A fully connected neural network modeling breakpoints in time series

Amy Pitts

Marist College

Wednesday, May 8, 2019

Introduction

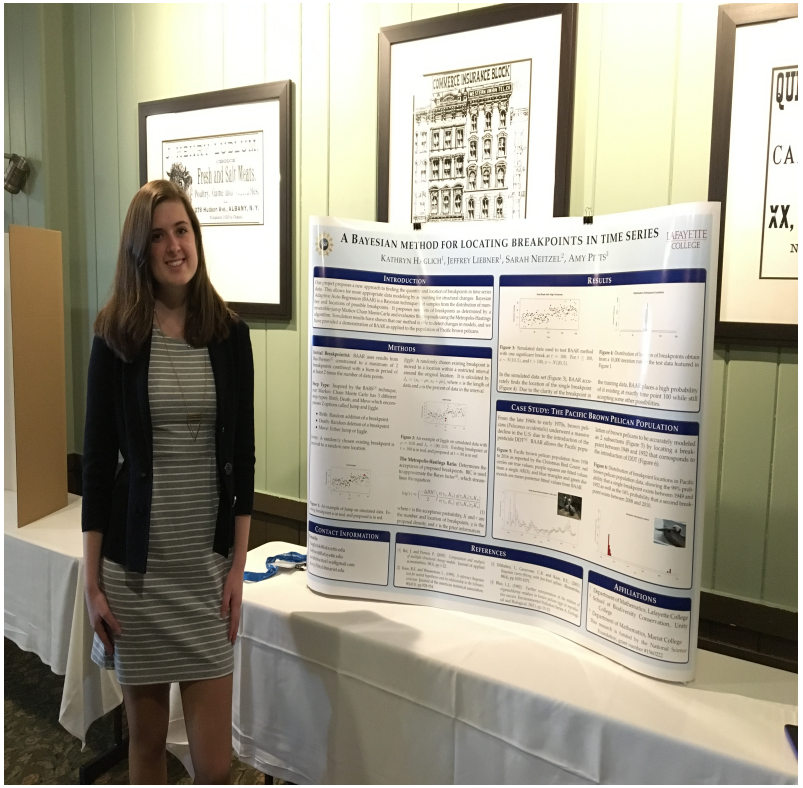
Goal: I created a dense fully connected neural network to be able to predict location of breakpoints in time series, learning off of simulated data.

Overview:

1. Initial Set up
2. Activation
3. Calculation of Loss
4. Gradient Descent
5. Introduction of Drop out
6. Output / Results Generated

