Utilizing Gaussian Processes for Numerical Image Classification

Ryan Honea

Austin Peay State University

April 19, 2018

Section 1

Introduction

Problem Statement

Analysis of MNIST Database

- 60,000 Training Samples of Integers 0-9
- 10,000 Testing Samples of Integer 0-9
- Obtain optimal classification accuracy while minimizing computational expense

Methods Attempted – See MNist Data Page

- Linear Classifiers (1-Layer Neural Nets)
- Support-Vector Machines
- K-Nearest Neighbors
- Deep Neural Nets

Proposed Method

Gaussian Process Classification

- Utilize the non-parametric (no specified distribution) qualities of Guaussian process
- Assesses data without need for excess parameters or prior knowledge of data
- Analyze if the strength of Gaussian Processes extends from small data sets to large datasets

What's a Gaussian Process?

Generating Random Gaussian Processes

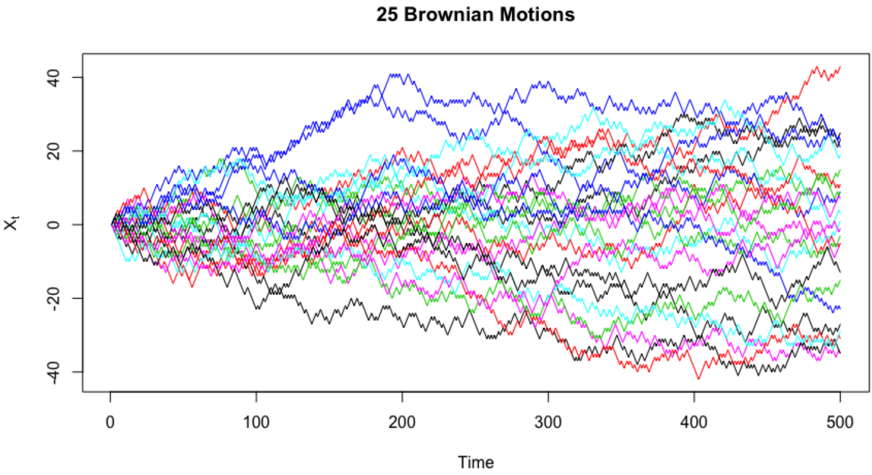
- Consider the Brownian Motion generated by flipping a coin,

$$X_t = \begin{cases} X_{t-1} + 1, & \text{If Heads} \\ X_{t-1} - 1, & \text{If Tails} \end{cases} \qquad X_0 = 0$$

,

- Expectation at any t is equal to 0
- Variance is t at time t

Brownian Motion Examples



The General Case

We define a Gaussian Process to be

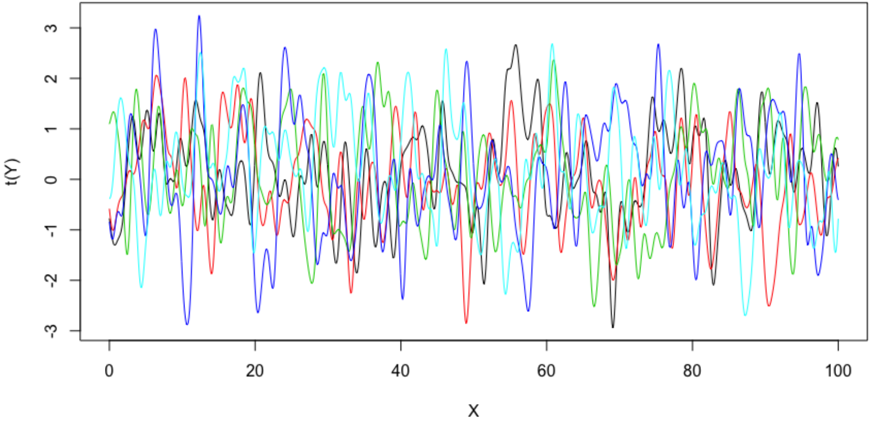
$$\{X_t\} \sim MVN(\vec{\mu}, \Sigma)$$

where $\vec{\mu}$ is a vector of means and Σ is the covariance matrix where $\Sigma_{i,j}$ is the covariance between X_i and X_j. A common covariance function can be defined by

$$\Sigma_{i,j} = exp\left(-||\vec{x_i} - \vec{x_j}||^2\right)$$

where $||\vec{x_i} - \vec{x_j}||$ is the Euclidean distance between two points. Note that this means increasing distance between points causes the covariance decreases exponentially.

