

Nonparametric Bayesian Classification For Fault Detection in an Electro-Hydrostatic Actuator

by

Nasim Feizinazhadgheshlaghi

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfilment of the requirements of the degree of

MASTER OF SCIENCE

Department of Statistics
University of Manitoba
Winnipeg

Copyright © 2025 by Nasim Feizinazhadgheshlaghi

Abstract

Hydraulic actuators are critical components in various industrial applications, and detecting internal leakage is essential for ensuring proper functioning and minimizing downtime. This thesis develops a nonparametric Bayesian framework for classifying leakage levels based on pressure signal data. Two modeling approaches are explored: (1) multinomial logistic regression with Dirichlet Process and Gaussian Process priors applied to extracted signal features from the Variance Fractal Dimension (VFD) method, and (2) the Infinite Hidden Markov Model (IHMM), which directly models temporal dynamics in raw signal sequences. Bayesian inference for the regression-based models is performed using the Metropolis-Hastings MCMC algorithm, while IHMM inference is conducted via collapsed Gibbs sampling. Posterior predictive distributions are used to make leakage-level predictions, and model performance is evaluated through k-fold cross-validation. Hyperparameters in all models are tuned through cross-validation and the performance evaluated with different classification evaluation metrics. The nonparametric nature of these models allows them to flexibly adapt model complexity to the data, supporting multi-class classification while capturing uncertainty.

Acknowledgment Page

I would like to express my sincere gratitude to Dr. Saman Muthukumarana for his unwavering guidance, continuous support (both financial and moral), and remarkable patience throughout the thesis process. His insightful feedback and steady encouragement were instrumental in shaping the direction and quality of this work.

I would also like to extend my appreciation to my committee members, Dr. Kevin McGregor and Dr. Hina Shaheen, for allocating their valuable time to review my thesis and provide constructive feedback. Their expertise and insights have significantly contributed to the improvement of this work. I am also sincerely grateful to Dr. Nariman Sepehri for generously providing the dataset used in this study and for sharing related domain knowledge.

I am grateful to my colleagues and friends for their support and camaraderie during my time at the University of Manitoba. Their encouragement and assistance have made this journey more enjoyable and fulfilling.

Finally, I would like to thank my family for their unwavering support, love, and encouragement throughout my academic journey. Their belief in me has been a constant source of motivation.

Dedication Page

This work is dedicated to my mother, father, and sister, in deep appreciation of their lifelong support, encouragement, and unwavering belief in me.

Contents

Contents	iii
List of Tables	vii
List of Figures	ix
1 Introduction	1
1.1 Objectives	2
1.2 Thesis Overview	3
2 Methodology	5
2.1 Bayesian Inference	5
2.1.1 Bayes' Theorem	5
2.1.2 Prior Impact	6
2.2 Binary Parametric Classification	7
2.3 Multinomial Logistic Regression with Nonparametric Priors	8

2.3.1	Dirichlet Process Prior	9
2.3.2	Gaussian Process Prior	11
2.3.3	Likelihood Function	12
2.4	Infinite Hidden Markov Model	13
2.4.1	Hidden Markov Models	13
2.4.2	Hierarchical Dirichlet Process Markov Model	16
2.5	Posterior Sampling using MCMC	19
2.6	Posterior Predictive Distribution	21
2.7	Classification Evaluation	22
2.7.1	K-fold Cross-Validation	25
3	Analysis	27
3.1	Data Description	27
3.2	Multinomial Logistic Regression with Nonparametric Priors	34
3.2.1	Preprocessing	34
3.2.2	Likelihood of Observed Data	36
3.2.3	Dirichlet Process Prior for Weights	38
3.2.4	Gaussian Process Prior for Weights	41
3.2.5	Prior for Bias Vector	42
3.2.6	Posterior Inference Using MCMC	43

3.2.7	Posterior Predictive Distribution	45
3.2.8	Prediction Pipeline	46
3.2.9	Hyperparameter Tuning	47
3.3	Infinite Hidden Markov Model (IHMM)	48
3.3.1	Hierarchical Structure and Priors	49
3.3.2	Likelihood Functions	50
3.3.3	Posterior Inference	51
3.3.4	Prediction	51
3.3.5	Hyperparameter Tuning	52
4	Results	53
4.1	Results of Dirichlet Process and Gaussian Process Models	53
4.1.1	Dirichlet Process Model Evaluation	54
4.1.2	Gaussian Process Model Evaluation	70
4.2	IHMM Model Evaluation	88
5	Conclusion	91
	Bibliography	95

List of Tables

4.1	DP Model Evaluation Results for Various Hyperparameters	54
4.2	GP Model Evaluation Results for Various Hyperparameters	71
4.3	IHMM Evaluation Results for Various Hyperparameters	88

List of Figures

2.1	Three coin HMM model	14
2.2	Confusion Matrix for binary classification	23
2.3	Confusion Matrix for 3-class classification	24
2.4	How cross-validation splits data into validation and training sets in each iteration	26
3.1	Abbreviation used for the datasets	27
3.2	(a) Experimental test bench. (b) Schematic of the arm (excavator link).	28
3.3	Typical test results for dataset OHHSt	29
3.4	Typical test results for dataset OHNST	30
3.5	Measured vs. denoised pressure signal P_A for datasets: CMSJo, CMMJo, CMHJo, CMNJo	31
3.6	Measured vs. denoised pressure signal P_B for datasets: CMSJo, CMMJo, CMHJo, CMNJo	32

3.7	Typical fault levels in the EHA for datasets: CMSJo, CMMJo, CMHJo, CMNJo	33
4.1	Fold 1, Trace Plot	55
4.2	Fold 1, Autocorrelation Plot	56
4.3	Fold 1, Posterior Predictive Check	57
4.4	Fold 2, Trace Plot	58
4.5	Fold 2, Autocorrelation Plot	59
4.6	Fold 2, Posterior Predictive Check	60
4.7	Fold 3, Trace Plot	61
4.8	Fold 3, Autocorrelation Plot	62
4.9	Fold 3, Posterior Predictive Check	63
4.10	Fold 4, Trace Plot	64
4.11	Fold 4, Autocorrelation Plot	65
4.12	Fold 4, Posterior Predictive Check	66
4.13	Fold 5, Trace Plot	67
4.14	Fold 5, Autocorrelation Plot	68
4.15	Fold 5, Posterior Predictive Check	69
4.16	Fold 1, Trace Plot	72
4.17	Fold 1, Autocorrelation Plot	73

4.18	Fold 1, Posterior Predictive Check	74
4.19	Fold 2, Trace Plot	75
4.20	Fold 2, Autocorrelation Plot	76
4.21	Fold 2, Posterior Predictive Check	77
4.22	Fold 3, Trace Plot	78
4.23	Fold 3, Autocorrelation Plot	79
4.24	Fold 3, Posterior Predictive Check	80
4.25	Fold 4, Trace Plot	81
4.26	Fold 4, Autocorrelation Plot	82
4.27	Fold 4, Posterior Predictive Check	83
4.28	Fold 5, Trace Plot	84
4.29	Fold 5, Autocorrelation Plot	85
4.30	Fold 5, Posterior Predictive Check	86
4.31	Log-likelihood Over Iterations for High Leakage Class	89
4.32	Log-likelihood Over Iterations for Small Leakage Class	89
4.33	Log-likelihood Over Iterations for No Leakage Class	89

Chapter 1

Introduction

Hydraulic actuators play an important role in various industrial and technological applications, making it necessary to ensure their proper functioning. Early fault detection can prevent costly downtime and preserve operational effectiveness. Internal leakage of the actuator is among the most common hydraulic system faults and is particularly difficult to diagnose without disconnecting the actuator or dismantling its components. Therefore, detecting internal leakage by monitoring the system performance using available signals is highly desirable, as it enables efficient diagnostics without interrupting the operation of the actuator ([Hosseinpour et al., 2024](#)); however, interpreting these signals reliably remains challenging due to noise, variability, and the complexity of real-world data.

This thesis aims to classify the level of internal leakage in hydraulic actuators based on signal monitoring. Classification algorithms and predictive models detect relationships between predictive variables and response variables, allowing the estimation of leakage levels from observed signal data ([Ye, 2014](#)). Bayesian inference provides a probabilistic

framework for classification, taking uncertainty into account by treating unknown parameters as random variables, offering an advantage over traditional machine learning models that typically provide deterministic outputs ([Hastie et al., 2009](#))([Bolstad and Curran, 2016](#)).

Previous research ([Hosseinpour et al., 2024](#)) ([Hosseinpour et al., 2024](#)) has shown that parametric binary Bayesian classification can effectively predict whether an actuator experiences no leakage or small leakage. While these models perform well for simple binary decisions, their structure is limited by a fixed number of parameters, restricting their adaptability to more diverse and complex datasets. This limitation makes parametric models less suitable for multiclass leakage detection tasks.

1.1 Objectives

To overcome the constraints of parametric models, this study employs nonparametric Bayesian classification. Unlike parametric models, Bayesian nonparametric approaches allow model complexity to grow with the available data, enabling the capture of more detailed patterns without solid assumptions ([Weng et al., 2019](#)). By adopting this flexible framework, the goal is to develop a more robust and adaptable leakage classification system that can better handle uncertainty and variability inherent in industrial environments.

1.2 Thesis Overview

This thesis is structured as follows:

In Chapter 2, we introduce the theoretical foundations of the two primary modeling approaches used in this work:

- Multinomial logistic regression with Dirichlet Process and Gaussian Process priors, applied to extracted features from actuator signals.
- Infinite Hidden Markov Models (IHMM), applied directly to raw signal sequences.

Chapter 3 provides an overview of the datasets, including visualizations to explore their structure, describes the preprocessing steps performed, and outlines the implementation procedures for the models.

In Chapter 4, we evaluate the performance of all models in terms of posterior convergence and classification accuracy, examining parameter stability, sampling efficiency, model fit, and predictive performance.

Finally, Chapter 5 presents concluding remarks and discusses the implications of our findings for future research and practical applications in condition monitoring of hydraulic actuators.

Chapter 2

Methodology

2.1 Bayesian Inference

2.1.1 Bayes' Theorem

Bayes' theorem is stated mathematically as the following equation:

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

where A and B are events and $P(B) \neq 0$.

- $P(A)$ and $P(B)$ are the probabilities of observing A and B . They are known as the prior probability and the marginal probability, respectively.
- $P(B | A)$ is the conditional probability of event B occurring given that A is true. It can also be interpreted as the likelihood of A given a fixed B because $P(B | A) = L(A | B)$.

- $P(A | B)$ is the conditional probability of event A occurring given that B is true. It is also known as the posterior probability of A given B .

By conditioning on the known value of the data X , and using Bayes' rule, we can obtain the posterior density:

$$P(\theta | X) = \frac{P(X | \theta)P(\theta)}{P(X)}$$

where $P(X) = \sum P(X | \theta)P(\theta)$ for discrete θ , or $P(X) = \int P(X | \theta)P(\theta) d\theta$ for continuous θ .

An alternative form of the equation eliminates the term $P(X)$, which is independent of θ and can be treated as a constant for a fixed X , leading to the unnormalized posterior density:

$$P(\theta | X) \propto P(\theta)P(X | \theta)$$

$$\text{posterior} \propto \text{prior} \times \text{likelihood}$$

More details about Bayes' Theorem and Bayesian Inference can be found in ([Gelman et al., 2013](#)).

2.1.2 Prior Impact

The prior distribution encapsulates beliefs about the unknown density function f before any data are observed. It is a probability measure defined on a suitable space of density functions, including all possible shapes and forms of the true density. It plays

a crucial role in shaping the posterior distribution, which represents updated beliefs about the density function after observing data (Walker, 2010). Although Bayesian models allow us to use important prior knowledge that we have about data, choosing an appropriate prior can be challenging (Ghaderinezhad et al., 2022).

2.2 Binary Parametric Classification

In (Hosseinpour et al., 2024), a parametric binary Bayesian classification approach was employed to detect whether the actuator had no or small leakage level. The binary response Y_i is modeled with a Bernoulli distribution, having a success probability p_i :

$$Y_i \sim \text{Bernoulli}(p_i), \quad i = 1, 2, \dots, n$$

where n represents the number of training datasets for classification. The logistic regression model is expressed as a linear function of x_{1i} and x_{2i} :

$$\log \left(\frac{p_i}{1 - p_i} \right) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i}$$

Here, β_0, β_1 , and β_2 are regression coefficients estimated via Bayesian inference. x_{1i} and x_{2i} are feature values extracted from pressure signals P_A and P_B . Rewriting the logistic regression equation for the probability p_i :

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i})}{1 + \exp(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i})}$$

This equation ensures p_i is within $[0, 1]$. A threshold classifies the signal as faulty if $p_i \geq 0.5$, otherwise, it's healthy. Normal priors for the coefficients are:

$$\beta_0 \sim \text{dnorm}(0, 10^{-3}), \quad \beta_1 \sim \text{dnorm}(0, 10^{-3}), \quad \beta_2 \sim \text{dnorm}(0, 10^{-3})$$

where $\text{dnorm}(\mu, \tau)$ denotes a normal distribution with mean μ and precision $\tau = 1/\sigma^2$.

In this thesis, we aim to apply nonparametric multiclass Bayesian classification for detecting no, small, medium, or high leakage.

2.3 Multinomial Logistic Regression with Nonparametric Priors

Parametric Bayesian approaches rely on models with a fixed number of parameters for prior and posterior distributions, which can restrict their flexibility when data varies. In contrast, Bayesian nonparametric methods get rid of the need for fixed parameters, allowing the model to adapt its complexity based on the data. This means the model can become more complex as more data is introduced, automatically adjusting the dimensionality of the parameter space in response to sample size. However, nonparametric models have their limitations. They might not generalize well to unseen data and may run into problems with the curse of dimensionality. Fortunately, if the dataset covers a wide range of variables, and has a manageable number of features, these problems can be reduced. This adaptability and automatic complexity adjustment make Bayesian nonparametric methods highly useful for applications requiring dynamic model structures ([Weng et al., 2019](#)).

2.3. MULTINOMIAL LOGISTIC REGRESSION WITH NONPARAMETRIC PRIORS⁹

All Bayesian statisticians, regardless of whether they utilize parametric or nonparametric models, inherently construct priors on spaces of density functions. A comprehensive prior should be large enough to have room for all uncertainties related to the unknown density function and if additional uncertainty is identified, the prior should be enlarged to encompass this new information before observing the data. This ensures that the posterior distribution reflects the complete picture of prior beliefs and data-driven updates. Using a nonparametric model might be the only way to create a prior large enough to avoid the need for post-data adjustments.

The key distinction between parametric and nonparametric priors lies in the flexibility of the density function's shape. Parametric Bayesian models restrict the shape of the density function to a specific family, like the normal distribution. In this case, the prior is essentially assigning a probability of one to the chosen shape. On the other hand, Bayesian nonparametric priors are defined on a space of density functions with unrestricted shapes. They achieve this flexibility by employing stochastic processes, such as Gaussian and Dirichlet Processes ([Walker, 2010](#)).

2.3.1 Dirichlet Process Prior

The Dirichlet process ([Ferguson, 1973](#)) draws inspiration from the conjugacy property observed between the multinomial likelihood and the finite-dimensional Dirichlet distribution. In the multinomial model, data are categorized into a finite number of groups. The probability of observing specific counts within these groups is modeled using the multinomial likelihood function. Because the Dirichlet distribution has a

similar mathematical structure to this likelihood function, it is commonly used as a prior for the probabilities assigned to each category (Ghosal, 2010).

In the standard framework of Dirichlet Process (Neal, 2000), the data points $y_1, y_2, \dots, y_n$ are treated as part of an exchangeable sequence, which is equivalent to assuming that the data are conditionally independent and identically distributed (i.i.d.) given an underlying distribution. Each data point y_i is associated with a latent variable θ_i , which is sampled from a mixing distribution G . The mixing distribution G itself is modeled as a random draw from a Dirichlet Process, enabling a flexible and adaptive representation of the data-generating process.

The Dirichlet Process (DP) is defined by two parameters:

$$G \sim \text{DP}(\alpha, G_0)$$

- The base distribution G_0 , which reflects prior beliefs about the parameter space.
- The concentration parameter α , which controls diversity. Smaller values of α encourage fewer, larger components, while larger values promote more, smaller components.

Dirichlet Processes can be represented through different constructions, such as the Stick-Breaking construction, the Chinese Restaurant Process (CRP), and the Limit of Finite Mixtures. The Stick-Breaking Process explicitly constructs a DP as a weighted sum of atoms, where weights are sequentially generated by breaking a unit-length stick. The Chinese Restaurant Process (CRP) provides a clustering-based perspective,

where data points are assigned to clusters with probabilities proportional to existing counts. The Limit of Finite Mixtures represents that a DP mixture model can be derived as the limit of a sequence of finite mixture models, where the number of mixture components approaches infinity (Teh et al., 2006).

In the context of multinomial logistic regression with a DP prior, the Stick-Breaking Process is preferable because it allows for an explicit representation of the DP-distributed weights, facilitating the nonparametric modeling of regression coefficients.

2.3.2 Gaussian Process Prior

A Gaussian process (Choudhuri et al., 2007) is a collection of random variables, any finite number of which have a joint Gaussian distribution. For a set of input points $x_1, \dots, x_k$, the corresponding random variables $\eta(x_1), \dots, \eta(x_k)$ are jointly multivariate normal. The Gaussian process is used as a prior over the space of real-valued functions, meaning it places a probability distribution over possible functions $\eta(x)$. This is different from the Dirichlet process, which is prior on cumulative distribution functions.

Mean Function: A Gaussian process is specified by its mean function, $\mu(x)$. The mean function represents the expected value of the process at each point in the input space and reflects the prior guess about the function $\eta(x)$.

Covariance Kernel: The other component of a Gaussian process is the covariance kernel (also referred to as a covariance function or kernel function), $\sigma(x, x')$. The

covariance kernel defines the covariance between the function's values at any two input locations, x and x' . It determines the smoothness and overall shape of the function. The covariance kernel must satisfy the condition that for every $k \geq 1$ and $x_1, \dots, x_k \in X$, the $k \times k$ matrix $((\sigma(x_i, x_j)))_{i,j=1,\dots,k}$ is positive definite.

The correlation kernel affects how smooth the function $\eta(x)$ looks and determines how much strength is borrowed from neighboring points.

The correlation kernel can be adjusted to be locally adaptive: If quick changes are expected in part of the covariate space, the correlation $r(x, x')$ should drop quickly as x' moves away from x . If the function is expected to be flat in some areas, the correlation should decrease more slowly. The prior can also be covariate adaptive so that in areas with fewer covariate values, the correlation should decay slowly so that information from farther points can still be used.

2.3.3 Likelihood Function

Both of the discussed priors, the Dirichlet Process and the Gaussian Process, will be used as prior belief for weights W in 2.1, which is our likelihood function. The likelihood function for our model is based on multinomial logistic regression, which is commonly used for classification problems where the response variable can take one of

multiple categorical outcomes. It is formulated as follows:

$$P(y_i \mid \mathbf{X}_i, W, b) = \prod_{j=1}^C \left(\frac{\exp(z_{i,j})}{\sum_{m=1}^C \exp(z_{i,m})} \right)^{\mathbb{1}(y_i=j)}, \quad \text{where} \quad z_{i,j} = \sum_{l=1}^d X_{il} W_{lj} + b_j. \quad (2.1)$$

2.4 Infinite Hidden Markov Model

Another nonparametric model used in this thesis is the Infinite Hidden Markov Model (IHMM). The IHMM ([Beal and Rasmussen, 2001](#)) is a nonparametric Bayesian extension of the traditional Hidden Markov Model (HMM) that allows for an unbounded number of hidden states. This flexibility is particularly useful in applications where the number of underlying states is not known and can vary based on the observed data. The IHMM is constructed using a Dirichlet Process (DP) prior, which provides a mechanism for generating an infinite number of states while maintaining a coherent structure.

2.4.1 Hidden Markov Models

Hidden Markov Models (HMMs) ([Rabiner, 1989](#)) are statistical models used to characterize the properties of signals. They have become popular because they are mathematically rich and work well in practice for important applications. Hidden Markov models fall into the category of stochastic signal models. They are called

“hidden” because the underlying stochastic process that produces the signal is not directly observable. Instead, it is observed through another set of stochastic processes that generate the sequence of observations. This concept is best illustrated by the classic coin toss example: Imagine someone behind a curtain flipping coins. You can only hear the results (“heads” or “tails”), but not see which coin is being flipped or how the coin selection is made. This scenario can be modeled by an HMM where each state represents a different biased coin, each with its own probability of heads and tails. The transitions between states, representing the selection of a different coin, are governed by a state transition matrix. The observed sequence of “heads” and “tails” is the output of this hidden process. Figure 2.1 illustrates a simple HMM with three coins.

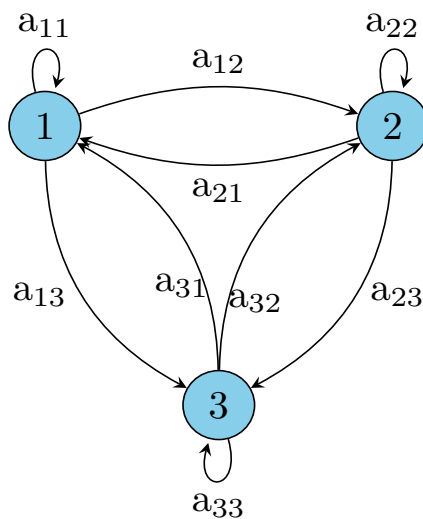


Figure 2.1: Three coin HMM model

Components of an HMM:

- **N**: Number of states in the model represented as $S_1, S_2, \dots, S_N$.
- **M**: Number of distinct observation symbols per state represented as $v_1, v_2, \dots, v_M$.
- **A**: State transition probability distribution: the probability of transitioning from one state to another. A matrix $A = a_{ij}$, where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$, representing the probability of transitioning from state S_i at time t to state S_j at time $t + 1$.
- **B**: Observation symbol probability distribution: the probability of observing a specific symbol in a given state. A matrix $B = b_j(k)$, where $b_j(k) = P(O_t = v_k | q_t = S_j)$, representing the probability of observing symbol v_k at time t given that the system is in state S_j .
- **π** : Initial state distribution: the probability of starting in a particular state. A vector $\pi = \pi_i$, where $\pi_i = P(q_1 = S_i)$, representing the probability of starting in state S_i at time $t = 1$.

For convenience, the complete parameter set of the model is denoted as: $\lambda = (A, B, \pi)$.

Given these components, an HMM can generate an observation sequence by:

1. Choose an initial state, $q_1 = S_i$, according to the initial state distribution, π .
2. Set $t = 1$.

3. Choose an observation symbol $O_t = v_k$ according to the observation symbol probability distribution in state S_i , $b_i(k)$.
4. Transit to a new state, $q_{t+1} = S_j$, according to the state transition probability distribution, a_{ij} .
5. Set $t = t + 1$ and repeat steps 3 and 4 until $t = T$.

