MODELO PARTIÇÃO PRODUTO PARA ATRIBUTOS CATEGÓRICOS

TULIO LIMA CRISCUOLO

MODELO PARTIÇÃO PRODUTO PARA ATRIBUTOS CATEGÓRICOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Wagner Meira Júnior
Coorientador: Renato Martins Assunção

Belo Horizonte
Agosto de 2019

TULIO LIMA CRISCUOLO

PRODUCT PARTITION MODEL FOR CATEGORICAL FEATURES

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Wagner Meira Júnior
Co-Advisor: Renato Martins Assunção

Belo Horizonte
August 2019

Criscuolo, Túlio Lima

C932p Product partition model for categorical features [manuscrito] / Túlio Lima Criscuolo. — Belo Horizonte, 2019
xxiv, 73 f. : il. ; 29cm

Orientador: Wagner Meira Junior
Coorientador: Renato Martins Assunção

Dissertação (mestrado) — Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação.
Referências: f. 35 - 42

1. Computação - Teses. 2. Análise de regressão Teses. 3. Categorias (Matemática) - processamento de dados - Teses. 4. Clustering - Teses. I. Meira Junior, Wagner. II. Assunção, Renato Martins. III. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação. IV. Título.

CDU 519.6*73(043)

Ficha catalográfica elaborada pela bibliotecária Irénquer Vismeg Lucas Cruz - CRB 6^a Região n^o 819.

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Product Partition Model for Categorical Features

TULIO LIMA CRISCUOLO

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. WAGNER MEIRA JÚNIOR - Orientador
Departamento de Ciência da Computação - UFMG

PROF. RENATO MARTINS ASSUNÇÃO - Coorientador
Departamento de Ciência da Computação - UFMG

PROFA. ROSANGELA HELENA LOSCHI
Departamento de Estatística - UFMG

PROF. DENIS DERATANI MAUÁ
Instituto de Matemática e Estatística - USP

PROF. FABRÍCIO MURAI FERREIRA
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 25 de Outubro de 2019.

Acknowledgments

I want to thank my advisor Wagner Meira, for giving me the freedom to explore distinct topics and mentoring me in the last couple of years. I want to thank my co-advisor Renato Assunção, for his inspiration and enthusiasm in handling this relevant but neglected problem. Also, I want to thank Rosangela Loschi, for her patience in helping me with the math and whose contribution made this work possible.

I wanna thank my family for their continuous support. A special thank you goes to my cousin Lucas Criscuolo, although no longer with us, for supporting me getting into university and inspiring me to get into machine learning area.

And lastly, but not less important I wanna to thank everyone in the SPEED lab, it was a great experience sharing the laboratory with all of you.

Finally, this work was partially supported by CNPq, CAPES, and FAPEMIG.

"Patience is a tree whose root is bitter, but its fruit is very sweet."
(Canadian proverb)

Resumo

O tratamento de atributos categóricos com uma grande quantidade de categorias é um problema recorrente em análise de dados. Existem poucas propostas na literatura para lidar com este problema importante e recorrente. Introduzimos um modelo generativo, que simultaneamente estima os parâmetros e o agrupamento do atributo categórico em grupos. Nossa proposta é baseada em impor um grafo no qual os nós correspondem a categorias e criando uma distribuição de probabilidade sobre partições deste grafo. Sendo um modelo Bayesiano, somos capazes de fazer inferência a posteriori sobre os seus parâmetros e o particionamento do atributo categórico. Comparamos nosso modelo com métodos estado da arte e mostramos que obtemos uma capacidade preditiva igualmente boa e melhor interpretação dos resultados obtidos.

Palavras-chave: Modelo Hierárquico, Regressão, Clusterização, Redução de dimensão.

