CSE583/EE552 Pattern Recognition and Machine Learning: Project #1

Due on January 31st, 2022 at 11:59pm

PROFESSOR Yanxi Liu Spring 22

Anish Phule **asp5607@psu.edu**

Problem 1

MLE Estimator derivation

To find the closed form solution, we need to minimize the following Error function E(w)[1],

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n^2\} \tag{1}$$

where,

$$y(x_n, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + ...w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{2}$$

The standard approach would be differentiating the error function w.r.t. w and equating the expression to 0. However, since w here is a vector, we need to perform partial derivative w.r.t each element in w, i.e. w_i.

$$\frac{\delta E}{\delta w_i} = \frac{1}{2} * 2\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}\frac{\delta y}{\delta w_i} \tag{3}$$

for some i $\leq$ M,

$$\frac{\delta y}{\delta w_i} = x^i \tag{4}$$

$$\frac{\delta E}{\delta w_i} = \sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}x_n^i \tag{5}$$

Equating with zero,

$$0 = \sum_{n=1}^{N}\{\sum_{j=0}^{M} w_j x_n^j - t_n\}x_n^i \tag{6}$$

Converting the individual elements into vector forms,

$$0 = \{X^T X\}\mathbf{w}^* - X^T T \tag{7}$$

Final equation for $\mathbf{w}^*$

$$\mathbf{w}^* = \{X^T X\}^{-1} X^T T \tag{8}$$

MAP estimator derivation

For MAP estimator, we introduce a prior estimation p(w/α)[1].
We consider a Gaussian distribution for the prior. That is,

$$p(w|\alpha) = N(w|\alpha^{-1}I) \tag{9}$$

Using Baye's rule,

$$p(w|x,t) = \frac{p(t|x,w)p(w)}{p(t|x)} \tag{10}$$

$$p(w|\alpha) = \frac{p(\alpha|w)p(w)}{p(\alpha)} \tag{11}$$

from the above two equations,

$$p(w|x,t) = \frac{p(t|x,w,\beta)p(\alpha)p(w|\alpha)}{p(\alpha|w)p(t|x)} \tag{12}$$

Here, terms like p(α) and p(t/x) are constants. Hence, we can now change the equality sign into proportionality.

$$p(w|x,t,\alpha,\beta) \propto p(t|x,w,\beta)p(w|\alpha) \tag{13}$$

$$ln(p(w|x,t,\alpha,\beta)) = ln(p(t|x,w,\beta)) + ln(p(w|\alpha)) + constant \tag{14}$$

$$ln(p(w|x,t,\beta)) = \frac{-\beta}{2}\sum_{n=1}^{N}\{y(x_n,w) - t_n\}^2 + \frac{N}{2}ln\beta - \frac{N}{2}ln(2\pi) \tag{15}$$

$$ln(p(w|\alpha) = ln(N(w|0,\alpha^{-1}I)) = \frac{-\alpha}{2}w^Tw + constant \tag{16}$$

$$ln(p(w|x,t,\alpha,\beta)) = \frac{-\beta}{2}\sum_{n=1}^{N}\{y(x_n,w) - t_n\}^2 + \frac{-\alpha}{2}w^Tw + constant \tag{17}$$

for optimum $\mathbf{w}^*$, we must minimize this function, that is, differentiate w.r.t. $\mathbf{w}$ and set it to zero.

$$\frac{d(ln(p(w|x,t,\alpha,\beta)))}{d\mathbf{w}} = -\beta(\mathbf{w}^TX^TX - t^TX) - \alpha\mathbf{w}^T = 0 \tag{18}$$

$$\mathbf{w}^T(X^TX - \frac{\alpha}{\beta}I) = t^TX \tag{19}$$

Therefore,

$$\mathbf{w}^* = (X^TX - \lambda I)^{-1}X^Tt \tag{20}$$

Where $\lambda = \alpha/\beta$.

Results:

The following results are for the MLE and MAP regression models for N = 50 points and M = 1.