This process acts as both a generator of observations and a model for how a given observation sequence was generated.

2.4.2 Hierarchical Dirichlet Process Markov Model

An Infinite Hidden Markov Model (IHMM) ([Beal and Rasmussen, 2001](#)) is a generalization of the traditional Hidden Markov Model (HMM) that allows for an unbounded number of hidden states. In contrast to standard HMMs, where the number of states must be predetermined, IHMMs can dynamically adapt the number of states based on the observed data. In many real-world applications of HMMs, determining the appropriate number of hidden states is a challenging task. IHMMs address this limitation by providing a framework that can accommodate an infinite number of states, enabling the model to automatically infer the complexity of the underlying process.

The key to building an IHMM lies in defining a mechanism that can generate a potentially infinite number of state-specific distributions while maintaining a connection

between them. The Hierarchical Dirichlet Process (HDP) provides an elegant solution to this problem.

Hierarchical Structure: The HDP (Teh and Jordan, 2010) establishes a hierarchical relationship between the state-specific distributions, ensuring that they share a common pool of potential states.

- **Global Base Distribution G_0 :** The foundation of the HDP is a global base distribution, denoted as G_0 , which is drawn from a Dirichlet Process (DP). This distribution defines the overall space of possible states the IHMM can explore.
- **State-Specific Distributions G_θ :** For each hidden state θ , the IHMM draws a state-specific distribution G_θ from another DP. Basically, the base measure of this DP is the global distribution G_0 . This hierarchical dependence ensures that the state-specific distributions are linked and share the same set of possible states as defined by G_0 .

State Transitions: Similar to standard HMMs, IHMMs model transitions between hidden states. However, due to the infinite state space, the transition probabilities cannot be represented by a finite transition matrix. Instead, the HDP provides a mechanism for defining transition probabilities between any pair of states.

Emission Distributions: Each hidden state is associated with an emission distribution that governs the generation of observations. The specific form of the emission distribution depends on the nature of the data.

1. HDP Prior:

- $G_0 \mid \gamma, H \sim \text{DP}(\gamma, H)$,
- $G_\theta \mid \alpha, G_0 \sim \text{DP}(\alpha, G_0)$,

where:

- γ and α are concentration parameters that control the variability of the DPs,
- H is a base distribution that defines the prior over the parameters of the emission distributions,
- $\text{DP}(\gamma, H)$ represents a Dirichlet Process with concentration parameter γ and base distribution H .

2. Generative Process:

- For each time step t :
 - $\theta_t \mid \theta_{t-1}, G_{\theta_{t-1}} \sim G_{\theta_{t-1}}$,
 - $x_t \mid \theta_t \sim F_{\theta_t}$,

where:

- θ_t represents the hidden state at time t ,
- x_t is the observation at time t ,
- F_{θ_t} is the emission distribution associated with state θ_t .

2.5 Posterior Sampling using MCMC

MCMC is a general method for sampling from arbitrary posterior distributions. It works by sampling parameter values θ from approximate distributions and then refining those samples through iterative adjustments to more closely match the target posterior distribution, $p(\theta | y)$. The sampling process is sequential, meaning each new draw depends on the immediately preceding one, forming a Markov chain. While the Markov property is inherent to the process, the key to the method's success is the convergence of these approximate distributions to the target posterior distribution over many iterations. MCMC is typically used when direct sampling from the posterior distribution is not feasible or computationally efficient. The goal is to create a Markov chain that has the desired posterior distribution as its unique stationary distribution and to run the simulation long enough for the distribution of the current draws to be a good approximation of this target. The basic Markov chain simulation methods are Metropolis-Hastings algorithm and Gibbs sampler ([Gelman et al., 2013](#)).

Metropolis-Hastings Algorithm:

The Metropolis-Hastings algorithm is a flexible family of Markov Chain Monte Carlo (MCMC) methods designed to generate samples from complex posterior distributions. It extends the basic Metropolis algorithm by permitting more general proposal (or jumping) distributions, which are not required to be symmetric.

Initialization: Start with an initial value θ_0 such that the posterior density $p(\theta_0 | y)$ is greater than zero.

Iteration: For each step $t = 1, 2, \dots$, perform the following:

a. Propose a new state: Draw a candidate value θ^* from a proposal distribution

$$J_t(\theta^* \mid \theta_{t-1}).$$

b. Compute the acceptance ratio:

$$r = \frac{p(\theta^* \mid y) \cdot J_t(\theta_{t-1} \mid \theta^*)}{p(\theta_{t-1} \mid y) \cdot J_t(\theta^* \mid \theta_{t-1})}$$

This ratio accounts for the asymmetry in the proposal distribution.

c. Accept or reject: With probability $\min(1, r)$, accept the proposed value and set $\theta_t = \theta^*$. Otherwise, retain the current value, setting $\theta_t = \theta_{t-1}$.

This algorithm constructs a Markov chain that, under mild regularity conditions, converges to a stationary distribution equal to the target posterior distribution $p(\theta \mid y)$ (Gelman et al., 2013).

Gibbs Sampling:

Gibbs sampling is particularly effective in scenarios involving multiple parameters. Although the Metropolis-Hastings (MH) algorithm can be generalized to accommodate multi-parameter problems, its efficiency tends to decline as the dimensionality increases, primarily due to a drop in acceptance rates. A common way to address this issue is by updating only a subset (or block) of parameters at each iteration, an approach known as *blockwise Metropolis-Hastings* (Bolstad and Curran, 2016).

Gibbs sampling (Geman and Geman, 1984) can be viewed as a special case of blockwise Metropolis-Hastings, where updates are performed by sampling directly from the full conditional distributions of the parameters. This is feasible when we can explicitly derive these conditionals. The algorithm is especially useful for hierarchical models, where the relationships among parameters are well-defined.

In Gibbs sampling, rather than using a generic proposal distribution, we draw each parameter θ_j from its true conditional distribution given all other parameters θ_{-j} . That is,

$$q(\theta_j; \theta_{-j}) = g(\theta_j \mid \theta_{-j}, y),$$

where θ_{-j} denotes the set of all parameters excluding θ_j . When using this approach, the acceptance probability simplifies to

$$\alpha = \min \left(1, \frac{g(\theta'_j \mid \theta_{-j}, y) q(\theta_j; \theta'_j, \theta_{-j})}{g(\theta_j \mid \theta_{-j}, y) q(\theta'_j; \theta_j, \theta_{-j})} \right) = 1,$$

meaning that each proposed sample is accepted without rejection.

2.6 Posterior Predictive Distribution

The posterior predictive distribution is used in Bayesian statistics to make predictions about a new, unobserved data point, denoted as X_{new} , after having observed some

data X . It reflects our uncertainty about both the new data point and the parameters of the underlying model.

Once the observed data X are available, we can update our beliefs about the model parameters using Bayes' theorem, resulting in the posterior distribution $p(\theta | X)$. To predict X_{new} , we do not rely on a single estimated value of the parameters θ . Instead, we average the predictions over all possible values of θ , weighted by how plausible each value is under the posterior distribution.

Mathematically, the posterior predictive distribution is given by:

$$p(X_{\text{new}} | X) = \int p(X_{\text{new}} | \theta) p(\theta | X) d\theta$$

This expression shows that the distribution of X_{new} is obtained by integrating the likelihood of X_{new} over the posterior distribution of the parameters. It provides a principled way to make predictions that fully incorporate the uncertainty in the model parameters, rather than conditioning on a single estimate ([Gelman et al., 2013](#)).

2.7 Classification Evaluation

To evaluate classifiers' performance and assess how accurate they are, there are some measures like accuracy, sensitivity or recall, precision, and F1 score. All of these measurements can be calculated with 4 values; True Positives (TP) which shows the number of instances that the model correctly predicted the positive class, True

Negatives (TN) are when the model correctly predicts the negative class, False Positives (FP) are when the model incorrectly predicts the positive class, and False Negatives (FN) are when the model incorrectly predicts the negative class. In our case, TP is the correct prediction of the fault, TN is the correct prediction of healthy signals, FP is the wrong prediction of fault, and FN is the wrong prediction of healthy signals.

Confusion Matrix

All these values can be summarized by the confusion matrix. Figure 2.2 shows the confusion matrix for binary classification, and Figure 2.3 shows it for 3-class classification.

		True class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

Figure 2.2: Confusion Matrix for binary classification

		True class		
		Class 1	Class 2	Class 3
Predicted class	Class 1	TP	FP	FP
	Class 2	FN	TP	FP
	Class 3	FN	FN	TP

Figure 2.3: Confusion Matrix for 3-class classification

Accuracy

Accuracy represents the ratio of correct predictions to the total number of predictions, and is calculated as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP + TN}{TP + FP + FN + TN}$$

Recall (Sensitivity)

Recall indicates the proportion of actual positive cases that were accurately predicted and is calculated as follows:

$$\text{Recall} = \frac{TP}{TP + FN}$$

High recall means fewer false negatives.

Precision

Precision measures how many of the predicted positive cases were correct, as follows:

$$\text{Precision} = \frac{TP}{TP + FP}$$

High precision means fewer false positives.

F1 Score

The F1 score integrates precision and recall into a single metric by calculating their harmonic mean, thereby treating both metrics with equal significance. The formula for the F1 score is given as follows:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

More details about evaluation metrics for classification can be found in ([Han et al., 2011](#)).

2.7.1 K-fold Cross-Validation

K-fold cross-validation ([James et al., 2023](#)) is a method used to assess the performance of a model by resampling the data. In this technique, the dataset is randomly divided into k approximately equal-sized folds. One fold is designated as the validation set to do the prediction with, while the model is trained on the remaining $k - 1$ folds. Then, the number of misclassified observations is calculated for the validation set. This process is repeated k times, with each fold serving as the validation set once.

The overall K-fold cross-validation (CV) estimate is obtained by averaging the errors across all iterations:

$$CV_k = \frac{1}{k} \sum_{i=1}^k \text{Err}_i$$

where $\text{Err}_i = I(y_i \neq \hat{y}_i)$.

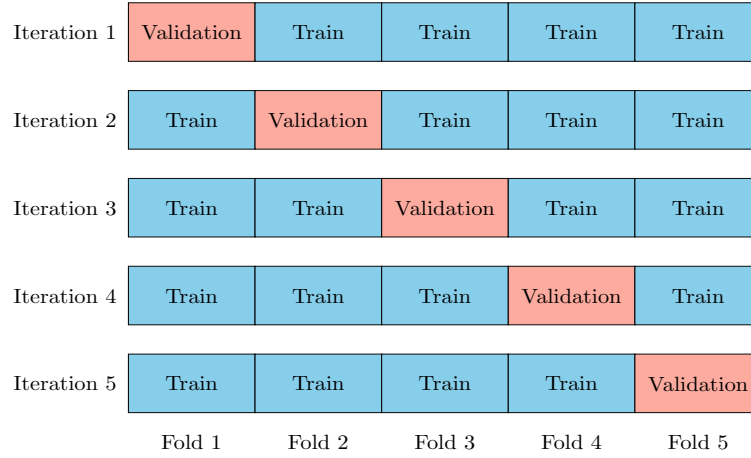


Figure 2.4: How cross-validation splits data into validation and training sets in each iteration

Chapter 3

Analysis

3.1 Data Description

According to (Hosseinpour et al., 2024), 48 datasets were extracted and obtained experimentally, each labeled with a five-letter code. The first letter represents the system mode: “O” for open-loop control and “C” for closed-loop control. The second letter indicates the load level: “N” for no load, “M” for medium load, and “H” for high load. The third letter denotes the leakage level: “N” for no leakage, “S” for small leakage, “M” for medium leakage, and “H” for high leakage. The final two letters specify the input type, either “Jo” (joystick) or “St” (step input).

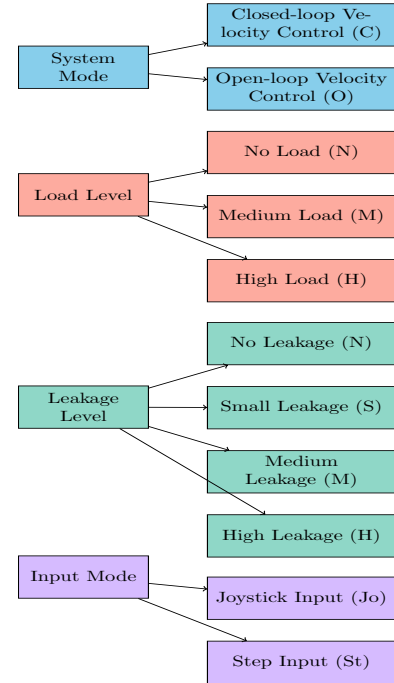


Figure 3.1: Abbreviation used for the datasets

Figure 3.2 shows the experimental setup used for data collection. The system includes a hydraulic actuator mounted on a backhoe arm, where a bidirectional pump drives an asymmetric cylinder. The rod position is measured with an encoder, and the load mass m_L is adjusted using mounting disks. The static force F_L applied to the actuator is calculated based on the arm geometry and angle θ .

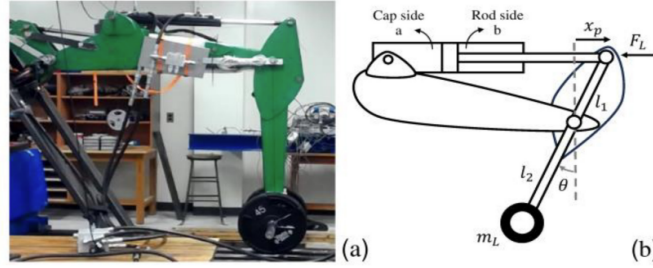


Figure 3.2: (a) Experimental test bench. (b) Schematic of the arm (excavator link).

Each dataset consists of 11 columns, including time, input, position, velocity, leakage, position, F_L , and both no-filter and denoised pressure signals P_A and P_B . Figures 3.3 and 3.4 display these columns for the OHHSt and OHNSt datasets, respectively.

Figures 3.5 and 3.6 show the comparison between the measured and denoised pressure signals P_A and P_B for the datasets CMSJo, CMMJo, CMHJo, and CMNJo. These plots illustrate how the denoising process affects the pressure signals in each data set. And figure 3.7 displays the leakage level from no-leakage to high leakage in the same four datasets.

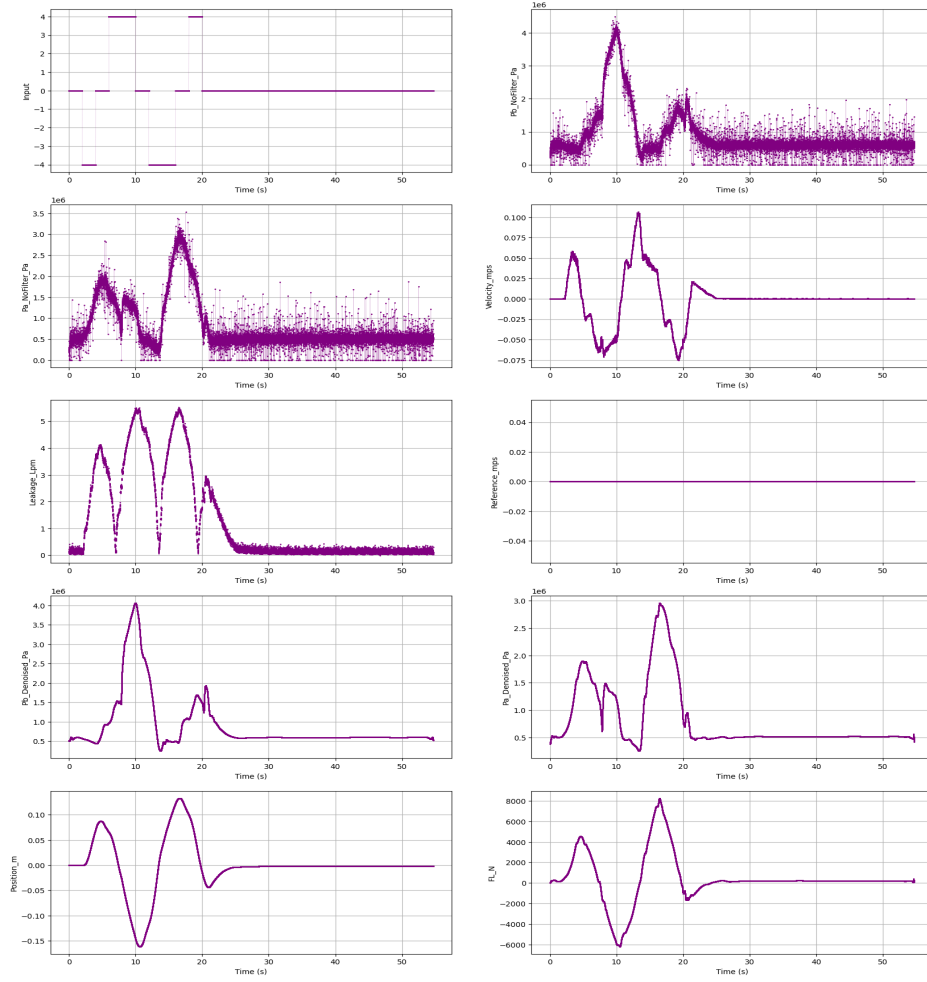


Figure 3.3: Typical test results for dataset OHHSt

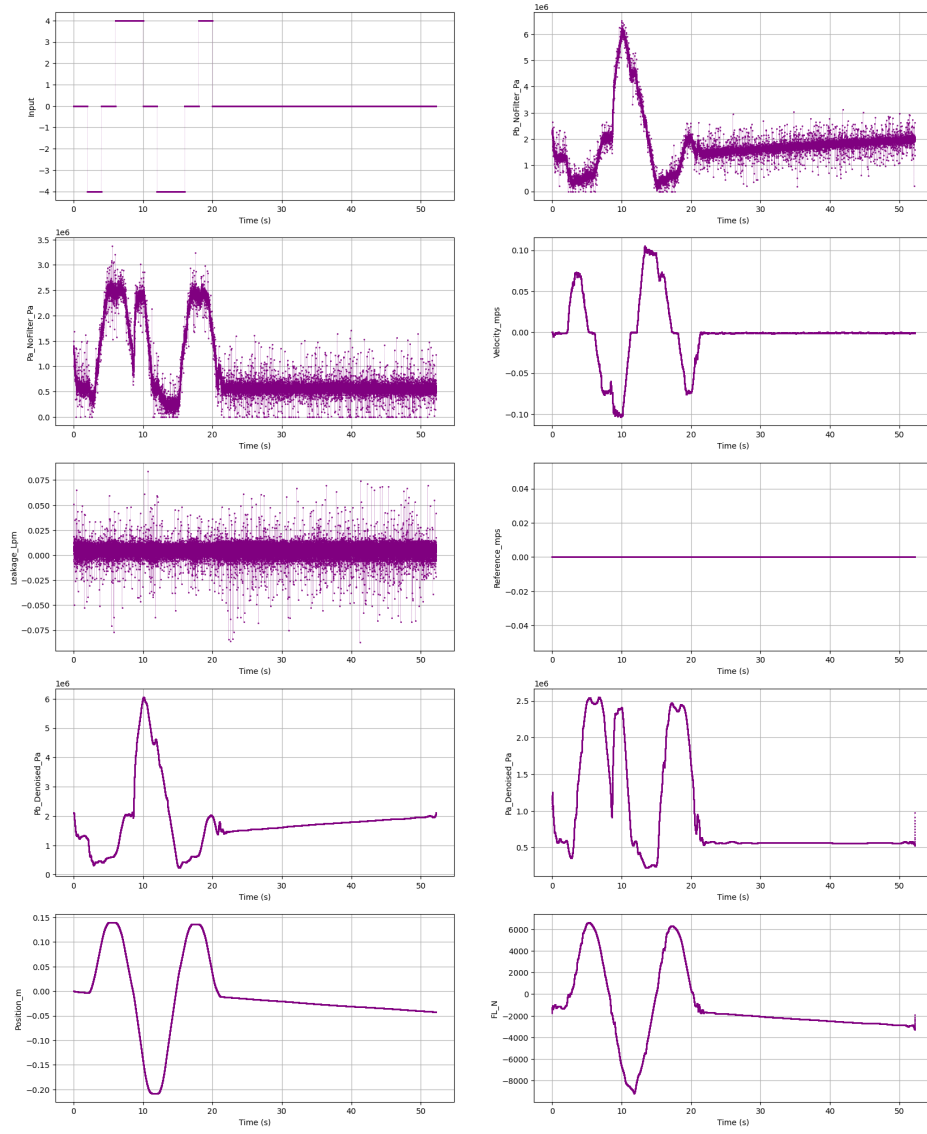


Figure 3.4: Typical test results for dataset OHNST

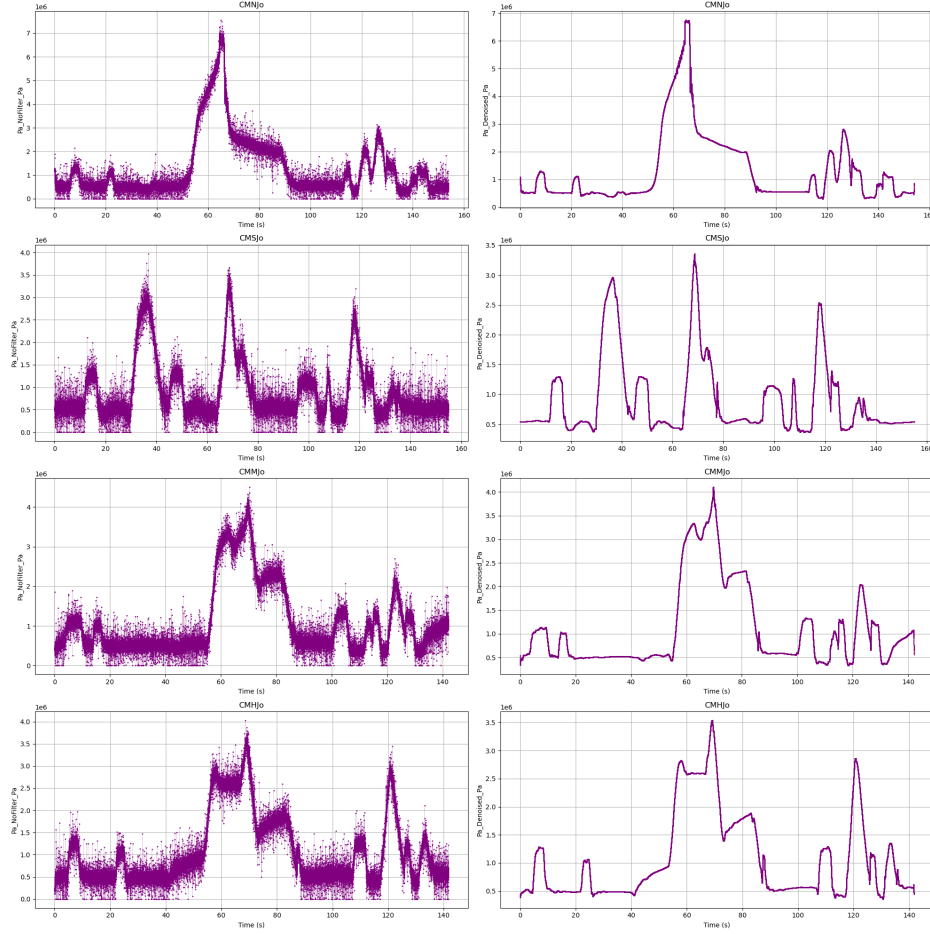


Figure 3.5: Measured vs. denoised pressure signal P_A for datasets: CMSJo, CMMJo, CMHJo, CMNJo

