PARTIALLY ORDERED LOGISTIC REGRESSION

By Taryn Laird

A Thesis Submitted to the Faculty of the

GRADUATE INTERDISCIPLINARY PROGRAM

IN STATISTICS AND DATA SCIENCE

In Partial Fulfillment of the Requirements

for the Degree of

MASTER OF SCIENCE

in the Graduate College

THE UNIVERSITY OF ARIZONA

2024

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Master's Committee, we certify that we have read the thesis prepared by: Taryn Laird
titled:

and recommend that it be accepted as fulfilling the thesis requirement for the Master's Degree.

Joseph C. Watkins

Joseph Watkins Date: Aug 9, 2024

Henry Scharf (Aug 9, 2024 10:27 PDT)

Henry Scharf Date: Aug 9, 2024

Xueying Tang (Aug 9, 2024 10:28 PDT)

Xueying Tang Date: Aug 9, 2024

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to the Graduate College.

I hereby certify that I have read this thesis prepared under my direction and recommend that it be accepted as fulfilling the Master's requirement.

Joseph C. Watkins

Joseph Watkins Date: Aug 9, 2024
Thesis Committee Chair
Graduate Interdisciplinary Program in Statistics and Data Science

Table of Contents

List of Tables

List of Figures

ABSTRACT

PARTIALLY ORDERED LOGISTIC REGRESSION

TARYN LAIRD

Partially ordered sets arise frequently in classification problems. Classification models that are currently used tend to either ignore the partial orderedness of the data by fitting nominal models or apply a strict ordering to the data by fitting ordinal models. In both cases, valuable information about the data is lost or overvalued in the model. Zhang and Ip created a framework for a multistep process in which a series of models can be used to classify data from partially ordered sets while maintaining the underlying structure of the data. While the framework of the model exists, it is not widely used. In this thesis, we provide an algorithm that fits the framework along with pseudocode to assist with implementation of the model. We then show an example of an application to a rare disease, SCN8A, which has a partially ordered structure of disease state.

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Taryn Laird
author

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

SIGNED: Joseph C. Watkins
Joseph Watkins

Aug 9, 2024
DATE

Chapter 1

Preliminaries

1.1 Introduction

Partially ordered data exist in many facets of life. Consider a math exam geared towards understanding what concepts a student understands and which concepts a student needs to master still, with different questions indicating gaps in different areas. The questions and concepts are related to each other, but not in a fully ordered manner. Alternatively, consider diagnosing a disease in which there are varying levels of severity but at each level of severity there are different subclassifications that can lead to that severity. The ability to model such phenomenon while leveraging the partial order can be quite powerful.

In a paper written by Zhang and Ip in 2012 [7], a new classification model was presented and derived allowing for the modeling of partially ordered sets. The framework provided, partitioned conditioned models for partially ordered data (POS-PCM), leverages known models for both nominal and ordinal data to create a multistep process to classify data based on the underlying structure of the data. These ideas were further examined by Peyhardi [6] in a literature review, which further generalized models such as those described by Zhang and Ip.

This thesis is organized in the following way. In Section 1.2 we cover the necessary background on regression, including all the models that will be used in POS-PCM, we follow that with a discussion of the necessary background material on partially ordered sets, posets, in Section 1.3. Moving into Chapter 2, we provide an overview of the algorithm and provide pseudocode for implementing the modeling process. Finally in Chapter 3, we apply the method to a rare disease, a genetic epilepsy resulting from pathogenic mutations in the SCN8A sodium channel gene, and we conclude with a discussion of future work in Section 3.3.

1.2 Regression

Partially ordered regression requires the use of several types of classification regression. Specifically logistic regression, multinomial logistic regression and conditional logistic regression. Each of these types of regression utilizes the framework of *generalized linear models*, GLM. For a model be considered a GLM, two criteria must be met [2]. First the response variable must be from an exponential family, meaning that the probability density (distri-

bution) function can be written in the form

$$f(y \mid \boldsymbol{\theta}) = h(y)c(\boldsymbol{\theta}) \exp\left(\sum_{j=1}^{k} w_j(\boldsymbol{\theta})t_j(y)\right) \tag{1.1}$$

where $h(y) \geq 0$, and $t_1(y), \ldots, t_k(y)$ are real valued functions of the observation y with no dependence on $\boldsymbol{\theta}$. Moreover, $c(\boldsymbol{\theta}) \geq 0$ and $w_1(\boldsymbol{\theta}), \ldots, w_k(\boldsymbol{\theta})$ are real valued functions of the (possibly) vector valued $\boldsymbol{\theta}$ and can not depend on y [1]. Second, there exists a link function, g, which describes how the mean response relates to a linear function of the covariates. For the purposes of this paper, we will denote this linear function as η_i, where

$$\eta_i = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_q X_{iq}$$

and $X_{i1}, \ldots X_{iq}$ are the covariates related to response Y_i. By definition of the link function, we have

$$g^{-1}(\eta_i) = \mu_i$$

where $\mu_i = E(Y_i | X_{i1}, \ldots X_{iq})$.

Example 1.2.1 (Linear Regression) *Let Y_i for $i = 1, \ldots, n$ be independent normal random variables with covariate set $(X_{i1}, \ldots, X_{iq})$. Then*

$$Y_i = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_q X_{iq} + \epsilon_i$$

where $\epsilon_i \sim N(0, \sigma)$ for some common value of σ is a generalized linear model. To verify this we must check the two criterion are met. It is well-known that the normal distribution is an exponential family, so the first criterion is met. Next we show that the link function is the identity function. That is, $\mu_i = E(\eta_i)$. To see this, note that

$$\begin{aligned} E(\eta_i) &= E(\beta_0 + \beta_1 X_{i1} + \cdots + \beta_q X_{iq} + \epsilon_i) \\ &= E(Y_i) \\ &= \mu_i. \end{aligned}$$

Thus, the mean response μ relates to a linear function of the covariates with the identity function. Therefore, both the conditions for a GLM are satisfied and linear regression is an example of a GLM.

1.2.1 Logistic Regression

Logistic Regression is used to classify objects into one of two groups based on underlying features of the object. We will show that logistic regression fits the GLM framework and then discuss briefly the implementation in R.

Let Y_i for $i = 1, \ldots, n$ be independent $\text{Ber}(p_i)$ random variables with corresponding covariate set $(X_{i1}, \ldots, X_{iq})$. First, we will show that a Bernoulli random variable is an exponential family. Recall that the probability distribution function is written as

$$f(y \mid p) = p^y (1-p)^{(1-y)}$$

for $y \in \{0, 1\}$ and $E[Y] = p$. Then with a little bit of algebra we can rewrite this as

$$f(y \mid p) = (1-p) \exp\left(y \log\left(\frac{p}{1-p}\right)\right).$$

Here $h(y) = 1$, $c(p) = (1-p)$, and $w_1(p) = \log\left(\frac{p}{1-p}\right)$, $0 < p < 1$ and $t_1(y) = y$. Thus, we have shown that a Bernoulli random variable is an exponential family. Next we must define an appropriate link function. For logistic regression there are three canonical link functions, we will use the *logit* link defined to be

$$\eta_i = \log\left(\frac{p_i}{1-p_i}\right),$$

the log odds. Isolating p_i in the link function we obtain

$$p_i = \frac{\exp(\eta_i)}{1+\exp(\eta_i)},$$

which is a logistic function with range (0,1). Thus, we have found a link which allows the mean response, p_i to be modeled as a function of the linear function of covariates, η_i. As both criterion have been met, we know that logistic regression is a GLM.

