Project 1

Prameth Gaddale (pqg5273@psu.edu)

January 30, 2023

Abstract

In this day and age, automation is crucial in various industries which involve laborious and repetitive tasks. This project covers regression and classification problems, which are considered to be one of basic techniques of *Pattern Recognition*. I've implemented linear models for regression, along with methods to optimize the parameters/weights of these models through the use of *Maximum Likelihood Estimation* and *Maximum A Posteriori Estimation*. Additonally, for classification models I've implemented the Fisher projection algorithm for the *Taiji* dataset. I've also introduced the key aspects of the *Central Limit Theorem*, in addition to answering few questions regarding estimating probability distributions.

Contents

1 Linear Regression

Supervised learning problems with the training data comprising of input vectors along with their corresponding target vectors, and the desired output consisting of one or more continuous variables is called *Regression*. Bishop[1] mentions about the use of polynomial curve fitting as an example to illustrate the usage and role of linear regression in real-world problems.
In polynomial curve fitting, we are given a training set (of N examples) of $\mathbf{x}$, where $\mathbf{x} \equiv (x_1, ..., x_N)^T$ and their corresponding target values $\mathbf{t} \equiv (t_1, ..., t_N)^T$.
Primarily, the objective of polynomial curve fitting is to find relevant parameters(represented by polynomial coefficients) $w_0, w_1, ...w_M$ that fit a polynomial function of degree M represented as 1,

$$y(x, \mathbf{w}) = w_0 + w_1 x + w2 + w^2 + ... + w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{1}$$

To arrive to our objective, we have to parameters that satisfy the minimization of an *error function*, that measures the magnitude of misfit between our targets and predictions.
A suitable criteria for optimizing the model parameters is to minimize the squared sum between the targets and predicted values, mentioned in 2.

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

The two most popular approaches to solve the error function and estimate the model parameters which take the uncertainty into consideration through the use of Bayesian statistics [2] are:

1. Maximum Likelihood Estimator
2. Maximum A Posteriori Estimator

1.1 Maximum Likelihood Estimator

In Maximum Likelihood Estimation, the main assumption is that the ground truth values (represented by t_i) follow a Gaussian distribution. The assumed distribution is spread over with mean value represented by the model prediction values, and the variance matrix (for multivariate datasets) as shown in (3).

$$p(t|x, \mathbf{w}, \beta) = \mathcal{N}(t|y(x, (w)), \beta^{-1}) \tag{3}$$

However, for a multivariate Gaussian distribution if we apply the i.i.d criterion which states that each example sampled from the dataset is independent and identically distributed we can write the joint conditional probability distribution in the product-form as (4)

$$p(t|x, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t|y(x, (w)), \beta^{-1}) \tag{4}$$

We also know that the standard Gaussian distribution is represented by (5),

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\} \tag{5}$$

Now taking the equation (4), applying the general equation (5) and taking the natural-logarithm across both the sides, we get,

$$\begin{aligned}
\ln p(t|x, \mathbf{w}, \beta) =& \sum_{n=1}^{N} \ln\{\mathcal{N}(t_n|y(x_n, \mathbf{w}), \beta^{-1})\} & (6)\\
=& \sum_{n=1}^{N} \ln\{\frac{1}{(2\pi\beta^{-1})^{1/2}} \exp\{-\frac{1}{2\beta^{-1}}\{t_n - y(x_n, \mathbf{w})\}^2\}\} & (7)\\
=& \sum_{n=1}^{N} \ln\{\frac{1}{(2\pi\beta^{-1})^{1/2}}\} + \sum_{n=1}^{N} \ln\{\exp\{-\frac{1}{2\beta^{-1}}\{t_n - y(x_n, \mathbf{w})\}^2\}\} & (8)\\
=& N \ln\{\frac{\beta}{(2\pi)}\}^{1/2} + \sum_{n=1}^{N} -\frac{\beta}{2}\{y(x_n, \mathbf{w}) - t_n\}^2 & (9)\\
\ln p(t|x, \mathbf{w}, \beta) =& -\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) & (10)
\end{aligned}$$

From statistics [2], we know that (10) is represented as log-likelihood and the first term on the right-hand-side, $-\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$ resembles the ***sum of the squared error***. Hence, maximizing the log-likelihood function is equivalent to minimizing the negative log-likelihood function.
Taking negative on both sides, we get the equation of negative log-likelihood which is used to

$$-\ln p(t|x, \mathbf{w}, \beta) = \frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi) \tag{11}$$

We can represent the error function of *Maximum Likelihood Estimator* by (12)

$$E_{ML}(\mathbf{w}) = \frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi) \tag{12}$$

One of the fundamental operation to find critical points of a function with respect to a specific parameter is to equate the partial derivative of the function to zero and solving for the parameter. Applying that strategy to equation (12) with respect to $\mathbf{w}$ we get,

$$\frac{\partial E_{ML}(\mathbf{w})}{\partial \mathbf{w}} = 0 \tag{13}$$

$$\frac{\partial}{\partial \mathbf{w}*}\left(\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2-\frac{N}{2}\ln\beta+\frac{N}{2}\ln(2\pi)\right) = 0 \tag{14}$$

$$\frac{1}{2}.2.\sum_{n=1}^{N}\frac{\partial y_n}{\partial \mathbf{w}_i}(y_n-t_n) = 0 \tag{15}$$

$$\sum_{n=1}^{N}\frac{\partial y_n}{\partial \mathbf{w}_i}(\sum_{j=0}^{M}w_j x_n^j - t_n) = 0 \tag{16}$$

$$\sum_{n=1}^{N}x_n^i(\sum_{j=0}^{M}w_j x_n^j - t_n) = 0 \tag{17}$$

$$\sum_{n=1}^{N}x_n^i\sum_{j=0}^{M}w_j x_n^j - \sum_{n=1}^{N}x_n^i t_n = 0 \tag{18}$$

$$\sum_{n=1}^{N}\sum_{j=0}^{M}x_n^i w_j x_n^j - \sum_{n=1}^{N}x_n^i t_n = 0 \tag{19}$$

$$\sum_{n=1}^{N}\sum_{j=0}^{M}x_n^{i+j} w_j - \sum_{n=1}^{N}x_n^i t_n = 0 \tag{20}$$

Converting the result from the summation-scalar format present in (20) to matrix-format we get,

$$\mathbf{X}^T\mathbf{X}\mathbf{w}_{ML} - \mathbf{X}^T t = 0 \tag{21}$$

$$\mathbf{X}^T\mathbf{X}\mathbf{w}*_{ML} = \mathbf{X}^T t \tag{22}$$

$$\mathbf{w}_{ML} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T t \tag{23}$$

Where, (23) represents the optimal weights for the polynomial with respect to minimizing the negative log-likelihood.
We can find the predictions of the model function by applying,

$$\mathbf{y} = \mathbf{w}_{ML}^T\mathbf{X} \tag{24}$$

Design Matrix Input

However, in real-life situations the input is in the form of multiple dimensions and is a function of the input vector either through pre-processing steps or through the use of higher dimensional polynomial for the model fit. In that case we can consider the design matrix in the form of $\phi(x_n)$ and modify our objective function as,

$$\nabla \ln p(t|w,\beta) = \nabla\left(\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\phi(x_n)\}^2\right) \tag{25}$$

Applying partial derivative with respect to $\mathbf{w}$ and solving for $\mathbf{w*}$ we get

$$\frac{\partial E_M L(\mathbf{w})}{\partial \mathbf{w*}} = 0 \tag{26}$$

$$\frac{\partial}{\partial \mathbf{w*}}\{\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\phi(x_n)\}^2\} = 0 \tag{27}$$

$$\sum_{n=1}^{N} t_n\phi(x_n)^T - \mathbf{w}^T\left(\sum_{n=1}^{N}\phi(x_n)\phi(x_n)^T\right) = 0 \tag{28}$$

$$\mathbf{w}^T\left(\sum_{n=1}^{N}\phi(x_n)\phi(x_n)^T\right) = \sum_{n=1}^{N} t_n\phi(x_n)^T \tag{29}$$

$$\mathbf{w}^T(\Phi\Phi^T) = \mathbf{t}\Phi^T \tag{30}$$

$$\mathbf{w}^T = (\Phi\Phi^T)^{-1}\mathbf{t}\Phi^T \tag{31}$$

$$\mathbf{w}_{ML} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{t} \tag{32}$$