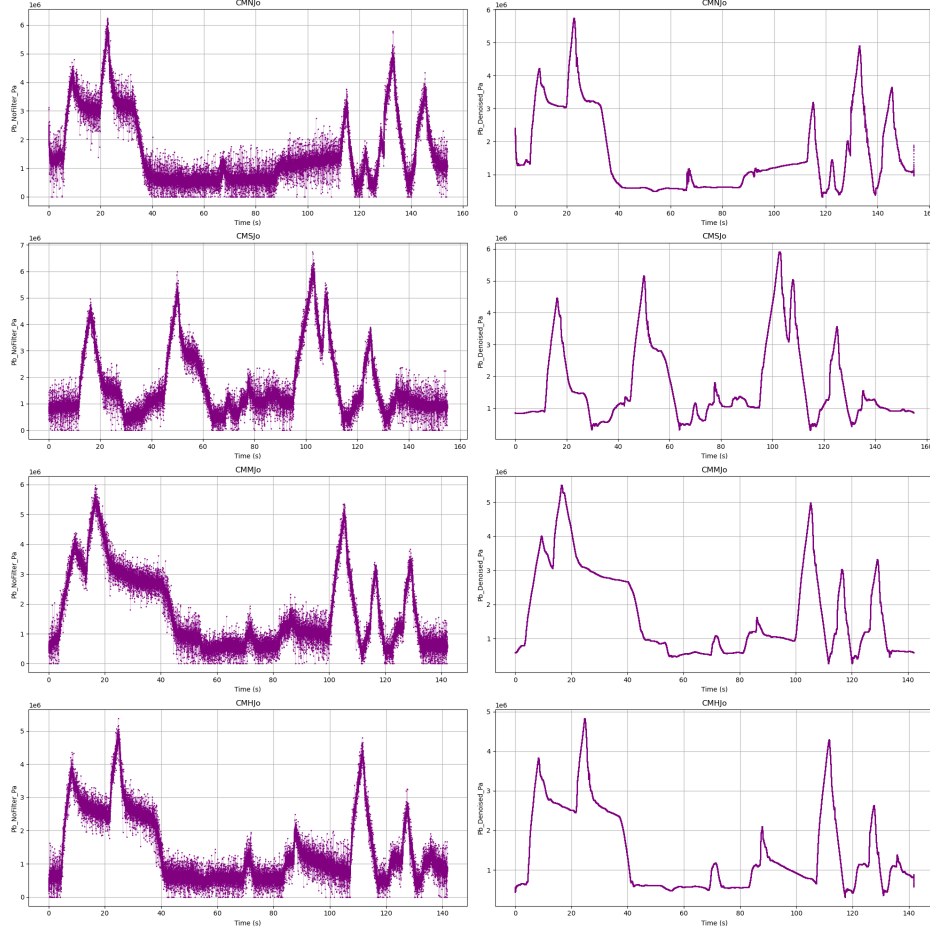


Figure 3.6: Measured vs. denoised pressure signal P_B for datasets: CMSJo, CMMJo, CMHJo, CMNJo

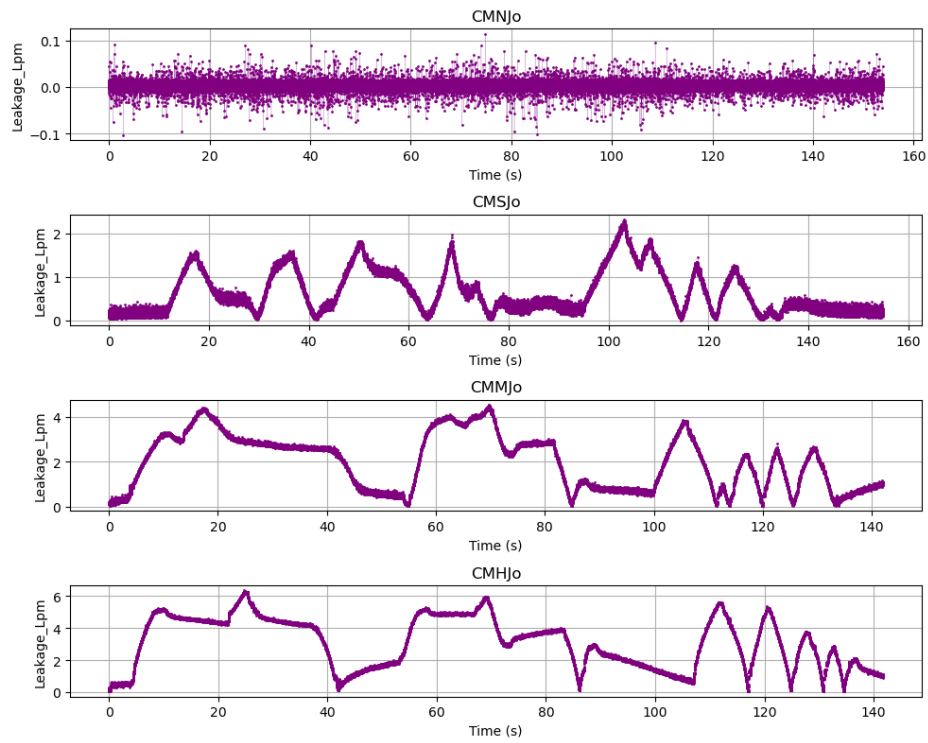


Figure 3.7: Typical fault levels in the EHA for datasets: CMSJo, CMMJo, CMHJo, CMNJo

3.2 Multinomial Logistic Regression with Nonparametric Priors

3.2.1 Preprocessing

Feature Selection

Following the approach in (Hosseinpour et al., 2024), we began with two features derived from the denoised pressure signals P_A and P_B for our classification task. Additional candidate features were added individually to evaluate their impact on classification accuracy. However, none of the added features improved the model’s performance. Therefore, we proceeded using only those two features.

Feature Extraction

To perform Bayesian classification on denoised pressure signals, we extract features that effectively characterize their time-series properties. The Variance Fractal Dimension (VFD) (Ghanbari et al., 2022) is selected as a key feature due to its ability to quantify the complexity of self-affine time series. A time series is termed *self-affine* if smaller sections appear as scaled-down versions of larger sections. In such cases, the variance of the differences between points in the series exhibits a power-law relationship with the time interval, expressed as:

$$\text{Var}[y(t_r) - y(t_s)] \sim |t_r - t_s|^{2H},$$

where H is the Hurst exponent, which measures the degree of self-similarity in the

time series. By taking the logarithm of both sides and fitting a line to the resulting log-log plot, H can be estimated.

In this work, H is computed using the following method: for each lag k , the square root of the variance of differences between time series values separated by k points is calculated. A linear regression is then performed on the log-log plot of these variances versus lag values, and the slope of the fitted line determines H .

Once H is obtained, the VFD is computed as:

$$D_\sigma = E + 1 - H,$$

where $E = 1$ for time series with a single independent variable. This yields $D_\sigma \in [1, 2]$. The VFD is then used as a feature for fault detection in the signals. By analyzing the computed VFD values, distinctions between healthy and faulty conditions can be reliably made. The VFD of the pressure signals is consistently lower in faulty cases, suggesting a decrease in signal complexity compared to healthy ones. This pattern can be explained by understanding that leakage can reduce the system's working pressure, which may also affect the variability of pressure signals.

This methodology is adapted from ([Hosseinpour et al., 2024](#)), where VFD was demonstrated to be an effective tool for time-series feature extraction and fault detection.

Standardizing

As part of the preprocessing step, we standardized the data using z-score normalization. Standardization involves subtracting the mean and scaling to unit variance for each

feature. This process ensures that all features contribute equally to the model by removing the influence of differing scales (Pedregosa et al., 2011).

Combining Labels

Initial experiments with four-class classification resulted in significantly lower accuracy compared to binary classification. To address this, we merged the *Medium* leakage class ('M') with the *High* leakage class ('H'), treating both as a single *High leakage* category. This reduced the classification problem from four to three classes, with the aim of improving model performance. This decision is motivated by the observation that distinguishing between medium and high leakage conditions is inherently more difficult due to their overlapping characteristics in the feature space.

Label Encoding

To make the string-based class labels compatible with classification models, they are converted into numerical values. This is done by assigning a unique integer to each distinct label, allowing categorical target variables to be represented in a numerical form that can be used effectively during model training.

3.2.2 Likelihood of Observed Data

The likelihood was assumed to follow a multinomial distribution, given that the response variable (leakage level) is categorical with multiple classes. Given a dataset $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$, where each $\mathbf{X}_i \in \mathbb{R}^d$ represents a feature vector and y_i denotes the

3.2. MULTINOMIAL LOGISTIC REGRESSION WITH NONPARAMETRIC PRIORS 37

corresponding class label, the probability of assigning y_i to class j is determined using a softmax function. The likelihood is defined as:

$$P(y_i \mid \mathbf{X}_i, W, b) = \prod_{j=1}^C \left(\frac{\exp(z_{i,j})}{\sum_{m=1}^C \exp(z_{i,m})} \right)^{\mathbb{1}(y_i=j)}$$

where $z_{i,j}$ represents the linear combination of features and weights:

$$z_{i,j} = \sum_{l=1}^d X_{il} W_{lj} + b_j.$$

Here, $\mathbb{1}(y_i = j)$ is an indicator function that equals 1 if class j is the true class for observation i , and 0 otherwise. The log-likelihood for the entire dataset is:

$$\log P(y \mid \mathbf{X}, W, b) = \sum_{i=1}^n \sum_{j=1}^C \mathbb{1}(y_i = j) \log \left(\frac{\exp(z_{i,j})}{\sum_{m=1}^C \exp(z_{i,m})} \right).$$

This formulation ensures that the probabilities $P(y_i \mid \mathbf{X}_i, W, b)$ sum to 1 across all classes.

Multinomial logistic regression can be parameterized in two ways: full parameterization for K categories and reduced parameterization for $K - 1$ categories. For C -class classification, the full parameterization is used, meaning each of the C classes has its own weight vector and bias term. Class scores $z_{i,j}$ are transformed into valid probabilities via the softmax function, which automatically ensures that probabilities sum to one across classes. This approach is well suited to the Bayesian framework,

as it allows priors to be placed on the parameters for all classes and enables direct uncertainty quantification for each class without constraining any class’s parameters to be fixed or excluded.

3.2.3 Dirichlet Process Prior for Weights

The Dirichlet Process (DP) prior is a nonparametric prior used to encode uncertainty about the distribution of the weight matrix W . Formally, the prior on the distribution G is given by:

$$G \sim \text{DP}(\alpha, G_0),$$

where G_0 represents the prior belief about the underlying structure of W , and α determines the concentration of G around G_0 . The DP is a distribution over distributions, meaning G itself is a discrete probability measure. This discreteness arises naturally from the stick-breaking construction, which represents G as a weighted sum of point masses.

The stick-breaking construction offers an explicit representation for probability measures sampled from a Dirichlet Process, which are known to be discrete with probability one. This method, introduced by (Sethuraman, 1994), relies on independent sequences of identically distributed random variables. The construction involves two sequences: one for the weights and another for the locations of the probability mass.

The stick-breaking construction expresses G as:

$$G = \sum_{k=1}^{\infty} w_k \delta_{\phi_k},$$

where $\{\phi_k\}$ are the atoms drawn independently from the base distribution G_0 , and δ_{ϕ_k} is the Dirac delta function centered at ϕ_k . The weights $\{w_k\}$ are recursively defined using Beta-distributed random variables $\{\beta_k\}$, such that:

$$w_k = \beta_k \prod_{j=1}^{k-1} (1 - \beta_j), \quad \text{with } \beta_k \sim \text{Beta}(1, \alpha).$$

Each β_k represents the fraction of the remaining “stick” allocated to the k -th component. The stick-breaking process ensures that:

$$\sum_{k=1}^{\infty} w_k = 1,$$

making G a valid probability measure.

For computational purposes, the infinite sum in the stick-breaking process is truncated at a finite level K . This yields the approximation:

$$G_K = \sum_{k=1}^K w_k \delta_{\phi_k}.$$

The base distribution G_0 is selected as a multivariate normal distribution. Specifically, the atoms ϕ_k are drawn independently from:

$$\phi_k \sim \mathcal{N}(0, \sigma^2 I),$$

where σ^2 is a hyperparameter controlling the variance of the base distribution. This choice allows G_0 to act as a prior over the weight space, encouraging smaller values for W in regions with limited data support.

The weights W are modeled as a linear combination of the stick-breaking weights $\{w_k\}$ and the atoms $\{\phi_k\}$:

$$W = \sum_{k=1}^K w_k \phi_k.$$

Here, $\{w_k\}$ represent the probabilities assigned to each component, while $\{\phi_k\}$ are the parameters for those components. The stick-breaking process naturally induces sparsity in the model because the weights decrease exponentially as a result of the multiplicative term $\prod_{j=1}^{k-1} (1 - \beta_j)$. This property allows the Dirichlet Process Prior to allocate most of the probability mass to a small number of components, even when K is large.

In the context of the model, the Dirichlet Process Prior provides a flexible, non-parametric framework for modeling the weight matrix W . By representing W as a mixture of components, the prior captures complex relationships in the data while avoiding overfitting. Additionally, the concentration parameter α regulates the distribution of the stick-breaking weights, with larger values of α leading to more evenly

distributed weights and smaller values encouraging sparsity. This parameter acts as a regularization mechanism, controlling the complexity of the model.

The truncation level K further governs the trade-off between computational feasibility and model expressiveness. While larger values of K allow for greater flexibility, they also increase computational cost.

3.2.4 Gaussian Process Prior for Weights

In this model, a Gaussian Process (GP) prior is placed over the weight matrix of a multinomial logistic regression classifier. The GP prior allows us to incorporate smoothness assumptions into the weight structure by encoding correlations between class-specific weights through a covariance function (or kernel). This enables the model to flexibly capture dependencies among weights in a nonparametric Bayesian framework.

Unlike the Dirichlet Process prior, instead of using a stick-breaking construction, the weights are drawn from a Gaussian Process prior with a covariance matrix defined by a Radial Basis Function (RBF) kernel.

Let W be the weight matrix of shape (d, C) , where d is the number of features and C is the number of classes. For each feature $i \in \{1, 2, \dots, d\}$, the corresponding weight vector $\mathbf{W}_i \in \mathbb{R}^C$ is modeled using a multivariate Gaussian distribution with zero mean and a covariance matrix defined by a Radial Basis Function (RBF) kernel:

$$\mathbf{W}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{K}),$$

where $\mathbf{K} \in \mathbb{R}^{C \times C}$ is the covariance matrix computed using the RBF kernel. The elements of $\mathbf{K}$ are given by:

$$\mathbf{K}_{ij} = \sigma^2 \exp \left(-\frac{(i-j)^2}{2\ell^2} \right),$$

with:

- $i, j \in \{1, 2, \dots, C\}$ representing class indices,
- σ^2 denoting the variance parameter of the kernel,
- ℓ denoting the length scale, which controls the smoothness of the weight correlations across classes.

A separate prior is placed on the weights for each feature, which together form a joint prior over the entire weight matrix W that is both structured and adaptable. The Gaussian Process prior introduces smoothness across weights for different classes, making the Bayesian logistic regression model more expressive while still allowing efficient inference.

3.2.5 Prior for Bias Vector

The bias vector b is assigned a Normal prior with precision τ_{prior} . The probability density function (PDF) for each element b_j is:

$$P(b_j) = \frac{1}{\sqrt{2\pi/\tau}} \exp\left(-\frac{\tau b_j^2}{2}\right).$$

This prior assumes that biases are centered around zero, with variance $1/\tau$. This prior is used for the bias in both the Dirichlet Process and Gaussian Process models.

3.2.6 Posterior Inference Using MCMC

The posterior distribution combines the likelihood of the observed data with the prior distributions over model parameters. Using Bayes' Theorem, the posterior is given by:

$$P(W, b \mid \mathbf{X}, \mathbf{y}) \propto P(\mathbf{y} \mid \mathbf{X}, W, b) \cdot P(W) \cdot P(b),$$

where:

- $P(\mathbf{y} \mid \mathbf{X}, W, b)$ is the likelihood from multinomial logistic regression.
- $P(W)$ represents the prior over the weight matrix. Two types of priors are considered:
 - **Dirichlet Process (DP) Prior:** The weight matrix is modeled as a sparse combination of basis vectors drawn from a base distribution $G_0 = \mathcal{N}(0, \sigma^2 I)$, using stick-breaking weights governed by concentration parameter α .
 - **Gaussian Process (GP) Prior:** Each row of the weight matrix (i.e., feature-specific weight vector across classes) is drawn from a multivariate

Gaussian distribution with zero mean and RBF kernel covariance. This induces smoothness across class-specific weights.

- $P(b)$ is the over bias terms, typically modeled as independent normal distribution with a fixed variance.

Since the posterior is intractable due to the nonlinearity of logistic regression and the complexity of nonparametric priors, we use Markov Chain Monte Carlo (MCMC), specifically the Metropolis-Hastings algorithm, to sample from the posterior.

In both models, the sampling procedure involves:

1. Propose new values W^* and b^* from a symmetric proposal distribution (e.g., Gaussian random walk).
2. Compute the acceptance probability:

$$A(\theta^* | \theta) = \min \left(1, \frac{P(\mathbf{y} | \mathbf{X}, \theta^*)P(\theta^*)}{P(\mathbf{y} | \mathbf{X}, \theta)P(\theta)} \right),$$

where $\theta = (W, b)$.

3. Accept the proposal with probability A , else retain the previous sample.

The Metropolis-Hastings algorithm is run for a sufficient number of iterations, discarding the initial samples (burn-in) to ensure convergence. The resulting samples approximate the true posterior over weights and biases, incorporating model uncertainty.

3.2.7 Posterior Predictive Distribution

After obtaining posterior samples for the weight matrix W and bias vector b , we can make probabilistic predictions on new observations using the posterior predictive distribution. This applies to both models, whether W was drawn using a Dirichlet Process prior or a Gaussian Process prior. For a new input $\mathbf{X}_{\text{new}}$, the posterior predictive probability for each class is given by:

$$P(\mathbf{y}_{\text{new}} \mid \mathbf{X}_{\text{new}}, \mathbf{X}, \mathbf{y}) = \int P(\mathbf{y}_{\text{new}} \mid \mathbf{X}_{\text{new}}, W, b) \cdot P(W, b \mid \mathbf{X}, \mathbf{y}) dW db.$$

Since the posterior distribution $P(W, b \mid \mathbf{X}, \mathbf{y})$ is approximated using MCMC samples, the integral is replaced by a summation over the posterior samples. Let $\{W^{(s)}, b^{(s)}\}_{s=1}^S$ represent S samples from the posterior. The predictive probabilities are then approximated as:

$$P(\mathbf{y}_{\text{new}} \mid \mathbf{X}_{\text{new}}) \approx \frac{1}{S} \sum_{s=1}^S P(\mathbf{y}_{\text{new}} \mid \mathbf{X}_{\text{new}}, W^{(s)}, b^{(s)}).$$

For multinomial logistic regression, the class probabilities are given by the softmax function:

$$P(\mathbf{y}_{\text{new}} = j \mid \mathbf{X}_{\text{new}}, W^{(s)}, b^{(s)}) = \frac{\exp(z_j^{(s)})}{\sum_{c=1}^C \exp(z_c^{(s)})},$$

where $z_j^{(s)}$ is the linear combination of features and weights:

$$z_j^{(s)} = \mathbf{X}_{\text{new}} W^{(s)} + b^{(s)}.$$

The final predicted class is the one with the highest posterior probability:

$$\hat{y}_{\text{new}} = \arg \max_j P(\mathbf{y}_{\text{new}} = j \mid \mathbf{X}_{\text{new}}, \mathbf{X}, \mathbf{y}).$$

3.2.8 Prediction Pipeline

The prediction process for both the Dirichlet Process and Gaussian Process models follows the same general steps:

1. Use each posterior sample $(W^{(s)}, b^{(s)})$ to compute the class scores $z_j^{(s)} = \mathbf{X}_{\text{new}} W_j^{(s)} + b_j^{(s)}$.
2. Apply the softmax function to convert the scores into class probabilities.
3. Average the resulting probabilities across all posterior samples.
4. Assign each observation to the class with the highest average predictive probability.

This Bayesian approach ensures that predictive uncertainty reflects the uncertainty in the model parameters, which is especially important given the small dataset. Both the Dirichlet and Gaussian process priors offer the flexibility to capture complex patterns without overfitting. The Dirichlet Process and Gaussian Process models were implemented in Python using the PyMC3 probabilistic programming library.

3.2.9 Hyperparameter Tuning

To improve classification performance and control model complexity, we performed hyperparameter tuning using a grid search strategy combined with k -fold cross-validation ($k = 5$). For each model, a set of key hyperparameters was explored over predefined ranges, and the combination yielding the highest average accuracy across folds was selected as optimal.

Dirichlet Process Model. The following hyperparameters were tuned:

- **Concentration parameter (α):** Controls the sparsity of the Dirichlet Process.
- **Base standard deviation (σ):** Determines the spread of the base distribution used in the stick-breaking construction.
- **Truncation level (K):** Sets the number of components used to approximate the infinite DP mixture.
- **Bias prior precision (τ_{prior}):** Affects the normal prior over bias terms in the classifier.

Gaussian Process Model. For the GP prior model, the following hyperparameters were considered:

- **Length scale (ℓ):** Controls the smoothness of the GP kernel, defining how quickly correlations decay between class-specific weights.

- **Kernel variance** (σ^2): Scales the magnitude of the covariance function.
- **Bias prior precision** (τ_{prior}): As above, influences the spread of the prior over the bias vector.

For each hyperparameter configuration, the model was trained and evaluated across all folds using standard performance metrics: accuracy, precision, and F1 score. The results were averaged across folds to assess generalization performance. In addition, various diagnostic plots were generated.

The best-performing set of hyperparameters was selected based on average accuracy and used for final evaluation and interpretation.

3.3 Infinite Hidden Markov Model (IHMM)

The training and evaluation of the Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM) for actuator leakage classification is grounded in a fully Bayesian framework. This section provides a comprehensive breakdown of the hierarchical structure, prior distributions, likelihood functions, posterior formulation, and prediction strategy, emphasizing the probabilistic meaning and choice of each parameter.