In **R**, logistic regression can be implemented using a variety of functions and packages. For our purposes we used the **glm** function in the base distribution. This function requires the response variables, Y_i's, to be coded as 0's and 1's, while the input variables can be any mixture of categorical data coded as factors and numerical data. Typical input for the function should follow the form

```
glm(Y ~ X, family=binomial(link=``logit''), data=``dataset'').
```

After fitting the model, the **predict** method can be used to assign probability to assist with assigning labels to a combination of covariates. Typically, values that are greater than 0.5 will be assigned to group 1 and values that are less than 0.5 will be assigned to group 0. The 0.5 boundary is not firm and may be tuned to fit the data if necessary.

1.2.2 Multinomial Logistic Regression

Multinomial Logistic Regression, is an extension of logistic regression. This form of regression is used to classify objects into more than two groups where the groups are unordered. We start by showing that multinomial logistic regression fits in the GLM framework and then discuss an implementation in R.

Let Y_i for $i = 1, \ldots, n$ be independent Multi$(1, \vec{p})$ random variables and let $(X_{i1}, \ldots, X_{iq})$ be the covariate set. Here Multi$(1, \vec{p})$ represents a multinomial random variable with one draw and vector or probabilities, $\vec{p} = (p_{i1}, p_{i2}, \ldots, p_{ik}$ with k total groups. Note that Y_i can also be considered a categorical random variable. We start by showing that multinomial distributions of this form are an exponential family. Recall the probability distribution function for a multinomial distribution with one draw per group is given by

$$P(Y = y) = P(Y_1 = y_1, Y_2 = y_2, \ldots, Y_k = y_k) = p_1^{y_1} \cdots p_k^{y_k}$$

where $\sum_{j=1}^{k} y_j = 1$ and p_j represents the probability of an object belonging to group j, and $\sum_{j=1}^{k} p_j = 1$. Then we see that

$$\begin{aligned} f(y \mid p_1, \ldots, p_k) &= p_1^{y_1} \cdots p_k^{y_k} \\ &= \exp\left(\log\left(p_1^{y_1} \cdots p_k^{y_k}\right)\right) \\ &= \exp\left(\sum_{j=1}^{k} \log\left(p_j^{y_j}\right)\right) \\ &= \exp\left(\sum_{j=1}^{k} y_j \log\left(p_j\right)\right). \end{aligned}$$

Define $h(y) = 1$, $c(p_1, \ldots p_k) = 1$, $w_j(p_j) = \log(p_j)$ and $t_j(y) = y_j$. Then $f(y \mid p_1, \ldots p_k)$ can be written in the form of Equation 1.1 and the distribution functions of the multinomial random variable form an exponential family.

Next we must define a link function. Like with logistic regression there are several possible choices depending on the situation. In this case as the groups are unordered a series of baseline-category logits will suffice as our links. First recall, there are k groups into which the objects are being classified, each with corresponding probability p_j with $\sum_{j=1}^{k} p_j = 1$. This indicates that there are $k-1$ unique p_j's with the last being determined by $1 - \sum_{j=1}^{k-1} p_j$. Choose m^* to be a baseline category, that is fix m^* from the set $\{1, \ldots, k\}$. Then for $m \neq m^*$ the logit link is given by

$$\log\left(\frac{p_{im}}{p_{im^*}}\right) = \beta_{m0} + \sum_{l=1}^{q} \beta_{ml} X_{il} = \eta_{im}$$

Then isolating p_{im} we obtain

$$p_{im} = \frac{\exp(\eta_{im})}{1 + \sum_{j \neq m^*} \exp(\eta_{ij})}$$

and for $m = m^*$

$$p_{im^*} = \frac{1}{1 + \sum_{j \neq m^*} \exp(\eta_{ij})}.$$

Note that given a baseline category, these functions relate the probabilities to a linear combination of the predictor variables. Thus, the baseline-category logits satisfy the properties of being link functions. Thus, both conditions are met, and multinomial logistic regression for a single draw is a form of GLM.

To implement this in **R** we use the function `multinom` in the `nnet` package. Typical input for this function should follow the following form:

```
library(nnet)
multinom(Y~X, data=``dataset'').
```

As with logistic regression, the **predict** method can be utilized to a combination of covariates to a category. In this case, **R** will provide output that includes the category

with the largest estimated probability value. If different thresholding could be useful, the individual probabilities can be seen for each group and corresponding assignments can be made.

1.2.3 Proportional Odds Regression

Building on the ideas of logistic and multinomial regression is *proportional odds regression*, a widely utilized method for classification when the response data are ordinal. Let Y_i for $i = 1, \ldots, n$ be independent Multi$(1, \vec{p})$ random variables, $\vec{p} = (p_1, \ldots, p_k)$, in which there is a natural ordering to the k categories. As there is a natural ordering, we have $P(Y \leq m) = p_1 + p_2 + \cdots + p_m = \sum_{j=1}^{m} p_j$. Let $(X_{i1}, \ldots, X_{iq})$ be the corresponding covariate set. As with multinomial regression, we will use a series of logit links to create the regression model. In this case, we will use a cumulative logit. That is,

$$\log\left(\frac{P(Y \leq m)}{P(Y > m)}\right) = \log\left(\frac{P(Y \leq m)}{1 - P(Y \leq m)}\right) = \log\left(\frac{p_1 + p_2 + \cdots + p_m}{p_{m+1} + p_{m+2} + \cdots p_k}\right)$$

where k is the total number of groups for the variable. In order to have the links satisfy the property that the mean response relates to a linear function of the covariates, one sufficient condition is that the slopes for each group are the same, that is the way in which a covariate interacts with the model is the same in each group, while the intercepts differ. Mathematically we can think of this as parallel hyperplanes. With the parallel slopes assumption, we have satisfied both the conditions and thus proportional odds regression satisfies the GLM framework.

To implement this in **R**, we use the `polr` function in the `MASS` package. Typical input for this function will follow the form:

```
polr(Y~X, data=``dataset'')
```

It should be noted that this particular function requires that the response variable be encoded as a factor. In addition, it requires more than 2 groups as otherwise ordering is not relevant.

1.2.4 Conditional Logistic Regression

In the case of classifying two groups with a structure, proportional odds regression is not possible to use as the idea of $P(Y \leq m)$ is not relevant and simplifies to an unordered case. Because of this, an alternative ordered regression scheme must be utilized. For the purposes of our discussion, we will use *conditional logistic regression*. Conditional logistic regression is utilized when observations can be matched to a control group or a different group in some way. Call these matches or groups *strata*. Consider a dataset with S strata. Let Y_i for $i = 1, \ldots, n$ be independent Ber$(1, p_i)$ random variables and with corresponding covariate set $(X_{i1}, \ldots, X_{iq})$. Then building on the ideas of logistic regression from Section 1.2.1 the model can be written as

$$logit(p_i) = \alpha_1 + \alpha_2 z_{i2} + \cdots + \alpha_S z_{iS} + \beta_0 + \beta_1 X_{i1} + \cdots + \beta_q X_{iq}$$

where the z_{ij}'s are indicator variables for the strata, the α's are the regression coefficients for the stratum and the β's are the regression coefficients for the covariates. Note that for each observation Y_i, $z_{ij} = 0$ for all $j \neq s$ where s represents the strata that the observation falls into. Thus, the logit links are parallel hyperplanes that differ in the intercept term for the linear model, notably the intercept is given by $\alpha_s + \beta_0$. Notice that the logit link is used which gives

$$p_i = \frac{\exp(\alpha_1 + \alpha_2 z_{i2} + \cdots + \alpha_S z_{iS} + \beta_0 + \beta_1 x_{i1} + \cdots + \beta_q x_{iq})}{1 + \exp(\alpha_1 + \alpha_2 z_{i2} + \cdots + \alpha_S z_{iS} + \beta_0 + \beta_1 x_{i1} + \cdots + \beta_q x_{iq})}$$

.

In **R**, we use the `clogit` in the `survival` package. Typical input for this function will follow the form:

```
library(survival)
polr(Y~X+strata(variable), data= ''dataset'')
```

It should be noted, that this does require a strata variable. If the data do not naturally have a matching or stratification to it, there must be one created to include in the model via random assignment or some other method.