Hence we can find the predictions of the model function by applying,

$$y = \mathbf{w}^T\phi(x_n) \tag{33}$$

Similarly, we can find the optimal value of the parameter β by (34),

$$\beta_{ML}^{-1} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - \mathbf{w}_{ML}^T\phi(x_n)\}^2 \tag{34}$$

Hence, the distribution represented by the Gaussian distribution assumption gets converted through the use of optimal values as,

$$p(t|x, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(y(x, \mathbf{w}_{ML}), \beta_{ML}^{-1}) \tag{35}$$

1.2 Maximum A Posteriori Estimator

Unlike *Maximum Likelihood Estimation* where the posterior distribution on the ground truth was assumed to be from a Gaussian distribution, Maximum A Posteriori Estimator introduces a prior distribution on the weight vector $\mathbf{w}$. That could be written mathematically as in (36). Using the general Gaussian distribution equation and applying natural logarithm to (39).

$$p(\mathbf{w}|\alpha) = \mathcal{N}((w)|0, \alpha^{-1}\mathbf{I}) \tag{36}$$

$$= \frac{1}{(2\pi\alpha^{-1})^{1/2}}\exp\{-\frac{1}{2\alpha^{-1}}(\mathbf{w} - 0)^2\} \tag{37}$$

$$p(\mathbf{w}|\alpha) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2}\exp\left\{-\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\} \tag{38}$$

$$\ln p(\mathbf{w}|\alpha) = \frac{(M+1)}{2}.\ln\left(\frac{\alpha}{2\pi}\right) - \left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\} \tag{39}$$

From probability theory we know that,

$$\begin{aligned}
\textit{Posterior} &\propto \textit{Likelihood} * \textit{Prior} && (40)\\
p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) &\propto p(\mathbf{t}|\mathbf{x},\mathbf{w},\beta)p(\mathbf{w}|\alpha) && (41)\\
\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) &\propto \ln p(\mathbf{t}|\mathbf{x},\mathbf{w},\beta) + \ln p(\mathbf{w}|\alpha) && (42)\\
\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) &= \ln p(\mathbf{t}|\mathbf{x},\mathbf{w},\beta) + \ln p(\mathbf{w}|\alpha) + \textit{constant} && (43)
\end{aligned}$$

Substituting the equations (10) and (39) in (43) we get,

$$\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) = \left(-\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2 + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi)\right) + \left(\frac{(M+1)}{2}.\ln\left(\frac{\alpha}{2\pi}\right) - \left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\}\right) + \textit{constant} \quad (44)$$

$$\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) = -\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2 + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) + \frac{(M+1)}{2}.\ln\left(\frac{\alpha}{2\pi}\right) - \left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\} + \textit{constant} \quad (45)$$

$$-\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) = \frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi) - \frac{(M+1)}{2}.\ln\left(\frac{\alpha}{2\pi}\right) + \left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\} + \textit{constant} \quad (46)$$

Ignoring the constants (for simplicity as they would be equate to zero on taking the partial derivative with respect to **w**) from (46) we can find the maximum of the posterior by minimizing the following equation,

$$-\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta) = \frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2 + \left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\} \quad (47)$$

$$\nabla\left(-\ln p(\mathbf{w}|\mathbf{x},\mathbf{t},\alpha,\beta)\right) = 0 \tag{48}$$

$$\nabla\left(\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2+\left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\}\right) = 0 \tag{49}$$

$$\nabla\left(\frac{\beta}{2}\sum_{n=1}^{N}\{\mathbf{w}^T\mathbf{X}-t_n\}^2+\left\{\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\right\}\right) = 0 \tag{50}$$

$$2*\frac{\beta}{2}\left(\sum_{n=1}^{N}\{\mathbf{w}^T\mathbf{X}-t_n\}^T\mathbf{X}\right)+\left\{2*\frac{\alpha}{2}\mathbf{w}\right\} = 0 \tag{51}$$

$$\beta(\mathbf{X}^T\mathbf{X}\mathbf{w}-\mathbf{X}^T\mathbf{t})+\alpha\mathbf{w} = 0 \tag{52}$$

$$\beta\mathbf{X}^T\mathbf{X}\mathbf{w}-\beta\mathbf{X}^T\mathbf{t}+\alpha\mathbf{w} = 0 \tag{53}$$

$$\mathbf{X}^T\mathbf{X}\mathbf{w}+\frac{\alpha}{\beta}\mathbf{w} = \mathbf{X}^T\mathbf{t} \tag{54}$$

$$\mathbf{w} = \left(\mathbf{X}^T\mathbf{X}+\frac{\alpha}{\beta}\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{t} \tag{55}$$

$$\mathbf{w}_{MAP} = \left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{t} \tag{56}$$

We can observe that, this optimization criterion is equivalent to minimizing the regularized sum-of-squares error, with the parameter $\lambda = \alpha/\beta$.

1.3 Report

1.3.1 Dataset

Given dataset (shown in 1) consists of input values(x), ground truth values(y), target values(t) and random noise values($sigma$) added to ground truth values to produce targets.

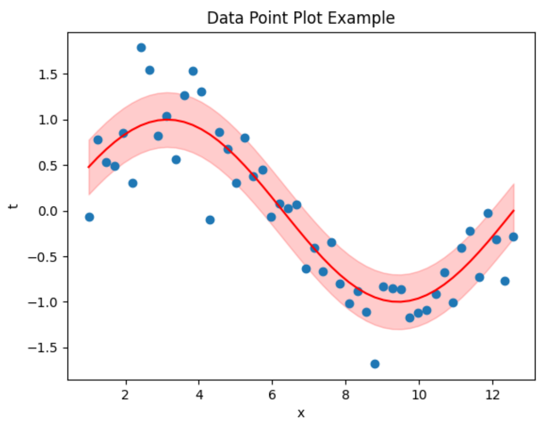


Figure 1: Given training dataset consists of (x), (y), (t) and($sigma$) values.

