E0:270 - Machine Learning - Gaussian Processes

Kunal Chelani [1] **Sruthi Gorantla** [1]

Abstract

Most of the methods that we observe in Machine learning aim at estimating a best-fit model for the Data. Bayesian Processes are one set of algorithms that instead compute a posterior distribution over models, or compute posterior predictive distribution for new inputs. However, they are restricted to be parametric. In this project, we study about Gaussian Processes which achieve this by being non-parametric at the same time.

1. Introduction

As discussed in (Bishop, 2006) and (Murphy, 2012), Bayesian processes allow for better quantification of the uncertainty in prediction. Our main focus in this project is understanding and implementing Gaussian processes regression and classification. As a warm up exercise, we start with Bayesian linear regression and Bayesian logistic regression as they also use the analytic properties of multivariate Gaussian distributions that are extensively used later. They differ from Gaussian processes methods in the sense that they are parametric.

1.1. Motivation

In fields like health care, stock market and similar, it is not sufficient that we just predict the risk. We also require the model to be confident about its prediction. This is not achieved by the traditional linear and logistic regression models. Hence we look at Gaussian processes linear and logistic regression models which achieve this with almost non existent overfitting(Rasmussen & Williams, 2005). The added advantage of these models is that they are non-parametric. This forms the motivation for our project to pursue these models in depth and provide a comparative study on the traditional models on the task of risk prediction using SVMs and neural networks.

[1]Department of Computer Science and Automation, Indian Insitute of Science, Bangalore. Correspondence to: Kunal Chelani <kunalchelani@iisc.ac.in>, Sruthi Gorantla <gorantlas@iisc.ac.in>.

Midterm Project Report, E0:270, Machine Learning, 2018.

2. Bayesian Linear Regression

2.1. Theory

As mentioned above, Bayesian Linear Regression aims to output a posterior distribution over the model parameters instead of outputting a posterior distribution over regression models. Assuming a Gaussian noise model on the data, the likelihood is distributed normally. In this setting a Gaussian prior is a conjugate one and results in a normal posterior distribution. Considering $t_i = w^T\phi(x_i) + n$ where n is zero mean i.i.d Gaussian with variance β^{-1}.

$$P(t|X, w, \beta) = \prod_{n=1}^{n=N} \mathcal{N}(t_n|w^T\phi(x_n), \beta^{-1}) \quad (1)$$

We assume a Gaussian prior of the form :

$$P(w) = \mathcal{N}(w|m_0, S_0) \quad (2)$$

This leads to a Gaussian posterior distribution:

$$P(w|t) = \mathcal{N}(w|m_N, S_N) \quad (3)$$

where

$$S_n^{-1} = S_0^{-1} + \beta\Phi^T\Phi \quad (4)$$

$$m_N = S_N(S_0^{-1}m_0 + \beta\Phi^T t) \quad (5)$$

Here Φ is NxM design matrix with each row containing the M basis terms. For example, if the function is of the form $y = ax^2 + bsin(x) + c$, one row of the design matrix will be $[x_i^2 \quad sin(x_i) \quad 1]$

2.2. Experiment

2.2.1. Setup

We consider 2-d data on a single variable following the equation:

$$y = x^3 - 0.5x - 1 \quad (6)$$

We add a Gaussian noise with mean 0 and variance 0.1 to this data. The aim is to study the effect of number of data points on the confidence of model parameters.

2.2.2. Results

Figure 1 through Figure 4 show results over different number of points. 5 samples from the obtained posterior distribution have been shown in each case. It can be easily

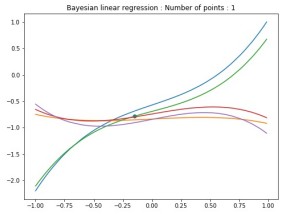

Figure 1: Num points = 1

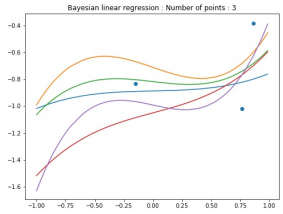

Figure 2: Num points = 3

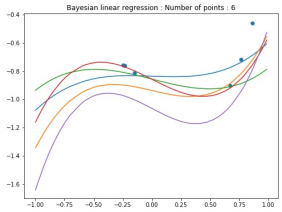

Figure 3: Num points = 6

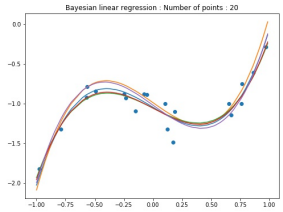

Figure 4: Num points = 20

observed the increase in confidence over a particular set of model parameters, which is in fact close to the originally chosen set of parameters.

