for this project.

- **Kappa Statistic** adjusts for agreement by chance/tells us how much better our model is compared to random guessing [6][7]
  - $\kappa = 0$: no better than random guessing
  - $\kappa = 1$: perfect agreement

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

where P_o is observed agreement (accuracy) and P_e is chance agreement, calculated by:

$$P_e = (P_{\text{nested}}^{\text{pred}} \cdot P_{\text{nested}}^{\text{actual}}) + (P_{\text{non-nested}}^{\text{pred}} \cdot P_{\text{non-nested}}^{\text{actual}})$$

Methods

- **Sensitivity** shows the proportion of actual positive cases that are correctly identified. Important if we overlook a positive/real nest observation. [6][7]

$$\frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- **Specificity** shows how well negatives are identified. Important if we predict a nest when there isn't one (an error) [6][7]

$$\frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

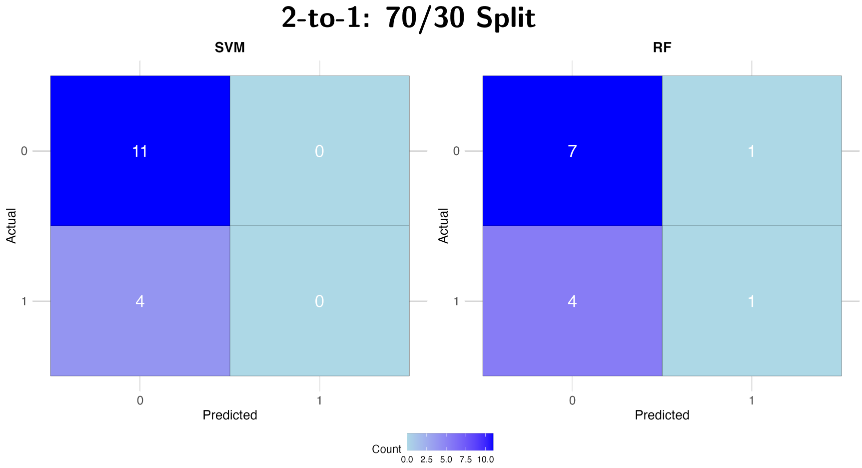
Methods

- **Balanced Accuracy** is useful to evaluate since our classes are imbalanced, and since sensitivity favors predicting nested and specificity favors predicting non-nested. [6][7]

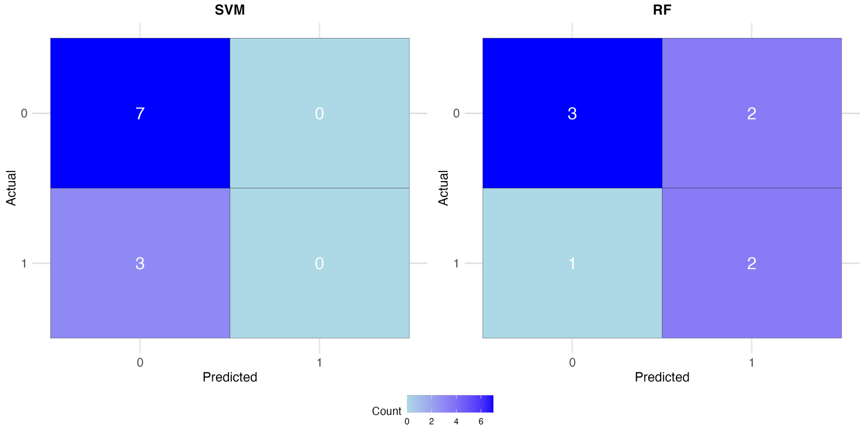
$$\frac{1}{2}\left(\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}+\frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}\right)$$

- **How does this differ from accuracy?**
  - Accuracy measures overall correctness, but can be biased toward majority class [6][7]
  - Balanced accuracy measures overall correctness as well, but gives equal importance to both classes [6][7]
  - If our model predicts all non-nested, accuracy will be high and balanced accuracy will be low
  - If our model predicts both classes equally, both accuracy and balanced accuracy will be high

Results

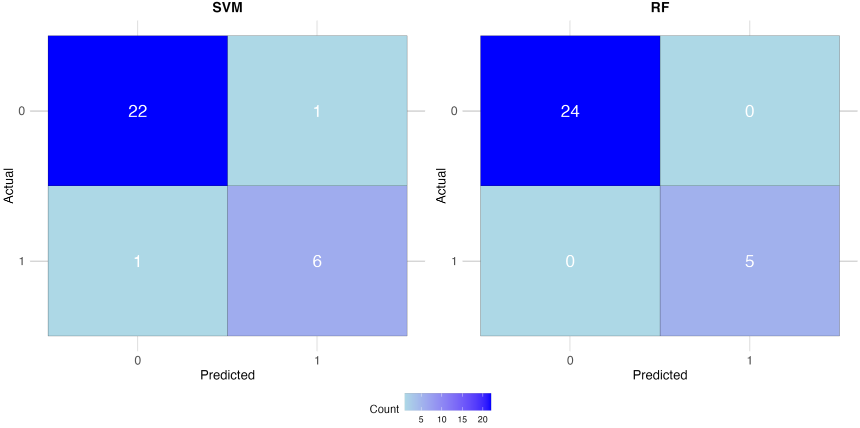

2-to-1: 70/30 Split
SVM
RF
Actual
Predicted
0
1
11
0
4
0
7
1
4
1
Count
0.0
2.5
5.0
7.5
10.0

