AIML - CS 337

Lecture 15: Kernels and Gaussian Processes

6th October 2022

Lecturer: Abir De *Scribe: Group 31 and Group 32*

1 Prologue

By now, we have studied various kernel tricks which can be for separating data having non-linear relationship by simply defining an appropriate Gram matrix representing the kernel. Further, the trick can be extended to non parametric regression[2], classification and PCA(kernel PCA[1]) as well. In this lecture we look at another application of kernels in the context of Gaussian Processes and how to deal with smaller training sets to still give fair results.

2 Problem

Consider the standard linear regression model as follows:

$$w^{\text{regression}} \rightarrow \min \left[\sum_{i \in D} (y_i - w^T x_i)^2 \right]$$

The solution to the above problem is:

$$w^{\text{regression}} = (\sum_{i \in D} x_i x_i^T)^{-1} \cdot (\sum_{i \in D} x_i y_i)$$

The predictions are made using function $f : \mathbb{R}^d \rightarrow \mathbb{R}$, $f(x_i) = w^T \cdot x_i$ Notice that when we substitute an input data from training set,

$$f(x_i) \neq y_i$$

An alternative approach to this could be to obtain a distribution on the function we are trying to predict such that every point in the training data must have exactly the same output in the hypothesis as the training label. More precisely, we would like to design a non linear estimator f to model the training data with the additional restriction that $\forall x_i \in D \; f(x_i) = y_i$; for the other points $x \notin D$, $f(x)$ is a random variable with an associated probability distribution, while having certain guarantees on accuracy on test set and assuming train and test set are from same distribution. We can visualise such a function as shown in the figure below:

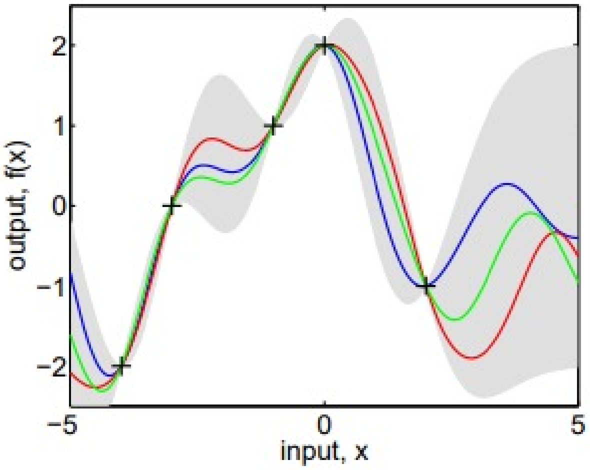

Figure 1: Graphical Representation

Here in this figure you can see that the points marked as + are the points in our dataset, for which the output is exactly one value while it is a distribution (as given by the shaded area) for all the other points

3 Gaussian Process

Gaussian processes are a method for non parametric estimation to provide confidence on the seen data and some kind of distribution on unseen data. For any subset of the training data, we must have that the joint prior distribution of this subset is normally distributed for some mean and covariance matrix. For any subset $\{x_1 ... x_m\}$ of the training data, the prior distribution follows:

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \quad \sim \quad \mathcal{N}(\vec{\mu}(x_1, \ldots, x_m), \Sigma(x_1, \ldots, x_m))$$

where $\vec{\mu}$ and Σ are deterministic functions.

On introducing a new data point into any subset of the training data, we expect the resulting conditional distribution to also follow the normal distribution. For the data point x^*

$$f(x^*)|(f(x_1), \ldots f(x_m), x^*) \sim \mathcal{N}(\vec{\mu}(x_1, \ldots, x_m, x^*), \Sigma(x_1, \ldots, x_m, x^*))$$

As described earlier, we expect that if a new data point introduced is already in the training data, then we expect that $f(x^*)$ takes the value that was present in the training set. This means that for any x^* such that $x^* \in \{x_1, \ldots, x_m\}$

$$f(x^*)|(f(x_1), \ldots f(x_m), x^*) \sim \mathcal{N}(f(x^*), 0)$$

Our aim is to design a matrix Σ that satisfies such a property i.e. posterior for any point in training data must have zero variance.