3. Bayesian Logistic Regression

In logistic regression, we aim at finding decision boundary for a data distributed into k classes. But this only gives a point estimate of the decision boundary. Hence we perform Bayesian Logistic Regression on the data to find the posterior distribution of the parameters. For this we use Laplace approximation (see appendix) since the exact evaluation of posterior distribution is intractable.

Given:
likelihood: $p(t|w) = \prod_{n=1}^{N} y_n^{t_n}(1-y_n)^{1-t_n}$

prior (Gaussian): $p(w) = \mathcal{N}(w \mid m_0, S_0)$

posterior is given by a Gaussian approximation

$$q(w) = \mathcal{N}(w \mid w_{MAP}, S_N) \tag{7}$$

where

$$S_N = -\nabla\nabla ln(p(w \mid t)) \tag{8}$$

$$= S_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\phi_n\phi_n^\top \tag{9}$$

The computation of predictive distribution is not straight forward. However, after approximations it simplifies down to the following:

$$p(C_1|\phi, t) = (k(\sigma_a^2)\mu_a) \tag{10}$$

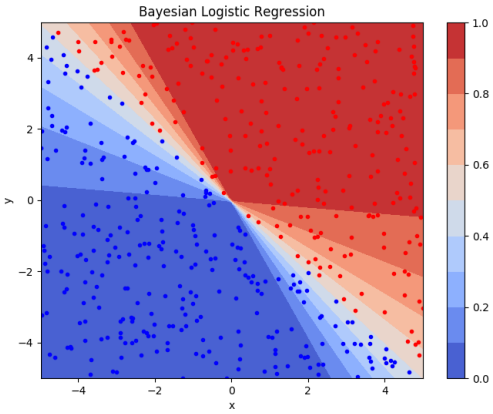

Figure 5: Bayesian Logistic regression of linearly separable data

where

$$k(\sigma^2) = (1 + \pi\sigma^2/8)^{-1/2} \tag{11}$$

$$\mu_a = w_{MAP}^\top \phi \tag{12}$$

$$\sigma_a^2 = \phi^T S_N \phi \tag{13}$$

$$p(C_2|\phi, t) = 1 - p(C_1|\phi, t) \tag{14}$$

3.1. Experiment

Here we perform Bayesian Logistic Regression for a two class classification task with 100 points in 2 dimensions. After applying Bayesian Logistic Regression, we find the posterior distribution of the parameters and plot the probability of the entire space. We observe that the decision boundary lies in the light white region where the probability of both the classes is equal i.e., $\frac{1}{2}$, which is the mean of the distribution of the parameters. The dark red region marks the space belonging to class 1, i.e., probability of the point belonging to class 1 and dark blue region has probability 0 which implies class 0.

4. Gaussian Processes

4.1. Theory

Multivariate Gaussian distributions are useful for modelling collections of real valued variables because of their nice analytic properties. Gaussian processes are extensions of multivariate Gaussian distributions to infinite sized collections. Formally, a Gaussian process is a stochastic process such that any finite sub-collection of random variables has a multivariate Gaussian distribution. This allows us to obtain expressions for probability density over all possible functions with the training and test data points as the domain.

4.2. Gaussian Processes Regression

We recall that in the simple Bayesian Linear Regression setting we aim at finding a posterior distribution on the parameters θ_0, θ_1 and θ_2 such that we model the function $y = \theta_0 x^3 + \theta_1 x - \theta_2$. The Gaussian process approach, on the other hand, aims at finding a distribution over the possible functions $f(x)$ that are consistent with the observed data. Hence, this is a *non-parametric* approach. Similar to the Bayesian methods, we assume a prior distribution of functions and update this as more data points are observed, giving us a posterior distribution over the functions.
Consider the linear regression model $f(x) = \phi(x)^\top w$ with a prior, say $p(w) = N(0, \Sigma_p)$. Then the distribution on the functions have the mean and covariance:

$$E[f(x)] = \phi(x)^\top E[w] = 0 \tag{15}$$

$$E[f(x)f(x')] = \phi(x)\Sigma_p\phi(x') \tag{16}$$

The covariance here is an exponential kernel function defined by:

$$cov(f(x_p), f(x_q)) = k(x_p, x_q) = exp(-\frac{1}{2}|x_p - x_q|^2) \tag{17}$$

A careful observation of this function gives us that the covariance is almost unity between variables whose corresponding inputs are very close, and decreases as their distance in the input space increases.
Let X be the training data and f be the joint distribution of the training outputs. Given X_*, the test data, we generate a random Gaussian vector with this covariance function.