Abstract

A common difficulty in data analysis is how to handle categorical predictors with a large number of levels or categories. There are few proposals developed in the literature to handle this important and frequent problem. We introduce a generative model that simultaneously carries out the model fitting and the aggregation of the categorical levels into larger groups. Our approach is based on imposing a graph where the nodes are categories and creating a probability distribution over meaningful partitions of this graph. Being a Bayesian model, it allows the posterior inference, including uncertainty measurement, on the estimated parameters and the categories partition. We compare our method with the state-of-art methods showing that it has equally good predictive performance and much better interpretation ability. Given the current concern on balancing accuracy versus interpretability, our proposal reaches an excellent result.

Palavras-chave: Hierarchical Model, Regression, Clustering, Dimensionality Reduction.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

A recurrent and difficult problem in predictive modeling is the presence of categorical variables with a large number of levels or categories. For example, an educational individual outcome may be analyzed using the college student major as a feature. This is a categorical variable with dozens of distinct and unordered categories. As another example, risk analysis involving automobile insurance claims data have a variable indicating the client car model involved in the event. This variable typically has about one hundred different instances. Occupational disease analysis with individual data uses the job occupation as a feature with a very large number of categories. Another example is the analysis of credit risk over a large region that considers the client's geographical location, such as the residence zip code. It is not uncommon for the location variable to have from hundreds to thousands of different categories.

Using such features with their many levels leads to harmful consequences for the machine learning algorithms. Usually, the feature is represented using the dummy-coding or one-hot-encoding that requires to introduce an indicator variable for each possible level of the categorical feature. Learned coefficients become very unstable making the results hard to interpret. It is a rare situation when we are interested in the effects of the feature-specific levels – such as specific car models or college majors. It is more usual that we are more interested in broad categories or subgroups of majors. However, it is not clear how to aggregate the levels into higher-level categories that are interpretable and statistically efficient. Besides the interpretation issue, the inclusion of categorical variables with many levels in a predictive model easily leads to a sparse design matrix. Bateni et al. [2019] shows that the number of parameters in the first layer in a neural network can dwarf the number of parameter in the remaining network due to a categorical variable having too many levels. The consequence may be an ill-conditioned optimization problem that results in an overfitted model.

The most common strategy recommended by practitioners (e.g., Kaggle participants) is to reduce the number of levels of the feature. When the feature levels are naturally organized in a hierarchical tree structure, one crude way is to cut the hierarchy at a high level. For example, considering a job occupation feature, one may consider law clerks, paralegals, and legal secretaries as a broad category named legal assistants, and web designer, web developer, network engineers and other computer jobs as a single category. The disadvantages of this crude strategy is the need of a pre-existing hierarchy and the arbitrary choice of a stopping level in a hierarchy, typically decided in advance, without concerning to the response to be predicted and risking to increase the model bias.

Another strategy is to reduce the encoding of the categorical variable based on the response variable, called *target encoding*, as proposed by Micci-Barreca [2001]. One may target encode the levels by replacing each category with its mean response and hence substituting the categorical variable by a continuous one. Yet another strategy is to sort the categories by their mean response and to retain only the categories that are responsible for, let's say, 90% of the examples, merging all remaining categories in an "Other" category. This can lead to fewer categories as many levels may appear only rarely. A difficulty with a response-based encoding is that any method using the response to guide the feature engineering will be subject to overfitting and it will lead to biased error bars for the coefficients and hypothesis testing.

Some more theoretically-based answers to this problem have been proposed recently and are reviewed in section 2.6. A Lasso-constrained regression approach for an analysis of variance proposed by [Bondell and Reich, 2009] aims at simultaneously estimating coefficients and collapsing levels of a covariate. The important novelty is to impose a Lasso-type regularization on pairwise coefficients' differences rather than the coefficients directly. Using a similar approach, Tutz and colleagues extended this idea to several different models in a series of papers [Gertheiss and Tutz, 2011; Oelker et al., 2014; Tutz and Berger, 2018]. Another proposal that is becoming popular was suggested in the context of extreme boosting [Prokhorenkova et al., 2018]. It relies on an artificial ordering of the training examples and it uses the "previous" examples in order to estimate a target encoding variable.