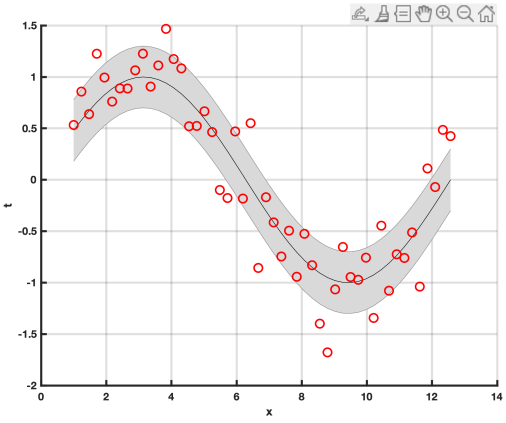

Figure 1: Original dataset

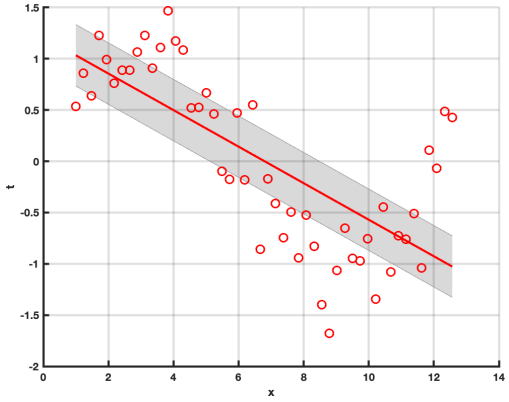

Figure 2: ML estimation model

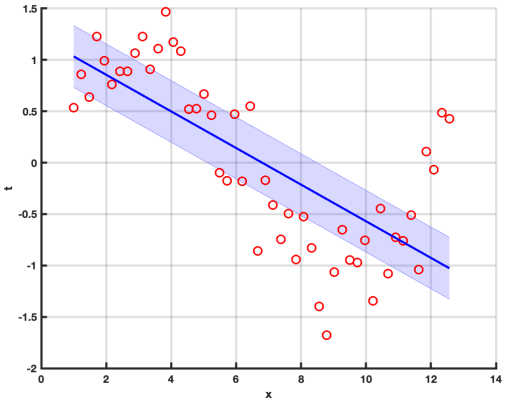

Figure 3: MAP estimation model

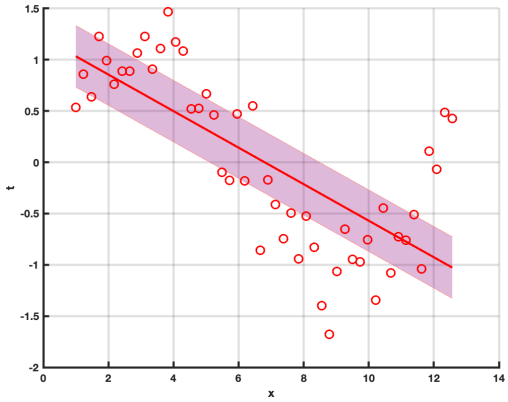

Figure 4: Comparison between the two model. For lower dimensions, there is very little or no difference.

The Bayesian models show approximate estimation of the original data from noisy data. For lower dimension, there isn't a lot of difference between the weights and therefore, our estimation through the two methods. However, when we increase the number of dimensions, the difference is clear. The MAP estimation is a lot smoother and closer to the actual curve (as seen in figure 5) as compared to the ML estimation. This is because of the introduction of the regularization in the error function, that controls the function from reaching large value, and therefore preventing phenomena like over-fitting.

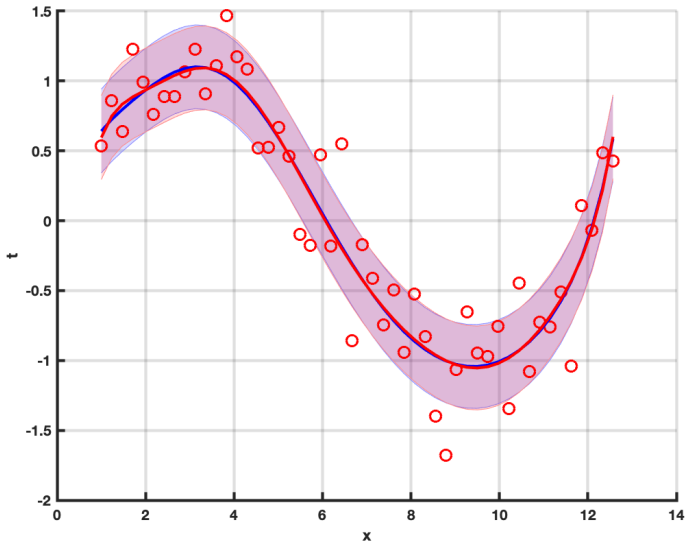

Figure 5: Comparison between ML(red) and MAP(blue) models for M=9. The ML model is clearly influenced by data, and therefore less accurate than MAP.