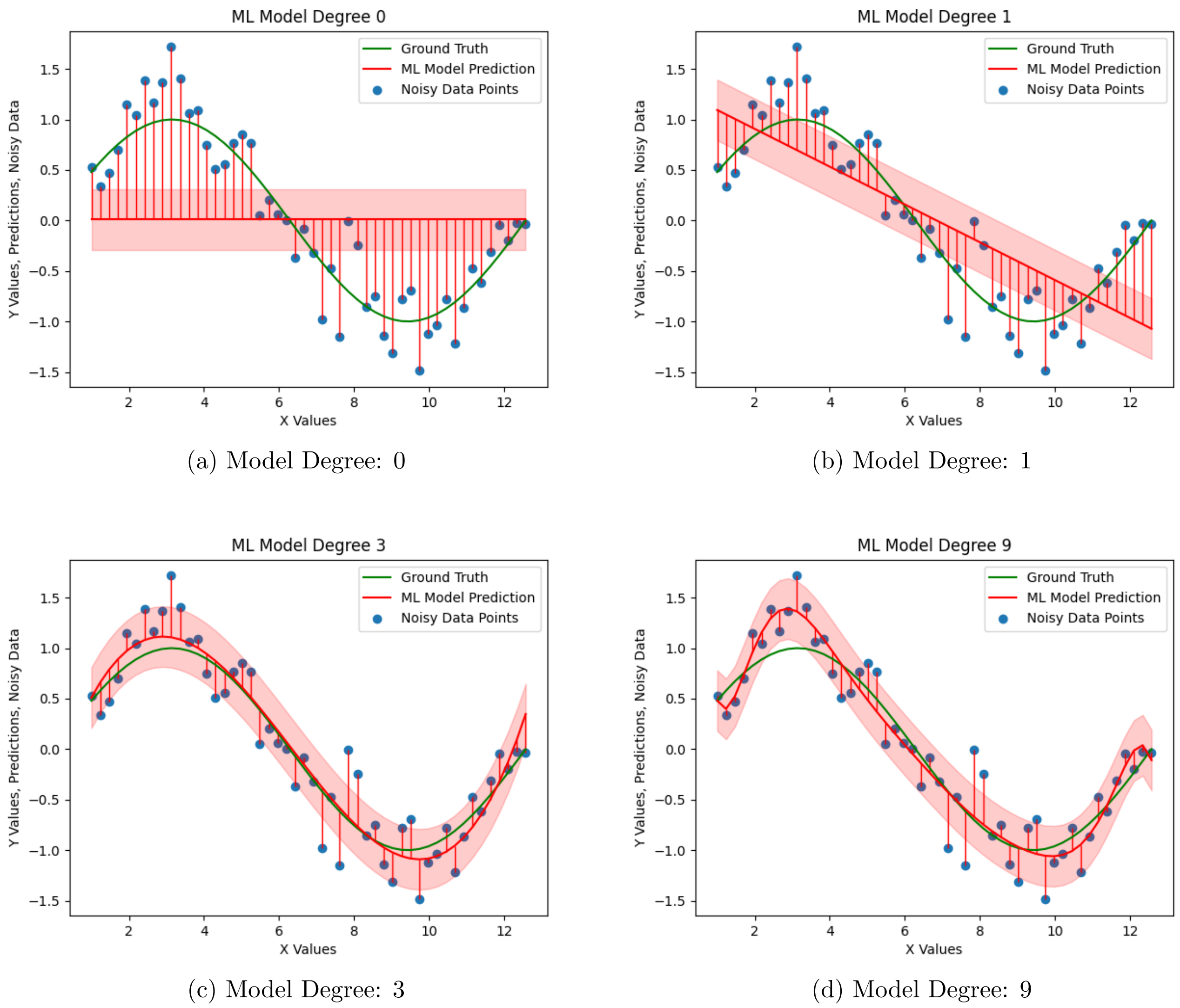
- The ground truth values of the curve represent the equation $y = sin(0.5 * x)$.
- The target values of the dataset are formed by adding a Gaussian noise to the ground truth points.

- The shaded part in 1 represents the Gaussian noise spanned over the ground truth.
- The given dataset has variable *num_points* that could be used to change the number of points used for the experiment.
- The generated datasamples are stored in *.npz* format.

1.3.2 Visualization

Maximum Likelihood Estimation

For the dataset with 50 training data points, here are the results observed for the *Maximum Likelihood Estimation* Model.



(a) Model Degree: 0

(b) Model Degree: 1

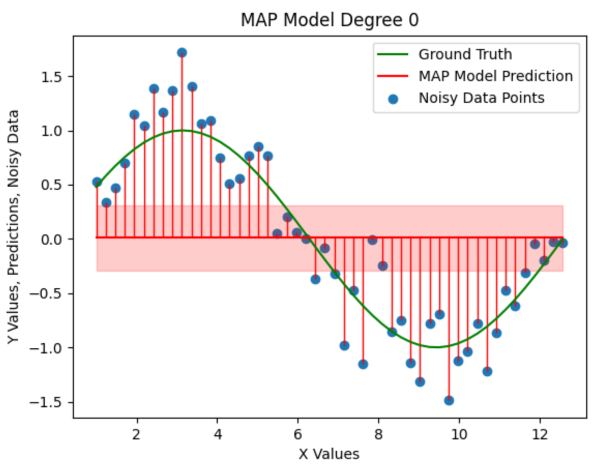
(c) Model Degree: 3

(d) Model Degree: 9

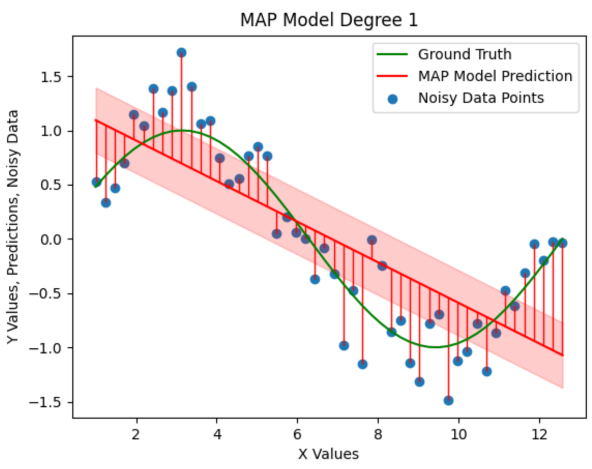
Figure 2: Maximum Likelihood Estimation Model Prediction Visualization

Maximum A Posteriori Estimation

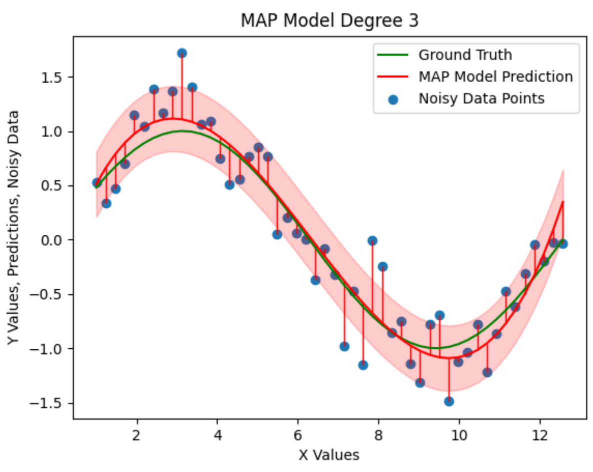
For the dataset with 50 training data points, here are the results observed for the *Maximum A Posteriori Estimation* Model.



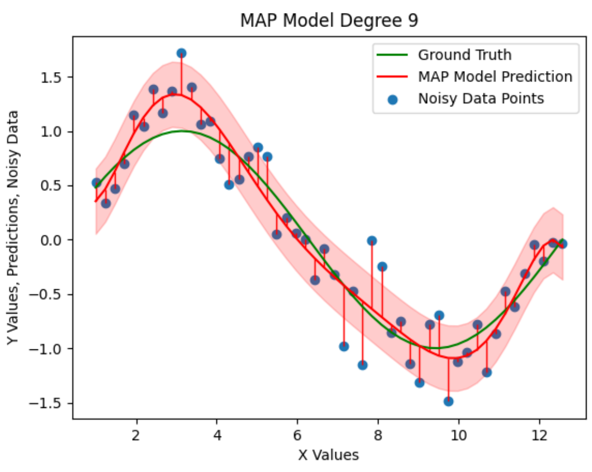
(a) Model Degree: 0



(b) Model Degree: 1



(c) Model Degree: 3

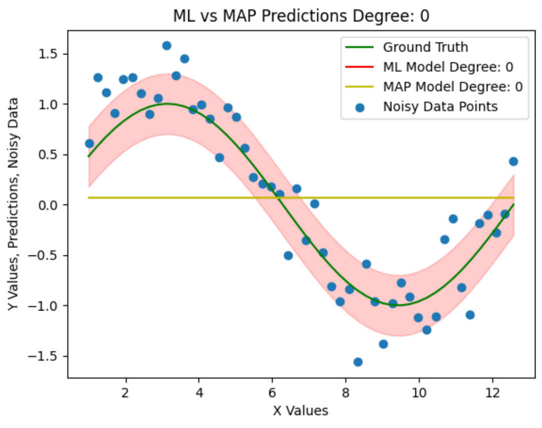


(d) Model Degree: 9

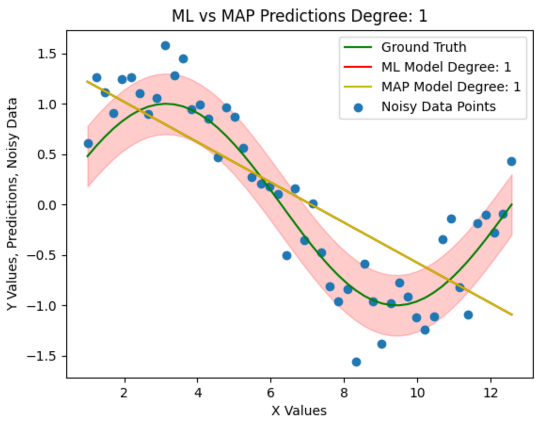
Figure 3: Maximum A Posteriori Estimation Model Prediction Visualization

