Decay Rate Estimation for System Classification

Jett Hollister
Duke University
jett.hollister@duke.edu

Introduction

Decay Rate Estimation

Physical processes are often modeled by a weighted sum of decaying exponentials,

$$\sum_{q=0}^{Q} A_q e^{-\alpha_q t}$$

where the decay rates, α_q, can uniquely characterize the system[1]. While the architecture of the systems themselves may be unknown, their decay rates can be estimated through model inversion of signal measurements produced by the system over set time intervals.

Relevance

Exponential analysis can be used to describe many physical phenomena, including nuclear physics (nuclear magnetic resonance), electrochemistry (reaction kinetics), and biophysics (fluorescence decay analysis)[2]. Classification models trained on the decay rate estimations can identify systems in real-time. For example, a classification model trained on marine animal sounds could be used by marine biologists to distinguish the communication signal of an endangered blue whale from other marine life.

Problem Overview

This analysis will produce a pipeline from raw signal measurements to system classification based on estimated decay rates. Using a dataset of 600 signal measurement trials from two distinct systems (Type A and Type B), the decay rates will be estimated and aggregated for all trials. The collection of these values will be used to train a model capable of classifying unseen signal decay measurements into their corresponding systems.

Part I – Decay Rate Analysis

This section will focus on the model inversion portion of the pipeline and will explore transformations of the decay rate dataset. The feasibility of classification will be assessed through analysis of decay rate distributions.

Part II – Classification

This portion will evaluate three classification methods and explore the predictive abilities of each. A recommendation will be suggested for the production model, and the output from the blind test data will be discussed.

[1]Tantum, S., & Collins, L. (2003). Performance bounds and a parameter transformation for decay rate estimation. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(10), 2224–2231. https://doi.org/10.1109/tgrs.2003.814660

[2]Istratov, A. A., & Vyvenko, O. F. (1999). Exponential analysis in physical phenomena. *Review of Scientific Instruments*, *70*(2), 1233–1257. https://doi.org/10.1063/1.1149581

Data

Signal Decay Measurements

The dataset contains 600 trials of signal decay measurements, equally representative of two systems, Type A and Type B. Each trial notes the sampling time interval, the signal, and the corresponding system label (Figure 1.1). The training data was produced from a weighted sum of decaying exponentials (Figure 1.2), but the original coefficients for each system are unknown.

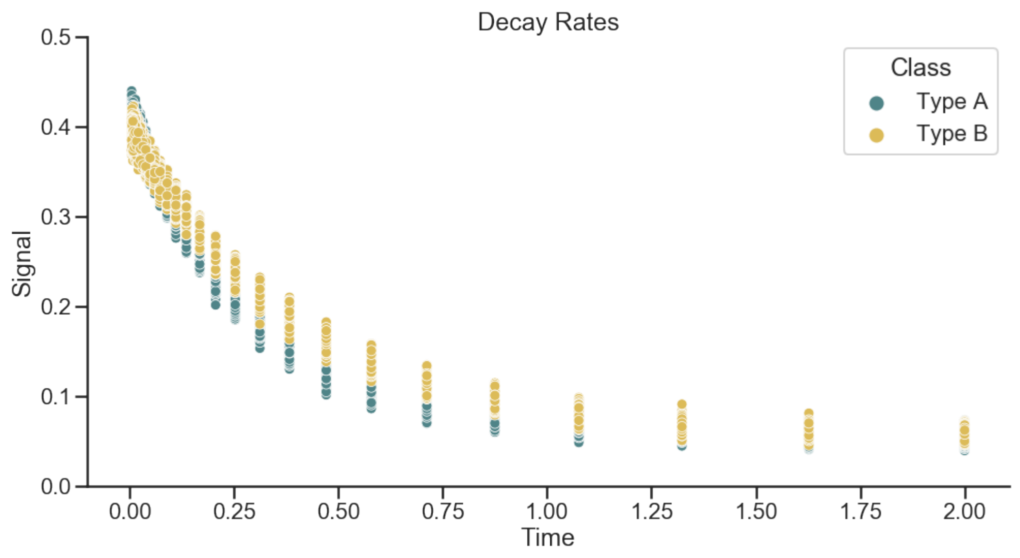

Figure 1.1: Plot of all 600 signal decay measurements

Time	Signal	Label
0.005	0.411405	Type A
0.006	0.419165	Type A
0.008	0.411505	Type A
0.009	0.397064	Type A
...	...	...
1.076	0.063966	Type A
1.323	0.063402	Type A
1.627	0.064123	Type A
2.000	0.054808	Type A