1.3 Partially Ordered Set

A *partially ordered set*, or poset, is a set, P, and a relation, $\preceq$, that satisfies the following properties:

- Reflexive: If $a \in P$, then $a \preceq a$.
- Antisymmetric: If $a, b \in P$ such that $a \preceq b$ and $b \preceq a$, then $a = b$.
- Transitive: If $a, b, c \in P$ such that $a \preceq b$ and $b \preceq c$, then $a \preceq c$.

Example 1.3.1 *Consider the set of natural numbers* $\mathbb{N}$ *with the relation* $\leq$. *Then* $(\mathbb{N}, \leq)$ *is a poset.*
Proof: We show that $(\mathbb{N}, \leq)$ *satisfies the properties described above. First, let* $a \in \mathbb{N}$. *Since* $a = a$, *we have* $a \leq a$ *and thus* $a \preceq a$. *Next let* $a, b \in \mathbb{N}$ *such that* $a \preceq b$ *and* $b \preceq a$. *Then* $a \leq b$ *and* $b \leq a$, *by the properties of the natural numbers, it must be that* $a = b$. *Finally, let* $a, b, c \in \mathbb{N}$ *such that* $a \preceq b$ *and* $b \preceq c$. *Then* $a \leq b$ *and* $b \leq c$, *which implies* $a \leq b \leq c$, *so* $a \leq c$ *and* $a \preceq c$. *Therefore, all three properties are satisfied and* $(\mathbb{N}, \leq)$ *is a poset.*

For larger posets it can be useful to visualize the elements of the set and the corresponding relations. A *Hasse diagram* is one such graphical representation of a poset. The vertices in the diagram represent elements in the set, and the edges represent the relation between elements. Typically, Hasse diagrams have an upward orientation, meaning that if $p \preceq q$ then p is lower in the diagram than q. Note that all Hasse diagrams are trees.

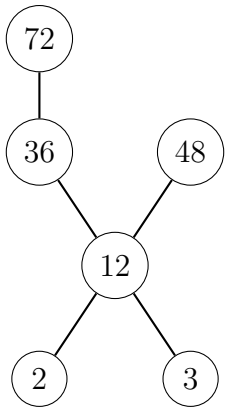

Figure 1.1: A Hasse diagram of the poset described in Example 1.3.2

Example 1.3.2 *Consider the set* $\{2, 3, 12, 36, 48, 72\}$ *with relation* $\preceq$ *defined to be division. That is* $a \preceq b$ *if* a *divides* b*. Then a possible corresponding Hasse diagram is seen in Figure 1.1.*

From the diagram, we are able to visualize the relations between the elements of the set. For example, we see that $2 \preceq 12$ *as* 12 *is divisible by* 2 *and* $12 \preceq 48$ *as* 48 *is divisible by* 12 *and by* 2 *and* 3*. We also see there is no relationship between* 2 *and* 3*.*

Consider a poset $(P, \preceq)$ and let $a, b \in P$. If $a \preceq b$, we say b *dominates* a. Moreover, a and b are *comparable*, if either $a \preceq b$ or $b \preceq a$, otherwise a and b are *incomparable*.

Example 1.3.3 *Consider the poset in Example 1.3.2. Since* $2 \preceq 12$*, we say that 12 dominates 2, and 2 and 12 are comparable. Since* 2 *and* 3 *have no relation between them, we say that* 2 *and* 3 *are incomparable.*

A poset is *finite* if the set P contains a finite number of elements, otherwise it is *infinite*. For our purposes we will restrict our discussion to finite posets. Moreover, an element $a \in P$ is *maximal* (*minimal*) if there is no element $b \in P$ such that $a \preceq b$ (respectively, $b \preceq a$), that is a has no elements that dominate it (respectively, there are no elements that a dominates).

Proposition 1.3.4 *Let* $(P, \preceq)$ *be a finite poset. Then* P *contains at least one maximal element and at least one minimal element.*

Example 1.3.5 *Consider the poset given by the Hasse diagram in Figure 1.1. As there are a finite number of elements in the set, this is a finite poset. A maximal element is* 72 *as there are no elements in the set that are divisible by* 72*. The minimal elements are* 2 *and* 3 *as there are no elements in the set that divide* 2 *and there are no elements in the set that divide* 3*.*

Let $(P, \preceq)$ be a poset. A *chain*, C, is a totally ordered subset C of P, where totally ordered means that all elements in C are comparable. An *antichain*, A, is a set of pairwise incomparable elements of P. We say that C (or A) are maximal if no other chain (or respectively antichain) covers it. In finite posets, chains and antichains can contain at most

one element in common. The *height* of P, $h(P)$ is the largest cardinality of a chain in the collection of chains in P, and the *width* of P, $w(P)$ is the largest cardinality of an antichain in the collection of antichains in P. It should be noted that, chains, antichains, height and width are easily identifiable through a Hasse diagram representation of the poset.

Example 1.3.6 *Consider the poset given in Figure 1.1.*

Then $C = \{2, 12, 36, 72\}$ *is an example of a chain, as all elements are comparable. Alternatively* $A = \{2, 3\}$ *is an example of an antichain as the two elements are incomparable. Moreover, we have* $h(P) = 4$ *as* C *is a maximal chain in the poset. Finally,* $w(P) = 2$ *as the maximal antichains in the set all have cardinality 2.*

Let $(P, \preceq)$ be a finite poset and let S_1 and S_2 be subsets of P. Then if at least one element in S_2 is dominated by elements in S_1 and no element in S_2 dominates any element in S_1, then S_1 and S_2 are *weakly ordered* and S_1 *weakly dominates* S_2. If every element in S_1 dominates all elements in S_2, then S_1 and S_2 are *strongly ordered* and S_1 strongly dominates S_2. A set of subsets is *totally weakly (strongly) ordered* if pairwise subsets are weakly (strongly) ordered.

Example 1.3.7 *Consider the poset given in Example 1.3.2. Let* $S_1 = \{48, 72\}$ *and* $S_2 = \{2, 36\}$*. Then* S_1 *and* S_2 *are weakly ordered as all elements in* S_2 *are dominated by* 72*. However, they are not strongly ordered as* 48 *does not dominate* 36 *which is an element in* S_2*.*

Finally, consider a set of subsets $\mathscr{D} = \{D_j : j = 1, \dots k\}$ of a poset $(P, \preceq)$. Then $\mathscr{D}$ is a *partition* of P if

(1) $\bigcup_{j=1}^{k} D_j = P$

(2) $D_u \cap D_v = \emptyset$ for all $u \neq v$.

Moreover, if $\mathscr{D}$ is totally weakly ordered, we call $\mathscr{D}$ an ordered partition.

Property 1.3.8 (Zhang and Ip, 2012 [7]) *A finite poset can always be partitioned into antichains that are totally weakly ordered.*

Example 1.3.9 *Consider the poset given in Example 1.3.2, with the Hasse diagram seen in Figure 1.1. Then* $\mathscr{D} = \{\{2, 3\}, \{12\}, \{36\}, \{48, 72\}\}$ *is a partition of totally weakly ordered antichains.*

Proof: Consider $\mathscr{D}$ *as described above. Note that for each* $D \in \mathscr{D}$ *for which* $|D| > 1$*, in the Hasse diagram, there is no single edge connecting the vertices, which indicates that there is no relation between the elements. Thus, each* D *is a set of pairwise incomparable elements, and therefore a collection of antichains. Next note that each* $D \in \mathscr{D}$ *comes from a different level in the Hasse diagram, with at least one element in each set having a relation to the elements in the level above. Since the Hasse diagram is oriented, we know that elements that come from lower levels are dominated by elements in levels above them. Thus, the sets in* $\mathscr{D}$ *are totally weakly ordered as pairwise subsets are weakly ordered. Finally, note that*

$$\bigcup_{D \in \mathscr{D}} D = P.$$

and clearly the elements of $\mathscr{D}$ are pairwise disjoint upon inspection, indicating that $\mathscr{D}$ is a partition of P. Therefore, $\mathscr{D}$ is a partition of totally weakly ordered antichains.