$$f_* \sim N(0, K(X_*, X_*)) \tag{18}$$

Then the joint distribution of f and f_* is given by

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,K) & K(X_*,X_*) \end{bmatrix}\right) \tag{19}$$

To get the posterior distribution over functions we restrict this joint prior distribution to contain only those functions which agree with the observed data points. Therefore we have:

$$f_*|X_*, X, f \sim N(K(X_*,X)K(X,X)^{-1}f,$$
$$K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*))$$

Function values f_* can now be sampled with this mean and covariance.

4.2.1. Synthetic Dataset

We create a synthetic dataset representing the sin functions. We sample three functions from the prior (restricted to -5 and 5) as shown in the figure 6. Now consider the figure 7. The blue dots represent the training data provided.

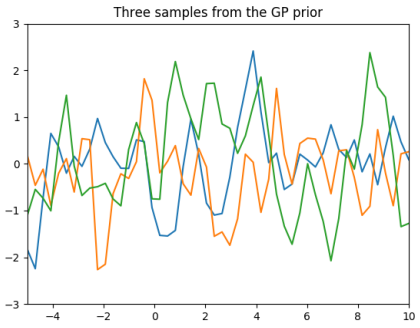

Figure 6: Three functions sampled from prior

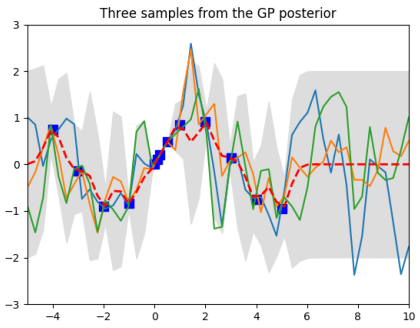

Figure 7: Three functions sampled from posterior

The red dotted line represents the mean of the posterior distribution and the light grey region is 2 standard deviations from the mean. From this posterior distribution, we sample three functions that represent the possible function for the given data.

4.2.2. Observations

We notice that near the region where many training data points are seen (near x = 0), the model is almost confident about the predictions nearby. Where as, near the regions where no training data has been seen (region $x > 5$), the standard deviation is too high, which means, the model cannot predict with high probability. The point to note here is, the model is said to have predicted correctly, even if the true label is in k-standard deviations from the mean. That way, this model will outperform the models that give just the point estimates.

4.2.3. Gaussian Processes Classification

Gaussian processes functions take real values over their domains. In a classification setting, we squash down these real values between 0 and 1 so that we can consider them as probabilities of belonging to a particular class. Here we will consider the derivation and results of a binary classi-

fication. Let y_N denote the set of class labels known at input points X_N and we need to predict the probability of a new point X_{N+1} belonging to class 1 π_{N+1}. Also let π be determined by squashing a latent function f by a sigmoid function.

$$\pi_i = \sigma(f_i) \tag{20}$$

where σ is the logistic function of the probit function. Therefore, the posterior probability:

$$P(f_{N+1}|X_N, y_N, X_{N+1}) \tag{21}$$

$$= \int P(f_{N+1}|f, y_N, X_{N+1})P(f|y_N, X_{N+1})df \tag{22}$$

The second term in the integral is not Gaussian and hence the integral becomes intractable. We therefore can use techniques like expectation propagation and Laplace approximation to approximate it to a Gaussian. We have used the latter approach, which involves calculating the mode of the distribution. This is done by maximizing the log of the posterior probability. Since we have assumed a G.P prior on the latent function with mean 0 and k as the kernel function, the expression to be maximised takes the form:

$$\nabla L(f) = \nabla P(y_N|f) - K^{-1}f \tag{23}$$

where K is the co-variance matrix calculated over the training data points. Since there isn't a closed form solution to this, we use Newton's gradient ascent to maximize the log likelihood, obtaining the parameters for the approximate distribution. The integral is then calculated using Dirac function and probit techniques. We skip the math here in interest of space. One can look at the appendix for more details.