In this work, we propose a principled Bayesian solution based on a generative model with the categorical variable levels organized in a graph. The set of features in our predictive model is partitioned into two classes. One contains the categorical variables with many levels. The other class contains the usual continuous features and the categorical features with a small number of levels. We propose a Bayesian regression model that simultaneously estimates the parameters of the second class as

well as aggregates the levels of the first class and estimates its coefficients.

More specifically, we propose a Product Partition Regression Model (PPRM) induced by spanning trees to deal with linear regression analysis involving a categorical covariate with a large number of levels. The model is represented in terms of graphs with vertices representing the different levels of the categorical variable and edges connecting neighbouring categories (a complete graph is allowed if no structure is available). We assume that categories with similar impact on the response variable should be merged thus reducing the dimension of the categorical covariate. The goal is to simultaneously learn the regression coefficients and to infer the effect and grouping of the categorical covariates. Our proposed model builds on the ideas proposed in [Teixeira et al., 2015, 2019] to carry out spatial clustering based on stochastically pruning spanning trees. Being a principled Bayesian solution, we obtain uncertainty measures associated with the coefficients *and the aggregation of the covariate levels in the first class*.

In Chapter 2 we review the technical background and previous related work. Chapter 3, we describe the proposed model, integrating the product partition model to aggregate levels of a categorical variable in a regression context. In Chapter 4, we present the empirical evaluation of the proposed model and compare its results against baselines strategies. Finally, chapter 5 concludes this thesis with final remarks and directions for future work.

Chapter 2

Background and Theoretical Framework

In this chapter we start by presenting concepts used in our proposed solution. Later, we review the literature related to the problem.

First, Section 2.1 provides a brief review of clustering concepts. Then, section 2.2 reviews Markov Chain Monte Carlo (MCMC) sampling algorithms. Later, in Section 2.3 we present necessary concepts from graph theory and how the clustering task is modeled as a graph partitioning problem. Next, sections 2.4 and 2.5, quickly review concepts of product partition model and Bayesian linear regression, that will be extended in the proposed solution in Chapter 3. Finally, Section 2.6 provides a literature review of related work.

2.1 Clustering

Cluster analysis (or Clustering) is the unsupervised classification of patterns (feature vector, observations or data items) into groups (clusters) based on similarity. Clustering is based on the intuitive sense that patterns within-cluster are more similar to each other than to patterns in a distinct cluster.

Clustering models have been applied in a wide range of research communities such as biology, geography, geology, psychology, marketing, information retrieval, and pattern recognition. Typical applications of clustering analysis are to get insight about the data or to preprocess the data for other algorithms.

Several techniques for clustering have been proposed. It is agreed on that clustering techniques can be divided into two main categories: hierarchical and partitional. Hierarchical clustering generates a sequence of partitions while partitional clustering

splits the data into some number (usually specified) of clusters without the hierarchical structure. An important concept is hard and fuzzy (soft) clustering. In hard clustering, each object is assigned to only one cluster, whereas in fuzzy clustering each object can belong to all clusters with a degree of membership.

There is also *constrained clustering*, where constraints are imposed for objects to belong to the same cluster. A constrained version of k-means is proposed in Wagstaff et al. [2001]. This work focuses on a constrained clustering problem with a hard assignment to clusters.

A great survey on clustering algorithms was written by Xu and Wunsch [2005].

2.2 Markov Chain Monte Carlo

In this section, we introduce some important Markov Chain Monte Carlo (MCMC) algorithms used in our solution. These algorithms, consists in constructing a Markov Chain with a specific target distribution, whose samples are obtained by observing samples from the specified Markov Chain. A comprehensive history review can be found on Brooks et al. [2011] (Chapters 1 and 2).

We begin by first reviewing the Metropolis-Hastings (MH) and next we review the Gibbs sampler, which is as a special case of MH.