Inspiration

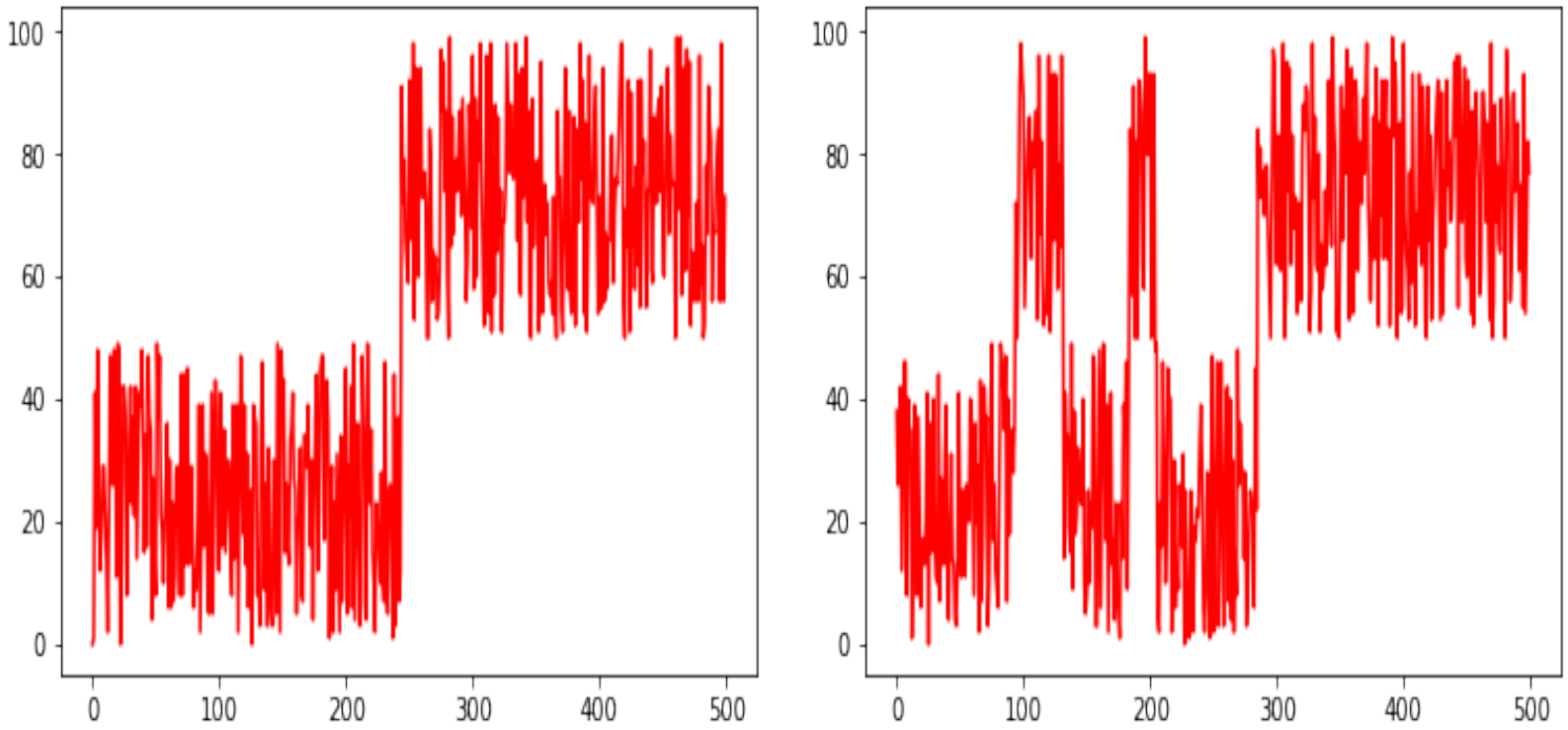
(a) Lafayette REU



(b) Poster Pres at NYCWiC

Figure 1: Project Inspiration

Breakpoint Example



(a) One Breakpoint

(b) Five Breakpoints

Figure 2: Simulated Time Series Examples

What is a Breakpoint Exactly!