The preprocessing steps for this model are the same as for the DP and GP models, except for feature extraction, as we want to use the raw pressure signals as input.

3.3.1 Hierarchical Structure and Priors

The HDP-HMM defines a generative process with two primary sources of variability: the hidden state transitions and the observed emissions. Both components are governed by their own likelihood functions and associated priors, forming a hierarchical Bayesian model. This implementation is based on the work of (Johnson and Willsky, 2013) and it is implemented using the `pyhsmm` library.

At the top level, we model an infinite number of potential hidden states using the *Griffiths–Engen–McCloskey* (GEM) distribution, which is the stick-breaking representation of a Dirichlet Process. In a $\text{GEM}(\gamma)$ distribution, the weights $\{\beta_k\}$ are generated by repeatedly “breaking” a unit-length stick into random pieces:

$$\beta_k = v_k \prod_{l=1}^{k-1} (1 - v_l), \quad v_k \sim \text{Beta}(1, \gamma), \quad \beta \sim \text{GEM}(\gamma).$$

Here, γ controls the expected number of active states. A larger γ leads to more states with smaller probabilities, encouraging model flexibility.

Each hidden state j then draws its transition probabilities from a Dirichlet Process:

$$\pi_j \sim \text{DP}(\alpha, \beta)$$

The concentration parameter α determines how strongly the transition distribution π_j is tied to the global distribution β . Higher α values allow more variability between transition behaviors of different states.

Together, this defines the prior over latent state sequences $Z = (z_1, z_2, \dots, z_T)$, where:

$$z_1 \sim \beta, \quad z_t \sim \pi_{z_{t-1}}, \quad t > 1.$$

For each hidden state k , the emission parameters μ_k, Σ_k are drawn from a conjugate Normal-Inverse-Wishart distribution:

$$\Sigma_k \sim \mathcal{W}^{-1}(\nu_0, \Psi_0), \quad \mu_k \mid \Sigma_k \sim \mathcal{N}(\mu_0, \Sigma_k / \kappa_0)$$

Here, $\mu_0 = \mathbf{0}$ reflects a non-informative prior mean for pressure. $\kappa_0 = 0.25$ encodes moderate confidence in this prior, balancing the prior belief and empirical observations. $\nu_0 = d + 2$ ensures a proper prior over covariance in d -dimensional space, while $\Psi_0 = I_d$ provides a weak prior over scale.

3.3.2 Likelihood Functions

The HDP-HMM involves two levels of likelihoods corresponding to the hidden state dynamics and emission generation. The complete-data likelihood of an observed sequence $X = (x_1, \dots, x_T)$ and latent states $Z = (z_1, \dots, z_T)$ is:

$$p(X, Z \mid \mu, \Sigma, \pi, \beta) = p(z_1 \mid \beta) \prod_{t=2}^T p(z_t \mid \pi_{z_{t-1}}) \prod_{t=1}^T p(x_t \mid \mu_{z_t}, \Sigma_{z_t})$$

The first product represents the likelihood of the hidden state trajectory given the transition distributions, and the second represents the Gaussian likelihood of pressure observations given hidden states.

3.3.3 Posterior Inference

Given a dataset $\mathcal{D} = \{X^{(i)}\}_{i=1}^N$, the posterior distribution over model parameters and latent states is:

$$p(Z, \mu, \Sigma, \pi, \beta \mid \mathcal{D}) \propto p(\beta) \prod_j p(\pi_j \mid \beta) \prod_k p(\mu_k, \Sigma_k) \\ \times \prod_i p(z_1^{(i)} \mid \beta) \prod_{t=2}^{T_i} p(z_t^{(i)} \mid \pi_{z_{t-1}^{(i)}}) \prod_{t=1}^{T_i} p(x_t^{(i)} \mid \mu_{z_t^{(i)}}, \Sigma_{z_t^{(i)}})$$

Because this distribution is analytically intractable, we use collapsed Gibbs sampling that integrates out π and β to simplify inference.

3.3.4 Prediction

Once one HDP-HMM is trained per class $c \in \{\text{No}, \text{Small}, \text{High}\}$, we compute the marginal likelihood of a test sequence X_{test} under each class model:

$$\hat{y} = \arg \max_c \log p(X_{\text{test}} \mid M_c)$$

The model M_c includes the inferred parameters $\hat{\mu}, \hat{\Sigma}, \hat{\pi}, \hat{\beta}$ specific to class c . The prediction is based on the class model that assigns the highest likelihood to the test sequence, reflecting the best generative fit.

This hierarchical Bayesian formulation allows the model to capture both local dynamics (via state transitions) and emission distributions specific to each leakage class, while leveraging global sharing of structure through the HDP. The priors are chosen to be weakly informative, giving the model flexibility while ensuring proper posterior distributions.

3.3.5 Hyperparameter Tuning

The hyperparameters for the HDP-HMM were tuned using 3-fold cross-validation. The following hyperparameters were considered: α (concentration parameter), γ (global stick-breaking concentration), the initial state concentration, and the maximum number of hidden states.

The hyperparameter tuning process involved evaluating the model’s performance using standard metrics such as accuracy, precision, and F1 score across different configurations. The set of hyperparameters that yielded the highest average accuracy across folds was selected. These best-performing hyperparameters were then used for final model training and evaluation.

Chapter 4

Results

4.1 Results of Dirichlet Process and Gaussian Process Models

We evaluate the performance of the DP and GP models using various hyperparameter combinations. All models were developed and executed using Python, leveraging its robust libraries for statistical modeling and data analysis. The results are presented in terms of accuracy, precision, and F1-score for each configuration. Number of MCMC iterations is set to 10000 for hyperparameter tuning and 20000 with burn-in of 8000 for one of the best hyperparameters. Then, to evaluate the quality of posterior inference, we use three diagnostic plots:

Trace Plot: Displays the sampled values across MCMC iterations.

Autocorrelation Plot: Shows how correlated samples are with their previous values.

Posterior Predictive Plot: Compares model-generated samples to observed data.

4.1.1 Dirichlet Process Model Evaluation

Table 4.1 summarizes the results for DP model (This is just some of the results).

α	σ	truncation K	τ	Accuracy	Precision	F1 score
0.1	0.1	5	1	0.6444	0.4997	0.5373
0.1	0.1	20	2	0.6667	0.5082	0.5571
0.1	0.5	5	0.1	0.6889	0.5560	0.5912
0.1	1	20	1	0.7111	0.6103	0.6280
0.1	2	10	0.1	0.6911	0.5848	0.6114
0.1	2	20	0.5	0.7111	0.6103	0.6280
0.1	5	15	0.5	0.6911	0.5848	0.6114
0.1	5	20	2	0.6689	0.5181	0.5744
1	0.1	5	0.5	0.6022	0.4865	0.4884
1	0.1	5	2	0.6444	0.4997	0.5373
1	0.1	15	0.5	0.6244	0.4928	0.5138
1	0.5	20	1	0.6889	0.5560	0.5912
1	1	20	2	0.7111	0.6103	0.6280
1	2	20	0.5	0.7111	0.6103	0.6280
1	5	10	1	0.6911	0.5848	0.6114
5	0.1	5	0.1	0.6667	0.5038	0.555
5	0.5	20	1	0.6667	0.5082	0.5571
5	1	20	1	0.6667	0.5038	0.5555
5	2	5	0.5	0.7111	0.6103	0.6280
5	2	20	1	0.6889	0.5560	0.5912
5	5	5	0.1	0.6911	0.5848	0.6114
5	5	20	1	0.7111	0.6103	0.6280
10	0.1	5	0.5	0.6022	0.4865	0.4884
10	0.5	5	1	0.6889	0.5560	0.5912
10	0.5	10	0.5	0.6667	0.5038	0.5555
10	0.5	20	1	0.6244	0.4928	0.5138
10	1	5	2	0.7111	0.6103	0.6280
10	1	20	0.5	0.6667	0.5082	0.5571
10	2	5	1	0.7111	0.6103	0.6280

Table 4.1: DP Model Evaluation Results for Various Hyperparameters

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS 55

In this section, we present the trace plots, autocorrelation plots, and posterior predictive checks for each fold of the Dirichlet Process prior model for the following hyperparameter set, which is one of the best-performing hyperparameter set: $(\alpha) = 0.1$, $(\sigma) = 0.5$, $(K) = 15$, $(\tau_{\text{prior}}) = 1.0$