Problem 1 Extra Credits:

1. Plot of Errors vs lnλ

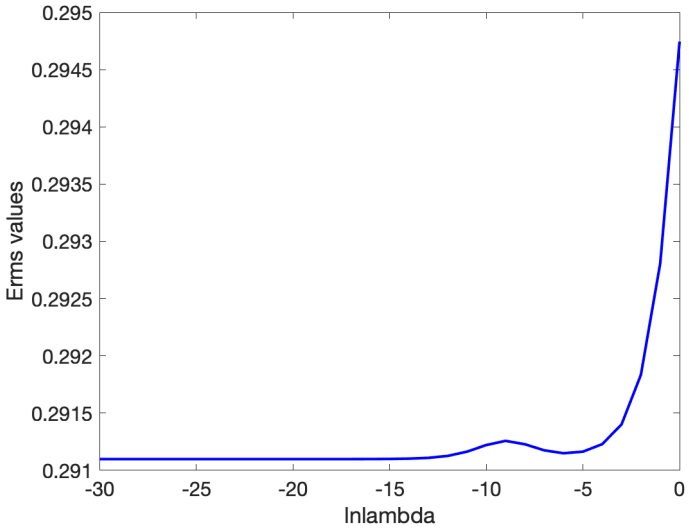

Figure 6: Erms vs lnλ

2. Table of the coefficients w* for polynomials of various order.

   - w* for various order polynomials for ML estimator

	M=0	M=1	M=3	M=6	M=9
w_0^*	0.0033	1.2121	-0.3072	0.9829	-2.8299
w_1^*		-0.1782	1.1198	-0.9751	8.0402
w_2^*			-0.2596	0.9075	-7.4567
w_3^*			0.014	-0.2833	3.7856
w_4^*				0.0378	-1.1257
w_5^*				-0.0023	0.2034
w_6^*				$5.609*10^{-5}$	-0.0227
w_7^*					0.0015
w_8^*					$-5.756*10^{-5}$
w_9^*					$9.214*10^{-7}$

   - w* for various order polynomials for MAP estimator

	M=0	M=1	M=3	M=6	M=9
w_0^*	0.0033	1.2121	-0.3068	0.9529	-2.519
w_1^*		-0.1782	1.1196	-0.9305	1.8786
w_2^*			-0.2596	0.8843	-1.6665
w_3^*			0.014	-0.2777	0.9319
w_4^*				0.0371	0.2999
w_5^*				-0.0023	0.0559
w_6^*				$5.5088*10^{-5}$	-0.0063
w_7^*					$4.20*10^{-4}$
w_8^*					$-1.5613*10^{-5}$
w_9^*					$2.488*10^{-7}$