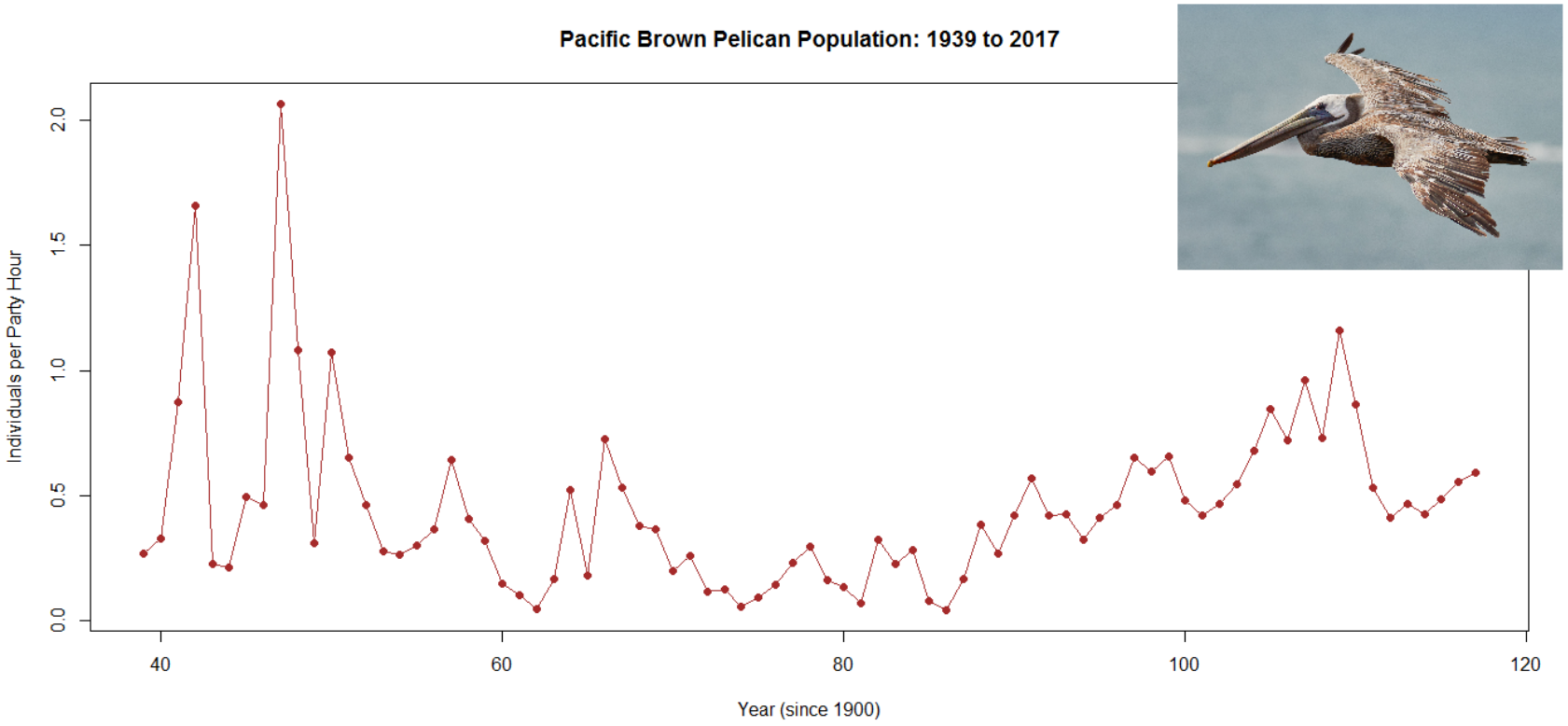
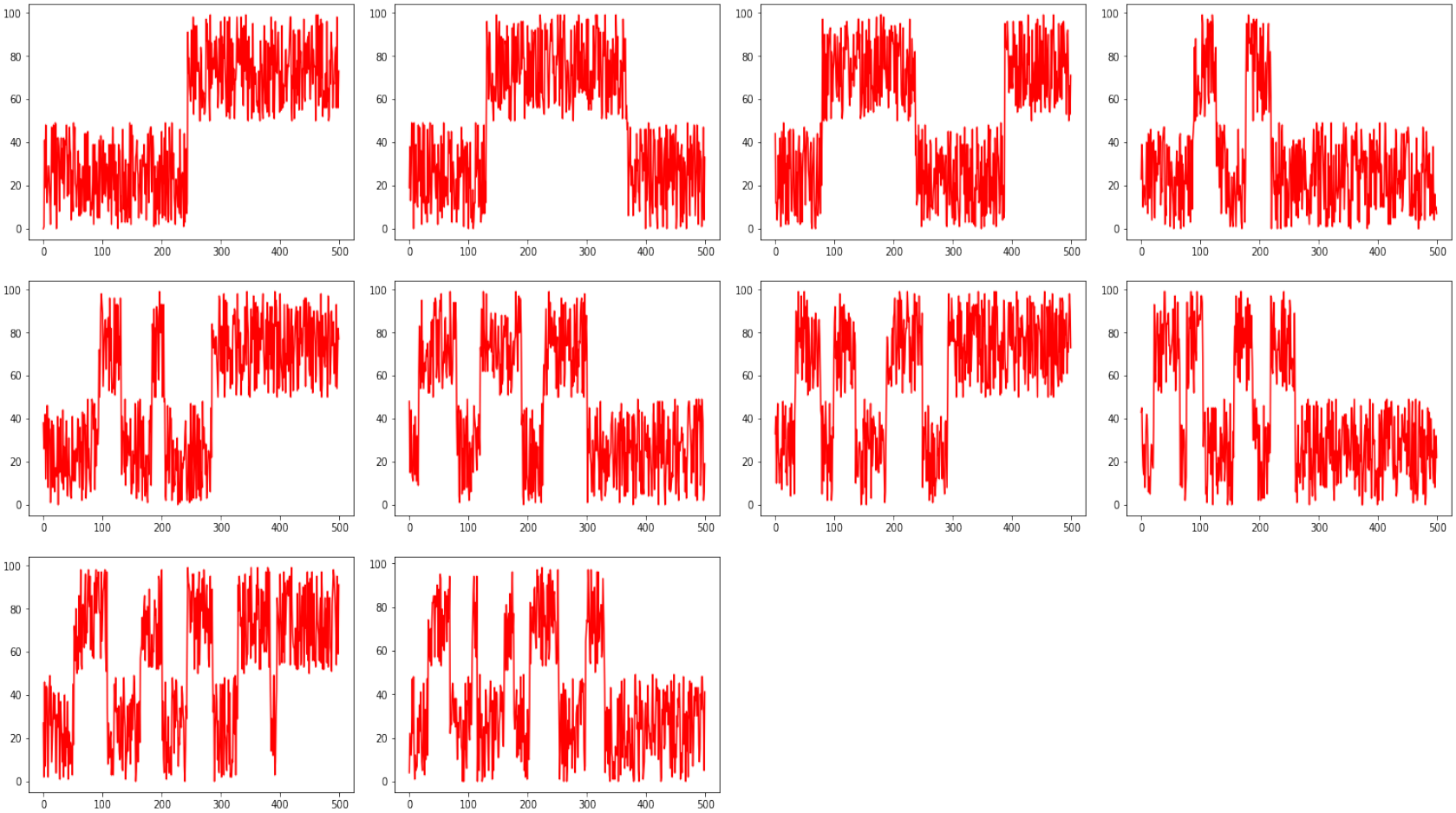