Figure 1.2: Sample of signal decay dataset

Model Inversion

Model Inversion

Model inversion refers to extrapolating a model's parameters only from its output, where the model, in this case, is known to be the weighted sum of two decaying exponentials:

$$s(t) = A_0 + A_1 e^{-\alpha_1 t} + A_2 e^{-\alpha_2 t}$$

For each trial, a function is fit to the data (Figure 2.1), and the estimated coefficients are extracted. While all unknown variables are estimated for each trial, only the decay rates, α_1 and α_2, are unique characteristics of the system. The decay rates for each trial are labeled with their given class (Type A or B) and merged into a single dataset (Figure 2.2).

SciPy Implementation

The non-linear least squares function, from SciPy's optimization toolbox[3], is used to perform the model inversion. To increase the probability of the optimization function converging, domain knowledge is used to assume $0 \leq \alpha_k \leq 10$ for $k = 1, 2$.

α_1	α_2	**Label**
4.165282	4.319087	Type A
...	...	...
2.612341	2.661444	Type B

Figure 2.2: Sample of decay rate dataset

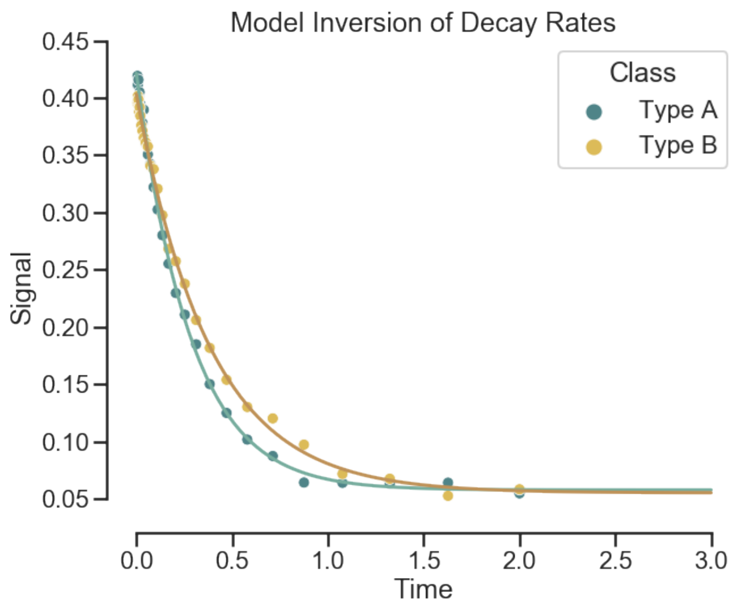

Figure 2.1: An example of model inversion for each class

[3]*Optimization and root finding (scipy.optimize)*. Optimization and root finding (scipy.optimize) - SciPy v1.5.4 Reference Guide. https://docs.scipy.org/doc/scipy/reference/optimize.html.