2.2.1 Metropolis-Hastings

The Metropolis-Hasting algorithm is a general term for a useful family of methods to draw samples from a probability distribution $p(\theta)$ (possibly multivariate), from which directly sampling is not viable (e.g: does not exists a closed formula or intractable integral). The required conditions for the convergence to the target distribution are irreducibility and aperiodicity [Roberts and Smith, 1994]. The irreducibility condition means that for every θ_a and θ_b in the domain of $p(.)$, we must be able to move from θ_a to θ_b in a finite number of steps with nonzero probability and the aperiodicity condition means that the number of steps from any state must not be a multiple of some integer.

The Metropolis algorithm was first introduced in the work of Metropolis et al. [1953]. It is an adaptation of a random walk, that uses an accepting/rejecting strategy to better represent a target distribution [Gelman et al., 2004]. The algorithm consists in defining a number of iterations S (length of the Markov Chain) and a symmetric proposal distribution (or jumping distribution) $q(\theta_a|\theta_b) = q(\theta_b|\theta_a)$, for every iteration s.

The pseudo-code 1 describes the steps of the Metropolis algorithm. It starts by with a valid starting point $\theta^{(0)}$, for which $p(\theta^{(0)}) > 0$. For a specified number of iterations, it draws a new state from the proposal distribution $q(.|.)$ and computes the Metropolis ratio r. The proposed state is accepted with probability $min(1, r)$, if the proposed state is rejected the current state is used at the next iteration. It is important to note that the chain may remain at the same state for many iterations.

Algorithm 1: Metropolis Algorithm

Initialize $\theta^{(0)}$
for $s = 1$ *to* S **do**
  Draw $\tilde{\theta}$ from $q(\tilde{\theta}|\theta)$
  Compute $r = \min\left(1, \frac{p(\tilde{\theta})}{p(\theta^{(s-1)})}\right)$
  Draw $u \sim U(0, 1)$
  if $u \leq r$ **then**
    Set $\theta^{(s)} = \tilde{\theta}$
  else
    Set $\theta^{(s)} = \theta^{(s-1)}$
return $\{\theta^{(1)}, \theta^{(2)}, ..., \theta^{(S)}\}$

The Metropolis-Hastings [Hastings, 1970] algorithm is a generalization of the Metropolis for when the proposal distribution q is not symmetric. To correct for the asymmetry, a correction term is introduced, thus replacing r to a ratio of ratios (2.1). The ratio is always defined, since the denominator can only be zero if $q(\tilde{\theta}|\theta) = 0$ or $p(\theta^{(s-1)}) = 0$. The first will never happen because it is not possible to propose a state with probability density equals to 0 and the second will not happen because a state with probability density equals to zero with probability one.

$$r = \min\left(1, \frac{p(\tilde{\theta})q(\theta^{(s-1)}|\tilde{\theta})}{p(\theta^{(s-1)})q(\tilde{\theta}|\theta^{(s-1)})}\right). \tag{2.1}$$

A few remarks can be made. The proposal distribution $q(.|.)$ has to have a positive density on the support of the target distribution $p(.)$. The probability densities $p(.)$ and $q(.|.)$ in the ratio can be substituted by an unnormalized densities, since the normalizing constant are common on both the numerator and denominator. In practice, for numerical reasons, it is useful to consider the log of the ratio as defined in (2.2) and accept the new state if $\log(u) \leq r_{log}$.

$$r_{log} = \min(0, \log(p(\tilde{\theta})) - \log(p(\theta^{(s-1)})) + \log(q(\theta^{(s-1)}|\tilde{\theta})) - \log(q(\tilde{\theta}|\theta^{(s-1)})) \quad (2.2)$$

2.2.2 Gibbs Sampler

The Gibbs Sampler (or Gibbs sampling) first introduced in Geman and Geman [1984] is a widely used MCMC algorithm to sample from a multidimensional distribution. It is know as Gibbs sampler, because it was used for a Bayesian study of a Gibbs random field in the context of discrete image processing.