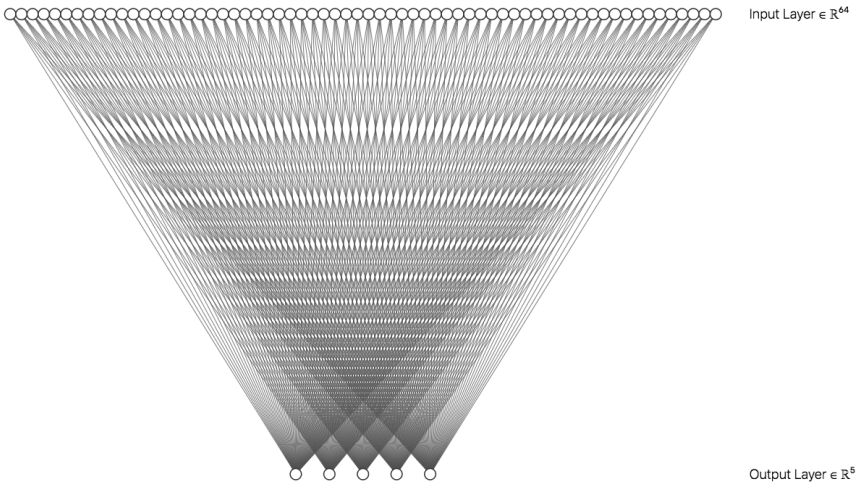
Figure 3: The Pacific brown pelican (*Pelecanus occidentalis*) population from 1939 to 2017 based on the Christmas Bird Count.

Lack of Data Problem / Solution



Method

Organizing the Dense Fully Connected Layers:

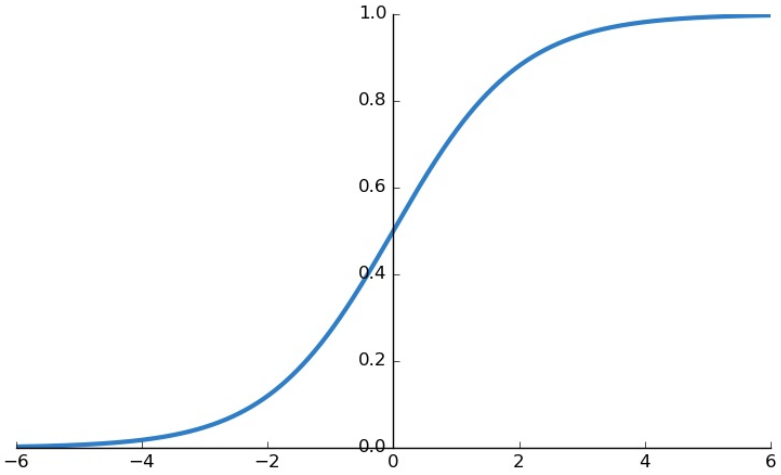


- Section the data into groups of 5.
- Only look at group of 5 that contain a breakpoint
- Dense Layer of 64 Neurons
- The 64 Neurons connected to 5 outputs

The picture displays 64 Neurons going to 5 Neurons. However, after experimentation the optimal number of neurons is 128 or 256.

Method: Activation

The Activation function used is the Sigmoidal Function



$$f\left(s_i\right) = \frac{1}{1 + e^{s_i}}. \qquad (1)$$

The sigmoidal function also known as the logistic function is used because it takes on values between 0 and 1.

Method: Calculation of Loss

To calculate the loss the method will use is the Mean Squared Error (MSE)

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{2}$$

where y_i is our predicted value and $\tilde{y}_i$ is the actual value.

Method: Calculation of Loss

To calculate the loss the method will use is the Mean Squared Error (MSE)

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{2}$$

where y_i is our predicted value and $\tilde{y}_i$ is the actual value.

Other Loss function tried:

- binary classification
- categorical cross entropy (when using softmax as the activation)

Method: Gradient Decent + Drop Out

Gradient Decent: 'atom'

- "an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments"

Drop Out: Rate of 10%

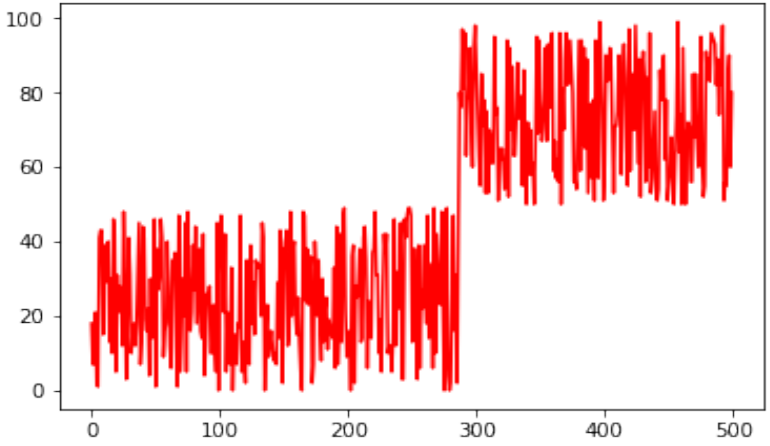
- This plays an important roll in avoid a single neuron gaining to much power controlling the model.

Results

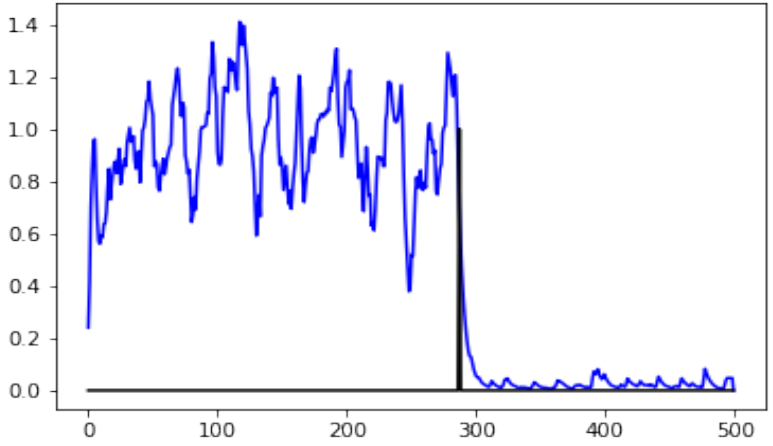
Train on 24748 samples, validate on 2749 samples