Decay Rate Analysis

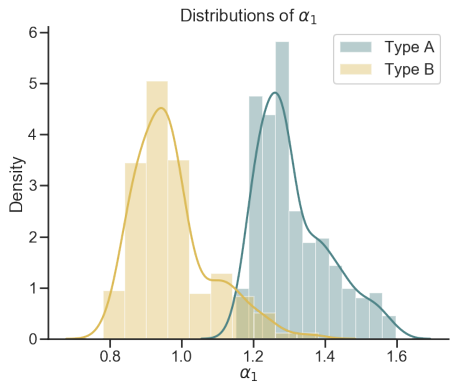

Figure 3.1: α_1 distributions

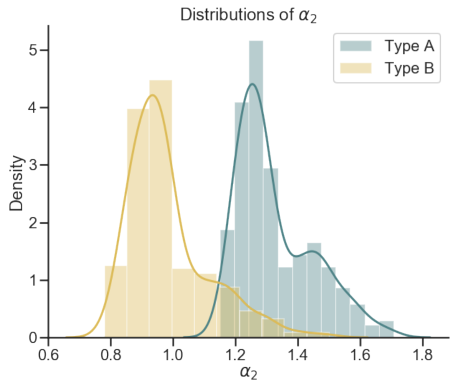

Figure 3.2: α_2 distributions

Decay Rate Distributions

The resulting distributions of the estimated decay rates demonstrated how well the they support classification of the system as Type A or Type B. In Figures 3.1 and 3.2, there is little overlap between the system classifications for α_1 and α_2, and this separation suggests an underlying, predictable relationship between decay rates and system classification. While the decay rate pairs are close in proximity across classes, they appear to be linearly separable (Figure 3.3). As a result, the classes should be most-accurately predicted by a linear model.

Log Transformation

The visualizations (left) represent the log-transformed decay rates. The data was log-transformed for two reasons: one of the classification models assumes normal distributions of each class, and the original data appeared to follow close to a log-normal distribution. The transformation decreased the difference in variance between classes from 0.06 to 0.001 and from 0.09 to 0.001 for Types A and B, respectively.

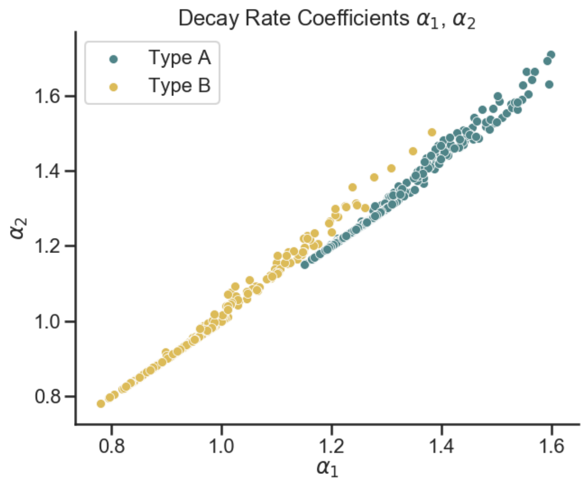

Figure 3.3: Decay rate pair distributions

Model Selection

Non-Linear Model

To explore the hypothesis that non-linear models will be less accurate than linear models for this analysis, the k-Nearest Neighbors algorithm (k-NN) was also evaluated. The algorithm uses the *n*-closest data points to determine the classification of an unknown instance and can be optimized with a variety of distance metrics and weights. The initial model used uniform weighting, the standard Euclidean distance metric, and 5-nearest neighbors.

Linear Models

As mentioned in Decay Rate Analysis, the almost-linear separability of the decay rate pairs suggested that a linear model would best distinguish between Types A and B.

Logistic Regression

Logistic regression (LR) measures the probability that an instance belongs to a given class based on the combination of its features. The model fits the data to a logistic function and returns the probability that a trial was Type A or Type B based on its position along the logistic function (Figure 4.1).

Linear Discriminant Analysis

Linear discriminant analysis (LDA) measures the probability of an instance's features based on its class. LDA makes two key assumptions of the data:

- The data is Gaussian for each class
- Each class has the same variance

Since we log-transformed the data, each class more-closely followed a Gaussian distribution and had nearly equal variances across both decay rates. LDA stems from Bayes' Theorem, which uses the probability of both classes and the data belonging to each class to return the probability of a single instance's class.