Simulated Gaussian Processes



Model Construction

Methods of Regression Model Construction

- Simplest model uses no additional hyperparameters outside the method of process generation
- Stronger models use hyperparameters that need to be estimated for strong fits
- Stronger model is less likely to overfit.
- Classification methods utilize a generalization of the logistic regression model called linear probit regression with a Gaussian prior.

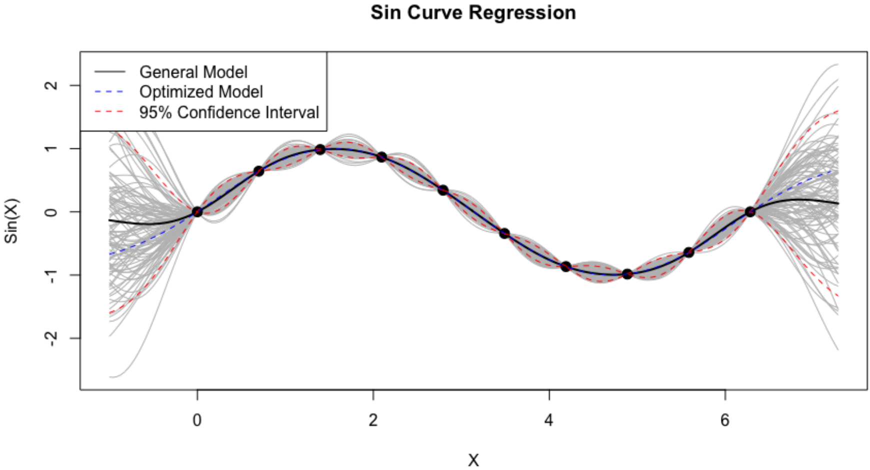
Full details of model construction will be available in paper, but are mostly outside the scope of this intro course.

Section 2

Examples of Gaussian Regression

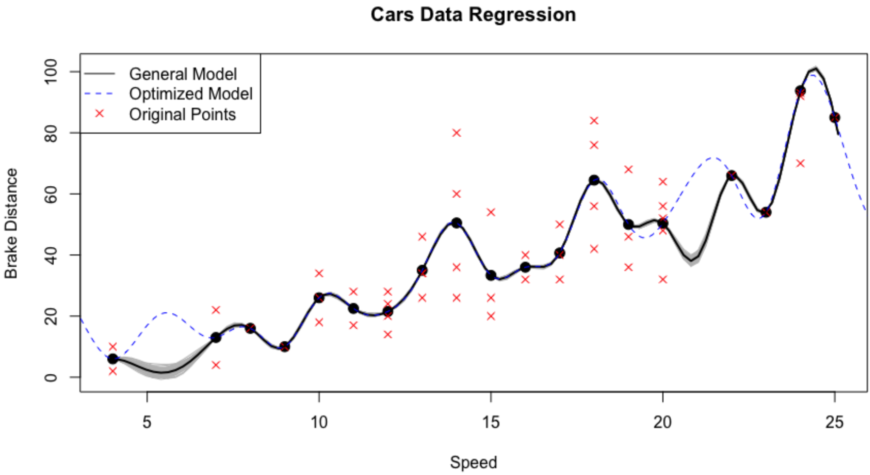
Regressing on a Sin Curve

Sin curve is a great example of a problem that linear regression doesn't solve well.