2-to-1: 80/20 Split

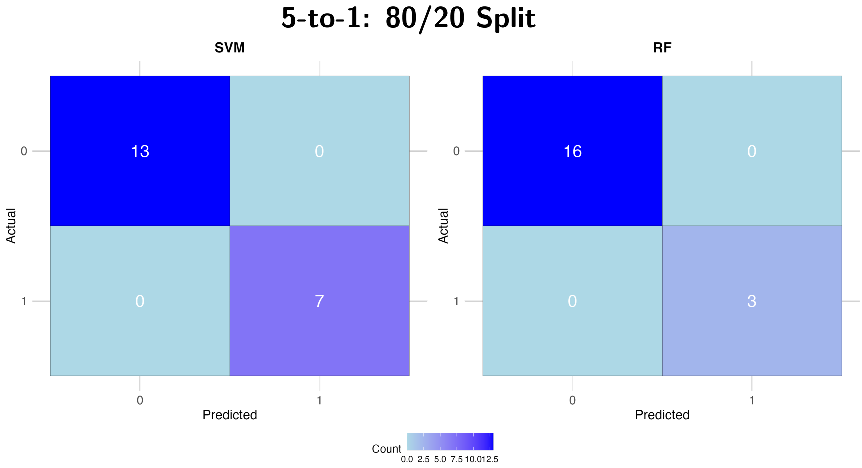


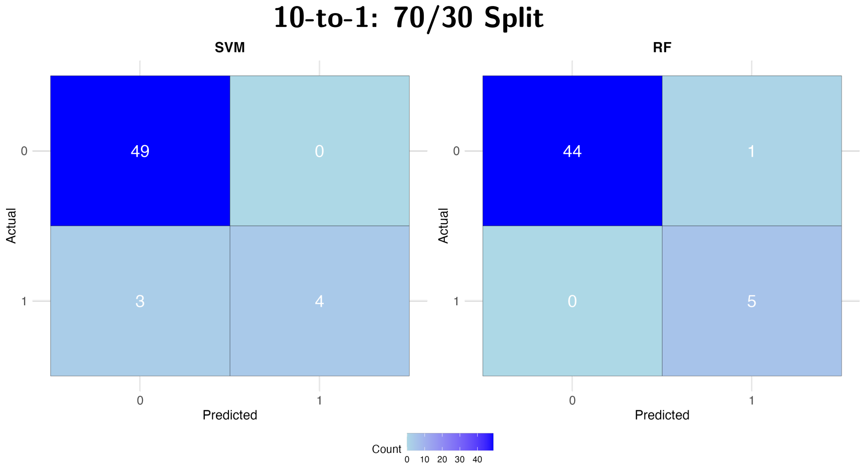
Results

5-to-1: 70/30 Split



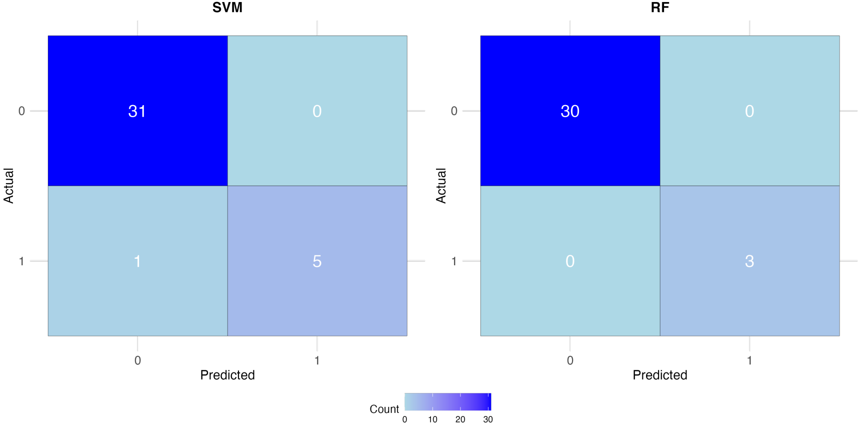
Results


5-to-1: 80/20 Split
SVM
RF
Actual
Predicted
0
1
13
0
0
7
16
0
0
3
Count
0.0 2.5 5.0 7.5 10.0 12.5