3. For fixed degree polynomial M=9, varying the sample size N.

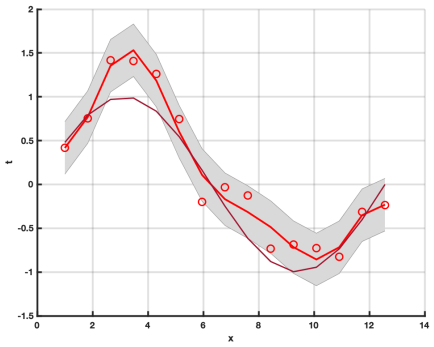

Figure 7: M = 15

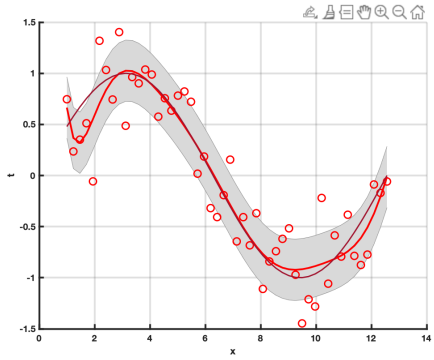

Figure 8: M = 50

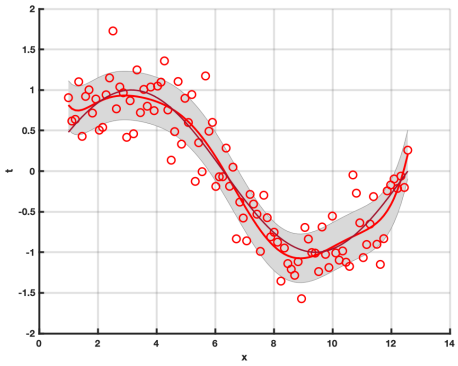

Figure 9: M = 100

Figure 10: For various orders of the polynomial, estimations done by the MLE model

As we see, increasing the number of points N makes the estimation better and closer to ground truth.

4. Curse of Dimensionality: The above examples prove that the curse of Dimensionality is indeed true. For M=9, for a lower N, the estimation is really poor, and even for N = 100, the estimation isn't exact. It still will give us lesser accurate values. The problem can be solved by increasing the number of data points. This however, is not always possible, hence the curse.

Problem 2

Classification

This problem deals with classification through the Fisher discriminant. For our case with multiple(K) classes, we define [1]

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \epsilon C_k} x_n \tag{21}$$

Where, N_k is the number of patterns in the class C_k.

$$\mathbf{m} = \frac{1}{N}\sum_{n=1}^{N} x_n = \frac{1}{N}\sum_{k=1}^{K} N_k \mathbf{m}_k \tag{22}$$

where $\mathbf{m}$ is the mean of the total data set.
We now therefore define,

$$S_k = \sum_{n \epsilon C_k} (x_n - \mathbf{m}_k)(x_n - \mathbf{m}_k)^T \tag{23}$$

$$\mathbf{S}_W = \sum_{k=1}^{K} S_k \tag{24}$$

$$\mathbf{S}_B = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T \tag{25}$$

In the above equations, $\mathbf{S}_W$ and $\mathbf{S}_B$ are the within-class and Inter-class covariances respectively[1]. Using these, we now define the Fisher Criterion J($\mathbf{w}$) as follows.

$$J(\mathbf{w}) = Tr(S_W^{-1} S_B) \tag{26}$$

The projection to lower dimensions is defined as,

$$\mathbf{y} = \mathbf{W}^T \mathbf{x} \tag{27}$$

The weight values $\mathbf{W}$ are defined by the the eigenvectors of $S_W^{-1} S_B$ corresponding to the largest eigenvalues[1].

Results:

- Confusion Matrices:

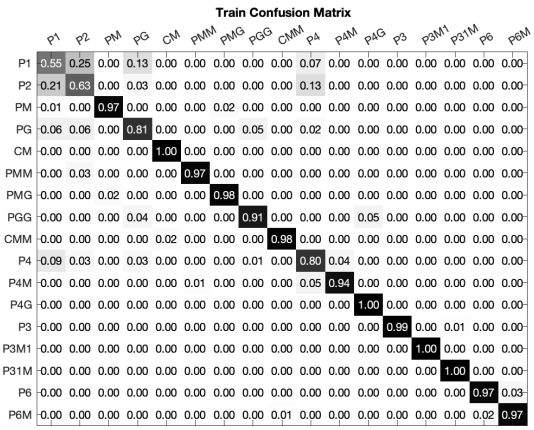

Figure 11: Train Confusion matrix for Wallpaper dataset

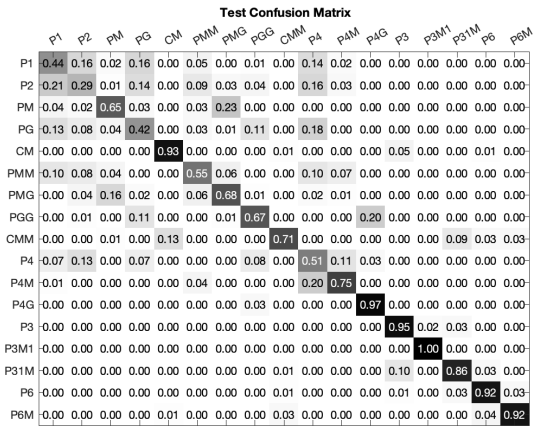

Figure 12: Test Confusion matrix for Wallpaper dataset

Problem 2 continued on next page...