Regressing on Cars Dataset

Cars dataset is a good set to practice on because linearity is suspect.



Notes on Cars Regression

- Great Example of the dangers of overfitting from the simple model
- Shows another issue with variance concerns on 2-dimensional datasets as the mean of points with equal x value must be taken.
- Likely not a candidate for Gaussian Process Regression

Section 3

Classifying on the MNist Database

Model Selection

Objectives in Model's Kernel Selection

- Needs to capture the large feature space of a 24x24 image (so 576 features!)
- Ideally, doesn't add much computational complexity to an already complex model ($O(n^3)$ computational time) Immediate thoughts on Kernels
- Radial Basis Function is commonly used on image recognition
- Gaussian Process Equivalent of Polynomial Kernel which is dot product kernel

Issues with Model Selection Phase

Issues on Computational Time Complexity

- Radial Basis function provided smallest addition in computation complexity but did very poorly in validation testing.
- Dot Product is also (for the most part) computationally inexpensive, but only has 82% accuracy (which isn't too awful, but we can do better!)
- Dot Product also is often used with some form of exponentiation, but long training times make it hard to test various values
- Rational Quadratic which can be seen as an infinite sum of RBF kernels is the most computationally inexpensive but gives the best accuracy, approximately 90%

Issues with Model Selection Phase

Issues on Computational Space Complexity

- With just using 5000 samples of the 60000 from the training set, my RAM would often fill up rendering my computer temporarily unusable. 2500 samples itself would often use 4gb of RAM at any given time
- The storage space complexity $O(n^2)$ was also often a barrier and would require dumping memory before testing new models

Final Model Choice

The big question: Do we accept reduced complexity and choose the dot product kernel, or do we choose the rational quadratic?

Rational Quadratic Kernel!

Additional Complexity is only marginal, and if it already takes forever, then an extra ten or so minutes of training won't do much harm.

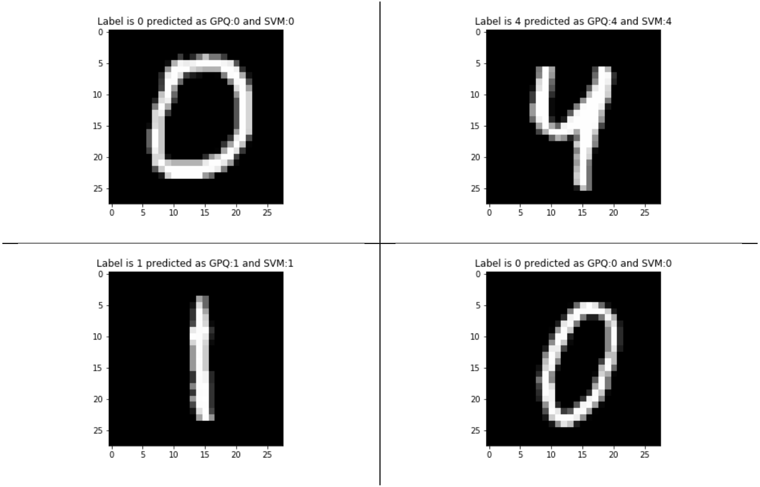
Comparison of Results against other Models

Using 2500 random samples from the training set and testing against the entire testing dataset of 10,000 samples, the following are the accuracies of various ML-Algorithms:

- SVM: 0.924
- KNN: 0.9068
- MLP: 0.7897
- GP-RBF: .1009
- GP Quadratic .8987
- GP Dot .8304