It should be noted that Property 1.3.8 assumes that the poset has one connected component, meaning that there is at least one relation between every node in the poset. In terms of the Hasse diagram this is seen when diagram only has a singular component. The poset shown in Figure 1.1 used throughout this section has one connected component. However, not all posets have a singular connected component. In this case, each component can be partitioned into a collection of totally weakly ordered antichains. The entire collection of antichains will not be totally weakly ordered. In Figure 1.2 we see an example of a poset with two components.

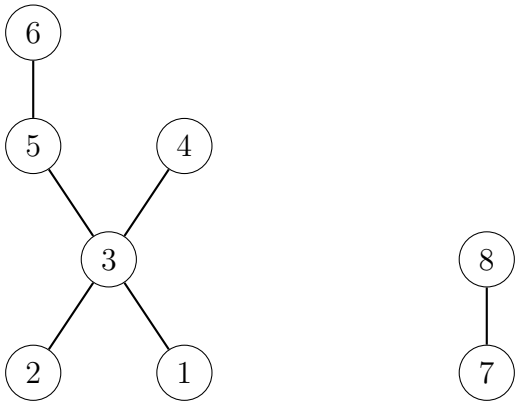

Figure 1.2: A poset with two connected components

Chapter 2

Partition Conditioned Models for Poset Responses

2.1 Introduction

Utilizing the ideas of partially ordered sets and the various types of regression, we are able to create the framework of partition conditioned models for poset responses (POS-PCM) originally introduced by Zhang and Ip [7]. In their paper [7], the authors introduced the concept of POS-PCM by describing the modeling procedure using an example. This thesis expands upon their work by providing an algorithmic approach that can be implemented for any dataset with a poset structure for the dependent variable. Moreover, we include pseudocode which can be adapted to a user's coding language of preference, allowing the user to more quickly interact with the model.

Consider a dataset with a partially ordered set of outcomes we wish to model using POS-PCM. Suppose the poset is one connected component. To implement the POS-PCM process, complete the following steps:

- Using Property 1.3.8, partition the poset into a collection of totally weakly ordered antichains.
- Using ordinal methods - proportional odds regression (more than two groups) or conditional logistic regression (two groups), assign the data to a specific antichain. Consider this Regression Step 1.
- Using use nominal methods - multinomial logistic regression (more than two groups) or logistic regression (two groups), to assign the data to the appropriate group within the antichains. Call this Regression Step 2.

It should be noted, that Regression Step 2 will need to be repeated for every antichain that has more than one element. The choice of ordinal models for Regression Step 1 is driven by the orderedness of the data within the poset itself. We can think of each level of the Hasse diagram as inducing an order on the data so it naturally follows that an ordinal method should be used. However, within each antichain, there is by definition no relation between the values indicating there is no natural ordering between the groups and nominal regression makes sense for Regression Step 2.

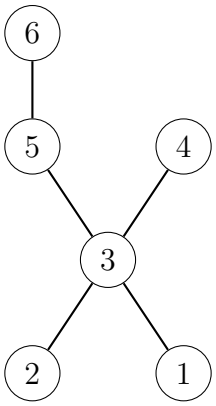

Figure 2.1: A poset with one connected component

Example 2.1.1 *Consider a poset with the Hasse diagram seen in Figure 2.1 representing the relations between the different categories. As there is one connected component in the Hasse Diagram we start by partitioning the poset into a collection of antichains using Property 1.3.8. From Example 1.3.9, we know that the poset can be partitioned into* $\mathscr{D} = \{\{1,2\},\{3\},\{5\},\{4,6\}\}$. *Next we would implement the Regression steps. Under the described methods, we would utilize a proportional odds regression model to first classify the data into one of the four antichains, completing Regression Step 1. Having done this we would complete two iterations of Regression Step 2. We would utilize logistic regression to classify the data placed into sets* $D_1 = \{1,2\}$ *and* $D_4 = \{4,6\}$ *respectively. A visual representation of this can be seen in Figure 2.2.*

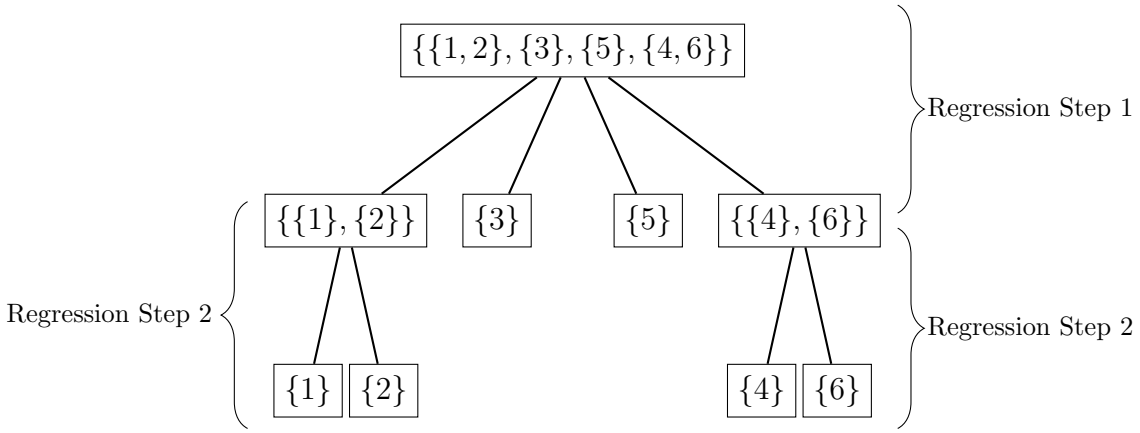

Figure 2.2: A tree representation of the POS-PCM process for a single component

In the case that the poset is more than one component, there are two additional steps that must be taken, one during the initial data organization phase and one in the regression phase. During the initial data organization, prior to partitioning the poset using Property 1.3.8, we first must partition the poset by distinct components. After doing that, each component will be partitioned using Property 1.3.8. Because of the additional partitioning that must occur, an additional step in the regression process is required. This additional

step in the regression process occurs before Regression Step 1, and uses nominal procedures to assign the data into the proper component of the poset. Consider this Regression Step 0. After Regression Step 0, Regression Step 1 and Regression Step 2 will occur separately within each of the components of the poset.

Example 2.1.2 *Now consider a poset with multiple components like that seen in Figure 1.2. As this poset has two components we first must partition the poset by its distinct components, in this case we have* $\mathscr{A} = \{1, 2, 3, 4, 5, 6\}$ *and* $\mathscr{B} = \{7, 8\}$*. For* $\mathscr{A}$*, the partition into antichains is seen in Example 2.1.1. For* $\mathscr{B}$ *we have the partition* $\mathscr{B}_1 = \{\{7\}, \{8\}\}$*. Using the described methods we would first create a model using a nominal method to label observations as group* $\mathscr{A}$ *or group* $\mathscr{B}$*. Once data are labeled using Regression Step 0, if the label is group* $\mathscr{A}$ *we would use a similar method as described in the regression steps from Example 2.1.1. If the label is group* $\mathscr{B}$*, we would utilize Regression Step 1 via conditional logistic regression to label the data as either group 7 or group 8. Visually this can be seen in Figure 2.3.*