Gibbs sampler is a MCMC scheme where the transition kernel is formed by the full conditional distributions. Let $p(\theta)$ be a distribution of interest, and the parameter vector θ be divided into d components $(\theta_1, \theta_2, ..., \theta_d)^T$ where each component can be a scalar, vector or matrix. Consider that the full conditional distribution for each component is know or can be sampled from, that is, $p(\theta_i|\theta_{-i})$ for $i = 1, 2, ..., d$ is available.

In a multidimensional problem, sampling directly from the target distribution $p(\theta)$ can be complicated, costly or unavailable. Gibbs sampling provides an efficient way to sample from the target distribution, by successively sampling from the full conditional distributions. Let S be the number of iterations of the Gibbs sampler; each iteration consists in cycling through each of the d components of θ and sampling from its full conditional distribution:

$$\theta_i^s \sim p(\theta|\theta_{-i}^{s-1}) \quad (2.3)$$

Where $\theta_{-i}^{s-1} = (\theta_1^s, \ldots, \theta_{i-1}^s, \theta_{i+1}^{s-1}, \ldots, \theta_d^{s-1})$ is the partially updated vector of parameters at iteration s. For every iteration, each parameter θ_i is updated given the most recent sampled value for θ_{-i}.

The Gibbs sampler is a special case of the Metropolis-Hastings. It consists in defining iteration s to be a sequence of d updates, updates one parameter conditioned in all other, with the proposal distribution as the full conditional distribution, such that the metropolis ratio is always 1 and the jumping is accepted. A detailed proof can be found on Brooks et al. [2011].

For complicated distributions, we might not be able to directly sample from the full conditional distribution of some parameters. For those parameters, which the FCD is not available, the Metropolis algorithm can be used. Thus, Gibbs and Metropolis algorithm can be viewed as a building block to draw samples from complex

distributions, we avoid using the term *Metropolis-within-Gibbs*, since Gibbs is a special case of a Metropolis algorithm.

It is known that Gibbs sampler can be computationally expensive and slow to converge to the target distribution. To overcome these difficulties, some strategies such as blocking and parameter collapsing are used. Blocked Gibbs sampler, consists in sampling from one or more variable at the same step, conditioned on all other variables, instead of sampling each one individually. Collapsing in Gibbs sampling, consists in integrating out (marginalize over) some parameters from the FCD, this is useful to increase the convergence rate of the sampler or making the sampling viable, but care must be taken to maintain the correlation of the parameters of interest; for more details we suggest Dyk and Park [2011].

2.3 Graph representation

In this section, we review basic concepts in graph theory used in our proposed solution and how we model the clustering task as a graph problem.

2.3.1 Basic definitions

A graph $G = (V, E)$, is a data structure composed by a set of nodes (or vertices) $V = \{v_1, v_2, \ldots, v_n\}$ and a set of edges $E \subseteq V \times V$, where an edge $e = (v_i, v_j) \in E$ indicates that there exists a relationship between nodes v_i and v_j, we say that v_i and v_j are neighbours in G. An *undirected graph* is a graph were the set of edges E is unordered, that is, the relation between two vertices does not depends on the direction of the edge. To our notation if G is undirected and $(v_i, v_j) \in E$, we can not have $(v_j, v_i) \in E$. Unless explicitly specified to be a connected graph, a graph will reference to a undirected graph. An example of an undirected and directed graph are show on figure 2.1.

A *walk* in a graph G, of length k between two vertices v_0 and v_k is a finite non-null alternating sequence of vertex and edges $W = v_0e_1v_1e_2v_2...e_kv_k$, such that the end of $e_i = (v_{i-1}, v_i), 1 \leq i \leq k$. If the edges of a walk W are distinct, then W is called a *trail*. In addition, if the vertex of a trail W are distinct we call W a *path*, it is not possible for a path to start and end at the same vertex. A *cycle* is defined to be a trail in which only the first and last vertex are equal.