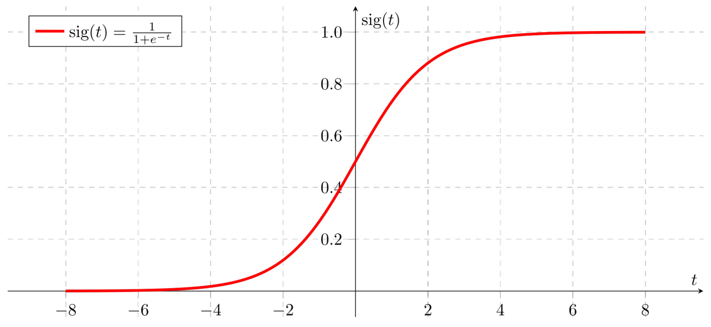

Figure 4.1: Example of logistic function

Performance

Evaluation

5-fold cross-validation was used to find the false positive and true positive rates for each classifier. The folds were stratified to preserve the percentage of samples for each class. Before tuning the hyperparameters of each classifier, LDA achieved the highest AUC score of 1.0, while the LR and k-NN classifiers fell slightly below perfect classification across all folds (Figure 5.1).

Hyperparameter Tuning

Scikit-learn's model selection toolbox[2] provides a cross-validated, exhaustive search function to optimize the hyperparameters of each classifier. The parameters, depending on the classifier, include iterations, penalization norms (L1 or L2), distance metrics (Manhattan, Euclidean, or Minkowski), and regularization strength. The fine tuning had the largest impact on the LR classifier, improving its AUC score from 0.981 to 1.0 (Figure 5.2).

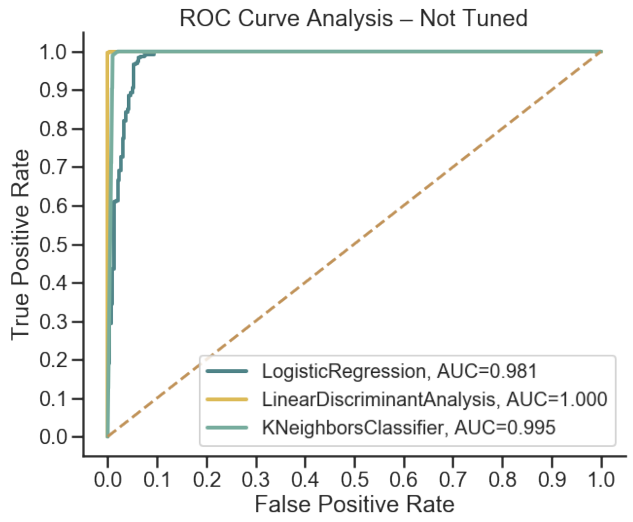

Figure 5.1: Pre-tuning ROC curve

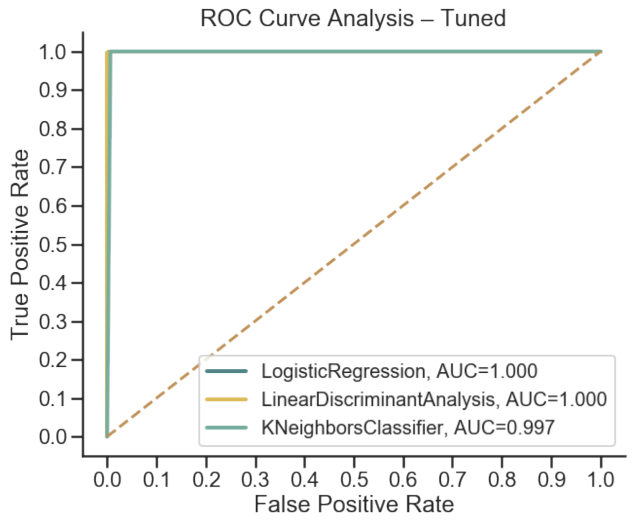

Figure 5.2: Post-tuning ROC curve

k-NN Evaluation

Predictive Ability

The k-NN decision boundary represents the algorithm's non-linear decision-making process (Figure 6.1). The model failed to capitalize on the linear separation of the two classes, and this failure resulted in the largest number of false positives across all classification methods (Figure 6.2). If future test data falls outside of the current range of values for either labels, the model could struggle to accurately classify these edge cases. As suspected in Decay Rate Analysis, the non-linear k-NN algorithm is unfit for the linearly-separated distribution of decay rate pairs.