Fold 1

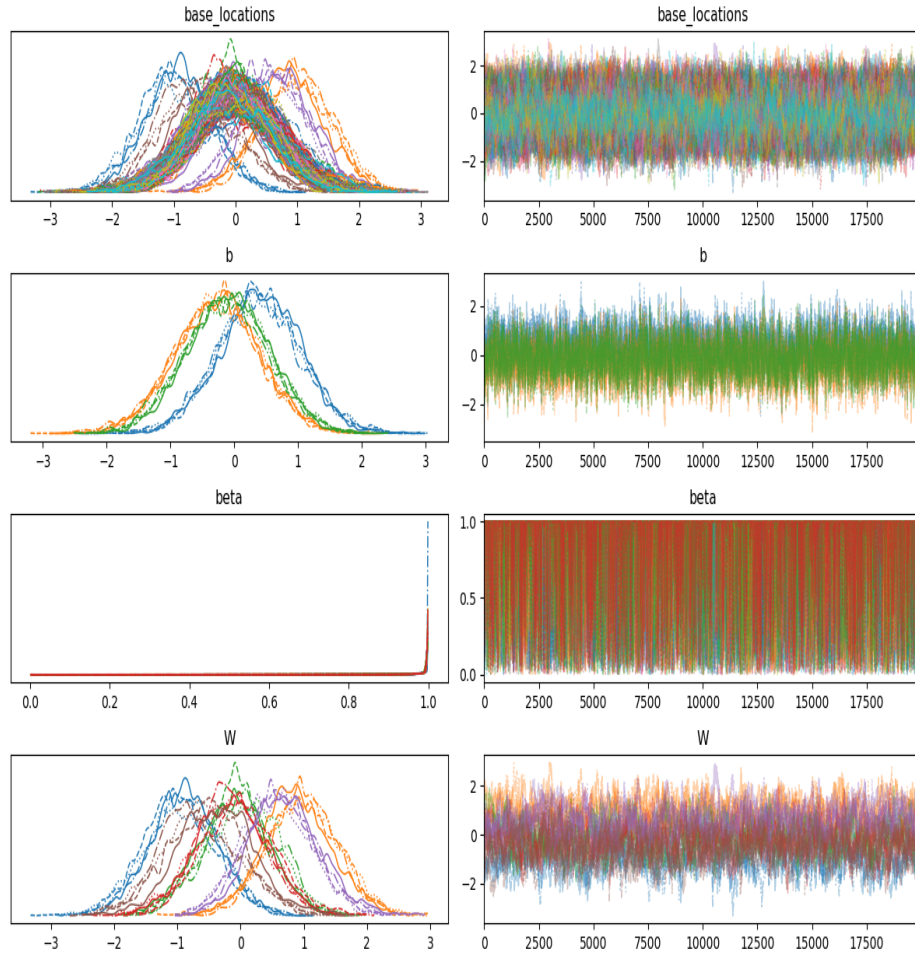


Figure 4.1: Fold 1, Trace Plot

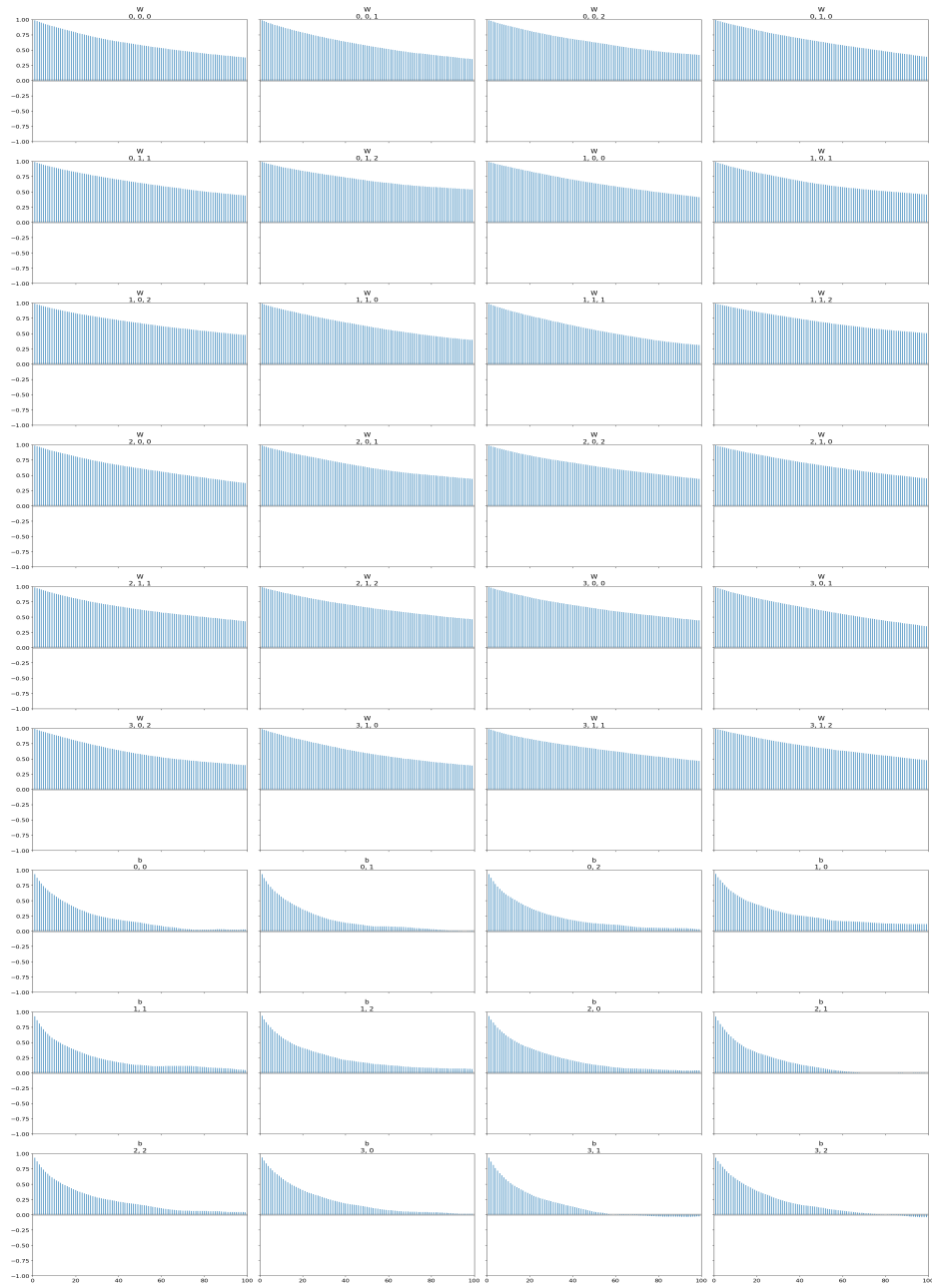


Figure 4.2: Fold 1, Autocorrelation Plot

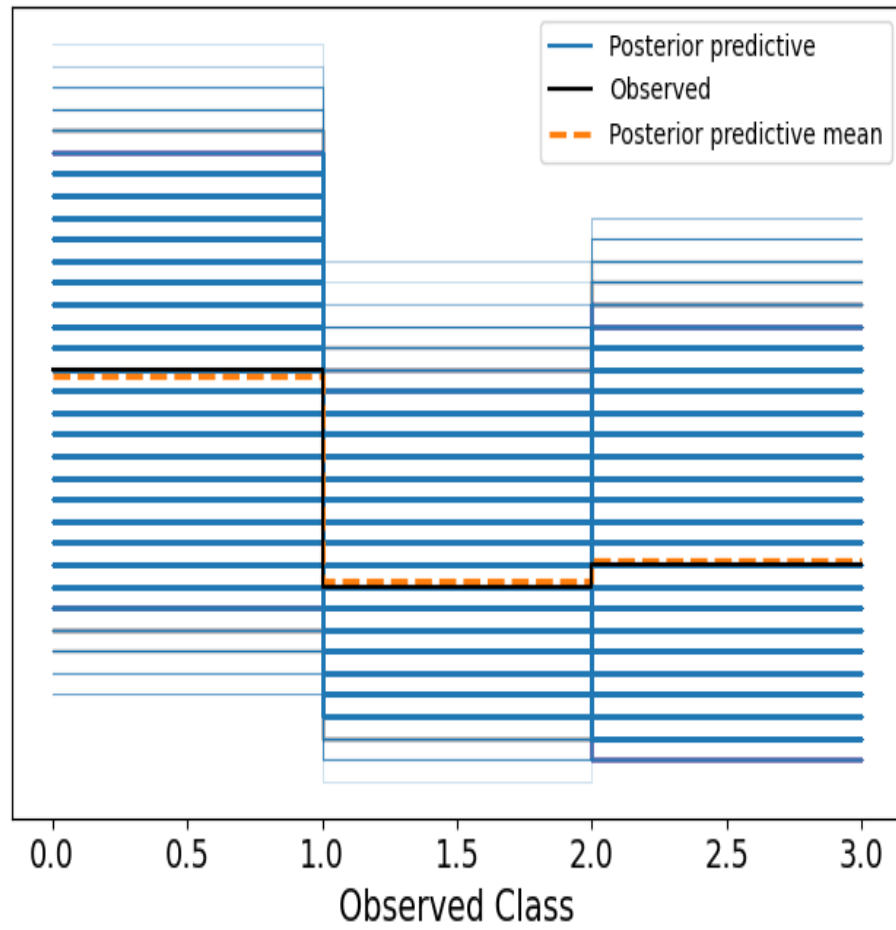


Figure 4.3: Fold 1, Posterior Predictive Check

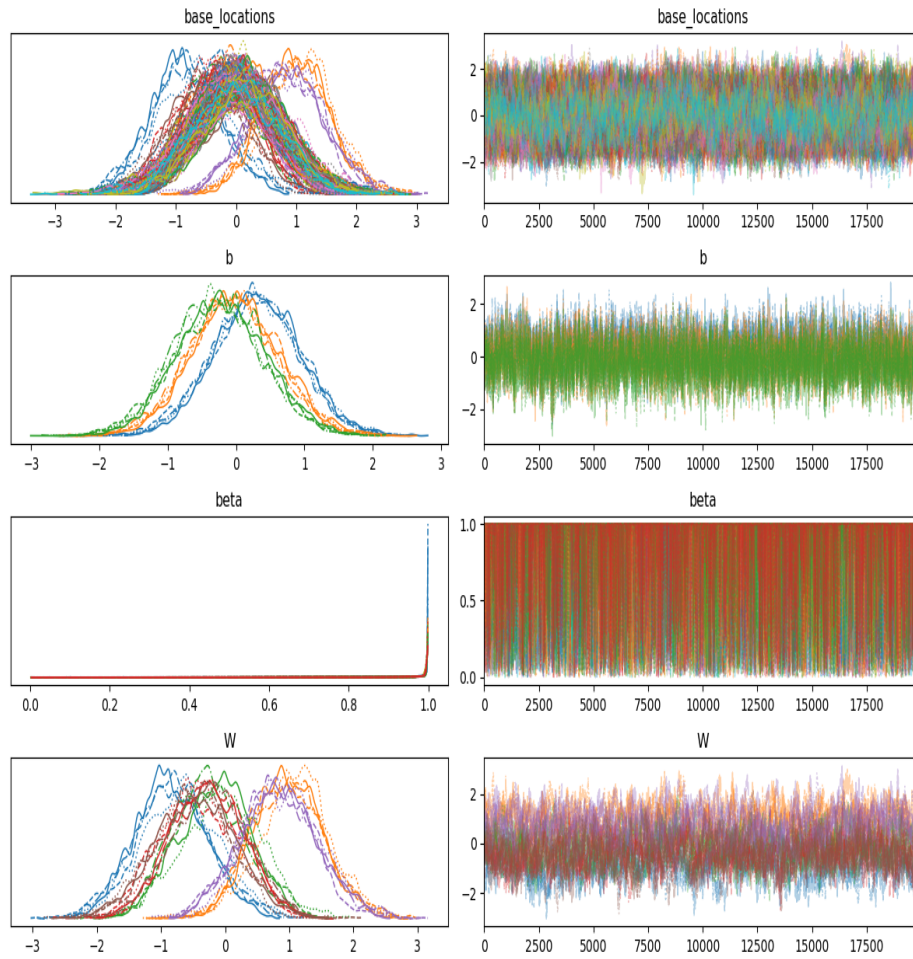
Fold 2

Figure 4.4: Fold 2, Trace Plot

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS 59

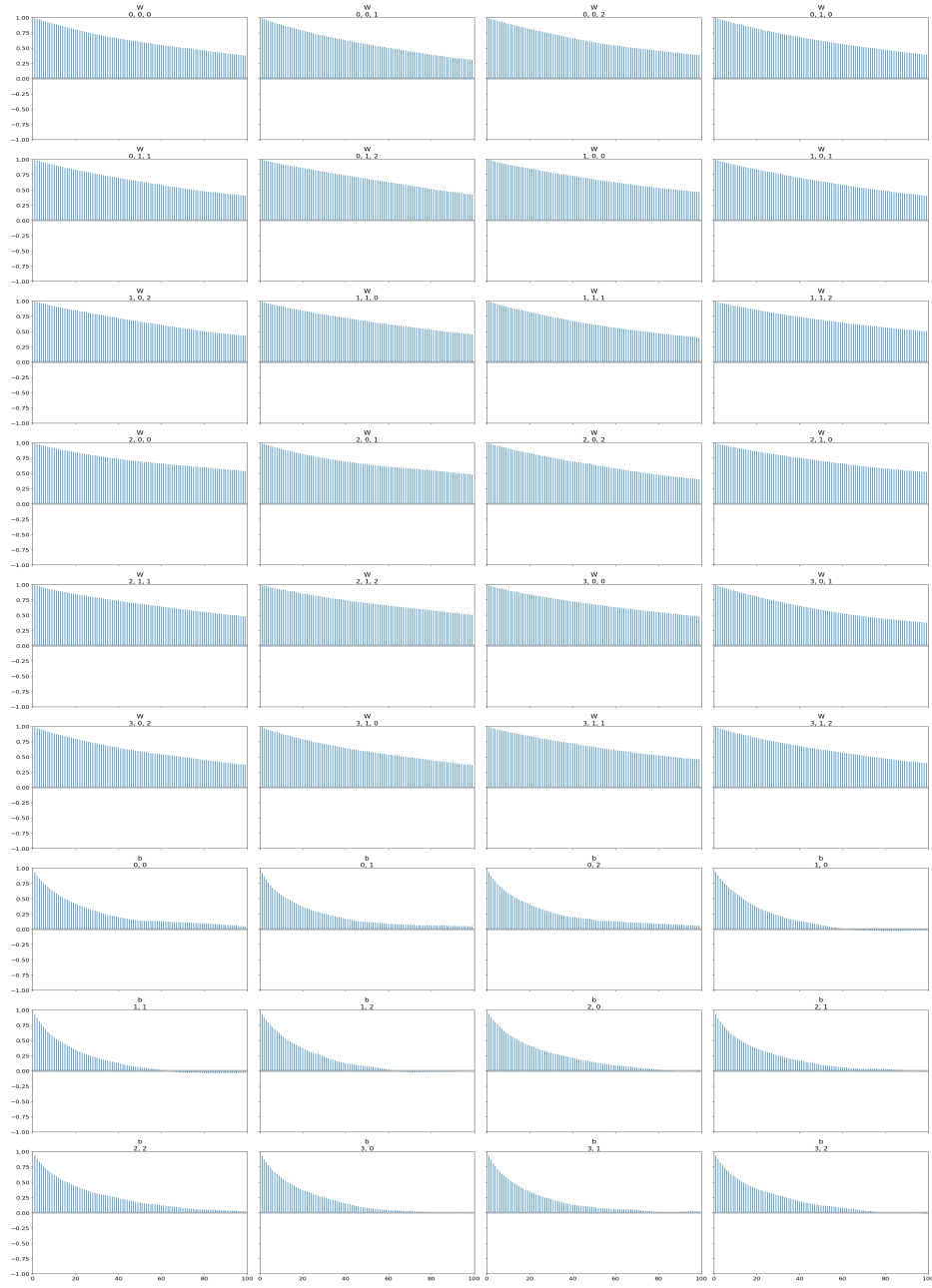


Figure 4.5: Fold 2, Autocorrelation Plot

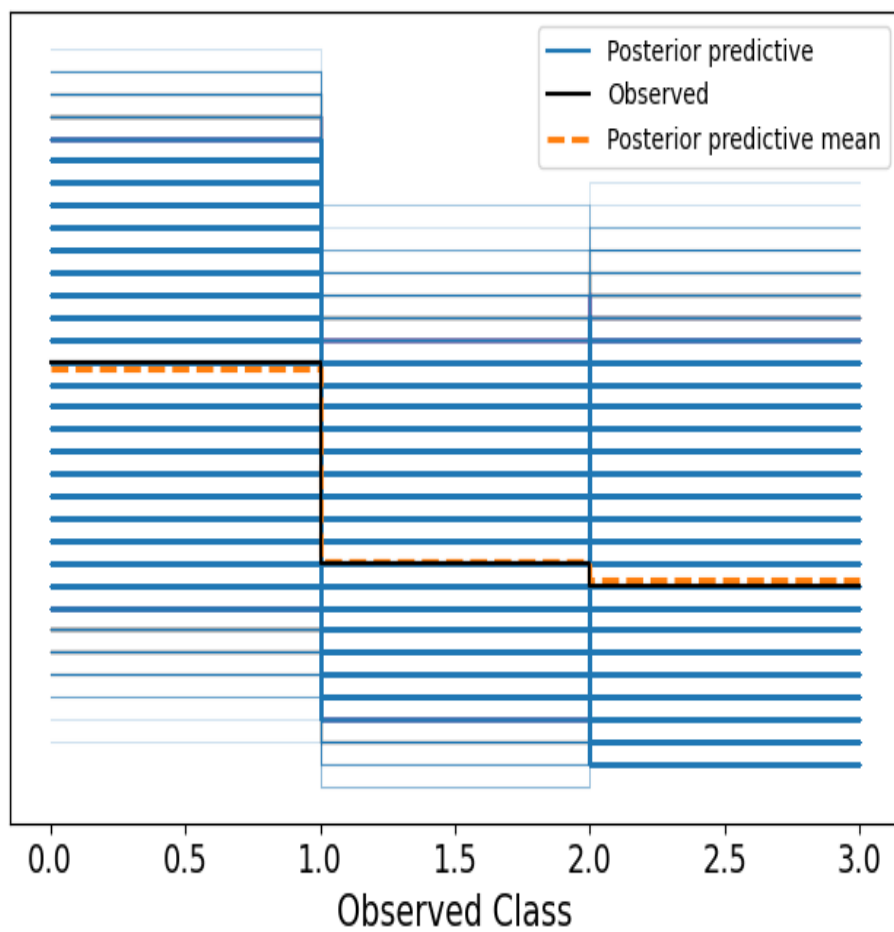


Figure 4.6: Fold 2, Posterior Predictive Check

Fold 3

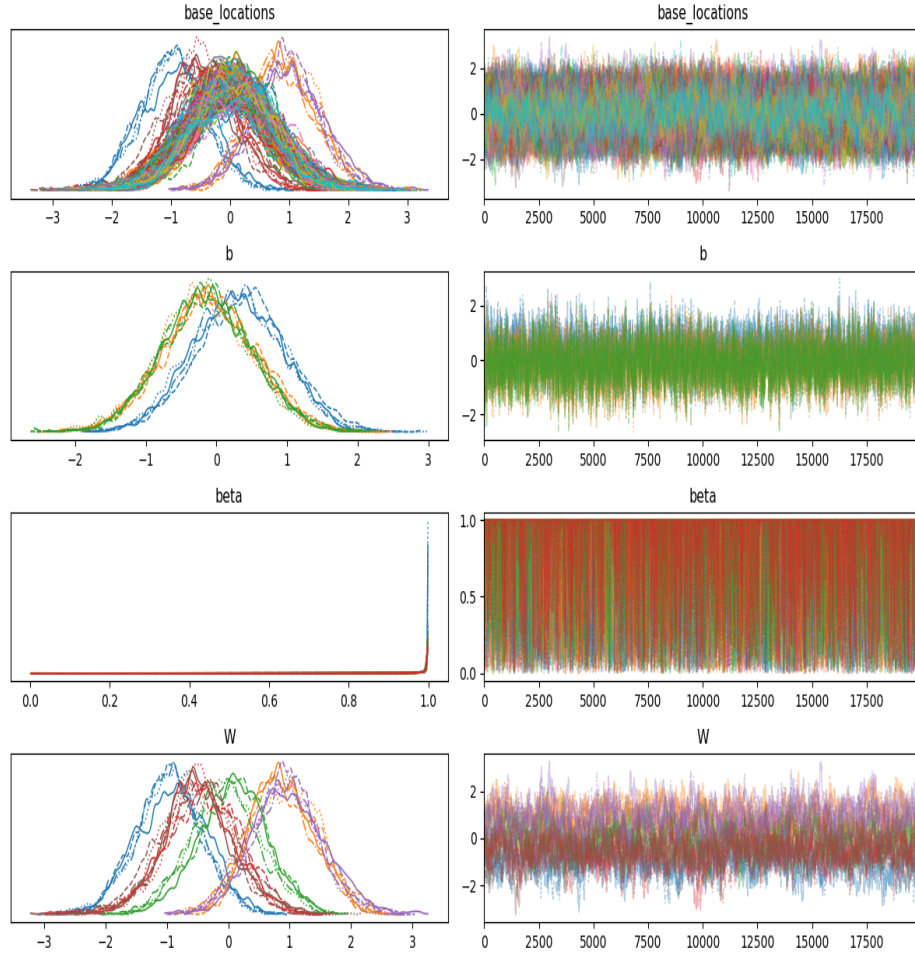


Figure 4.7: Fold 3, Trace Plot

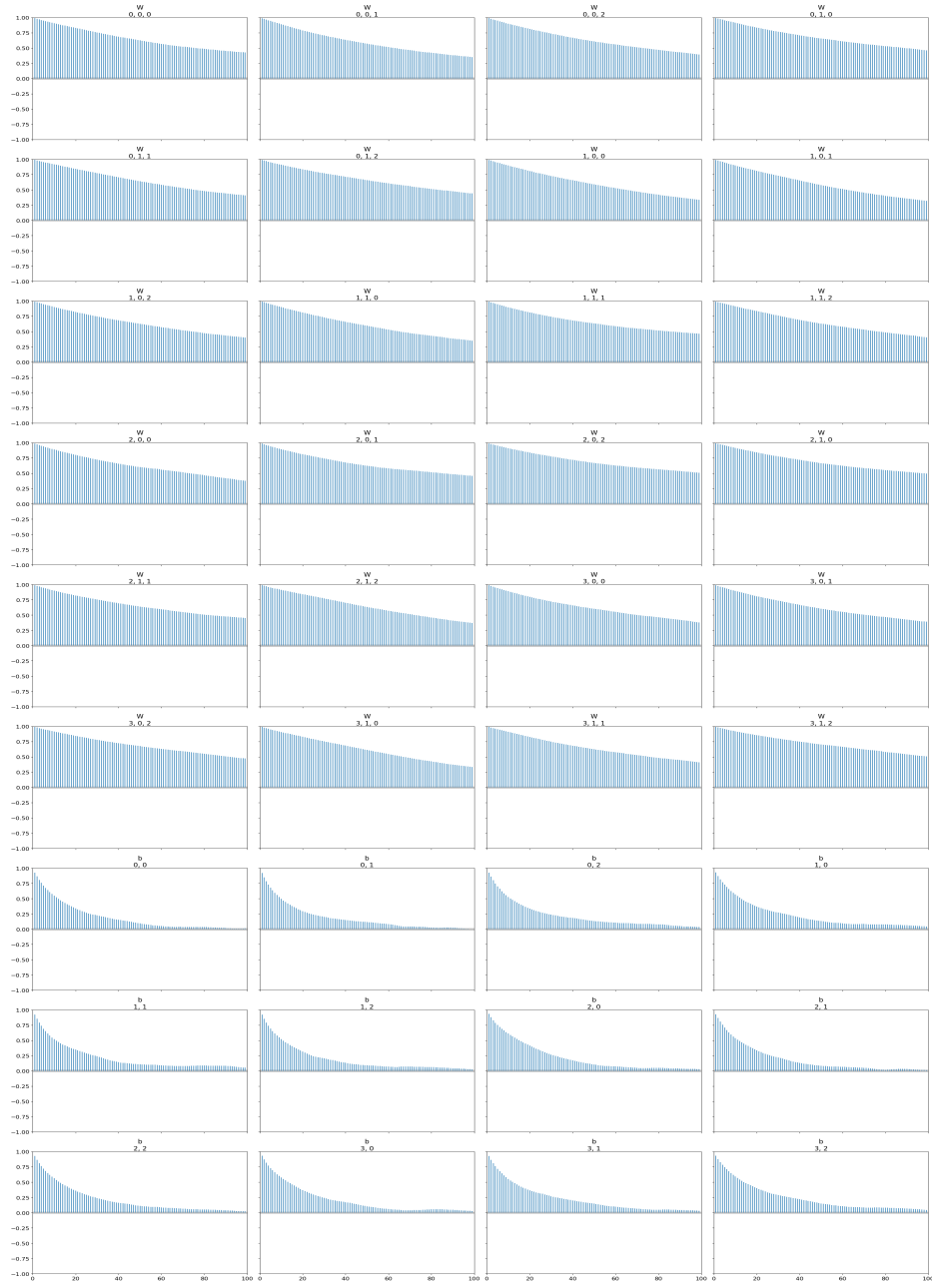


Figure 4.8: Fold 3, Autocorrelation Plot

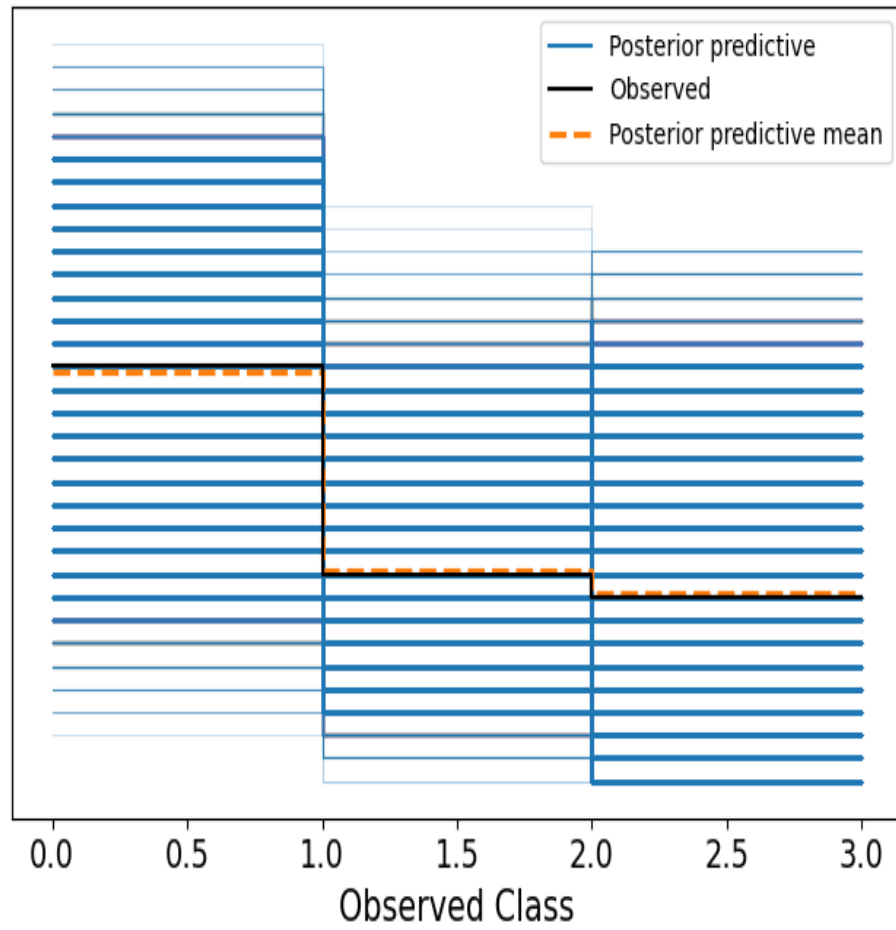


Figure 4.9: Fold 3, Posterior Predictive Check

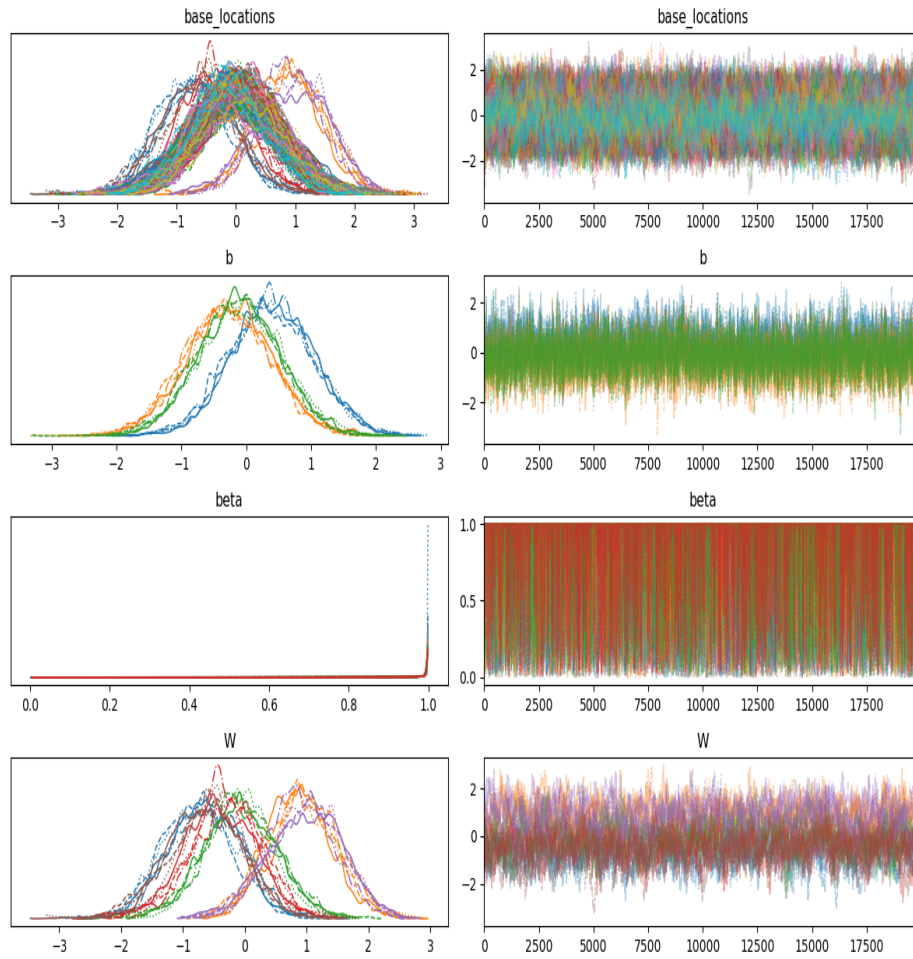
Fold 4

Figure 4.10: Fold 4, Trace Plot

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS65

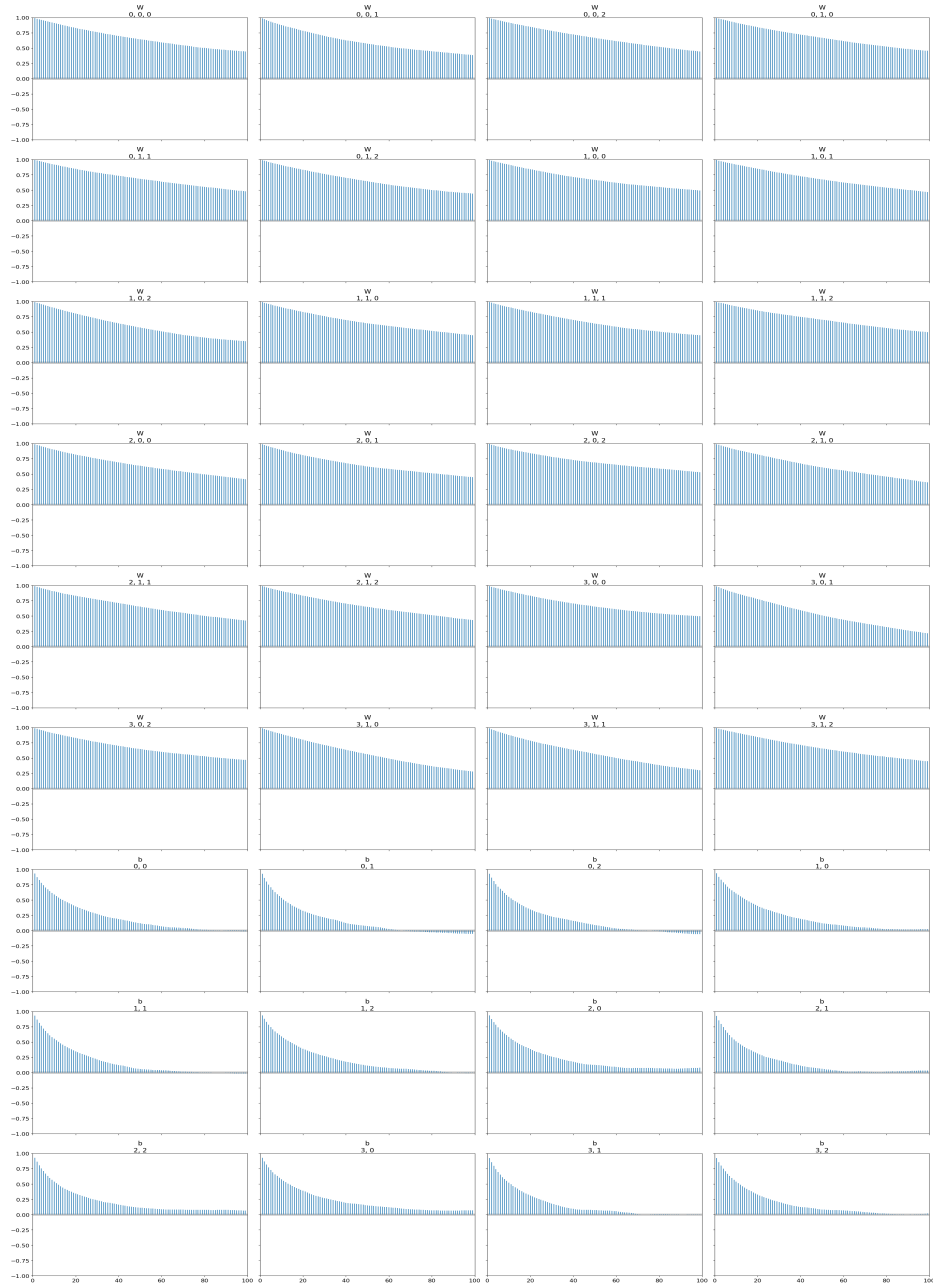


Figure 4.11: Fold 4, Autocorrelation Plot

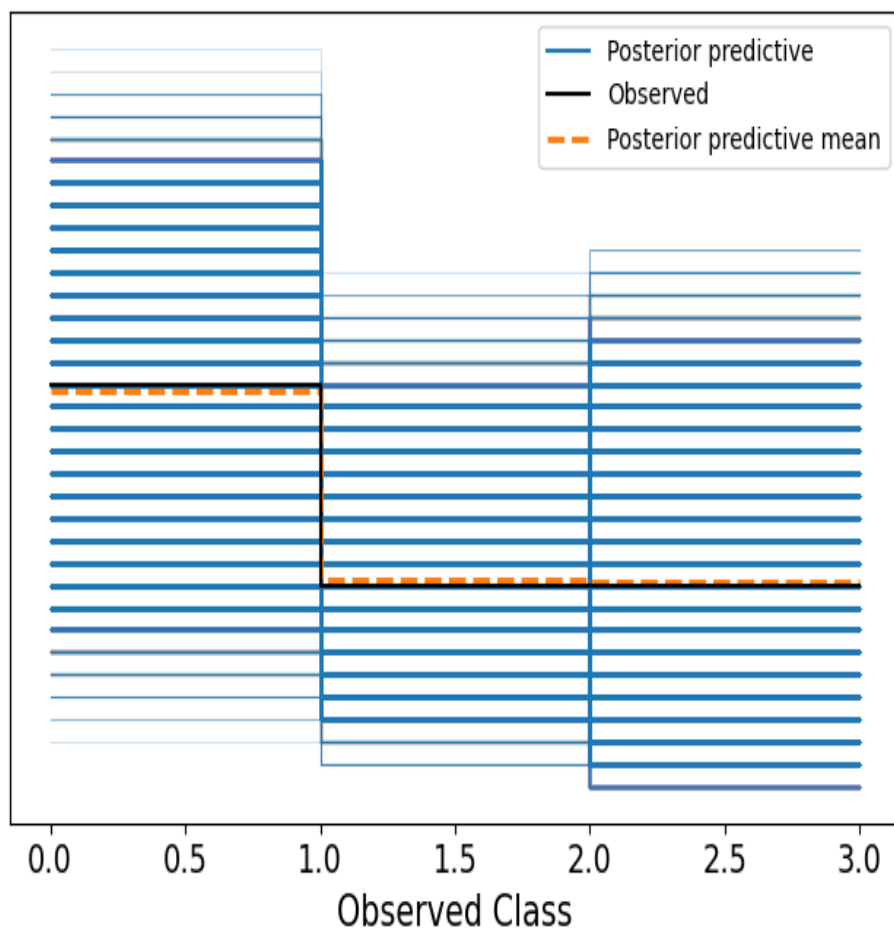


Figure 4.12: Fold 4, Posterior Predictive Check

Fold 5

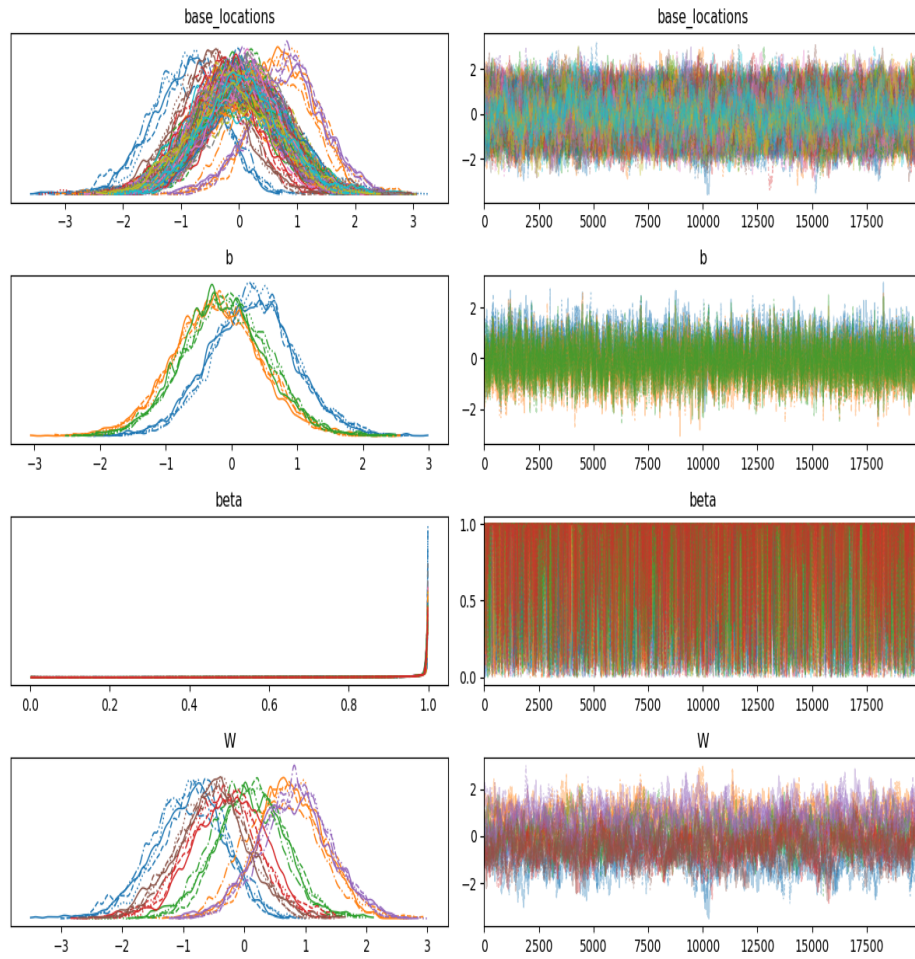


Figure 4.13: Fold 5, Trace Plot

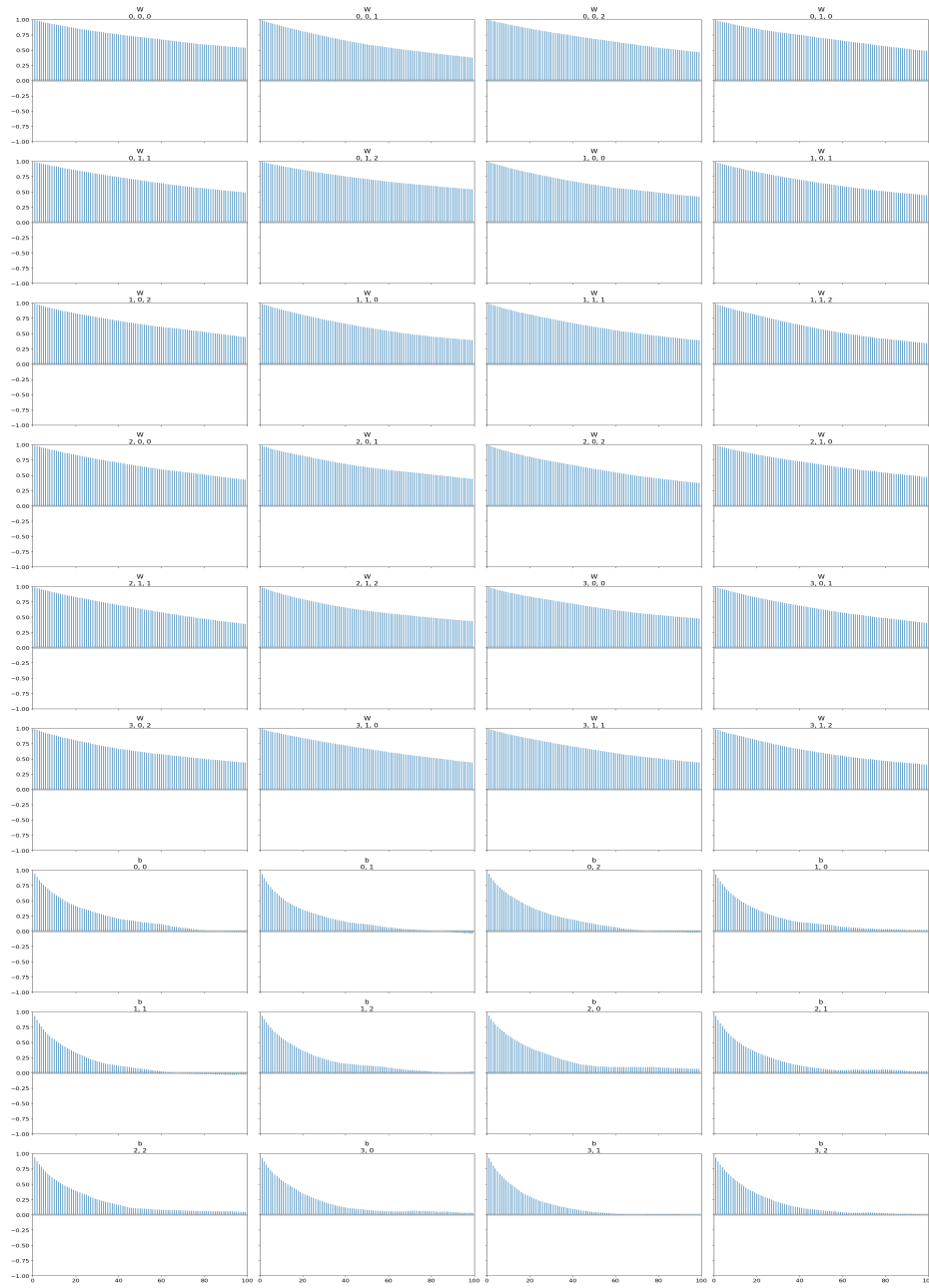


Figure 4.14: Fold 5, Autocorrelation Plot

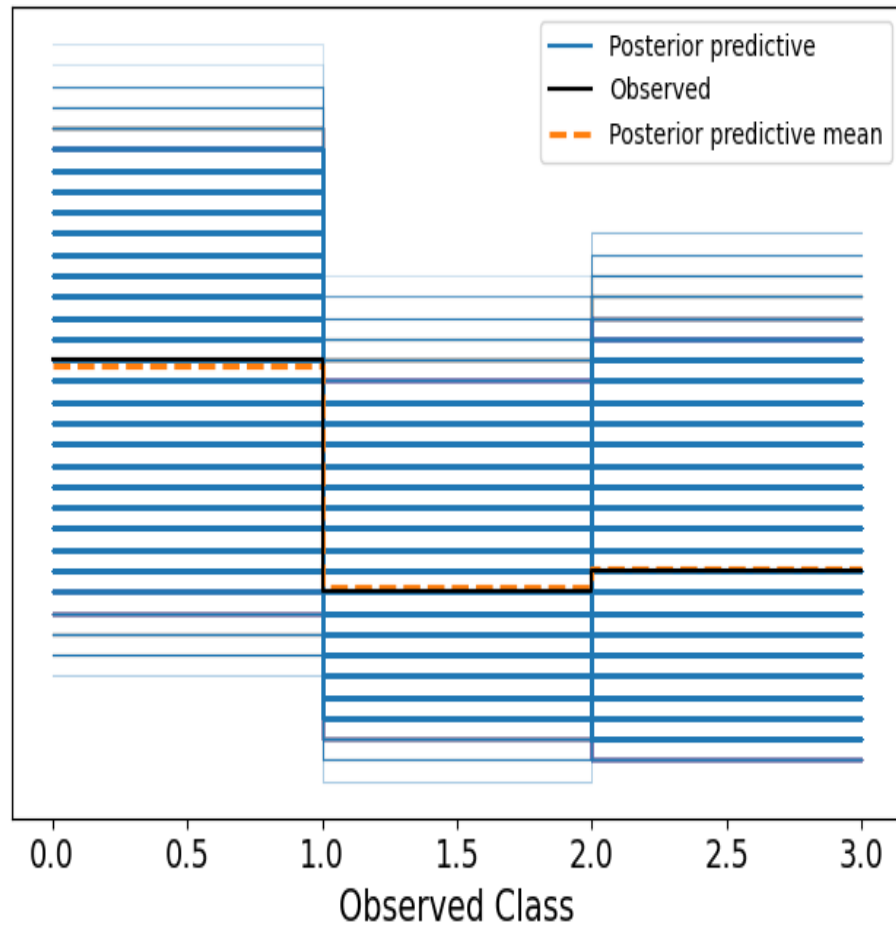


Figure 4.15: Fold 5, Posterior Predictive Check

Interpretation of Results: Dirichlet Process Model

Trace Plots: These plots show the sampled values of the regression weights W , biases b , stick-breaking weights β that control the importance of atoms, and base locations (atoms) across MCMC iterations. Each color corresponds to a different dimension of that parameter.

Conclusion: The chain appears to be well-mixed with no visible trends, indicating convergence.

Autocorrelation Plots: These plots measure correlation between successive MCMC samples. Fast decay toward zero indicates that samples are nearly independent. In our plots, most W and b components show a rapid drop-off, supporting good mixing.