1.3.3 Comparison

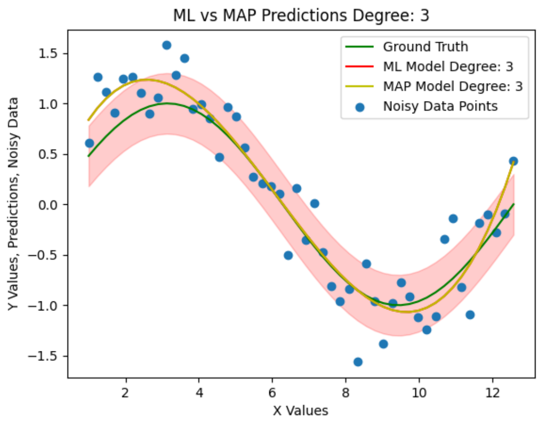
In 4, comparisons of the Maximum Likelihood Model and the Maximum A Posteriori Model are observed for varying model degrees corresponding to 50 data points.



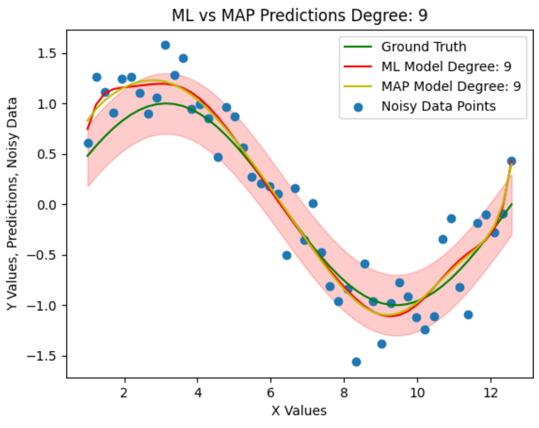
(a) Model Degree: 0



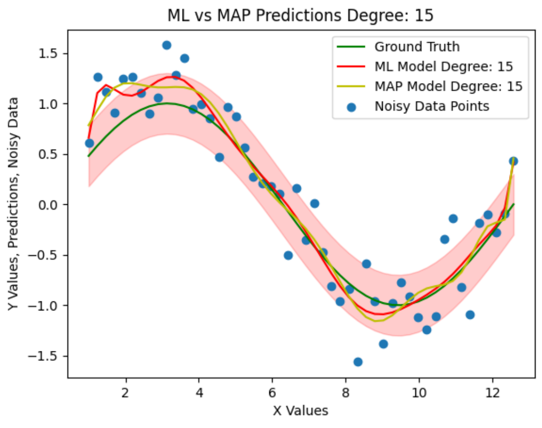
(b) Model Degree: 1



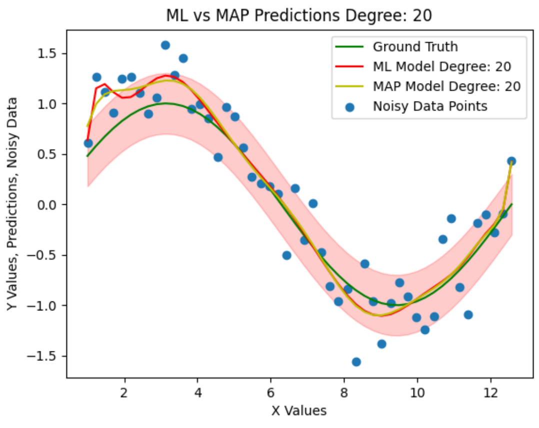
(c) Model Degree: 3



(d) Model Degree: 9



(e) Model Degree: 15



(f) Model Degree: 20

Figure 4: Maximum A Posteriori Estimation Model Prediction Visualization

1.3.4 Summary

- Predictions of both the *Maximum Likelihood Estimation* and *Maximum A Posterirori* models have been made and plotted.

- From 2, we can observe the polynomial degrees 0 and 1 weren't good fits of the underlying ground truth curve. The polynomial function representing were lines, and weren't able to capture the data distribution accurately.

- However, in 2 we know that polynomial with degree 3 was able to provide a good fit for the data distribution, being *just right* for the data modelling. But on the other hand, the polynomial with degree 9 was over-fitting the ground truth curve, but was trying to go through most of the target data points.

- Similar behaviour is observed in the case of *Maximum A Posteriori Estimation* model as shown in 3.

- Higher the model degree, the model fits the noisy data pretty well. However, when it comes to degree 9 polynomial, it overfits to the examples and isn't able to generate a proper sine-like curve to capture the ground truth distribution.

- Hence, in this case as well, the model degree 3 performed most optimally in terms of fitting to the noisy data distribution.

- Similarly, we can observe the model based comparison based on the varying degrees as shown in 4. We can see that the MAP model has smoothed out the overfit-effects at higher order compared to ML model which tries to capture every datapoint in the target space.

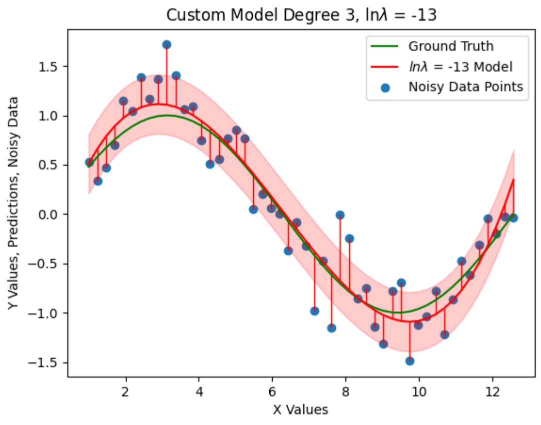
- Hence, we can say that higher order degrees for polynomials have overfitted the noisy training data, especially in the case of *Maximum Likelihood Estimation*. However, *Maximum A Posteriori Estimation* did a good job of smoothing out the noise at a certain extent without letting overfit the data distribution.

- This is because of the inherent regularization being applied by MAP model.

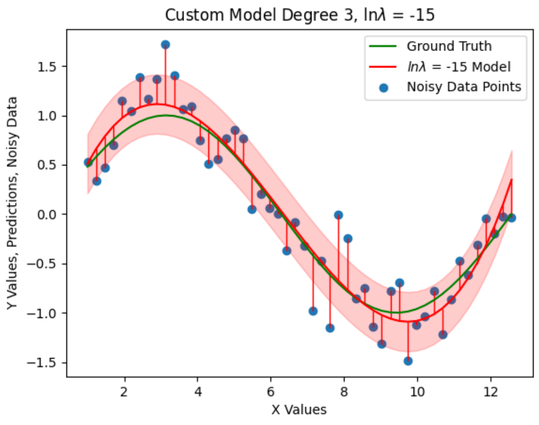
1.4 Extra Credit

1.4.1 Additional Lambda Values

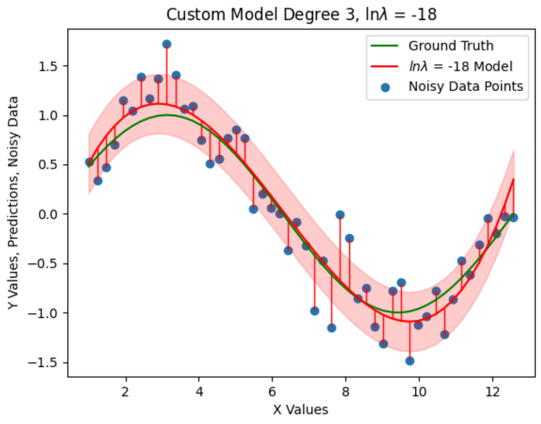
The results for the regularized custom model with varying values is shown in 5.
We can observe that higher the value of regularization, the curve tends to become more linear and penalizes the weight vectors at a higher extent.