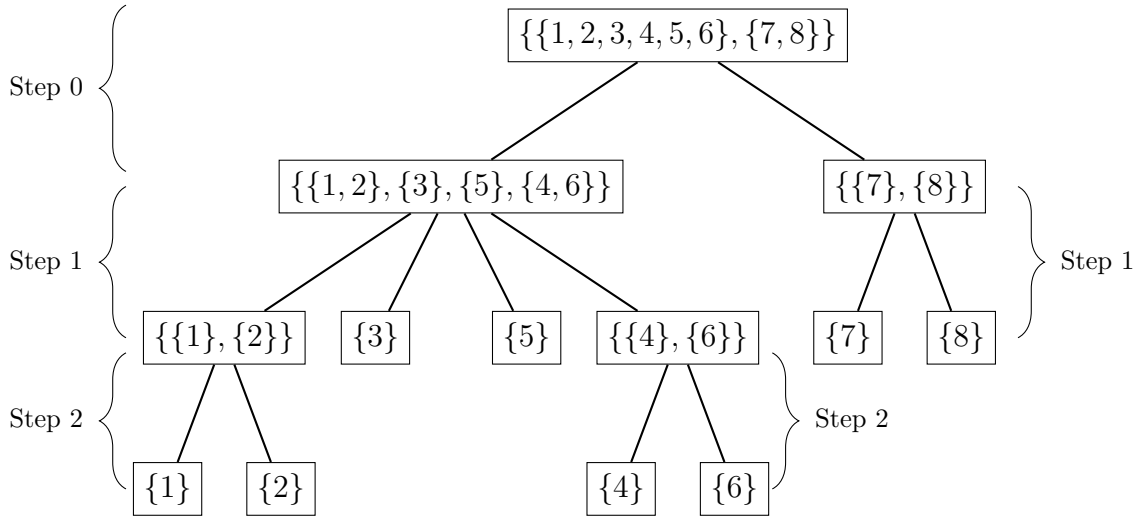

Figure 2.3: A tree representation of the POS-PCM process for multiple components

2.2 Pseudo-Code

Based on the algorithm presented above, we can now consider the methods of implementation. In this section, we will present our novel generalized approach that could be used with any software program. In Chapter 3 we will provide a specific implementation of the algorithm.

In the initial steps of the algorithm, the data must be partitioned into totally weakly ordered antichains using Property 1.3.8 and the natural partitions created by disjoint components of the posets. This leads to the first step in the process, data labeling. In the latter steps of the algorithm, the regression is performed iteratively.

2.2.1 Data Labeling Steps

For the first step of the process, we proceed with adding three alternative labels, necessary for each step of the regression process. The goal of this process it to work through the data adding a column to the dataset in such a way that each regression step can be implemented without further data manipulation. The first step described below can be skipped if the dataset contains only one connected component.

Labeling Step 0 - Required only for datasets with more than one component

For this step we add a column to the data that categorizes the data into the appropriate group for the first level of the tree. As we are working under the assumption that the underlying poset has more than one component, this column would categorize the data into its respective component. This can be thought of as the categorization provided for Regression Step 0, described above. Algorithm 1 describes an overall flow to the work. Note, we assume the poset has L components.

Algorithm 1 Data Labeling Based on Partition

```
for each index of string do
    if category is component 1 then
        New Variable is component 1
    ⋮                                   ▷ Extend else for each component
    else category is component L
        New Variable is component L
    end if
end for
```

Example 2.2.1 *Consider the poset provided in Figure 1.2 which contains two connected components. To implement Algorithm 1, we would create a new variable in our dataset that contains two values as factors, without loss of generality call them A and B. The data that gets label A are the data that are categorized as $1, 2, 3, 4, 5, 6$, and the data that get label B are the data that are categorized as $7, 8$.*

Labeling Step 1

In this step we continue our adding columns to our data set by adding a column to represent the data label based on the partition of the poset. Note that in the case of a poset with a single connected component, this is the first step of data labeling that would take place. For the case of a poset with more than one component, we will repeat this step for each component.

Example 2.2.2 *Revisiting the poset provided in Example 2.1.2 in this step we would be completing the labeling for Regression Step 1. We would create a new variable in our dataset that contains 6 new labels $A1$, $A2$, $A3$, $A4$, $B1$, and $B2$, with the knowledge that in later steps we would consider $A1$, $A2$, $A3$ and $A4$ as a connected component and $B1$ and $B2$ as another connected component. The labeling would happen as follows:*

Algorithm 2 Data Labeling Based on Partition

```
for each index of string do
    if category is new group 1 then
        New Variable is new group 1
    ⋮                                          ▷ Extend Else for each antichain
    else category is new group M
        New Variable is new group M
    end if
end for
```

- $A1$ *for groups* 1 *and* 2
- $A2$ *for group* 3
- $A3$ *for group* 5
- $A4$ *for groups* 4 *and* 6
- $B1$ *for group* 7
- $B2$ *for group* 8

Labeling Step 2

In the final step of the data labeling process, we split any remaining non-singleton antichains into their corresponding labels. It follows the same procedure as seen in Algorithm 2 with the labeling happening for Regression Step 2 for the sets of the antichains that contain more than one element. All elements that have been classified into a group as a singleton will be left with a blank as they will not be used in this particular classification step in the later procedures.

Example 2.2.3 *Continuing with our relabeling of the poset from Example 2.1.2 we would need to further label data that is in the set labeled as* $A1$ *and* $A4$ *in the previous step. Here we would create four new labels* $A11$, $A12$, $A41$ *and* $A42$. *The labeling would be completed in the following way:*

- $A11$ *for group* 1
- $A12$ *for group* 2
- $A41$ *for group* 4
- $A42$ *for group* 6

2.2.2 Regression Steps

Since the data have been appropriately labeled for each step of the regression process, we now move into a discussion regarding the regression steps. We start with Step 2.1 which need only be considered if the underlying poset structure has multiple components. In the case of a poset with a single component, the regression step process begins in Step 2.2.

Regression Step 0 - For posets with multiple connected components

In this step, we begin the regression process by first assigning the data into the components of the poset using nominal regression methods as described in Section 2.1. Without loss of generality, assume the variable created in Algorithm 1 is named Y_1.

Algorithm 3 Nominal Regression to Classify into Components

Regress Y_1 against the predictors $\vec{X}$
Store the model as to use for future predictions

Example 2.2.4 *Continuing with Example 2.1.2, the poset has 2 components - labeled in Example 2.2.1 as A and B. As there are only two connected components, utilize logistic regression to create a model that will assign the data into one of the two categories.*

Regression Step 1

For this step will complete Regression Step 1 as described in Section 2.1. In the case of a poset that contains a single connected component, this will be the first step in the regression process. For this step we utilize an ordinal regression method to assign the data to the appropriate antichain. Note that for a poset with multiple connected components this process will need to be repeated for each component. Without loss of generality, assume the variable created in Algorithm 2 is called Y_2. Pseudocode for this step can be seen in Algorithm 4.

Algorithm 4 Ordinal Regression to Classify into Ordered Components

For each component complete the following steps
Regress Y_2 against the predictors $\vec{X}$
Store the models to be used later in classification

Example 2.2.5 *In this step, we will train the models that will assign observations to the appropriate antichains within their respective components, as defined in Example 2.2.2. To do so, we will complete the steps outlined in Algorithm 4, two times, once for the component labeled as A and once for the component labeled as B. For component A we will utilize proportional odds regression, or something similar, as there are 4 labels (A1, A2, A3, A4) that can be applied to the data within this subset. After the assignment for component A is completed, we move onto the assignment for component B. Since B contains only two labels B1 and B2, we must use conditional logistic regression or something similar. This may require an additional column being added to the dataset to emulate a matched pairs situation.*

This can be done randomly by assigning indices to the data. After completing the regression, component B has now been fully labeled and trained in the model, and will no longer need to be considered. Component A contains a couple of labels, A1 and A4 that contain multiple values which still need to be handled.

Regression Step 2

For any antichain that contains more than one element, we now will train a model to assign observations to the appropriate group. This will be done using nominal regression techniques, as by definition, antichains contain no relations between objects within them. To do this, we would complete the following process. Without loss of generality, assume the variable created in the dataset in Algorithm 2 for the third level of classification is labeled Y_3.