Two vertices u and v in G are said to be connected, if there exists a path between the vertices u and v in G. Let $V^k \subseteq V$, we define $G[V^k]$ or G^k to be an vertex induced subgraph of G, it is obtained by deleting the edges of G that are not in V^k

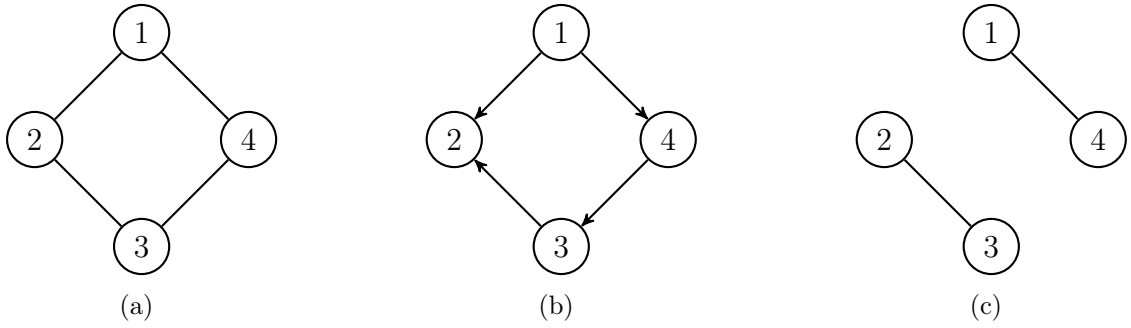


Figure 2.1. Illustration of types of graph: (a) an undirected graph; (b) a directed graph; (c) a disconnected graph

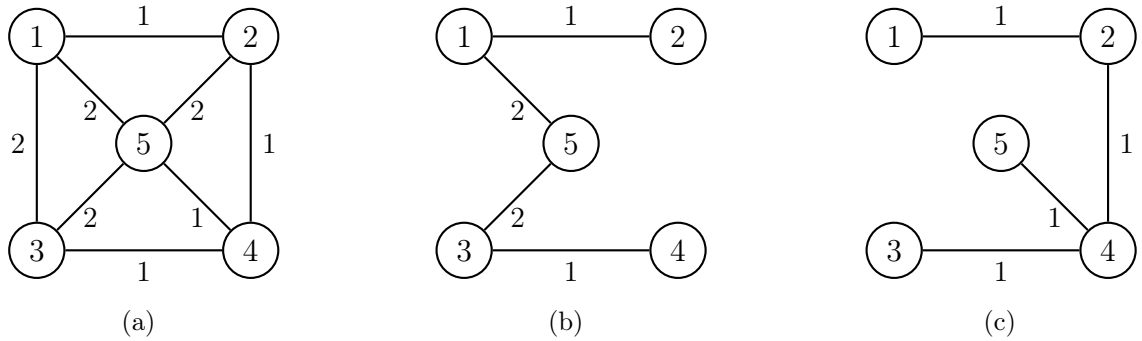


Figure 2.2. (a) A weighted graph; (b) a spanning tree; (c) a minimum spanning tree.

and its incident edges. An equivalence relation V in V, such that uRv if vertex u and v are connected in G defines a partition of the set of vertex V in non-empty subsets $V^1, V^2, \ldots, V^{\mathcal{F}}$. The sub-graphs induced by each subset $G^i, 1 \leq i \leq \mathcal{F}$ is called a **connected component**. If $\mathcal{F} = 1$, we call G a *connected* graph, otherwise we call G *disconnected*. Connected and disconnected graphs are show in Figure 2.1.

A **tree** graph is a connected graph with no cycles. A tree has several properties, but the properties used in this work are: (1) any two vertices in a tree are connected by a unique path; (2) the number of edges in a tree is equal to the number of vertices minus one; (3) every edge is a cut edge, that is, the removal of any edge creates a new component. A **spanning tree** of a graph G is a *spanning subgraph* of G, that is a tree. That is a tree $T = (V, E')$ is a spanning tree of $G = (V, E)$, if $E' \subseteq E$, thus T contains all vertices in G and the minimum number of edges of G to be connected.