4.3. Experimental Results

We consider a toy dataset shown in figure ?. The task is identify probabilities of test points belonging to each of the binary classes. Figure ? shows the result. The space has been divided into pixels, each of them actually representing a test point. The scale on the right of the figure shows probability of belonging to class red. As expected, the Gaussian Process is almost immune to over-fitting and shows very little confidence in predicting about data it has never come across.

4.4. Advantages of GP methods

A natural question that pops up is about the need for Gaussian Processes methods. The answer to that is actually the motivation section in the beginning of this report, but we try to give a somewhat quantifiable answer to that here. Consider the data shown in figure 10 distributed by the following rules:

$$\sin(x_1) - 5\sin(x_2) - 0.9 > 0 \implies x \in +1 \tag{24}$$

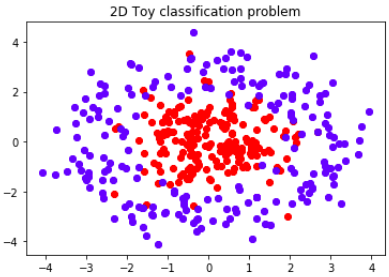

Figure 8: Data for classification experiment

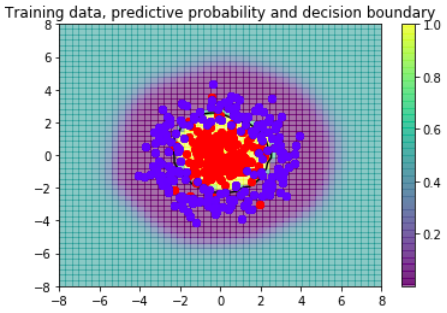

Figure 9: GP posterior probabilities and decision boundary

$$\sin(x_1) - 5\sin(x_2) - 0.9 <= 0 \implies x \in -1 \tag{25}$$

It is clear from figure 11 that SVM predicts with very high probability that the points lying anywhere other than inside the decision boundary to belong to the blue class. Similar results are achieved by a neural network based classifier shown in figure 12 . These models can therefore be seen to predict unseen data wrongly with very high probability whereas the Gaussian process classification is robust, and it allows to reject the prediction in case of low confidence. This can be seen in figure 13. This is one huge advantage of using Gaussian processes in areas where confidence estimation over prediction is necessary