10-to-1: 70/30 Split
SVM
RF
Actual
Predicted
0
1
49
0
3
4
44
1
0
5
Count
0
10
20
30
40

Results

10-to-1: 80/20 Split



Results

Accuracy:

	SVM		RF	
Ratio	**70/30**	**80/20**	**70/30**	**80/20**
2:1	73.33	70	61.54	62.5
5:1	93.33	100	100	100
10:1	94.64	97.3	98	100

- Best performing:
  - 2-to-1: SVM 70/30 Split
  - 5-to-1: SVM 80/20 Split and RF (Both splits)
  - 10-to-1: SVM 80/20 Split and RF 80/20 Split

Results

Kappa:

	SVM		RF	
Ratio	**70/30**	**80/20**	**70/30**	**80/20**
2:1	0.00	0.00	0.08	0.25
5:1	0.81	1.00	1.00	1.00
10:1	0.70	0.89	0.90	1.00

- Best performing:
  - 2-to-1: RF 80/20 Split
  - 5-to-1: SVM 80/20 Split and RF (Both Splits)
  - 10-to-1: RF 80/20 Split

Results

Sensitivity:

	SVM		**RF**	
Ratio	**70/30**	**80/20**	**70/30**	**80/20**
2:1	1.00	1.00	0.88	0.60
5:1	0.96	1.00	1.00	1.00
10:1	1.00	1.00	0.98	1.00

- Best performing:
  - 2-to-1: SVM (Both Splits)
  - 5-to-1: SVM 80/20 Split and RF (Both Splits)
  - 10-to-1: SVM (Both Splits) and RF 80/20 Split

Results

Specificity:

	SVM		RF	
Ratio	**70/30**	**80/20**	**70/30**	**80/20**
2:1	0.00	0.00	0.20	0.67
5:1	0.86	1.00	1.00	1.00
10:1	0.57	0.83	1.00	1.00

- Best performing:
  - 2-to-1: RF 80/20 Split
  - 5-to-1: SVM 80/20 Split and RF (Both Splits)
  - 10-to-1: RF (Both Splits)

Results

Balanced Accuracy:

	SVM		RF	
Ratio	**70/30**	**80/20**	**70/30**	**80/20**
2:1	0.50	0.50	0.54	0.63
5:1	0.91	1.00	1.00	1.00
10:1	0.79	0.92	0.99	1.00

- Best performing:
  - 2-to-1: RF 80/20 Split
  - 5-to-1: SVM 80/20 Split and RF (Both Splits)
  - 10-to-1: RF 80/20 Split

Conclusions

Observations on SVM Performance:

- Performs well, but only if we have enough data points.
  - SVMs are especially sensitive to small and unbalanced data sets.
  - The 2:1 ratio had the smallest sample size of all three data sets.
  - This is why our SVM model did not make consistently accurate predictions.
- SVMs can be relied on for our data.

Conclusions

Observations on RF Performance:

- Performed better than SVMs.
- More consistently accurate (higher accuracies), higher kappa values, higher balanced accuracies across all ratios
- Confusion matrices had fewer errors overall.
- RF is preferred over SVMs for our data.

Future Study

Future plans for data analysis:

- Analyze other data splits (60/40, 75/25, etc.)
- Perform hyperparameter tuning to control training process
  - Want to select the best model settings (e.g., tree depth, kernel type) to improve model performance and prevent overfitting or underfitting.
- Our models will have more data to train and test on soon - data generation is ongoing.

References

1. Culver, M., Gibeaut, J. C., Shaver, D. J., Tissot, P., & Starek, M. (2020). *Using Lidar data to assess the relationship between beach geomorphology and Kemp's ridley (Lepidochelys kempii) nest site selection along Padre Island, TX, United States*. *Frontiers in Marine Science*, 7, Article 214. https://doi.org/10.3389/fmars.2020.00214
2. Scikit-Learn developers. (n.d.). *Support vector machines*. In *Scikit-Learn: Machine Learning in Python (User Guide)*. https://scikit-learn.org/stable/modules/svm.html
3. Breiman, L. (2001). *Random forests*. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324
4. RISHU_MISHRA. (2025). *K-fold cross validation in R programming*. GeeksforGeeks. https://www.geeksforgeeks.org/r-language/k-fold-cross-validation-in-r-programming/
5. Grolemund, G., & Wickham, H. (2017). *R for data science*. O'Reilly Media. https://r4ds.had.co.nz/index.html
6. Gatto, L. (2020, February 28). *An introduction to machine learning with R*. ~/. https://lgatto.github.io/IntroMachineLearningWithR/index.html
7. Géron, A. (n.d.-b). *Hands-on machine learning with scikit-learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Online Learning. https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/

Questions? Thank you!