Last Iteration Output: 1s 22us/step

- loss: 0.1263
- binary_accuracy: 0.8646
- val_loss: 0.1269
- val_binary_accuracy: 0.8643



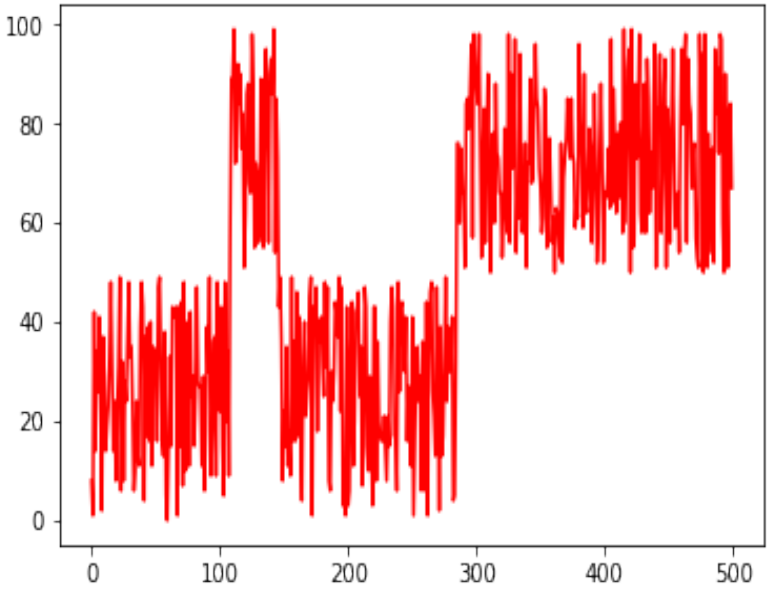
(a) Data with One Breakpoint before time 300.



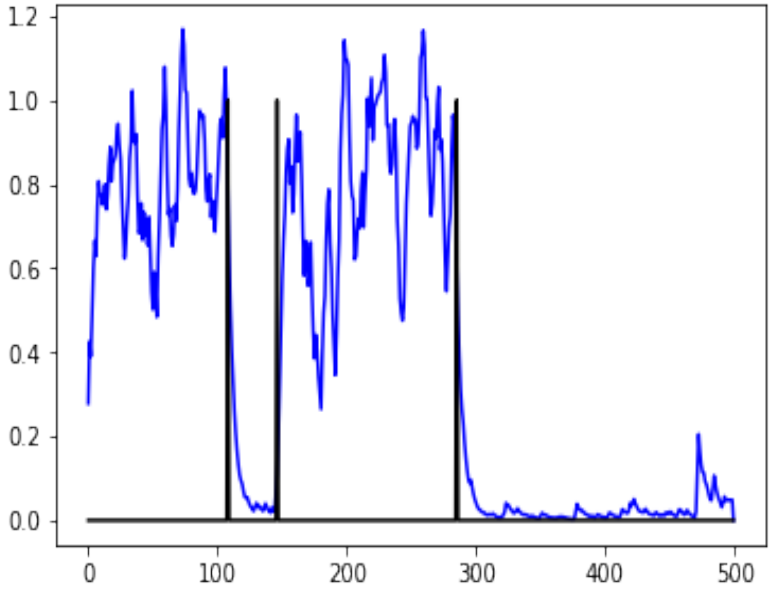
(b) Blue = predicted break set and black = actual break set.