Posterior Predictive Check: Compares the model’s predictions (blue) to the observed class frequencies (black). When the predictive lines align with the observed values, it suggests the model has learned the class structure. In our plots, the orange dashed line (posterior predictive mean) tracks the observed frequencies well, indicating reasonable predictive performance.

4.1.2 Gaussian Process Model Evaluation

Table 4.2 summarizes the results for GP model (This is just some of the results).

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS71

ℓ	σ^2	τ	Accuracy	Precision	F1 score
0.1	0.1	5	0.6667	0.5082	0.5571
0.1	0.1	20	0.6889	0.5560	0.5912
0.1	0.5	20	0.7111	0.6103	0.6280
0.1	1	5	0.7111	0.6103	0.6280
0.1	1	10	0.6911	0.5848	0.6114
0.1	1	20	0.6911	0.5848	0.6114
0.1	2	5	0.6911	0.5848	0.6114
0.1	2	20	0.6911	0.5848	0.6114
0.5	0.1	5	0.6667	0.5082	0.5571
0.5	0.1	15	0.6889	0.5560	0.5912
0.5	0.1	20	0.6889	0.5560	0.5912
0.5	0.5	5	0.7111	0.6103	0.6280
0.5	0.5	20	0.7111	0.6103	0.6280
0.5	1	5	0.7111	0.6103	0.6280
0.5	1	20	0.7111	0.6103	0.6280
0.5	2	5	0.6911	0.5848	0.6114
0.5	2	15	0.6911	0.5848	0.6114
1	0.1	10	0.6444	0.4997	0.5373
1	0.1	20	0.6667	0.5082	0.5571
1	0.5	10	0.6889	0.5437	0.5909
1	0.5	15	0.7111	0.6103	0.6280
1	1	10	0.6889	0.5437	0.5909
1	1	20	0.6889	0.5437	0.5909
1	2	5	0.6889	0.5437	0.5909
2	0.1	5	0.5	0.2568	0.3388
2	0.1	10	0.52	0.3073	0.3748
2	0.1	15	0.52	0.3128	0.3796
2	0.1	20	0.52	0.3198	0.3851
2	0.5	10	0.6244	0.5090	0.5264
2	0.5	20	0.6644	0.5849	0.5863
2	1	20	0.6867	0.6170	0.6156
2	2	5	0.6867	0.6170	0.6156
2	2	20	0.6867	0.6170	0.6156

Table 4.2: GP Model Evaluation Results for Various Hyperparameters

In this section, we present the trace plots, autocorrelation plots, and posterior predictive checks for each fold of the Gaussian Process prior model for the following hyperparameter set: $(\ell) = 0.1$, $(\sigma^2) = 1.0$, $(\tau_{\text{prior}}) = 5.0$