Hyperparameter Improvements

When tuning the hyperparameters of the k-NN classifier, prediction using the 3-nearest neighbors and Manhattan distance metric resulted in an increase in the AUC score by 0.002. This slight increase in performance and corresponding hyperparameters, like the Manhattan distance metric, could easily be random, but it could also demonstrate the model attempting to account for the linear separation of the classes. For example, by weighting closer points more-heavily, the algorithm may have been able to overcome interference resulting from the proximity of clusters.

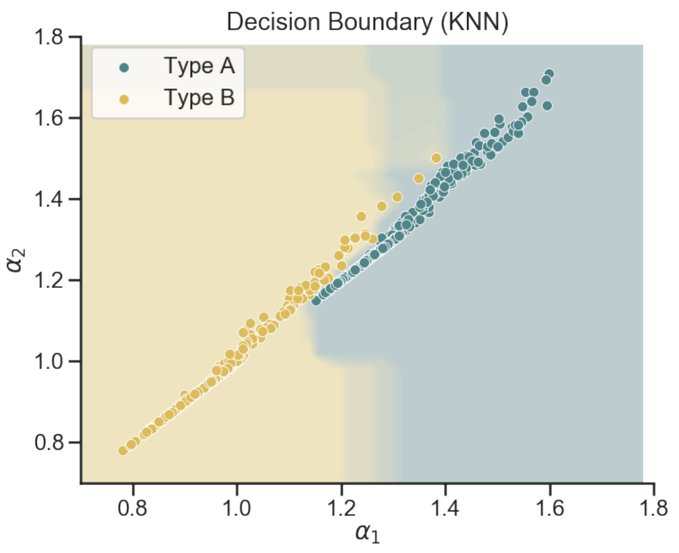

Figure 6.1: k-NN decision boundary

		Actual	
		+	–
Predicted	+	TP = 300	FP = 5
	–	FN = 0	TN = 295

Figure 6.2: k-NN confusion matrix

LR Evaluation

Decision Confidence

The LR classifier correctly predicted 599 of the 600 trials, with only one false positive. Due to the accuracy of the model, the decision statistic for each prediction was used to further distinguish the classifier's performance (Figure 7.1). On average, the model was generally confident in its predictions, but the distribution of decay rate pairs (Figure 7.2) naturally has the potential for edge cases between classes. The false positive prediction, found at approximately (3.55, 3.55), resulted from the close relation of decay rate pairs in that range.

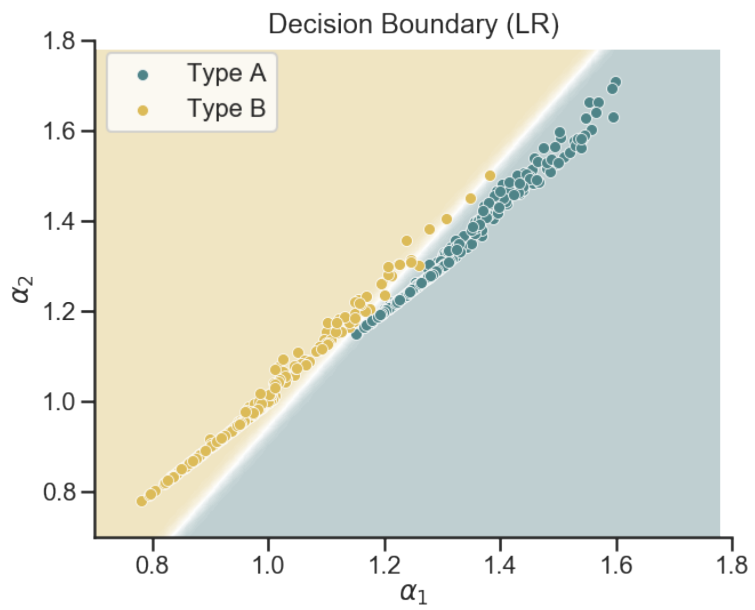

Figure 7.1: LR decision boundary

Label	Mean Dec. Stat.
Type A	0.981
Type B	0.016

Figure 7.2: LR mean decision statistics

Hyperparameter Tuning

Regularization refers to reducing the variance of a model to better handle noisy data. During fine tuning, the LR regularization strength was decreased from scikit-learn's default value of 1.0 to 0.001. Due to the lack of noise in the data and similarity between decay rate pairs, it makes sense that reducing regularization would improve the model's ability to distinguish between classes.