Results

From the Same Predictive Model

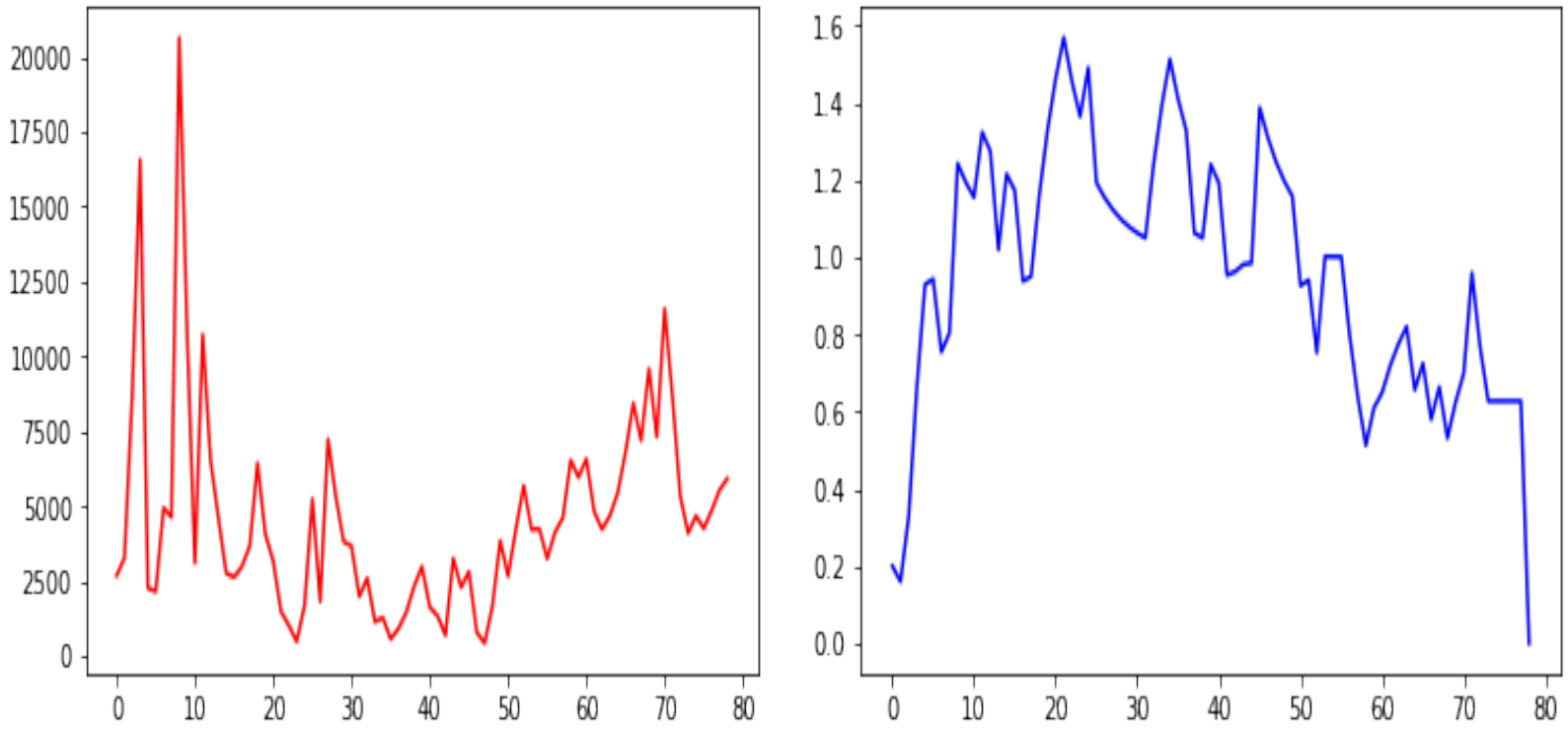


(a) Data with Three Breakpoint.



(b) Predicted and Actual breaks

Pelican Data



(a) Pelican Data (b) Predicted Breakpoints

Figure 6: Running the model on the Pelican data

References

DiMatteo, I., Genovese, C.R. and Kass, R.E., (2001). *Bayesian curve−fitting with free−knot splines*. Biometrika, 88(4), pp.1055-1071.

Kass, R.E. and Wasserman, L., (1995). *A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion*. Journal of the american statistical association, 90(431), pp.928-934.

Zeileis, A., Leisch, F., Hansen, B., Hornik, K., Kleiber, C. and Zeileis, M.A., (2007). *The strucchange Package*. R manual.

Pelican images courtesy of Frank Schulenburg and Pacific Southwest Region USFWS under Creative Commons.

Thank You

Questions?