Fold 1

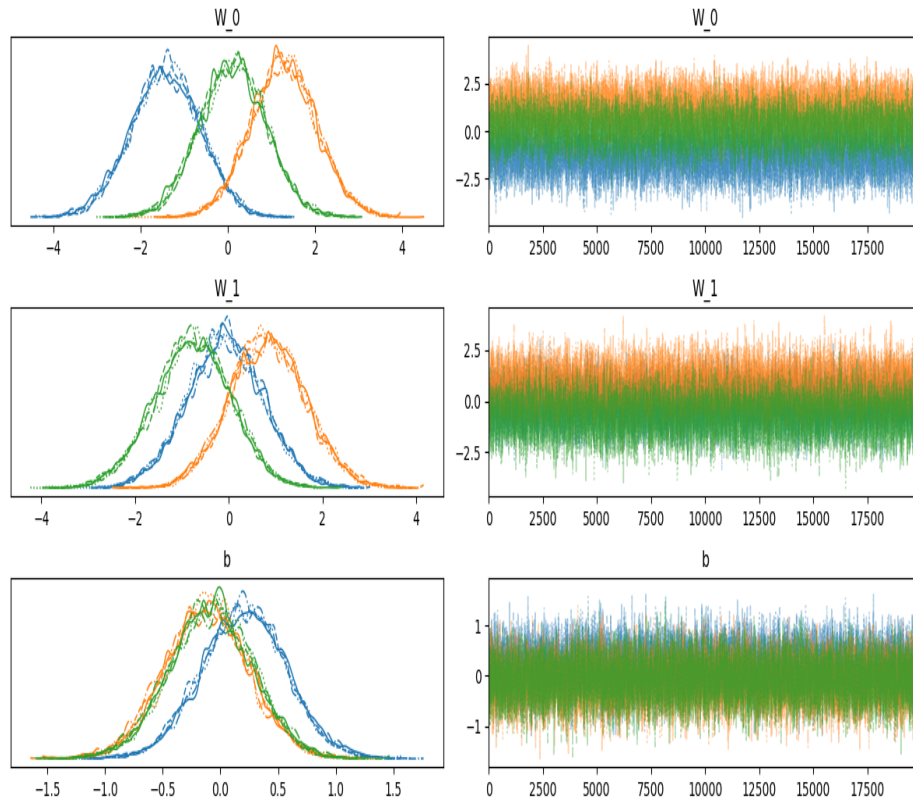


Figure 4.16: Fold 1, Trace Plot

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS73

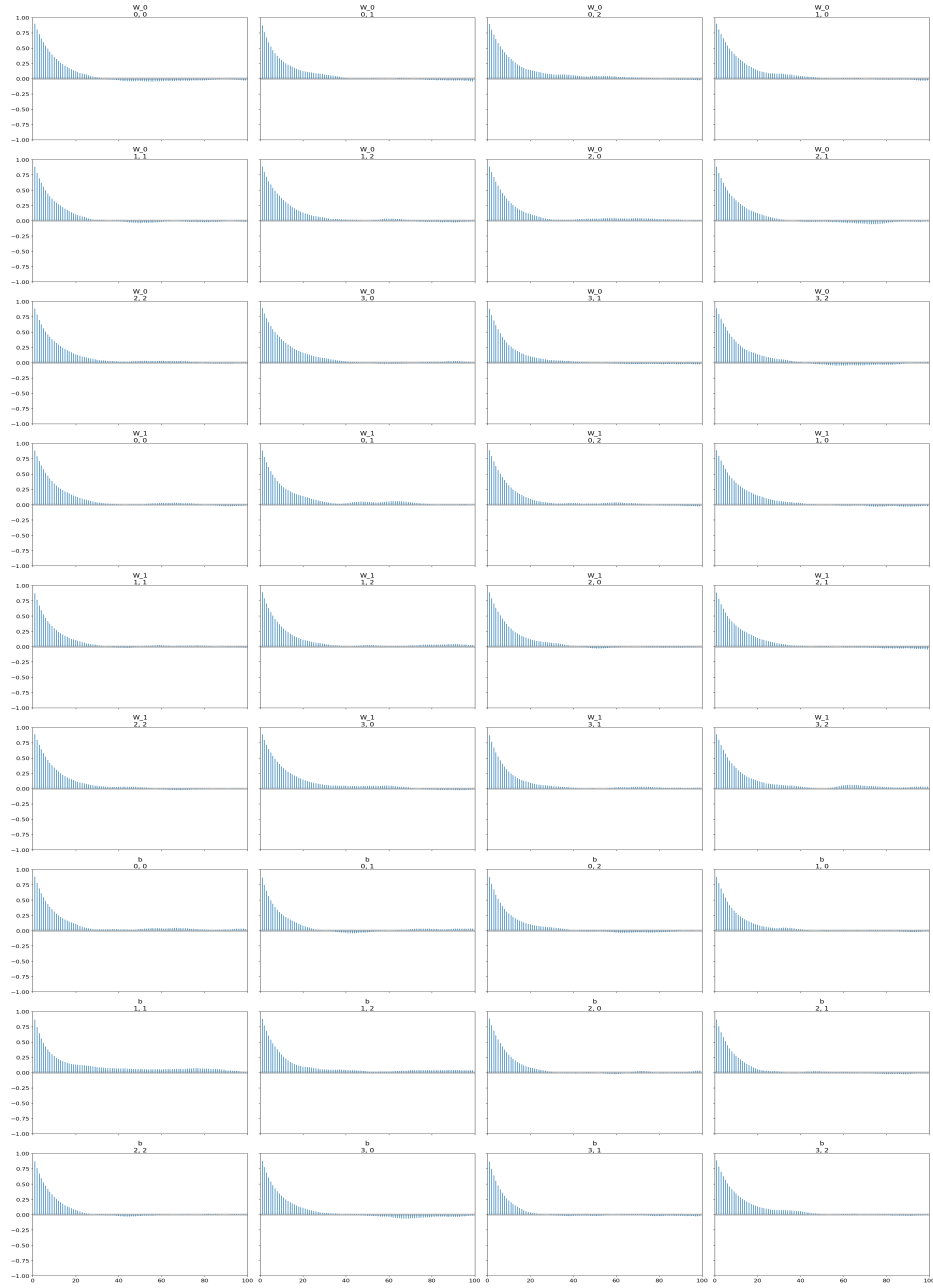


Figure 4.17: Fold 1, Autocorrelation Plot

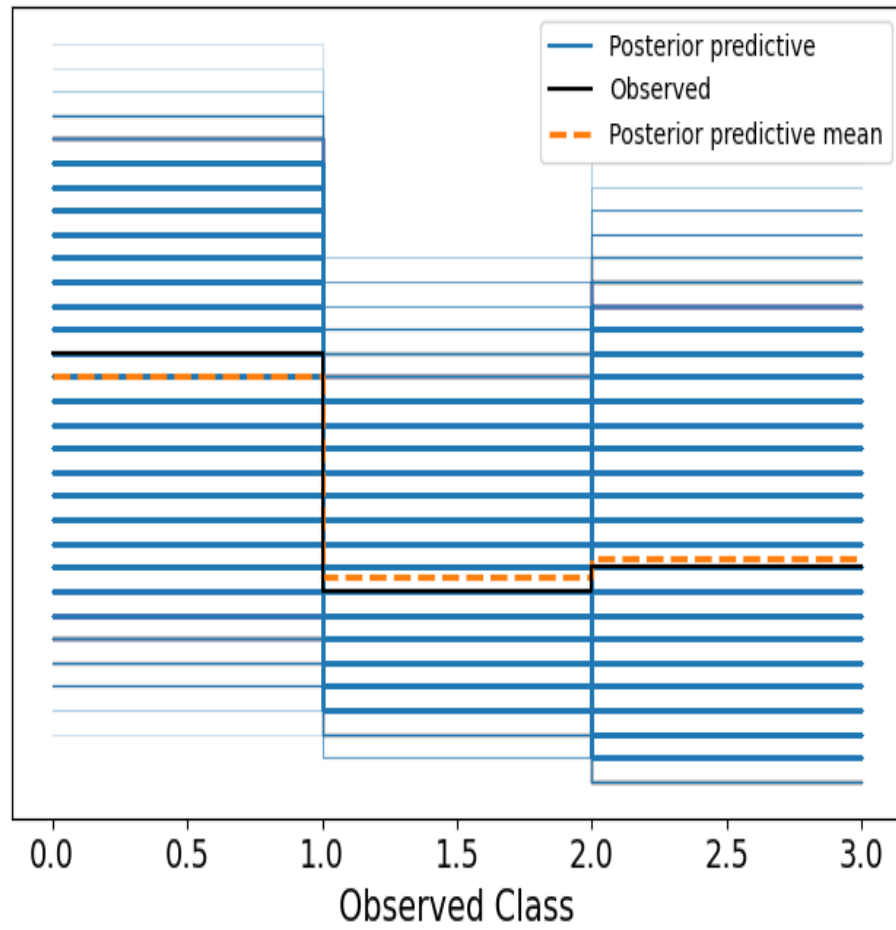


Figure 4.18: Fold 1, Posterior Predictive Check

Fold 2

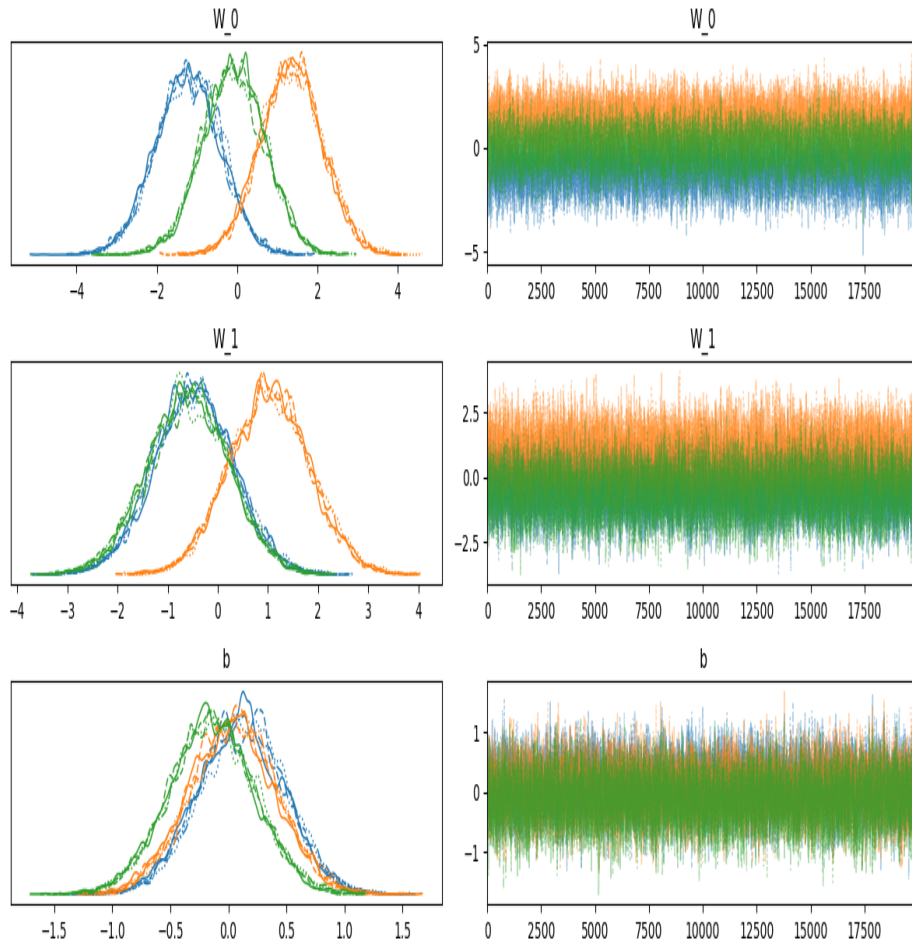


Figure 4.19: Fold 2, Trace Plot

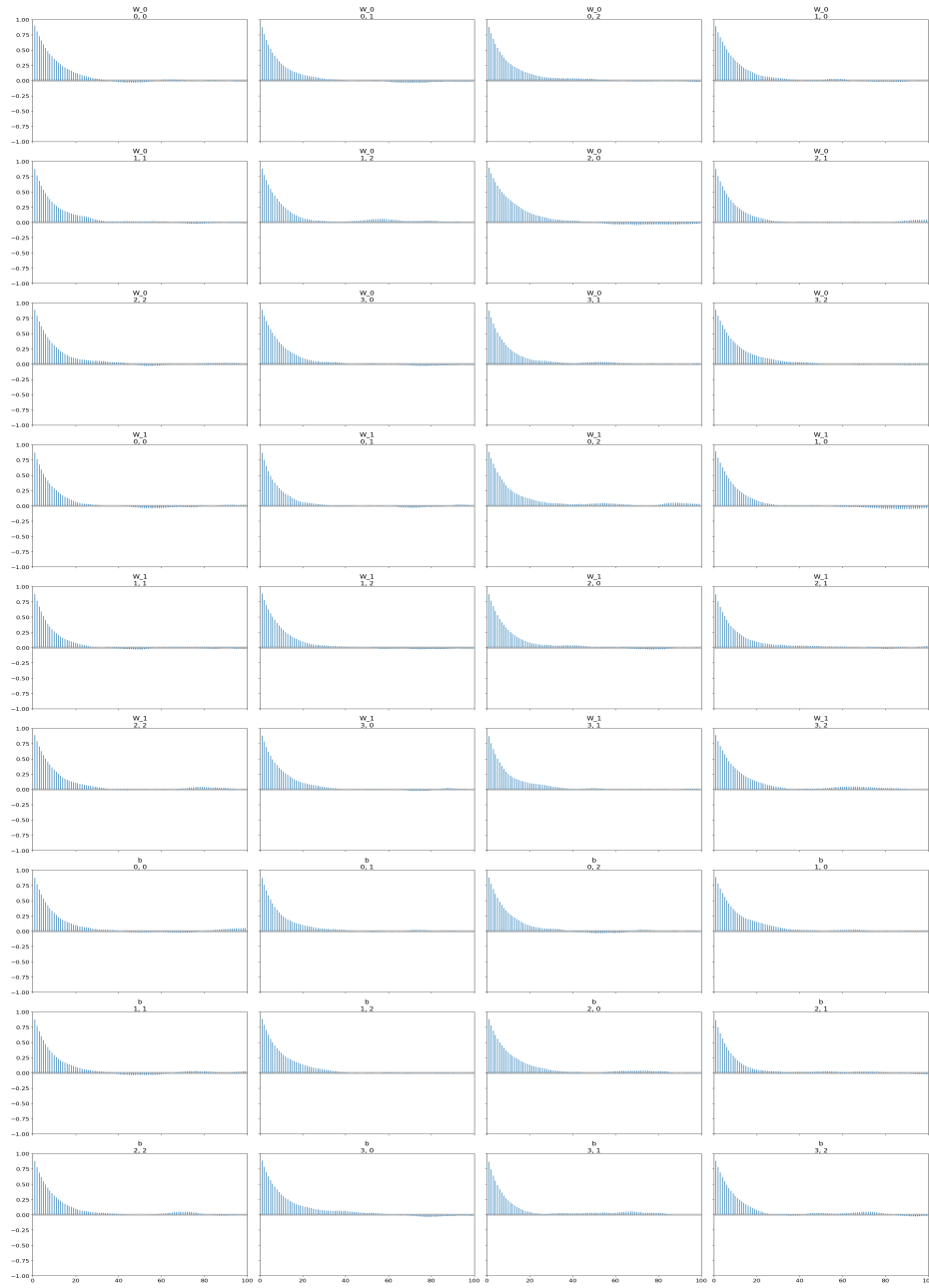


Figure 4.20: Fold 2, Autocorrelation Plot

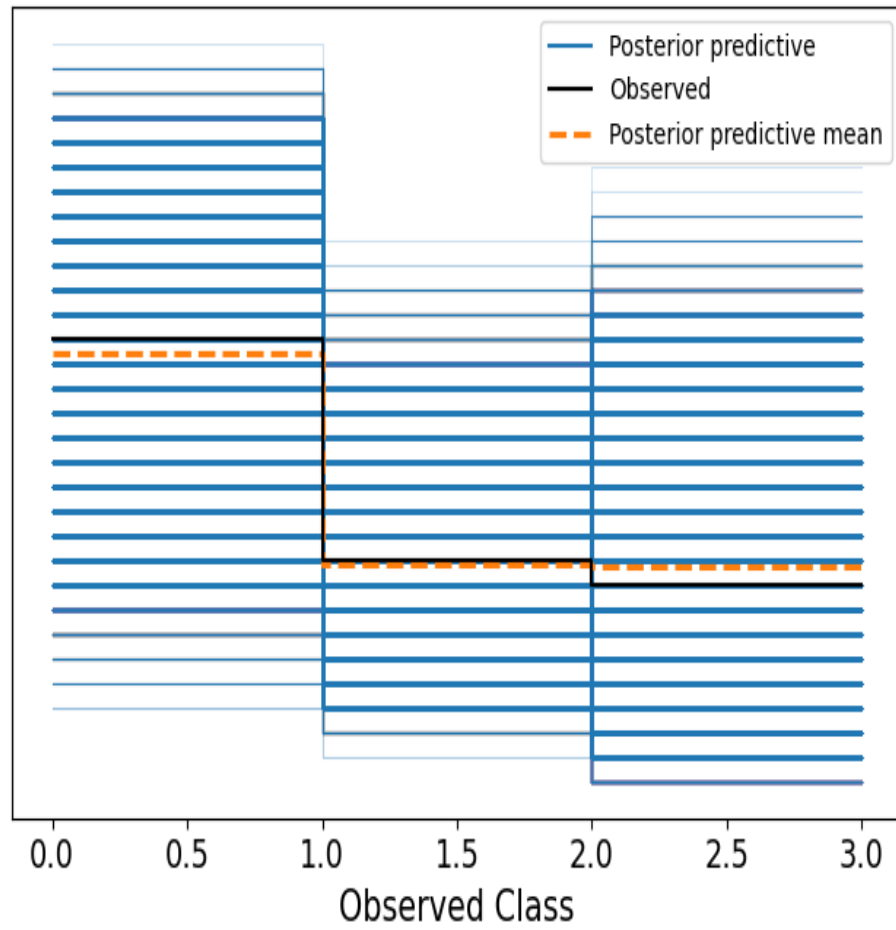


Figure 4.21: Fold 2, Posterior Predictive Check

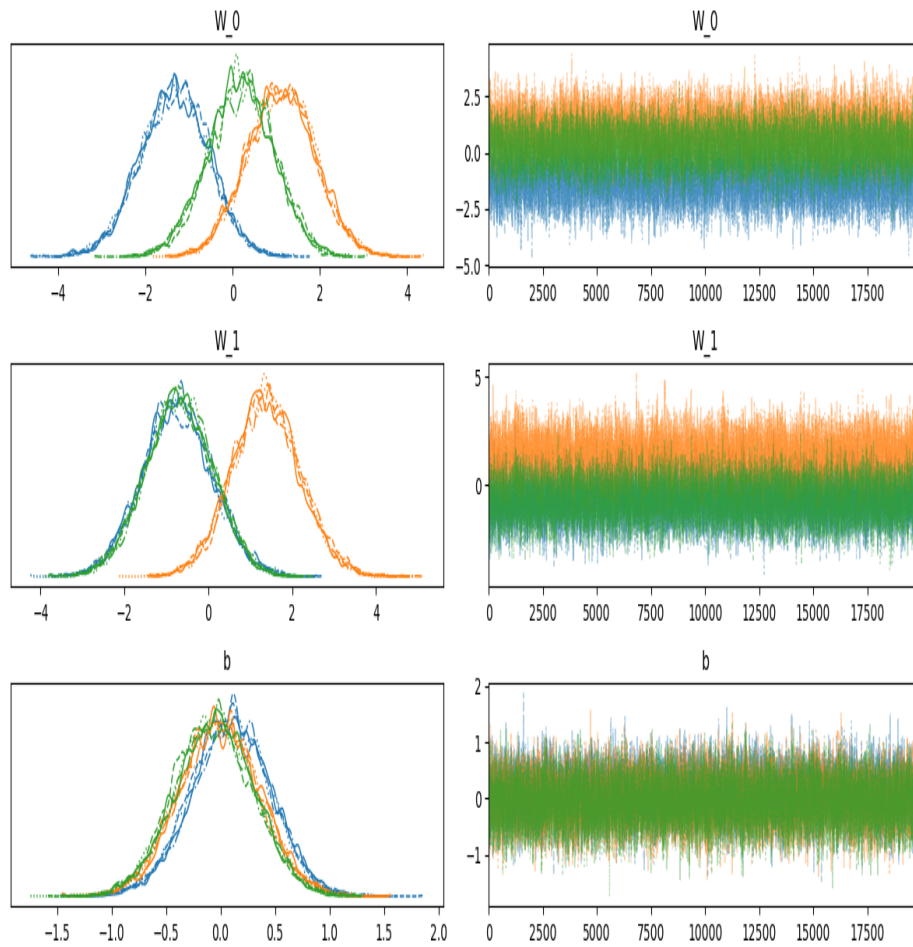
Fold 3

Figure 4.22: Fold 3, Trace Plot

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS 79

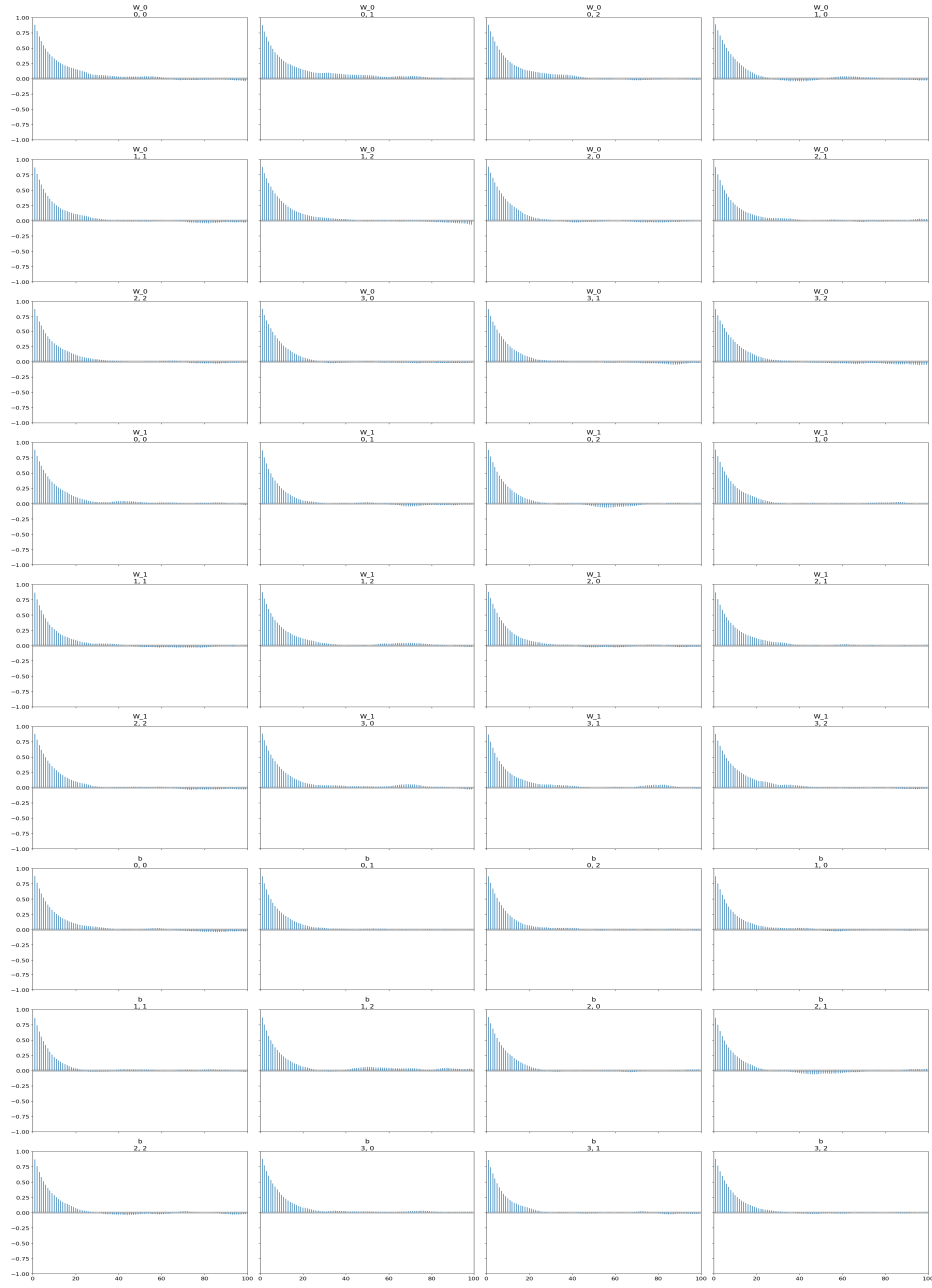


Figure 4.23: Fold 3, Autocorrelation Plot

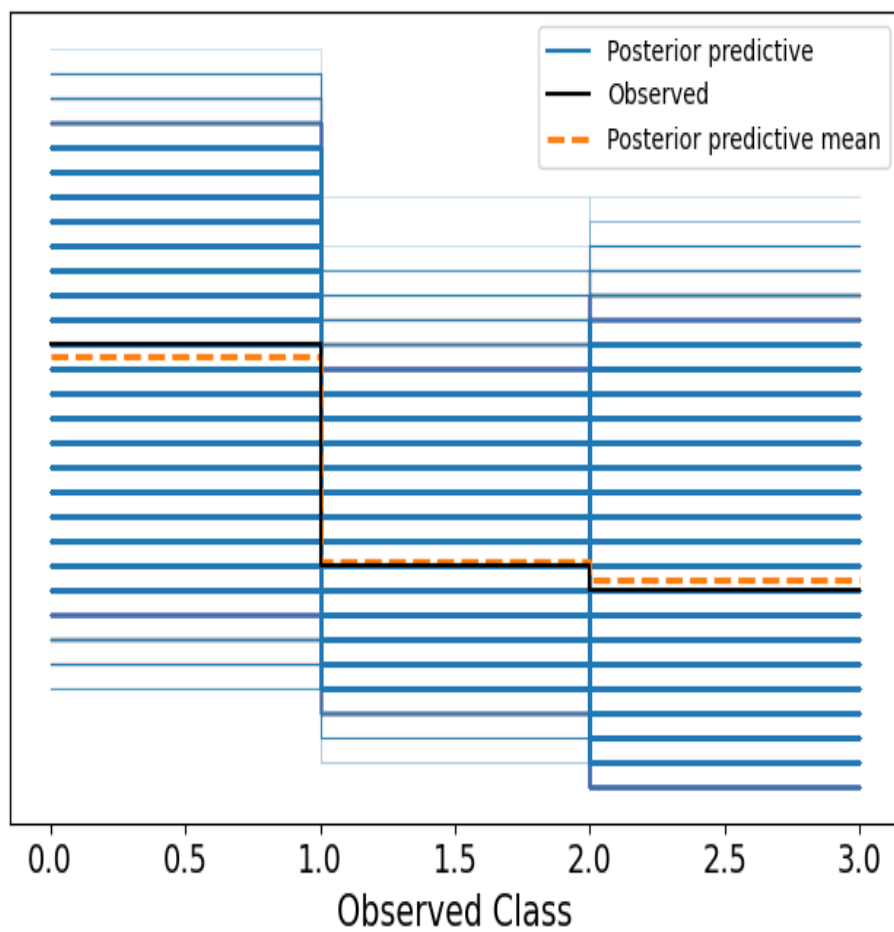


Figure 4.24: Fold 3, Posterior Predictive Check

Fold 4

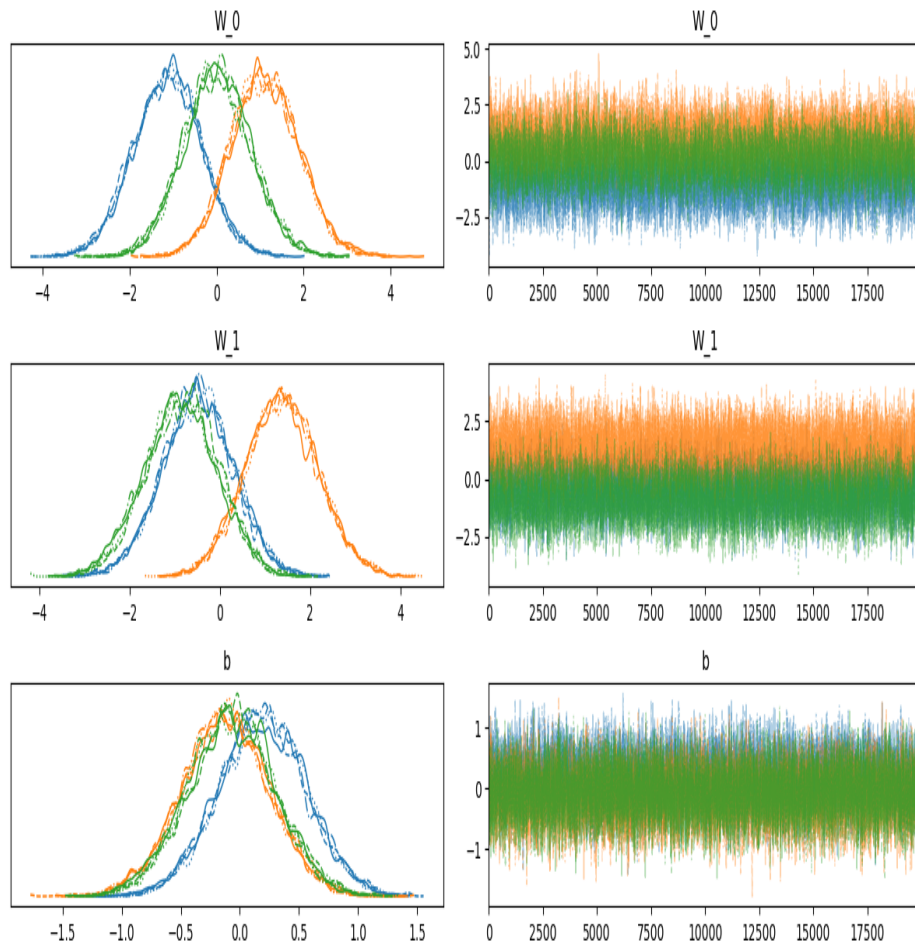


Figure 4.25: Fold 4, Trace Plot



Figure 4.26: Fold 4, Autocorrelation Plot

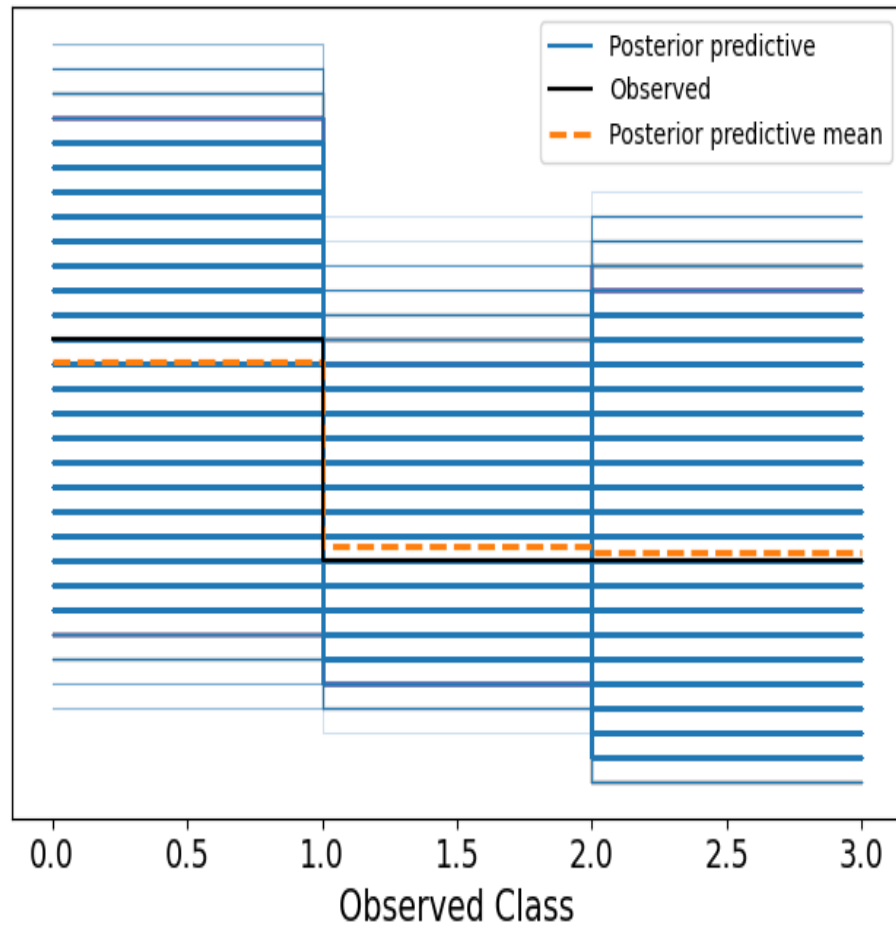


Figure 4.27: Fold 4, Posterior Predictive Check

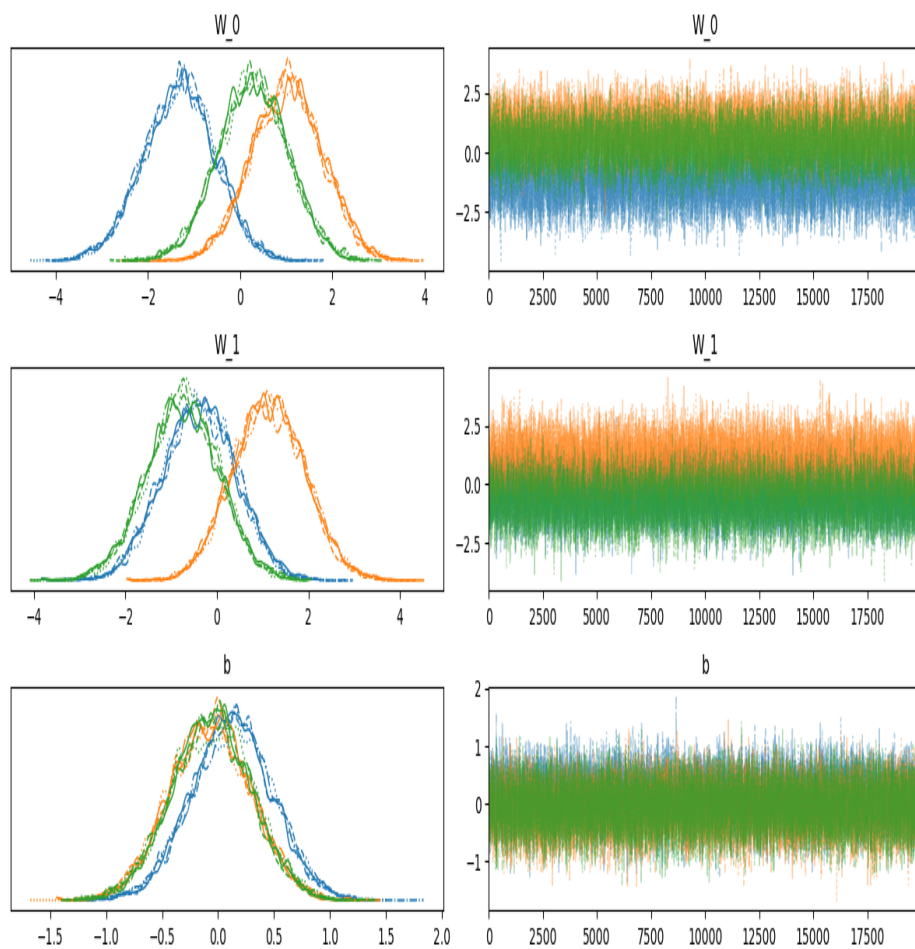
Fold 5

Figure 4.28: Fold 5, Trace Plot

4.1. RESULTS OF DIRICHLET PROCESS AND GAUSSIAN PROCESS MODELS85

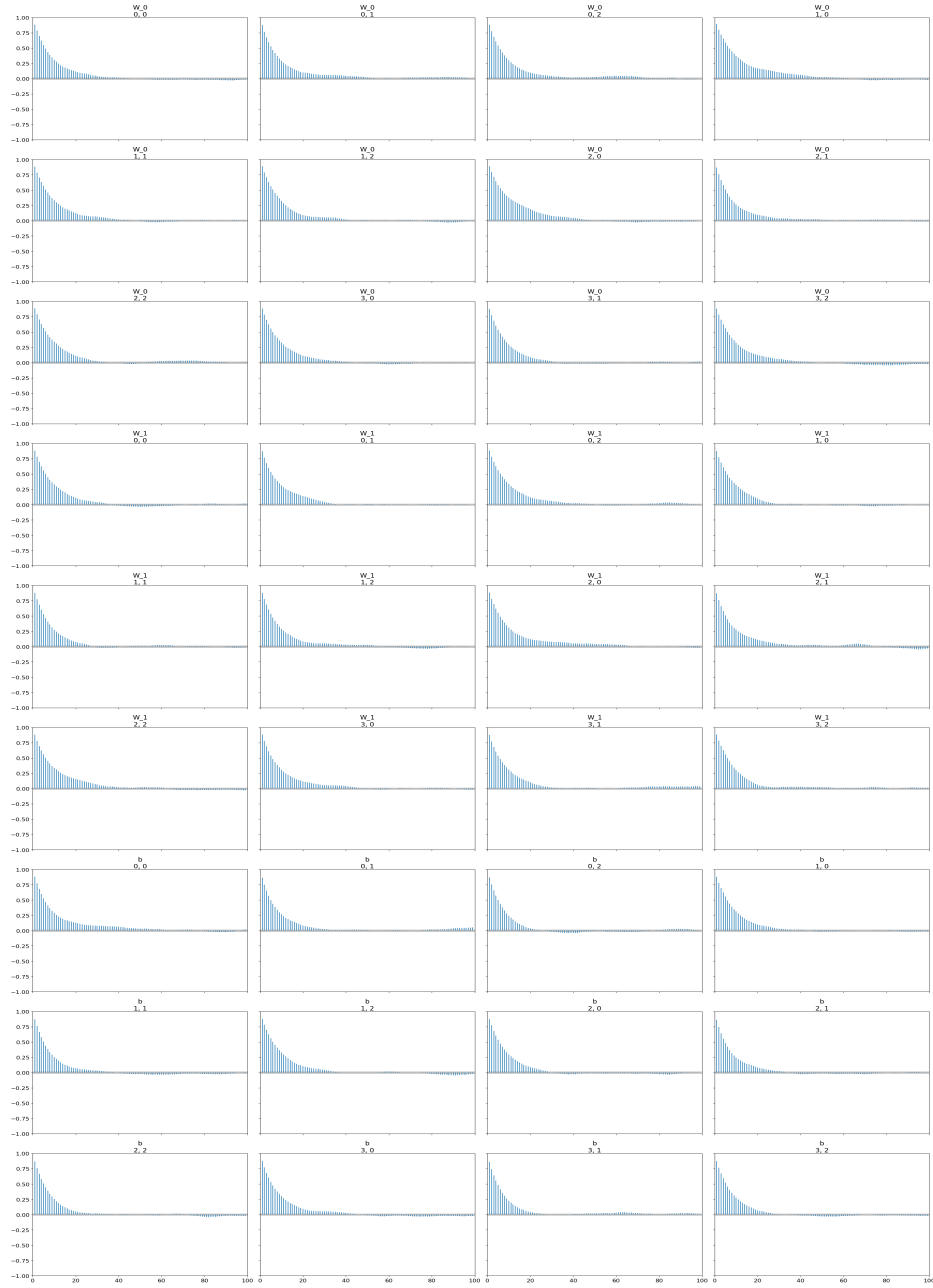


Figure 4.29: Fold 5, Autocorrelation Plot

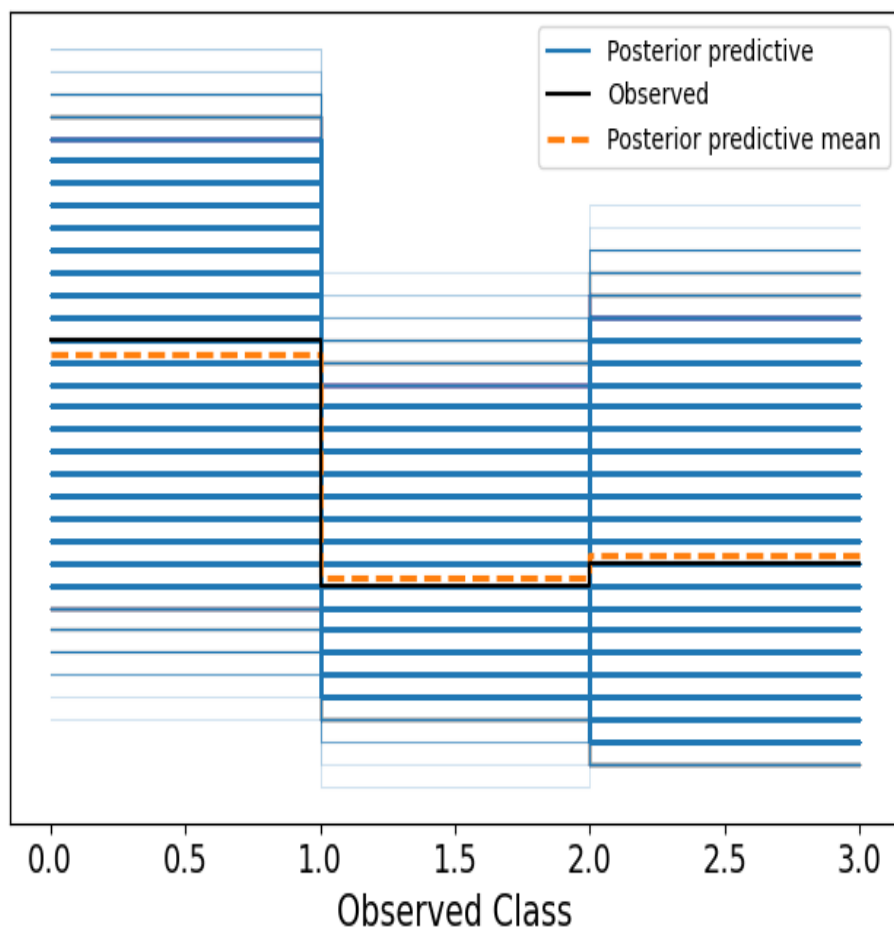


Figure 4.30: Fold 5, Posterior Predictive Check

Interpretation of Results: Gaussian Process Model

Trace Plots: These display sampled values of weight vectors W_0 , W_1 , and biases b across different dimensions of each parameter during MCMC sampling.

- W_i : For each input feature i , the weight vector across classes is modeled using a Gaussian Process prior. Each color represents one dimension (i.e., a class-specific component) of the vector.
- b : Class-specific biases. Each line represents the trajectory of one class bias term across samples.

Conclusion: The parameter traces appear well-mixed and stationary, indicating good convergence to the posterior distribution.

Autocorrelation Plots: Quick decay for the parameters implies efficient sampling and low autocorrelation.

Posterior Predictive Check: A Visual comparison of predicted versus actual class proportions shows that posterior predictive distributions align with observed frequencies, suggesting that the GP prior generalizes well despite data limitations.

4.2 IHMM Model Evaluation

We evaluate the performance of the IHMM model using various hyperparameter combinations. The results are presented in terms of accuracy, precision, and F1 score for each configuration, in 4.3. Number of MCMC iterations is set to 500 for hyperparameter tuning and 1200 for the best hyperparameters. Then, to diagnose whether the training is stable and converging as expected, we plot the log-likelihood over iterations for each class model.

α	γ	Initial State	Max States	Accuracy	F1 score	Precision
1.5	2.0	1.5	8	0.5000	0.4151	0.4804
1.8	2.0	2.0	25	0.5417	0.4408	0.4934
2.0	2.0	2.0	25	0.7500	0.6796	0.6888
2.0	3.0	2.0	10	0.6042	0.5436	0.6117
2.0	4.0	3.0	15	0.5000	0.4355	0.5097
2.0	5.0	5.0	30	0.6250	0.4909	0.4778
2.2	2.5	2.0	30	0.5417	0.4131	0.4157
3.0	2.0	3.0	15	0.5625	0.4722	0.4881
3.0	3.0	3.0	15	0.5208	0.4487	0.4733
4.0	2.0	2.0	20	0.5417	0.4098	0.3787
4.0	4.0	2.0	20	0.4583	0.3221	0.3037

Table 4.3: IHMM Evaluation Results for Various Hyperparameters

According to the results, the best-performing hyperparameter set is: $(\alpha) = 2.0$, $(\gamma) = 2.0$, (initial state) = 2.0, (max states) = 25.

Therefore, the final model is trained with the above hyperparameter set with 1200 MCMC iterations and the accuracy increased to 0.7708 ± 0.1062 . The log-likelihood over iterations for each class model for fold 1 is plotted below.

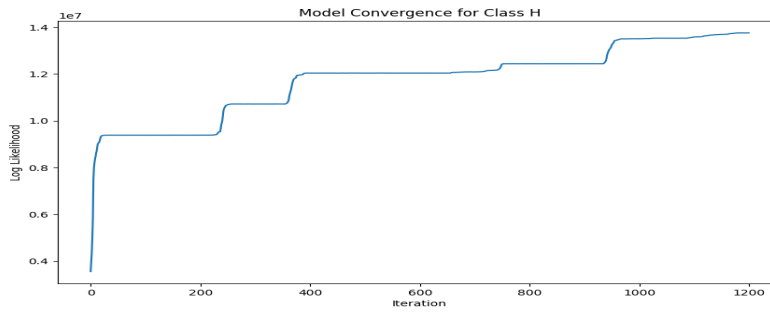


Figure 4.31: Log-likelihood Over Iterations for High Leakage Class

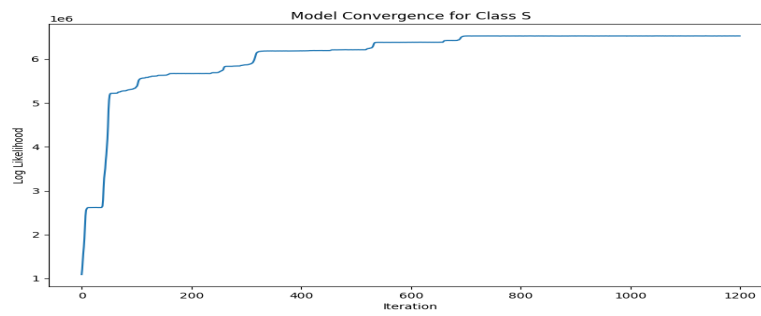


Figure 4.32: Log-likelihood Over Iterations for Small Leakage Class

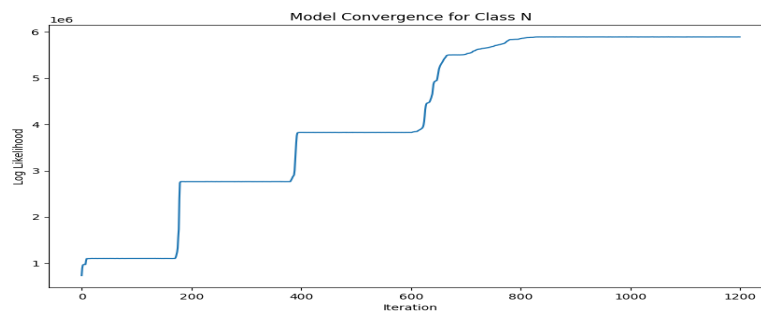


Figure 4.33: Log-likelihood Over Iterations for No Leakage Class

Interpretation of Results: IHMM Model

The convergence plots for classes H, N, and S illustrate how the HDP-HMM model improves over training iterations. For all three classes, the log-likelihood increases sharply during the initial iterations, indicating that the model quickly starts to learn useful patterns in the data. After this early phase, the rate of increase slows down, showing that the model is fine-tuning its parameters. In the plots for classes H and N, we see noticeable jumps followed by flat sections, suggesting that the model makes major updates at certain points, possibly when it changes the number of hidden states or reassigns data more effectively. These jumps are followed by periods of stability as the model settles into a better configuration. The plot for class S shows a steadier increase with smaller jumps, indicating a more gradual learning process. In all three cases, the log-likelihood levels off by around 1000 iterations, suggesting that the model has reached convergence and further training would not lead to significant improvements.

Chapter 5

Conclusion

This thesis explored nonparametric Bayesian approaches for detecting internal leakage faults in electro-hydrostatic actuators (EHA) using pressure signal data. We implemented and evaluated three models: multinomial logistic regression with Dirichlet Process (DP) and Gaussian Process (GP) priors, and the Infinite Hidden Markov Model (IHMM).

The DP and GP prior models were motivated by the limitations of fixed-structure parametric models, especially in handling multiclass classification and adapting to complex, nonlinear patterns in the data. These models were applied to extracted time-series features (specifically, the Variance Fractal Dimension of pressure signals) and trained using Metropolis-Hastings MCMC. The DP prior introduced a discrete, sparse prior over weight vectors, while the GP prior encoded smooth correlations between class-specific weights.

Despite strong convergence diagnostics and posterior predictive checks, both models

achieved a classification accuracy of approximately **71%**. This performance can be attributed to the following factors:

- Limited dataset size (48 sequences), which constrains model generalization.
- Loss of temporal information due to feature extraction, which reduces the model's ability to capture fault dynamics over time.