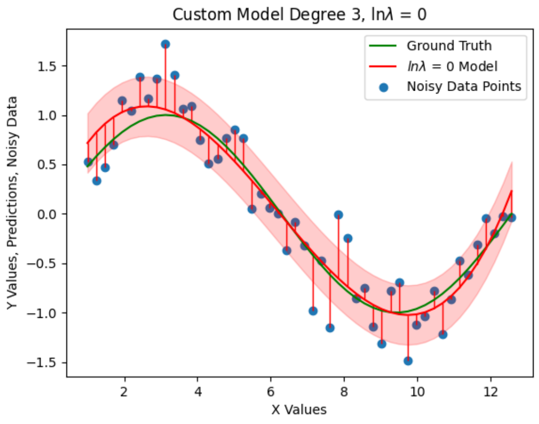
(a) Regularized Model $\ln \lambda = -13$



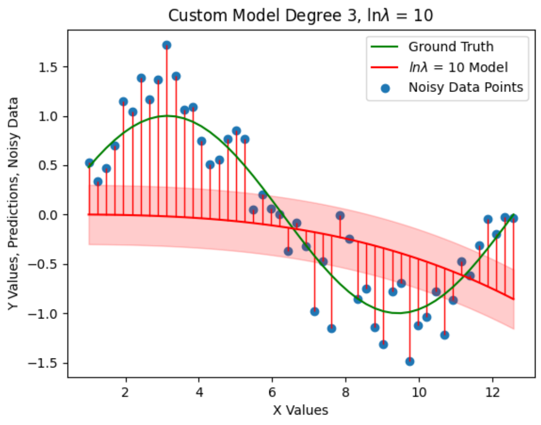
(b) Regularized Model $\ln \lambda = -15$



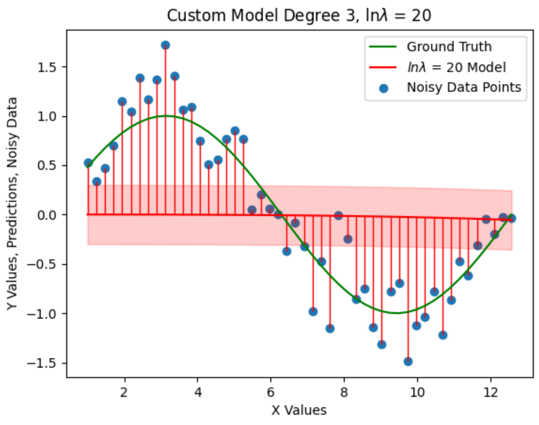
(c) Regularized Model $\ln \lambda = -18$



(d) Regularized Model $\ln \lambda = 0$



(e) Regularized Model $\ln \lambda = 10$



(f) Reguralized Model $\ln \lambda = 20$

Figure 5: Additional lambda values model predictions

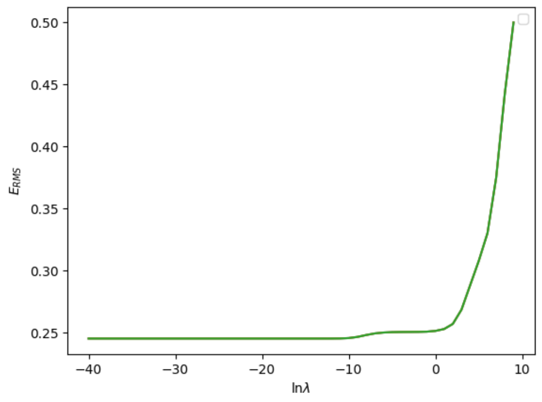


Figure 6: Root Mean Squared Error vs. $\ln\lambda$ plot.

The plot 6 represents the plot of Root Mean Squared Error vs $\ln\lambda$ values. For higher values of , models tends to underfit the training distribution and produces worse training error.

1.4.2 Varying Degree of Polynomial Order

The table 1 represents the weights of the Maximum Likelihood Model with varying degrees.

Weights	**M = 0**	**M = 1**	**M = 3**	**M = 6**	**M = 9**
w_0^{ML}	0.009549	1.204800	-0.185197	-0.713196	-1.367908
w_1^{ML}	-	-0.176208	0.958219	1.860958	4.790468
w_2^{ML}	-	-	-0.221176	-0.794732	-4.971345
w_3^{ML}	-	-	0.011743	0.184684	2.956968
w_4^{ML}	-	-	-	-0.026116	-1.027571
w_5^{ML}	-	-	-	0.001895	0.214036
w_6^{ML}	-	-	-	-0.000052	-0.027113
w_7^{ML}	-	-	-	-	0.002043
w_8^{ML}	-	-	-	-	-0.000084
w_9^{ML}	-	-	-	-	0.000001

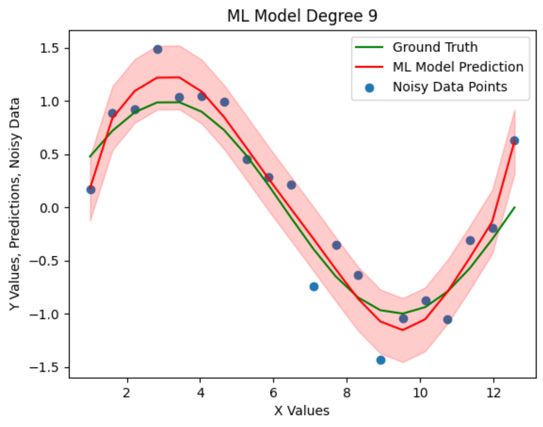
Table 1: Weights of the Maximum Likelihood model with varying degrees.

The table 2 represents the weights of the Maximum A Posteriori Model with varying degrees.

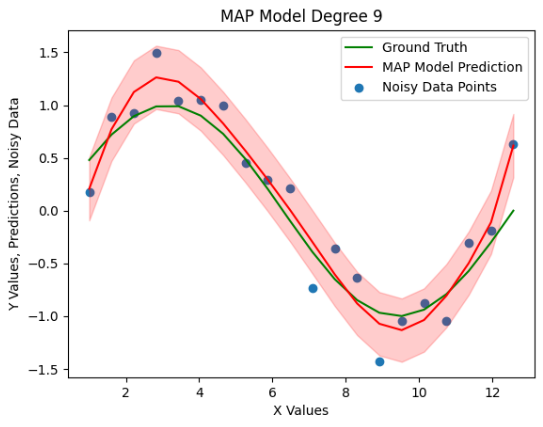
- We observe that Maximum Likelihood estimation model weights are higher compared to MAP model weights.
- The higher order coefficient are nearly non-existent in the case of the MAP model.

Weights	M = 0	M = 1	M = 3	M = 6	M = 9
w_0^{MAP}	0.009549	1.204745	-0.184890	-0.674439	0.263706
w_1^{MAP}	-	-0.176201	0.958032	1.803124	0.875506
w_2^{MAP}	-	-	-0.221146	-0.764588	-1.278764
w_3^{MAP}	-	-	0.011742	0.177347	1.131148
w_4^{MAP}	-	-	-	-0.025209	-0.497739
w_5^{MAP}	-	-	-	0.001839	0.119170
w_6^{MAP}	-	-	-	-0.000051	-0.016513
w_7^{MAP}	-	-	-	-	0.001324
w_8^{MAP}	-	-	-	-	-0.000057
w_9^{MAP}	-	-	-	-	0.000001

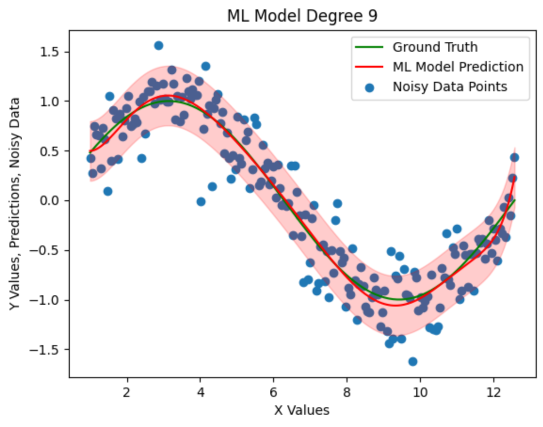
Table 2: Weights of the Maximum A Posteriori model with varying degrees.