Let $X_D = [X_1^d X_2^d \dots X_n^d]^T$ denote the points in train set and $X_T = [X_1^t X_2^t \dots X_n^t]^T$ denote the points in test set. $f(X_D)$ and $f(X_T)$ denote random variables depending on the input.

$$f(X) = \begin{pmatrix} f(X_1^d) \\ f(X_2^d) \\ . \\ . \\ f(X_n^t) \end{pmatrix}$$

These random variables are dependent on each other, and we need to model the dependency between them. We model $f(X)$ as multi-variate Gaussian distribution. Therefore,

$$\begin{bmatrix} f(X_D) \\ f(X_T) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(X_D) \\ m(X_T) \end{bmatrix}, \begin{bmatrix} k(X_D, X_D) & k(X_D, X_T) \\ k(X_D, X_T)^T & k(X_T, X_T) \end{bmatrix} \right)$$

Here, $m(\cdot)$ is a function denoting mean, and $k(\cdot, \cdot)$ is a kernel function used for creating the covariance matrix. The reason we are using kernel function here is because we want to model some sort of similarity between the random variables, high correlation implying higher similarity. With this model, we can determine the prior distribution of the random variables $f(\cdot)$

$$P(f(X)) = P\left(\begin{bmatrix} f(X_D) \\ f(X_T) \end{bmatrix} \right) = \frac{1}{(\text{some constant}) \cdot \det(K)^{0.5}} \exp(-0.5 f(X)^T \Sigma^{-1} f(X))$$

Our aim now is to get distribution of $f(X_T)$ given the train predictions $\{y_i\}$ and X_T. for which we can use Bayes' rule.

3.1 Evaluating the posterior

We would model the predictions $Y = [y_1 y_2 \dots y_n]^T$ as

$$Y = I \cdot f(X_D)$$

A more general model would be to include additive Gaussian noise such that $Y = f(X_D) + \eta$. But for simplicity, we assume noise to be zero. Before the actual derivation, we note two important results.

3.1.1 Gaussian Marginalisation Rule

If we marginalize out variables in a multivariate Gaussian distribution, the result is still a Gaussian distribution. Mathematically, if $X = [X_1, X_2, \dots X_n]^T$ is a multi-variate Gaussian random variable($\sim \mathcal{N}(\mu, \sigma)$), then any subset of X is a multi-variate Gaussian and the mean and covariance is given by $(A\mu, A\sigma A^T)$. A can be constructed by using e_i^T as rows. For example, if $n = 3$, and the subset is constructed using $[X_1 \; X_3]$, then,

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3.1.2 Conditional Rule for multi-variate Gaussian

Intuitively, if we start with a Gaussian distribution and update our knowledge given the observed value of one of its components(that is, find conditional probability distribution), then the resulting distribution is still Gaussian! Mathematically,
Let $[x\ y]$ jointly form multi variate Gaussian random variable,

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix}\right)$$

Here Σ_{ab} represents covariance matrix between random vectors a and b and $f(\cdot)$ represents the PDF.

$$f(x|y) = \frac{f(x,y)}{f(y)}$$

Now, we will substitute $f(x,y)$ with the expression for multi-variate Gaussian distribution($\mathcal{N}(\mu,\Sigma)$), and $f(y)$ with $\mathcal{N}(\mu_y,\Sigma_{yy})$. Simplifying the equations, we get

$$f(x|y) = \mathcal{N}(\Sigma_{xy}\Sigma_{yy}^{-1}y, \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx})$$

(μ is assumed to be zero for simplicity)
To get the detailed derivation, please see [3]