Algorithm 5 Nominal Regression to Classify into label for antichains

For each antichain with more than one element complete the following
Regress Y_3 against the predictors $\vec{X}$
Store the models to use for classification

Example 2.2.6 *Continuing with the example from this section, in Example 2.2.4 and Example 2.2.5 assigned the data into individual components using Algorithm 3 and further assigned data into individual antichains using Algorithm 4. At this point, there are two labels that need further assignment, namely labels A1 and A4. All other antichains, contain a single element and do not need further labeling. As there are two sets, we will complete two iterations of Algorithm 5. In both instances, as A1 and A4 contain two elements each, and we would utilize logistic regression to create the models. Having completed this step, the data have been fully grouped into individual labels.*

2.2.3 Model Prediction

Having completed the steps outlined in Algorithms 3, 4, and 5 a collection of models have been created and trained. At this time, it may be helpful to go back and look at performance of the model to potentially tune parameters and cutoffs for the prediction probabilities. Once the tuning is completed, the model can now be utilized to predict categories for new data. In order to utilize the model for prediction it can be useful to write a function, as the multiple steps for regression requires that all future data be run through multiple models. Having a function to complete this reduces the number of steps a user must take. It should be noted that as the data in this step are not being used to train the model, the data labeling steps outlined in Section 1.2 are not necessary. The following is an outline of a potential function that could be used for a poset with multiple components.

For a poset with a single component the algorithm can be simplified slightly as there is one less layer to the regression step. A function in this case to create predictions with data is shown in Algorithm 7.

Algorithm 6 Function to classify data from poset with multiple components

Function to predict a category using the POS-PCM
Pass in: the predictors $\vec{X}$
for each index of string **do**
  Predict the outcome using Algorithm 3
  if Predicted Value is in component 1 **then** ▷ Extend Else for each component
    Predict the outcome using the appropriate function from Algorithm 4
    if Predicted value is in an antichain with multiple elements **then**
      Predict the outcome using the appropriate function from Algorithm 5
    else Store the outcome for return
    end if
  else Predict the outcome using the appropriate function from Algorithm 4
    if Predicted value is in an antichain with multiple elements **then**
      Predict the outcome using the appropriate function from Algorithm 5
    else Store the outcome for return
    end if
  end if
end for
return Predicted Values

Algorithm 7 Function to classify data from poset with single component

Function to predict a category using the POS-PCM
Pass in: the predictors $\vec{X}$
for each index of string **do**
  Predict the outcome using Algorithm 4
  if Predicted value is in an antichain with multiple elements **then**
    Predict the outcome using the appropriate function from Algorithm 5
  else Store the outcome for return
  end if
end for
return Predicted Values

Chapter 3

Applications to Rare Disease Classification in SCN8A

3.1 An Overview of SCN8A

SCN8A is a gene that encodes a sodium channel in the brain, $Na_v1.6$. Genetic mutations of SCN8A can cause epileptic and developmental encephalopathy (DEE) [5] in individuals. Mutations of the SCN8A gene can cause gain of function (GOF) or loss of function (LOF) characteristics in the sodium channel within the patient. Progression, medication response and treatment of the disease vary widely based on the specific mutation and diagnosis of the patient. The ability to predict a patient's particular subgroup prior to a full genetic panel being completed could help physicians attend to a patient's needs more precisely early on in diagnosis.

In order to be able to complete the classification into appropriate subgroups, we first discuss a more refined scale of both the LOF and GOF categorizations. For mutations that cause LOF characteristics, currently there are two subgroups that patients can be categorized into LOF-, meaning the patient does not have seizures, or LOF+, meaning the patient does have seizures. For the GOF characteristics, traditionally patients were classified along a scale with mild, moderate and severe categories depending on the severity of the presentation of the developmental and epileptic encephalopathy. In (Hack 2024) [4], machine learning methods were used to complete supervised and unsupervised learning techniques applied to patient registry data allowing for a comparison of groupings of patients with a GOF variant. The supervised approach used the traditional mild, moderate, severe scale while the unsupervised approached created clusters via analysis of the data and not a predetermined classification scheme. The results from the unsupervised approach showed that the optimal division of the GOF groups is still three. However, these groups do not follow the mild, moderate, severe pattern. Instead, the patients were classified into groups entitled developmental encephalopathy (DE), epileptic encephalopathy (EE), and epileptic and developmental encephalopathy (DEE). Moreover, it was shown that this classification ultimately led to a more precise prediction of effective treatment than the traditional classification scheme.

Given the categorization from Hack, 2024 [4] and the LOF classifications, a partial order can be applied to the groups, and can be visualized using a Hasse diagram seen in Figure 3.1.

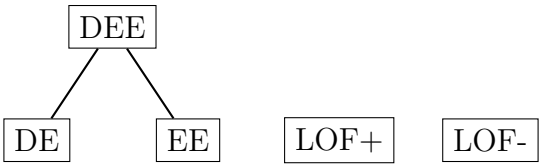

Figure 3.1: Hasse diagram representation of the SCN8A poset

A partial order exists between the DE, EE and DEE groups under the larger GOF category. This partial ordering comes from disease progression in patients. Typically, patients in the DE group, experience the onset of development encephalopathy prior to experiencing seizures. Patients in the EE group, experience the onset of epileptic encephalopathy prior to experiencing developmental delays. Patients in the DEE group experience developmental and epileptic encephalopathy onset around the same time. At this time, there is no partial ordering that exists in the LOF patients, and they can be seen as separate components in the poset. Given the poset structure of the categories, the methods discussed in Chapter 2 can be utilized to create a model in which clinical data of patients can be utilized to predict disease group and provide insight to treatments prior to knowing the patient's specific genetic mutation.

3.2 Application of Regression Methods to SCN8A

Based on the Hasse diagram of the poset for SCN8A seen in Figure 3.1, there are three distinct components for the underlying structure. As there is more than one component to the poset, to complete the regression, there will be three columns added to the dataset to aid with the regression process. These columns will take the following form:

- Step 1 Column
  - Group A: corresponds to DE, EE, and DEE labels, the GOF component
  - Group B: corresponds to LOF+
  - Group C: corresponds to LOF-
- Step 2 Column
  - Group A1: corresponds to DE and EE label
  - Group A2: corresponds to DEE label
- Step 3 Column
  - Group A1a: corresponds to DE label
  - Group A1b: corresponds to EE label

Before completing the regression portion of the algorithm, there is one more column that will be added to the dataset for use in Algorithm 4, which corresponds to Step 2 in the regression process. For this particular example, the ordinal step distinguishing between

Group A1 and Group A2 will be completed using conditional logistic regression. This requires the data to have pairs between groups, which ultimately will lead to a reduction of our data in this particular step. The column that will be added will be titled ID and will sequentially number the data points in the group with more entries, in this case Group A2, starting with 1 and ending with, N, the last entry. For the values in Group A1, we will randomly assign a value from the set $\{1, \ldots, N\}$ without replacement. It should be noted if the group sizes are reversed (more data points in Group A1 than in Group A2), the previous scheme to add an ID label will be reversed. Data that does not correspond to Group A1 and Group A2 does not need the additional label and should be left blank.

Now that the data labeling steps are complete the regression steps can occur. In the first step of the regression process following the procedures from Algorithm 3, multinomial logistic regression will be utilized to train the model to assign the data into Groups A, B and C. After completing this step, the data that corresponds to Group B or Group C have been fully assigned by the model and will be dropped from the data set. The next step of the regression process following the procedures from Algorithm 4 can now be completed. In this case, conditional logistic regression will be used to distinguish observations between Groups A1 and A2. At the end of this step, data that corresponds to Group A2 have been fully assigned and may be dropped from the data set for the next step. The final step of the regression process utilizes the procedures laid out in Algorithm 5. For this step, logistic regression will be utilized to train a model to assign data into Groups A1a and A1b. Once this step is completed, the model has been fully trained and can be utilized to make predictions for data.