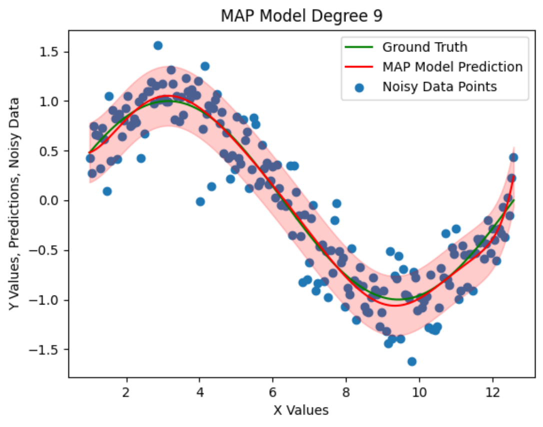
(a) ML Model, 20 Data Points



(b) MAP Model, 20 Data Points



(c) ML Model, 200 Data Points



(d) MAP Model, 200 Data Points

Figure 7: Maximum Likelihood and Maximum A Posteriori Estimation Model Prediction with varying data points.

1.4.3 Varying Number of Sample Points

Higher the data points used for fitting the polynomial, the model is able to perform better on capturing the underlying ground truth curve as shown in 7.
The results in 7 suggest that the gap between the ML and MAP model gets reduced with the increase in the data set size. The ML model performs much better (in terms of the model fit, the 20 dataset size proved to have under-fitted the ML model).

2 Classification

Unlike regression problems, the objective of classification models is to assign input vector $\mathbf{x}$ to one of K discrete classes where $k = 1, ..., K$. The *decision regions* or *decision boundaries* therefore represent the division of the input space, which are used to classify each input into disjoint classes.
Fisher Linear Discriminant Analysis is considered to be a part of the family of linear models used for classification, where the decision boundaries are linear functions of the provided input vector $\mathbf{x}$.

2.1 Fisher Projection

In *Fisher Linear Discriminant*, the algorithm tries to project the data from a high-dimensional space to a lower-dimensional space. However, inherently due to drop of dimensions, some of the key information is lost. The main goal of the algorithms is to find a **projection to a decision subspace that maximizes the *class separation***.
In Fisher projection, concurrently 2 operations are performed which could be considered as our objective function,

- **Maximizing the distance between projected inter-class means.**
- **Minimizing the distance between projected intra-class variances.**

However, optimizing those functions individually is not possible. Hence, the goal is defined to maximize their ratio. They are mathematically viewed as following,
We tend to construct a Fisher project matrix $\mathbf{W}$ which aids int he transformation of the input data $\mathbf{x}$ as shown in (57).

$$\mathbf{y} = \mathbf{W}^T\mathbf{x} \tag{57}$$

Applying the intra-class variance over all the given K classes, we get (58)

$$\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k \tag{58}$$

where of each class, the matrix $\mathbf{S}_k$ is represented by,

$$\mathbf{S}_k = \sum_{n \in C_k} (x_n - \mathbf{m}_k)(x_n - \mathbf{m}_k)^T \tag{59}$$

If we have only two classes we get,

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{n\in C_1} x_n, \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{n\in C_2} x_n \tag{60}$$

For multiple classes we have,

$$\mathbf{m}_k = \frac{1}{N_k}\sum_{n\in C_k} \mathbf{x}_n \tag{61}$$

The global mean of the given data is given by,

$$\mathbf{m} = \frac{1}{N}\sum_{k=1}^{K} N_k\mathbf{m}_k \tag{62}$$

Where the term $N = \sum_k N_k$ represented the total number of data points. And the matrices $\mathbf{S}_T$ and $\mathbf{S}_B$ that aid in the construction of the projection matrix are given by,

$$\mathbf{S}_T = \sum_{n=1}^{N}(\mathbf{x}_n - \mathbf{m})(\mathbf{x}_n - \mathbf{m})^T \tag{63}$$

$$\mathbf{S}_B = \sum_{n=1}^{N} N_K(\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T \tag{64}$$

Now, to find the most optimal $\mathbf{W}$ we find a criterion to minimize using the equations (63) and (64) as,

$$J(\mathbf{W}) = Tr\left((\mathbf{W}\mathbf{S}_W\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{S}_B\mathbf{W}^T)\right) \tag{65}$$

Minimizing this criterion, $J(w)$ is done by finding the eigen-vectors of $W = S_W^{-1}S_B$.
Hence, we can choose the eigenvectors that correspond to the D' largest eigenvalues.
I've considered 8 most significant eigen values in the Code Implementation *fisher_projection*().

2.2 Report

Following is the report for the Fisher Projection based classification for the Taiji dataset.

2.2.1 Dataset

Given dataset is of *Taiji Poses* with 7 different poses and one transitional pose.

- The dataset is split between training and test sets.

- The Figure 8 represents the decision boundary of the given dataset without Fisher projection applied.

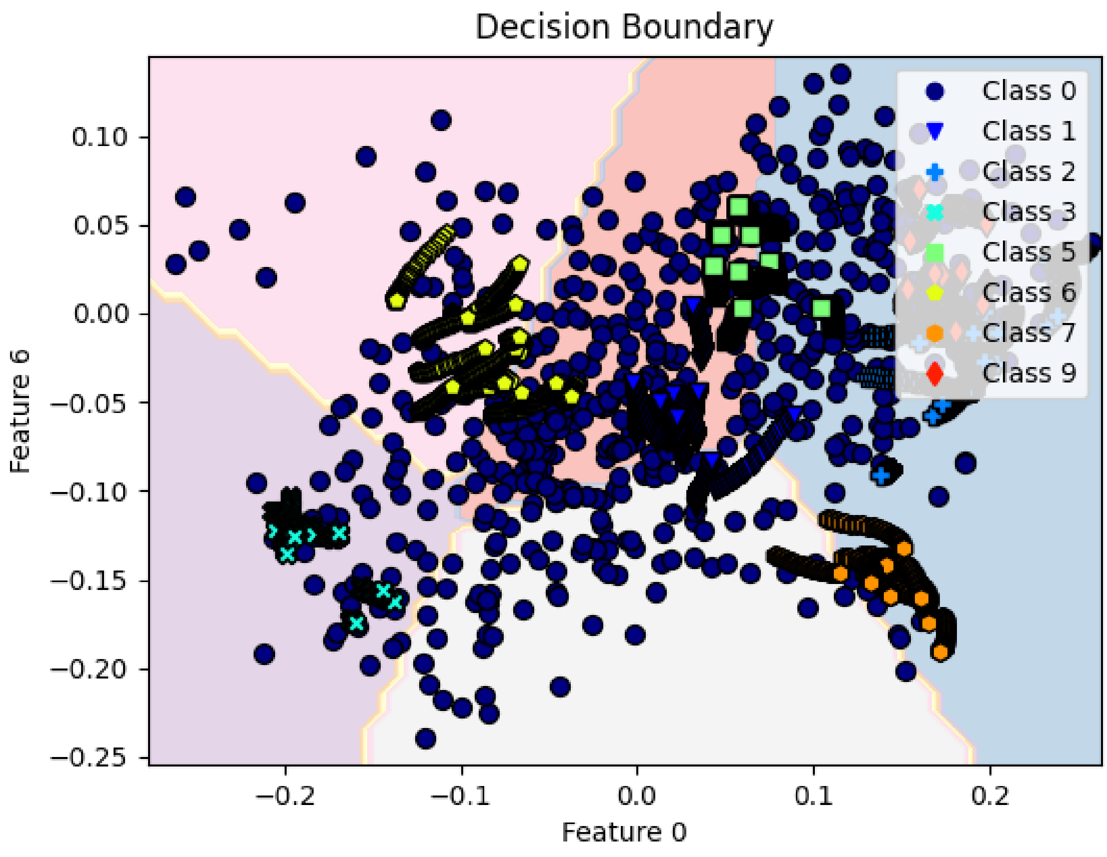
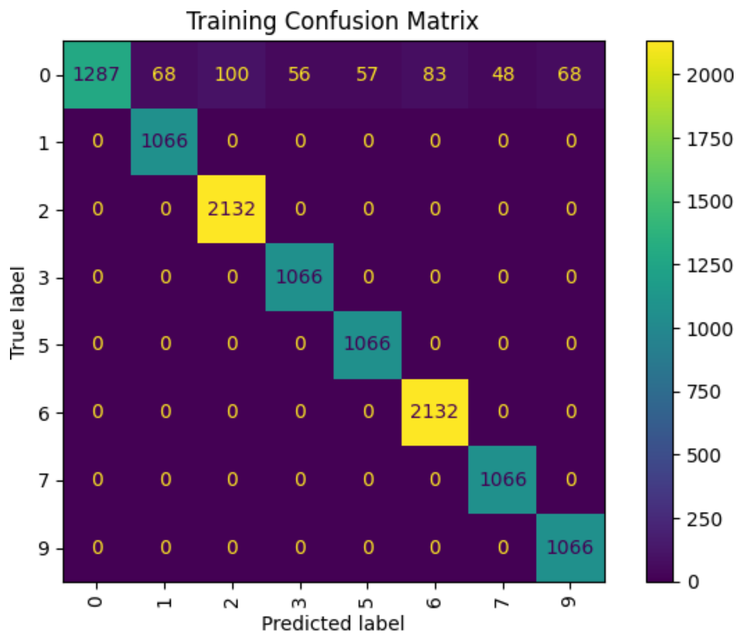