LDA Evaluation

Superior Model

Like the LR classifier, the LDA model also mislabeled the same single data point out of the 600 trials, and it was similarly a false positive (Figure 8.1). However, when comparing the average decision statistics, the LDA model was nearly 2% more confident in its Type A classifications and 94% more confident in its Type B classifications (Figure 8.2). While the model did have a single prediction error, it is likely that this value could be an outlier, as all models failed to accurately classify it. The model's relative accuracy, in combination with its AUC score of 1.0, solidify the LDA classifier as the most reliable predictor of a system from the estimates of its decay rates.

Label	Mean Dec. Stat.
Type A	0.999
Type B	0.001

Figure 8.2: LDA mean decision statistics

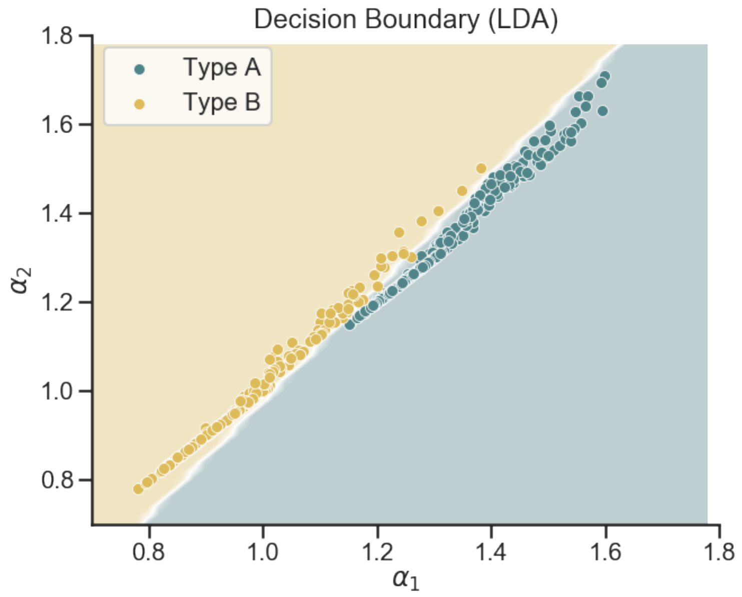

Figure 8.1: LDA decision boundary

Blind Data Confidence

Given its performance in comparison to the other classifiers, the LDA model was used to evaluate the final blind test cases. The model predicted the systems of each trial in the test set with a similar confidence as Figure 8.2, achieving a mean decision statistic of 0.995 and 0.0002 for Types A and B, respectively. While the accuracy of its predictions are unknown, the confidence of its classifications are promising.

References

Istratov, A. A., & Vyvenko, O. F. (1999). Exponential analysis in physical phenomena. *Review of Scientific Instruments*, *70*(2), 1233–1257. https://doi.org/10.1063/1.1149581

Optimization and root finding (scipy.optimize). Optimization and root finding (scipy.optimize) - SciPy v1.5.4 Reference Guide. https://docs.scipy.org/doc/scipy/reference/optimize.html.

scikit-learn. *3.2. Tuning the hyper-parameters of an estimator*. scikit. https://scikit-learn.org/stable/modules/grid_search.html.

Tantum, S., & Collins, L. (2003). Performance bounds and a parameter transformation for decay rate estimation. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(10), 2224–2231. https://doi.org/10.1109/tgrs.2003.814660

Collaboration

I collaborated with no classmates on this project.