3.2.1 Application Using Simulated Data

Given the general framework provided in Section 3.2, the method can now be implemented. Because SCN8A is a rare genetic mutation, there are not a lot of data available to work with. As a result the model was first fit to simulated data. This allowed for repeated trials and parameter tuning without fitting a model that only worked given a specific set of clinical data. In the discussion presented here, the simulated data come from summary statistics provided in Hack, 2023 [3] and from summary statistics gained from patient data used in Section 3.2.2. It should be noted that the simulated data discussed here differs from traditional simulation data. Traditionally, β coefficients are chosen based on the understanding of the structure and then the linear functions and link functions are used to create corresponding outputs based on plausible $\vec{X}$ values. However, in this case, we assign a label (DE, EE, DEE, LOF+, LOF-) based on simulated $\vec{X}$ values and knowledge of the disease. We do not consider the parameters and create link functions for assignment. More discussion about the creation of the variables follows below the description of the variables used in the model. Additionally, the number of observations per group, we will call it m, is the same. In actuality this is not the case and the groups are much more imbalanced. The variables created in the simulated dataset are as follows:

- A factor indicating whether seizures are present or not. (1 - yes, 0 - no)
- A factor indicating whether a patient has a seizure type of "motor/focal" or not (1 - yes, 0 - no)

- A factor indicating whether a patient has a seizure type of absence or not (1 - yes, 0 - no)
- Age at seizure onset (simulated as a normal random variable with mean and sd extrapolated from summary statistics)
- A factor indicating if severe developmental delay is present (1 - yes, 0 - no)

To provide an example of how the simulated data was created, consider the patients in the LOF+ group. A typical patient in the LOF+ category has seizures, with about 75% of patients experiencing "motor/focal" type seizures, 50% of patients experiencing "absence" type seizures, an average age of seizure onset of 40 months with a standard deviation of 15 months, and approximately 80% of patients experience severe developmental delay. To simulate the data for this group, we would generate m observations for each variable based on the information we have. In **R**, this would look like the following.

- seizures <−**rep**(1,m)
- mf <−**sample**(**c**(0,1), m, prob=**c**(0.75, 0.25),**replace**=TRUE)
- absence <−**sample**(**c**(0,1), m, prob=**c**(0.5,0.5), **replace**=TRUE)
- age <−**rnorm**(m, 40,15)
- **delay** <−**sample**(**c**(0,1), m, prob=**c**(0.8, 0.2), **replace**=TRUE)
- y <−**rep**(LOF+, m)

After generating the values for each variable, we would combine these into our data matrix by **data_matrix** <−**cbind**(y, seizures, mf, absence, age, **delay**). This was completed for each disease group, and the corresponding data matrices were combined using the **rbind**() function. While this differs from the traditional methods, it is more suitable for this case, as the β parameters are not known and not fully researched at this time for each of the levels of the model. This allowed us to get accurate data and pairings based on what is known clinically about the disease groups.

Having simulated the data for training the models, we can now move into the regression steps. Using the framework provided, the model was fit according to the procedures described in Section 3.2. After fitting the model, the model was used to make predictions for the simulated data. This process was implemented for various group sizes per disease label, $m = 13$, $m = 36$, $m = 45$ and $m = 100$. In Table 3.1 the confusion matrices for each of the group sizes is shown, $m = 13$ is in the top left, $m = 36$ is in the top right, $m = 45$ is in the bottom left and $m = 100$ is in the bottom right. It should be noted that these labelings are based on the highest probability after running the model through the POLR function. In reality, when presenting this analysis to clinicians, the probability of labeling a patient into each group should be provided. This will ensure that patients that are harder to classify mathematically can be classified according to the clinicians professional judgement, an example and discussion of this can be found at the end of Section 3.2.2.

	DE	EE	DEE	LOF+	LOF-
DE	6	7	0	0	0
EE	4	9	0	0	0
DEE	4	0	9	0	0
LOF+	0	0	0	13	0
LOF-	0	0	0	0	13

	DE	EE	DEE	LOF+	LOF-
DE	23	13	0	0	0
EE	8	28	0	0	0
DEE	0	22	14	0	0
LOF+	0	0	0	36	0
LOF-	0	0	0	0	36

	DE	EE	DEE	LOF+	LOF-
DE	36	5	4	0	0
EE	15	29	1	0	0
DEE	6	11	28	0	0
LOF+	0	0	1	44	0
LOF-	0	0	0	0	45

	DE	EE	DEE	LOF+	LOF-
DE	96	0	4	0	0
EE	0	97	3	0	0
DEE	0	0	100	0	0
LOF+	0	0	0	100	0
LOF-	0	0	0	0	100

Table 3.1: Confusion matrices various group sizes

Now consider the misclassification rates. For $m = 13$, there were 15 mislabeled data points giving a misclassification rate of 23.1%. When $m = 36$, the misclassification rate is 23.9%. When $m = 45$ and $m = 100$, the misclassification rates are 19.1% and 1.4% respectively. Notice that the rate of error decreases as the sample size increases.

Focusing for a moment on the confusion matrix for $m = 100$, we see the model correctly labels all patients with variants that cause LOF+, and LOF-. The model mislabels, four DE and 3 EE patients as having DEE.

Having verified the code works, additional data was simulated at the $m = 100$ level and run through the model to test the accuracy. The confusion matrices for the tests that were completed are shown in Table 3.2.

	DE	EE	DEE	LOF+	LOF-
DE	49	0	1	0	0
EE	0	47	3	0	0
DEE	0	0	50	0	0
LOF+	0	0	0	50	0
LOF-	0	0	0	0	50

	DE	EE	DEE	LOF+	LOF-
DE	47	0	3	0	0
EE	0	49	1	0	0
DEE	0	0	50	0	0
LOF+	0	0	1	49	0
LOF-	0	0	0	0	50

	DE	EE	DEE	LOF+	LOF-
DE	45	0	5	0	0
EE	0	50	0	0	0
DEE	0	0	50	0	0
LOF+	0	0	0	50	0
LOF-	0	0	0	0	50

	DE	EE	DEE	LOF+	LOF-
DE	49	0	1	0	0
EE	0	46	4	0	0
DEE	0	0	50	0	0
LOF+	0	0	0	50	0
LOF-	0	0	0	0	50

Table 3.2: Confusion matrices from simulated test data

Based on the tables, we can see that the model overall performs pretty well. The average misclassification rate amongst the four trials is 1.9%. Most of the errors occur in the labeling of the DE or EE group as the DEE group. There is one instance where a LOF+ is mislabeled

as a GOF classification with the label of DEE. In terms of the problem, this makes sense as individuals with DEE exhibit symptoms from both the DE and EE groups, so if a DE or EE patient has more severe symptoms from the other category it is likely for a misclassification error.

3.2.2 Application Using Patient Data

Having created the template in Section 3.2.1, the process can now be updated and used with actual patient data. For this section, the prediction will be limited to the gain of function patients only. Because of this, the poset will have a single connected component and the algorithm will not utilize the first step of the data labeling or the regression processes as outlined in Section 3.2. The dataset that is used contains the following variables:

- Developmental Quotient: A number expressing a child's development determined by dividing the child's development based on test scores by the child's age and multiplying by 100.
- Age at seizure onset: the age (in months) at which a child first experienced a seizure.
- Seizure delay gap: the number of months between when a child exhibited developmental delay and their first seizure. Positive values indicate delay was seen first and negative values indicate seizure activity first.
- Seizure free: An indicator variable to represent if a patient has every obtained seizure freedom for any significant period of time, where significant is considered to be six or more months without seizures. (1 - experienced seizure freedom, 0 - not able to experience seizure freedom).
- Developmental Delay: An indicator variable signifying if a patient exhibits developmental delay. (1 - exhibits developmental delay, 0 - does not exhibit developmental delay)
- A collection of indicator variables to classify a patient's first seizure type. These are calculated as (1 - first seizure type was this, 0 - this was not the first seizure type). The following are the list of seizures:
  - Absence Seizure
  - Tonic-clonic (GTC) Seizure
  - Atonic Seizure
  - Focal Seizure
  - Convulsive Seizure
  - Myoclonic Seizure
  - Infantile Spasms
  - Tonic Seizure

– Unknown Seizure Type

The model was fit using approximately 70% of the data as a training set and 30% of the data as a testing set. The confusion matrices for the training set are shown in Table 3.3 and the confusion matrices for the testing set are shown in Table 3.4.