Figure 8: Taiji Dataset Example Decision Boundary

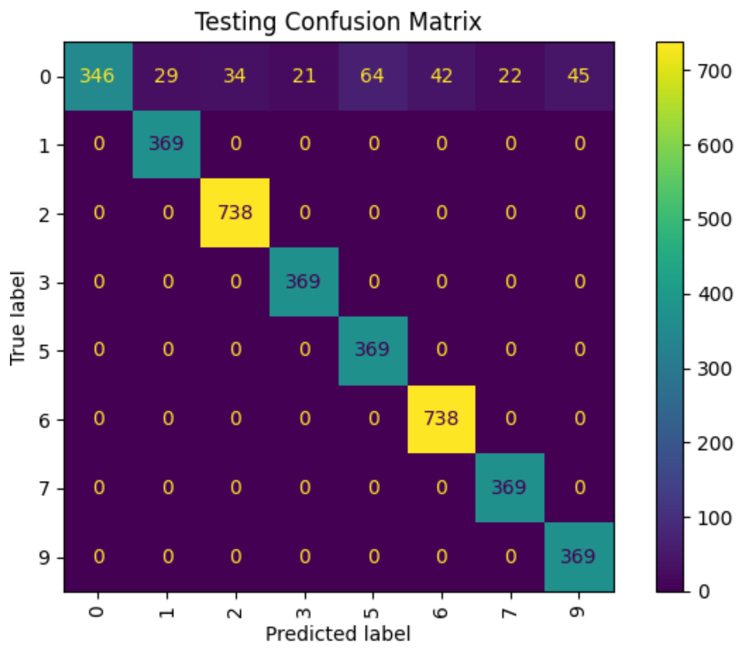
2.2.2 Confusion Matrix

The confusion matrices for the training and test sets after applying Fisher projection and *Linear Discriminant Analysis* classification.

The diagonal elements of the confusion matrix represent the correct classifications, with the non-diagonal elements of the matric represents mis-classification with the rows representing the true labels of each class category and columns representing the predicted labels with respect to the various classes.



(a) Taiji Training Data Confusion Matrix



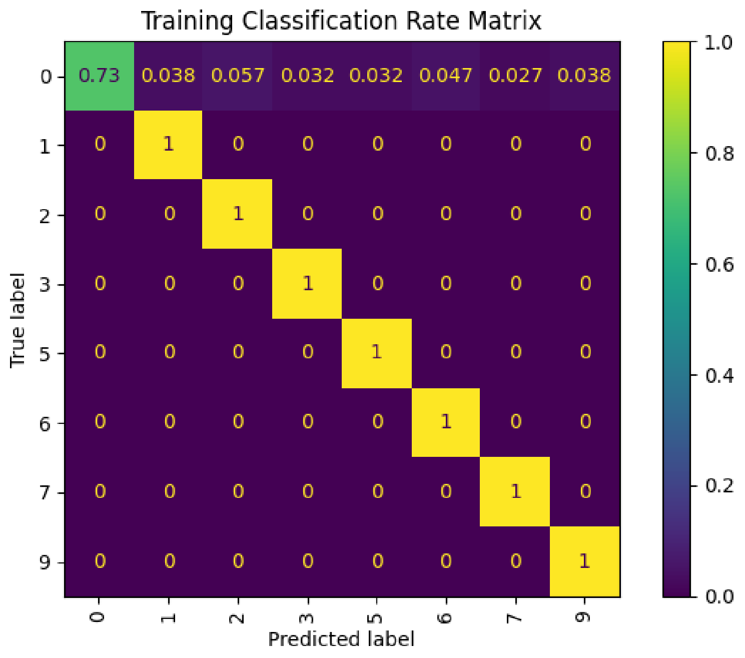
(b) Taiji Test Data Confusion Matrix

Figure 9: Taiji Training and Test Confusion Matrices

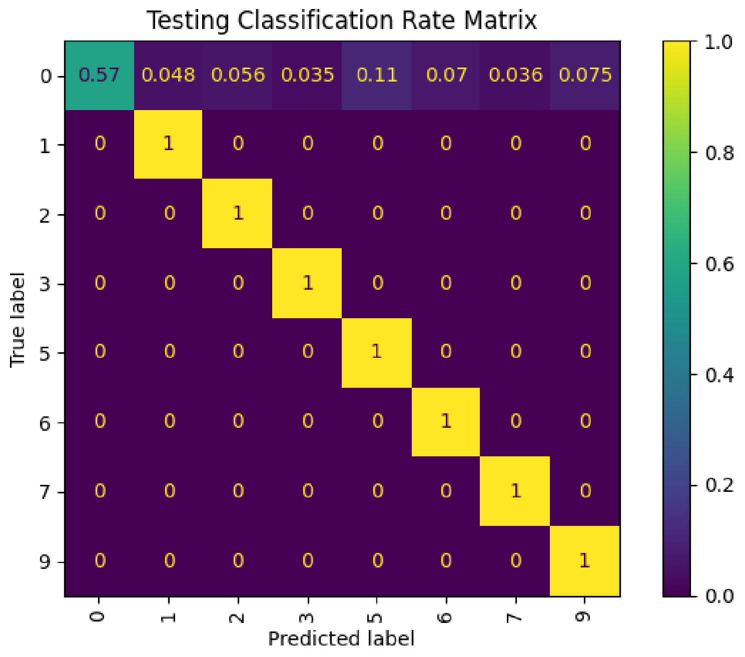
2.2.3 Classification Rates

The classification rate confusion matrices for the training and test sets after applying Fisher projection and *Linear Discriminant Analysis* classification.

The matrix plots in 10 as similar to the confusion matrices however, the elements in the matrix represent the per-class classification rates with respect to the



(a) Taiji Training Data Classification Rate Matrix



(b) Taiji Test Data Classification Rate Matrix

Figure 10: Taiji Training and Test Classification Rate Matrices

The Table 3 represents the overall classification rate data for various classifiers.