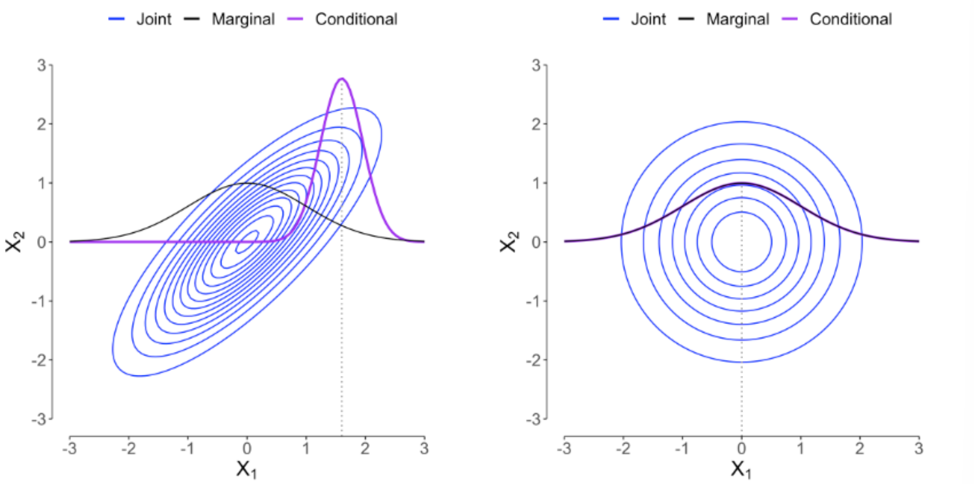

Figure 2: Joint, Marginal, Conditional for bivariate Gaussian. Source[3]

3.2 Getting to the posterior

We modelled $Y = f(X_D)$, now's the time to use it.

$$\begin{bmatrix} f(X_T) \\ Y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(X_T) \\ \mu(X_D) \end{bmatrix}, \begin{bmatrix} k(X_T, X_T) & k(X_T, X_D) \\ k(X_T, X_D)^T & k(X_D, X_D) \end{bmatrix}\right)$$

Using the conditional rule described above,

$$P(f(X_D)|Y) = \mathcal{N}(\mu_{posterior}, \sigma^2_{posterior})$$

where

$$\mu_{posterior} = \mu X_T + k(X_T, X_D)(k(X_D, X_D))^{-1}(Y - \mu(X_D))$$
$$\sigma^2_{posterior} = k(X_T, X_T) - k(X_T, X_D)(k(X_D, X_D))^{-1}k(X_T, X_D)^T$$

4 Aftermath[1]

.

4.1 Posterior mean

For the sake of investigation, let's assume $\mu(\cdot) = 0$.
Now, if $X_D = X_T$, $\mu_{posterior} = Y$ which is desirable because we want the mean for predictions at training points to be the same as the given predictions in train set. The mean value at a single test location, say x_i^t, is a weighted sum of all the observations Y. The weights are defined by the kernel between the test location x_i^t and all training locations in X.

4.2 Posterior variance

Observe that if $X_D = X_T$, we get $\sigma_{posterior} = 0$. This means that the prediction for a point in train set is exactly the mean, which in turn is the Y of the training set.

5 Conclusion

We started with a train set $\{(x_i^d, y_i^d)\}$ and test inputs $\{x_i^t\}$, and devised a function that would yield no error for inputs which are in train set, and low errors on other points. The described method is particularly useful for low data situations. A detailed study of Gaussian processes can be found in the reference [4].

[1] pun intended (credits Group 31)

References

[1] *Kernel PCA*. URL: https://www.geeksforgeeks.org/ml-introduction-to-kernel-pca/.

[2] *Non-parametric regression*. URL: https://en.wikipedia.org/wiki/Nonparametric_regression.

[3] *Properties of multi-variate Gaussian*. URL: https://fabiandablander.com/statistics/Two-Properties.html.

[4] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.