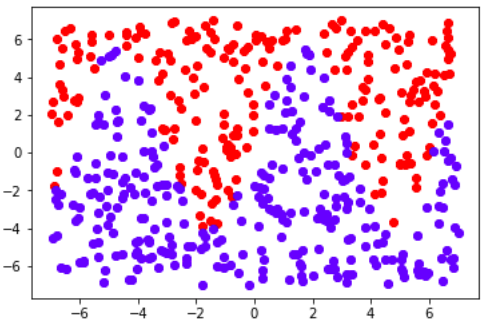

Figure 10: Sinusoidal Distributed data

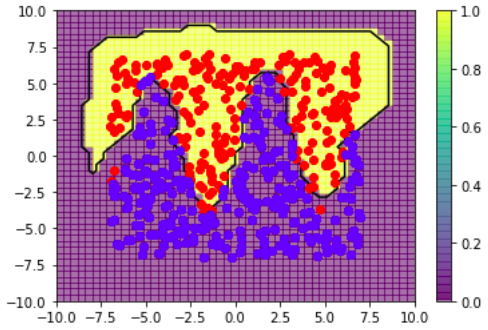

Figure 11: Classification using SVM

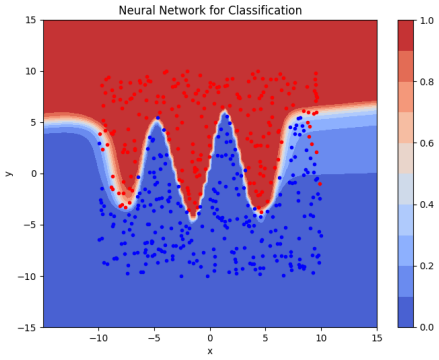

Figure 12: Classification using 3-layer NN

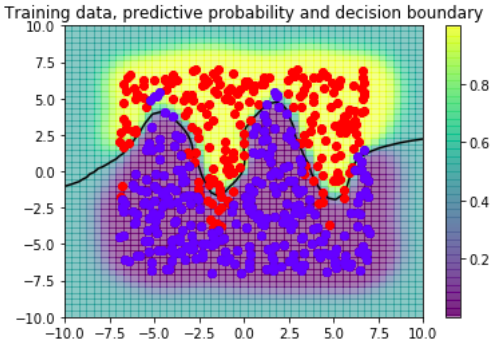

Figure 13: Classification using GP

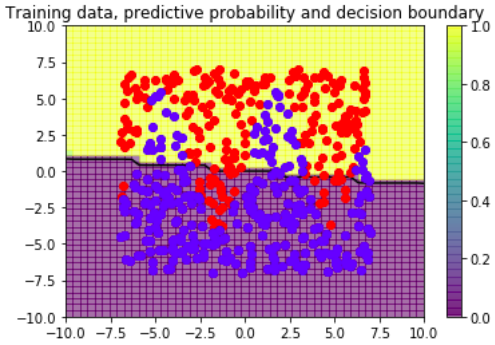

Figure 14: Linear kernel

4.5. Kernel Engineering

Another question that comes up is about the ability of GP methods to perform the way they do. To be precise, without fixing a model, how do we obtain the correct function with high probability. The answer to this lies in the engineering of the kernel. Without a proper kernel function in the GP prior, we can not expect to obtain good results. Consider the data in figure 10. Using a linear kernel, we obtain the results shown in figure 14, if we use a linear kernel instead of a Gaussian kernel. These results don't make any sense what so ever and have been provided to support the statement that the feature engineering in standard classification/regression approaches has been taken over by kernel engineering in case of Gaussian Process approaches.

5. Conclusion

Along with the prediction at a test data point, Gaussian Processes come with the information of how uncertain the model is about the prediction. This makes it useful in applications where not just the prediction but the confidence of the model about its prediction also plays a major role. In fields like health care, risk prediction is often expected to be as accurate as possible. Hence the model should be given a chance to say it cannot predict rather than giving false alarms. This way, Gaussian Processes can help reducing the false positives and false negatives. It can also be clearly seen that the model doesn't overfit on the training data.

References

Bishop, Christopher M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

Murphy, Kevin P. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.

Rasmussen, Carl Edward and Williams, Christopher K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Appendix

Laplace Approximation Laplace Approximation aims to find a Gaussian approximation to the probability density defined over a set of continuous variables.
Let $p(z) = \frac{1}{Z} f(z)$ where $Z = f(z)$. To find Gaussian approximation $q(z)$ such that it is centered on a mode of the distribution $p(z)$, we use Taylor's expansion.

$$ln(f(z)) = ln(f(z_0)) - \frac{1}{2}A(z - z_0)^2 \tag{26}$$

where $A = -\frac{d^2}{dz^2} ln(f(z))$. Then the Gaussian approximation $q(z)$ is given by:

$$q(z) = \Big(\frac{|A|}{2\pi^d}\Big)^{\frac{1}{2}} exp\{-\frac{1}{2}(z - z_0)^\top A(z - z_0)\} \tag{27}$$

Finding Posterior distribution
likelihood: $p(t|w) = \prod_{n=1}^{N} y_n^{t_n}(1 - y_n)^{1-t_n}$
prior (Gaussian): $p(w) = \mathcal{N}(w \mid m_0, S_0)$
posterior is given by:

$$p(w \mid t) \propto p(w)p(t \mid w) \tag{28}$$

$$ln(p(w \mid t) = ln(p(w)) + ln(p(t \mid w)) \tag{29}$$

$$= -\frac{1}{2}(w - m_0)^\top S_0^{-1}(w - m_0) \tag{30}$$

$$+ \sum_{n=1}^{N}\{t_n ln(y_n) + (1 - t_n)ln(1 - y_n)\} + c$$

where $y_n = \sigma(w^\top \phi_n)$.
Now we find the MAP estimate which gives the mean the negative hessian of the log likelihood gives the covariance.

$$\hat{w} = \underset{w}{\operatorname{argmax}}\, ln(p(w \mid t)) \tag{31}$$

$$S_N = -\nabla\nabla ln(p(w \mid t)) \tag{32}$$

$$= S_0^{-1} + \sum_{n=1}^{N} y_n(1 - y_n)\phi_n\phi_n^\top \tag{33}$$

Therefore Gaussian approximation to the posterior is

$$q(w) = \mathcal{N}(w \mid w_{MAP}, S_N) \tag{34}$$