Model	**Overall Training Classification Rate**	**Overall Test Classification Rate**
LDA	0.95775	0.93451
KNN: 3 Neighbors	0.98380	0.80224
KNN: 5 Neighbors	0.97888	0.83231
KNN: 7 Neighbors	0.97562	0.87156
KNN: 10 Neighbors	0.97324	0.88583

Table 3: Overall Classification Rates of various classifiers, after performing Fisher projection.

2.2.4 Analysis and Conclusion

- Performing the fisher projection on the given Taiji dataset through the use of 8 significant eigen vectors/eigen values has provided the classification confusion matrices as shown in Figure 9.
- It is observed that the Linear Discriminant Analysis has captured the classification pattern pretty well for all the classes except the Class-0 which represents the transitional pose.
- Fisher projected data has provided benefits in terms of generalization for the classification. We observe in the classification rates for per-class in the Figure 10 that the training and test errors are not that far-off except for the Class-0.
- This suggests that performing Fisher projection prior to the Linear Discriminant Analysis classification has provided a good model fit.
- This is evident in the overall classification rates of the LDA model in the Table 3. The overall training classification rate is found to be 0.95775 and the overall test classification rate is found to be 0.93451. Judging by that criterion, there isn't a large difference suggesting LDA model to have provided a reasonable model fit.
- However, in the case of K-Nearest Neighbors classifier the model performed worse for lower k-values.
- Looking at the overall classification rates for $k = 3$, there is a huge difference between the training and test classification rates suggesting the model being hugely over-fitting the training dataset. The presence of very large training set error and low test set error suggests the issue of an over-fitted model.
- However for higher values of k-values for the KNN classifier, the model tends to perform reasonably well compared to its lower k-value counterparts. Even though, the overall test classification rate of KNN with 10 neighbors being considered cannot match the potential of the LDA model with about 93.451% overall test classification rate.

3 Central Limit Theorem

3.1 Introduction

According to Fischer [3], classical central limit theorem states that for a large enough sample size the population distribution can be assumed to be Gaussian. However, its essential to note that the sequence of random variables forming the population must be drawn from a distribution with finite mean and variance and must follow the i.i.d. criterion.
Bishop [1], mentions that the posterior distribution for a model is expected to become increasingly better approximated by a Gaussian distribution as the number of data points is increased.

Let us consider an unknown distribution with n independent, identically distributed random variables $X_1, X_2, ..., X_n$, where is n is very large with mean, μ and variance, σ^2.

Hence, we can find the mean, μ_N and variance, σ_N^2 as,

$$\mu_N = \frac{1}{n}(X_1 + X_2 + ... + X_n) \tag{66}$$

$$= \frac{1}{n}(n * \mu) \tag{67}$$

$$\mu_N = \mu \tag{68}$$

$$\sigma_N^2 = Var\left(\frac{1}{n}(X_1 + X_2 + ... + X_n)\right) \tag{69}$$

$$= \left(\frac{1}{n}\right)^2 (Var(X_1) + Var(X_2) + ... + Var(X_n)) \tag{70}$$

$$= \left(\frac{1}{n}\right)^2 \left(\sigma^2 + \sigma^2 + ... + \sigma^2\right) \tag{71}$$

$$= \left(\frac{1}{n}\right)^2 (n\sigma^2) \tag{72}$$

$$\sigma_N^2 = \frac{\sigma^2}{n} \tag{73}$$

Hence, the approximate Gaussian distribution is formed with a mean, μ_N and variance, σ_N^2.

3.2 Question 1

Question:

For a coin with a probability of heads of 0.6 in each flip, calculate the exact distribution of the total number of heads when the coin is flipped 5 times. Provide the equation(s) used to generate the result and a brief explanation of you reasoning.

Solution:

When a coin is flipped, there are two possible outcomes: head and tails each carrying a fixed probability. Given is coin is imbalanced in nature with the probability of head in each flip being 0.6.
Binary outcome experiments such as coin-toss follow the **Binomial Distribution**. Let X be the binomial random variable over the coin-toss experiment.
Hence, the probability mass function for the experiment representing the binomial random variable X is given by,

$$F(\mathbf{X}=\mathbf{x}) = \begin{cases} \binom{n}{x} p^x q^{n-x} & , x = 0, 1, 2, ..., n \\ 0 & , otherwise \end{cases} \tag{74}$$

The parameters in the binomial distribution are:

- n: The total number of trials.
- p: The probability of success on a single trial.
- q: The probability of failure on a single trial.
- x: Represents the trial number(which is a whole number, $0 \leq x \leq n$)

In our case, the probability of heads and tails adds up to 1. Hence we can write,

$$q = 1 - p$$

By incorporating that change in (74) we get,

$$F(\mathbf{X}=\mathbf{x}) = \begin{cases} \binom{n}{x} p^x (1-p)^{n-x} & , x = 0, 1, 2, ..., n \\ 0 & , otherwise \end{cases} \tag{75}$$

We get $n = 5$(as total number of coin flips is 5), $p = 0.6$ and $q = 0.4$.

$$F(\mathbf{X}=\mathbf{x}) = \begin{cases} \binom{5}{x} (0.6)^x (0.4)^{5-x} & , x = 0, 1, 2, ..., n \\ 0 & , otherwise \end{cases} \tag{76}$$

Therefore (76) represents the probability distribution of total number of heads when the coin is flipped 5 times.
In short hand notation, we can also write that,

$$S_n \sim Binomial(n = 5, p = 0.6)$$

Hence, we can say that the exact distribution of the total number of heads if coin is flipped 5 times is represented by a Binomial Distribution with parameters $n = 5$ and $p = 0.6$.

3.3 Question 2

Question:

Using the central limit theorem, approximate the distribution of the total number of heads when the coin is flipped 5000 times. Provide the equation(s) used to generate the result and a brief explanation of you reasoning.

Solution:

The coin-toss experiment as we know from 3.2, follows the Binomial Distribution for the probability estimation. However, for 5000 coin flips, calculation of probability through the use of factorials in the combination terms becomes a computationally expensive. From central limit theorem we know that, for a large number of trails in an experiment, we can approximate the population to be of a normal distribution.

- Let F_n represent the probability distribution we need to estimate for 5000 coin flips, which comes from a **Binomial Distribution**.

$$F_n \sim Binomial(n, p)$$

- Let the approximate normal distribution have a mean, μ_N and variance σ_N^2.
- Let the approximate normal distribution have a mean, μ_N and variance σ_N^2.

From central limit theorem, we know that,

$$\mu_N = \mu \tag{77}$$

$$\sigma_N^2 = \frac{\sigma^2}{n} \tag{78}$$

Some useful properties are:

- **Mean**: $\mu = np$
- **Variance**: $\sigma^2 = np(1-p)$

For our case, we get,

$$F_{n=5000} \sim Binomial(n = 5000, p = 0.6)$$

$$\begin{aligned} Mean, \mu =& n * p \\ =& 5000 * 0.6 \\ \mu =& 3000 \end{aligned}$$

$$\begin{aligned} Variance, \sigma^2 =& n * p * (1-p) \\ =& 5000 * 0.6 * 0.4 \\ \mu =& 1200 \end{aligned}$$

From (77) and (78), we get

$$\mu_N = 3000$$

$$\begin{aligned}\sigma_N^2 =& \frac{\sigma^2}{n}\\ =& \frac{1200}{5000}\\ \sigma_N^2 =& 0.24\end{aligned}$$

$$F_{n=5000} \sim Normal(\mu_N = 3000, \sigma_N^2 = 0.24)$$

Hence, we can say that the exact distribution of the total number of heads if coin is flipped 5000 times can be approximated using Central Limit Theorem by a Gaussian Distribution with parameters $\mu_N = 3000$ and $\sigma_N^2 = 0.24$.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer Sciences Media, 2006. 2, 21

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001. 2, 3

[3] H. Fischer, *A History of the Central Limit Theorem: From Classical to Modern Probability Theory*, 01 2011. 21