To address these limitations, we extended our methodology by applying the Infinite Hidden Markov Model (IHMM), which directly models the sequential structure of the raw time-series data. The IHMM's use of the Hierarchical Dirichlet Process (HDP) enabled automatic adaptation of the number of hidden states based on the data. As a result, it outperformed both previous models and achieved an improved classification accuracy of **77%**, highlighting the advantage of modeling temporal dynamics for sequence-based fault detection.

The results of this study confirm the value of nonparametric Bayesian models in fault classification tasks, particularly when the underlying data distribution is complex or when labeled data is limited. The ability of DP and GP priors to adapt model complexity is a key strength, although their performance may be limited when important structural aspects, such as temporal dependencies, are not directly modeled.

The IHMM, on the other hand, demonstrates superior predictive capability by capturing time-dependent patterns in actuator behavior. Its flexibility in discovering latent state transitions provides a powerful framework for analyzing sequential industrial signal data.

For future work, more advanced feature extraction techniques could be explored to improve the performance of the feature-based models. While Variance Fractal Dimension (VFD) offers useful information about signal complexity, alternative methods may capture richer signal characteristics relevant to fault detection. Additionally, although the Infinite Hidden Markov Model (IHMM) achieved the highest classification accuracy by leveraging the temporal structure of the raw signals, its computational cost remains a limitation. Optimizing the inference process could make the IHMM more scalable and applicable.

Another promising extension would be to place prior distributions on the concentration parameters, such as α in the Dirichlet Process model and γ in the iHMM. Treating these as random variables within a hierarchical Bayesian framework would allow the model to infer appropriate values from the data, potentially improving adaptability and reducing the need for manual hyperparameter tuning.

Bibliography

Beal, M. J. and C. E. Rasmussen (2001). The infinite hidden markov model. In *Advances in Neural Information Processing Systems*, Volume 14, pp. 577–584. MIT Press. (Cited on pages 13 and 16.)

Bolstad, W. M. and J. M. Curran (2016). *Introduction to Bayesian Statistics*. New York: John Wiley & Sons, Incorporated. (Cited on pages 2 and 20.)

Choudhuri, N., S. Ghosal, and A. Roy (2007). Nonparametric binary regression using a gaussian process prior. *Statistical Methodology* 4(2). (Cited on page 11.)

Escobar, M. D. and M. West (1995). Bayesian density estimation and inference using mixtures. *90*(430), 577–588. (Not cited.)

Ferguson, T. S. (1973). A bayesian analysis of some nonparametric problems. *The Annals of Statistics* 1(2). (Cited on page 9.)

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin (2013). *Bayesian Data Analysis*. CRC Press LLC. (Cited on pages 6, 19, 20 and 22.)

- Geman, S. and D. Geman (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-6*(6). (Cited on page 21.)
- Ghaderinezhad, F., C. Ley, and B. Serrien (2022). The wasserstein impact measure (wim): A practical tool for quantifying prior impact in bayesian statistics. *Computational statistics & data analysis* 174. (Cited on page 7.)
- Ghanbari, M., W. Kinsner, and N. Sepehri (2022). Adaptive regularization in ann for condition monitoring and fault detection in heavy-duty hydraulic machines. In *2022 IEEE 21st International Conference on Cognitive Informatics and Cognitive Computing (ICCIICC)*, pp. 175–182. IEEE. (Cited on page 34.)
- Ghosal, S. (2010). *The Dirichlet process, related priors and posterior asymptotics*, pp. 35–79. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press. (Cited on page 10.)
- Han, J., M. Kamber, and J. Pei (2011). *Data Mining: Concepts and Techniques, 3rd Edition*. Morgan Kaufmann. (Cited on page 25.)
- Hastie, T., R. Tibshirani, and J. Friedman (2009). *Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Second Edition ed.). Springer Series in Statistics. Springer. (Cited on page 2.)
- Hosseinpour, S., W. Kinsner, S. Muthukumarana, and N. Sepehri (2024). Fault detection in an electro-hydrostatic actuator using polyscale complexity measures and

- bayesian classification. *IEEE Open Journal of Instrumentation and Measurement* 3. (Cited on pages 1, 2, 7, 27, 34 and 35.)
- Hosseinpour, S., W. Kinsner, and N. Sepehri (2024). A Bayesian optimized neural network for fault detection in electro-hydrostatic actuators. *Journal of Machine Intelligence and Data Science (JMIDS)* 5(1), 144–151. (Cited on page 2.)
- James, G., D. Witten, T. Hastie, R. Tibshirani, and J. Taylor (2023). *An Introduction to Statistical Learning: With Applications in Python*. Springer Texts in Statistics. Springer International Publishing. (Cited on page 25.)
- Johnson, M. J. and A. S. Willsky (2013, February). Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research* 14, 673–701. (Cited on page 49.)
- Neal, R. M. (2000). Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics* 9(2). (Cited on page 10.)
- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12. (Cited on page 36.)
- Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2), 257–286. (Cited on page 13.)

- Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica Sinica* 4(2). (Cited on page 38.)
- Teh, Y. W. and M. I. Jordan (2010). *Hierarchical Bayesian nonparametric models with applications*, pp. 158–207. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press. (Cited on page 17.)
- Teh, Y. W., M. I. Jordan, M. J. Beal, and D. M. Blei (2006). Hierarchical dirichlet processes. *Journal of the American Statistical Association* 101(476). (Cited on page 11.)
- Walker, S. G. (2010). *Bayesian nonparametric methods: motivation and ideas*, Volume Series Number 28. Cambridge University Press. (Cited on pages 7 and 9.)
- Weng, Y., Y. Fang, H. Yan, Y. Yang, and W. Hong (2019). Bayesian non-parametric classification with tree-based feature transformation for nippv efficacy prediction in copd patients. *IEEE access* 7. (Cited on pages 2 and 8.)
- Ye, N. (2014). *Data Mining: Theories, Algorithms, and Examples*. Human factors and ergonomics series. CRC Press. (Cited on page 1.)