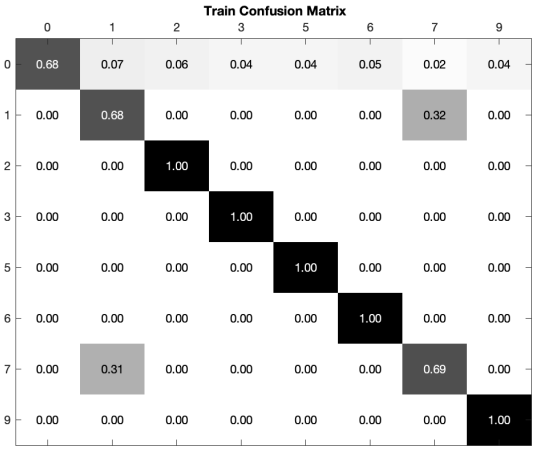

Figure 13: Train Confusion matrix for Taiji dataset

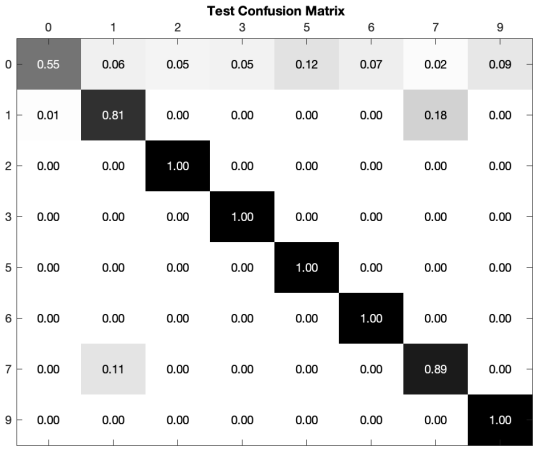

Figure 14: Test Confusion matrix for Taiji dataset

- Classification rates and overall Classification rate:
  - Wallpaper Dataset:

Class No.	Train rate	Test rate
1	0.55	0.44
2	0.63	0.29
3	0.97	0.65
4	0.81	0.42
5	1	0.93
6	0.97	0.55
7	0.98	0.68
8	0.91	0.67
9	0.98	0.71
10	0.80	0.51
11	0.94	0.75
12	1	0.97
13	0.99	0.95
14	1	1
15	1	0.86
16	0.97	0.92
17	0.97	0.92

Overall train rate: 91%
Overall test rate: 71.88%

Problem 2 continued on next page...

– Taiji Dataset:

Class No.	Train rate	Test rate
1	0.68	0.55
2	0.68	0.81
3	1	1
4	1	1
5	1	1
6	1	1
7	0.69	0.89
8	1	1

Overall train rate: 88.11%
Overall test rate: 90.63%

Analysis and Summary:

We see two kinds of datasets here, one with a lot of classes to segregate in, and one with less or reasonable number of classes.

The classification for the wallpaper dataset although has high training rate, has very low test rate. On the other hand, the classification for the Taiji dataset has good training as well as test rates. This shows us that increasing the number of dimensions can have a negative effect on the overall result, and we can clearly see overfitting in the wallpaper datatset classification. Also, the Taiji dataset has more number of data points, which makes the classification even better.
The Confusion matrices are a reflection of these. The classifier gives wrong result due to over-fitting, thus creating a kind of 'confusion', hence the filled confusion matrix in the wallpaper data.

Regarding Outliers:

In both the datasets there have to be some outliers. What outliers do is mislead the data into thinking it belongs to a different class, and therefore causing errors in data results and performance. For example, in the Class 0 of the Taiji data, while all other points are clearly classified, there is an error in the class aforementioned. In the wallpaper data, apart from the overfitting problem, we see some classes being severly affected, to which outliers can be the possible reason why.

References

[1] 'Pattern Recognition and Machine Learning book - Christopher M. Bishop'