	DE	EE	DEE
DE	14	0	2
EE	0	27	3
DEE	3	7	69

	DE	EE	DEE
DE	13	0	0
EE	0	26	0
DEE	7	10	69

	DE	EE	DEE
DE	14	0	2
EE	0	26	2
DEE	0	3	78

	DE	EE	DEE
DE	16	0	1
EE	0	24	4
DEE	3	3	74

Table 3.3: Confusion matrices for patient data - training set

In the case of the training data, the model had an average error of 10%. Inspecting the confusion matrices, indicates that the model performs pretty uniformly across the groups, with the most incorrect labelings coming from the DEE patients. This makes sense, as there are about twice as many patients with this label. This perhaps could be reduced by tuning the parameter used when predicting with the first step of the model process using conditional logistic regression.

	DE	EE	DEE
DE	7	0	0
EE	0	9	3
DEE	1	6	24

	DE	EE	DEE
DE	10	0	0
EE	0	15	1
DEE	0	5	19

	DE	EE	DEE
DE	7	0	0
EE	1	11	2
DEE	0	3	26

	DE	EE	DEE
DE	6	0	0
EE	0	12	2
DEE	0	5	25

Table 3.4: Confusion matrices for patient data - test set

In the case of the testing data, the model had an average error rate of 14.5%, constituting a slight increase from the training data, which is to be expected. The confusion matrices show the same general trends as the ones for the training data, with a majority of mislabeling happening with the DEE patients. Of interesting note, none of the DE patients were mislabeled when using the test data.

While this model shows that there is room for improvement, another potential way to structure the output would be to provide the clinician with the probability of being in a particular category. For example, we will work with a patient in the data set. This patient has an EE classification and has the following characteristics:

- Developmental Quotient: 95
- Age onset: 11
- Seizure Delay Gap: 2.0
- Seizure Free: 0
- Developmental Delay: 0
- First Seizure Type: Tonic-clonic

Using the models, the conditional probabilities for each category are shown in Table 3.5. In the way the model works, the first step is to classify the data into either the group that contains the DE and EE patients or the group of DEE patients. Then if the DE and EE group is chosen the model makes predictions based on that. As the probability is so high for the patient being categorized as DEE, in the process described, the model would have stopped. However, as can be seen in Table 3.5, there is also a very high probability that the patient should be classified as an EE patient. In providing a clinician with data like this, they are able to see which categories a patient is likely to fall into. From there, they can use their judgement to make a choice that seems more likely, based on other clinical features that may not be incorporated into the model.

Classification	(DE	EE)	DEE
Probability	0	1	0.998

Table 3.5: Probabilities of diagnosis for patient

It should be noted that the probabilistic outputs provided in the table will not add to one. The reason being the multiple levels in the model. At each step of the regression process, the classification of the data in the groups will add to 1. However, the results for each level are not taken in whole. At each level, the results that are added to the table are those that assign the data point into a singleton set - that is, the data have been labeled into a group with only one possible outcome. Those that are not fully assigned to a group with only one possible outcome must be run through the next level, and again those probabilities will sum to 1. Thus, the probabilities provided in the table will always sum to greater than 1 unless the data is completely unordered. In Table 3.5, we see that the probabilities sum to 1.998 overall. In this case, there are two levels that are worked through, the assignment of the patient as an individual that has a DE/EE variant or a DEE variant, and a secondary assignment of a DE variant vs and EE variant. For this particular example the sum of the DE and EE categories will always be 1 with the DEE variant pushing the results larger than one. Visually, we can think about this process in a tree diagram, with probabilities as weights on branches. For this particular patient, a tree representing the probabilities is seen in Figure 3.2.

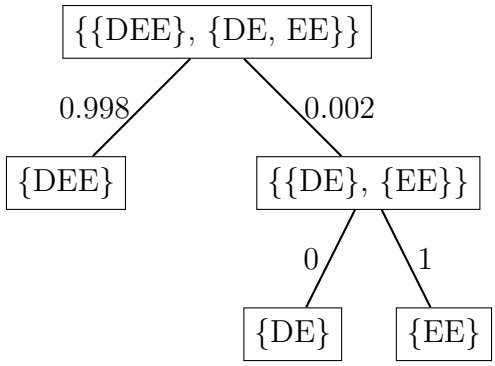

Figure 3.2: Tree of probabilities for disease diagnosis

3.3 Future Work

In Sections 2.1 and 2.2 we introduced a novel algorithm and showed pseudocode that can be used to model partially ordered data. While many things in the world can be modeled by POS-PCM, it is not widely used at this time, in part because it is a newer development but also in part because it is not the most straightforward algorithm to implement. As future work, it would be beneficial to clinicians and those that could use POS-PCM to model their data to have an interface built that would complete the process for them. The interface would allow the user to enter information about the structure of the poset and provide the data, and in return the user would get a working model to help make predictions - either exact labelings or probabilities for each category to make conclusions regarding their data.

The discussion of a user interface, above, assumes that a user knows the underlying poset structure of the data. However, this is not always the case. As an extension of the user interface described, it could be beneficial to allow a user to experiment with different poset structures based on their knowledge of the subject matter to compare different qualities of fit. In Peyhardi [6], a detailed method is discussed regarding fitting the best structure to the data, which could perhaps be incorporated to the interface as well. This would be particularly useful for users, that perhaps think there is a structure underlying the data, but they are not sure what that structure might be beyond broad strokes.

In Zhang, 2012 [7] and Peyhardi, 2013 [6], the model is described, and some statistical properties are discussed. However, there has not been a lot of work published regarding power of the model or making comparisons to other types of modeling. Further research into this area could open up the use of it in more areas.

Bibliography

[1] G Casella and R. Berger. *Statistical Inference*. Duxbury Resource Center, 6 2001.

[2] Julian James Faraway. *Extending the linear model with R : generalized linear, mixed effects and nonparametric regression models*. 2nd edition edition, 2006.

[3] Joshua B. Hack, Kyle Horning, Denise M. Juroske Short, John M. Schreiber, Joseph C. Watkins, and Michael F. Hammer. Distinguishing Loss-of-Function and Gain-of-Function SCN8A Variants Using a Random Forest Classification Model Trained on Clinical Features . *Neurology Genetics*, 9(3), 6 2023.

[4] Joshua B. Hack, Joseph C. Watkins, and Michael F. Hammer. Machine learning models reveal distinct disease subgroups and improve diagnostic and prognostic accuracy for individuals with pathogenic SCN8A gain-of-function variants. *Biology Open*, 13(4), 4 2024.

[5] Hammer MF, Xia M, and Schreiber JM. SCN8A-Related Epilepsy and/or Neurodevelopmental Disorders. Technical report, 2023.

[6] Jean Peyhardi. A new generalized linear model (GLM) framework for analysing categorical data; application to plant structure and development. Technical report, Universite Montpellier, 2013.

[7] Qiang Zhang and Edward Haksing Ip. Generalized linear model for partially ordered data. *Statistics in Medicine*, 31(1):56–68, 1 2012.