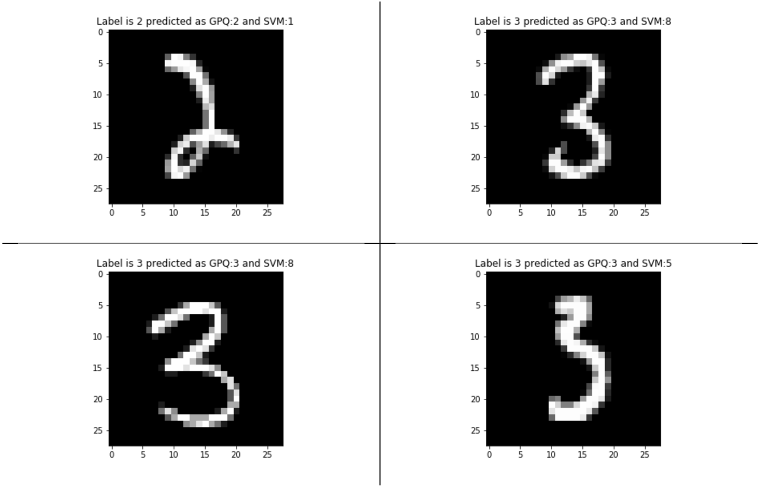
Comparison Against SVM

Both SVM and the Guassian Process Classification got it right.



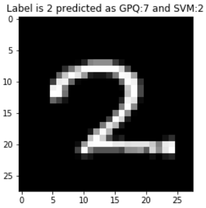
Comparison Against SVM

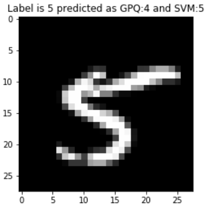
SVM got it wrong and Gaussian process Classification got it right.

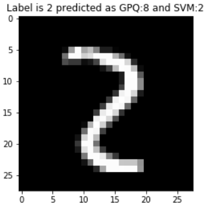


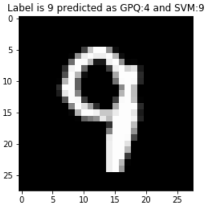
Comparison Against SVM

SVM got it right and Gaussian process Classification got it wrong.



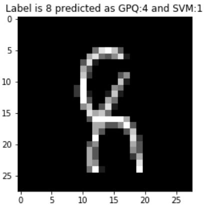


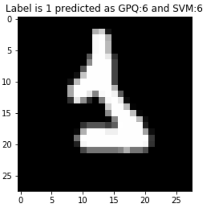


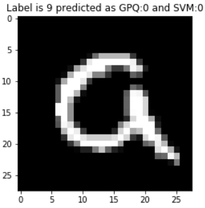


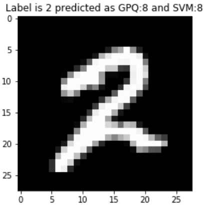
Comparison Against SVM

Both SVM and the Guassian Process Classification got it wrong.









Comparison against SVM

Notes on the algorithms

- Not really clear which features each captures better
- SVM is completely nonparametric and Gaussian Process Classification is nonparametric for the most part, so it's difficult to see what each is doing best
- SVM performed much faster, but this could be due to Scikit Learns additional optimization with the algorithm
- SVM has much less space complexity in prediction

Section 4

Conclusion

Remarks

To put it bluntly: Probably not best for image classification...

- Has high space complexity which is compounded by images
- Images have a lot of noise, and so the parametric component of the classification will reduce accuracy
- Extremely strong for nonparametric regression (especially time series!)
- SVM and KNN's performance being much stronger and faster means we should probably use them.

Future Work on Gaussian Processes

1. Continue reading Rasmussen and William's "Gaussian Processes for Machine Learning" (It's free!)
2. Explore a possible mixed model that utilizes a trained Gaussian Process Classifier on edge cases of an SVM model on MNIST data
3. Test additional kernels on MNIST including mixed kernels
4. Write a model in a more optimized language in order to include more training samples

Acknowledgments

- Dr Mayo for excellently teaching a new (and difficult to approach) course at APSU
- Joseph Mathews for making fun of the initial model's failures (Everything was predicted to be a 9)
- The many youtube videos that aided my understanding of the processes (Dr. Bobby Grammacy at Virginia Tech most notably, and MIT's various Open Courseware courses)
- My cat Tuna for sitting on my laptop and reminding me to occasionally take a break from coursework