Let $w(u, v)$ be a weight function for the edge connecting vertices u and v. A **minimum-weight spanning tree** (MST) is a spanning tree with the minimal sum of weights. There can be several spanning tree with a minimum sum of weights. Figure 2.2 shows a weighted connected graph, a spanning tree and a minimum spanning tree.

A **minimum spanning forest** (MSF),is a generalization of MST for disconnected graphs. It is defined as the union of minimum spanning trees of each component of a disconnected graph.

2.3.2 Computational representation

A graph structure is commonly represented by an adjacency matrix or an adjacency list. An adjacency matrix is a square matrix A, with rows and columns labeled by the vertices in a graph G. The matrix index $A_{(i,j)} = 1$, if vertices v_i and v_j are neighbours in G and it is 0 otherwise. The adjacent matrix is symmetric for undirected graphs and its diagonal elements are all equal to 0. An adjacency list A is a list of list. Each element A_i is a list containing the index of the vertices that are neighbor of the vertex v_i. Unless otherwise stated, we use an adjacency list to represent a graph.

2.3.3 Modeling as graph problem

We model the task of clustering levels of a categorical variable as a graph problem. Consider a categorical variable with V distinct levels, we model that variable as a graph with V vertices, where each vertex corresponds to a level. A *prior-knowledge* of clustering of two levels of the categorical variable is introduced by adding an edge connecting their respective vertices in G, therefore if there exists a path between two vertices, they can be clustered (belong to the same component). Thus, the task of clustering the levels of a categorical variable is to find a partitioning of the set of vertices of a graph.

For a more formal definition, let G be a graph, with $\mathcal{F} \geq 1$ components (possibly disconnected), that represents the prior clustering knowledge of a categorical variable. As previously described, a connected component in G defines an equivalence class, thus it defines a partitioning of G, we define each connected component to be a cluster. Thus, a partitioning of G in c components corresponds to a clustering of a categorical variable in c clusters.

2.4 Product Partition Model

The product partition model (PPM) is a probabilistic model for cluster identification introduced in A. Hartigan [1990] and later extended in Barry and Hartigan [1992] to the study of a change point identification problem. The PPM is a convenient framework to allow the data to weight partitions that are likely to hold.

Initially, we present the PPM for contiguous cluster identification that is useful in change point analysis. Assume a sequence $\boldsymbol{Y} = (y_1, y_2, \ldots, y_n)$ of n observations and let $I = \{1, \ldots, n\} \cup \{0\}$. Denote by $\boldsymbol{\pi}$ be a random partition of I into c contiguous blocks with endpoints $i_j, 1 \leq j \leq c$. Let G_j be the cluster $[i_{j-1} + 1, \ldots, i_j]$ such that

$$\boldsymbol{Y} = \left((y_1, y_2, \ldots, y_{i_1}), (y_{i_1+1}, y_{i_1+2}, \ldots, y_{i_2}), \ldots, (y_{i_{c-1}+1}, y_{i_{c-1}+2}, \ldots, y_{i_c})\right).$$

The random quantity $(\boldsymbol{\pi}, \boldsymbol{Y})$ is said to be a PPM if the following conditions are satisfied:

(i) The prior distribution for the partition $\boldsymbol{\pi} = \{G_1, G_2, \ldots, G_c\}$ is the following product distribution:

$$p(\boldsymbol{\pi}) = K \prod_{k=1}^{c} c(G_k), \tag{2.4}$$

where $c(G_k)$ is the prior cohesion for the cluster G_k and K is a normalization constant. The prior cohesion $c(G_k)$ is a non-negative component, subjectively chosen, that measures the strength of one priors beliefs that the elements in G_k should be in the same cluster.

(ii) Let $\boldsymbol{Y}_{G_k}$ be the set of observations corresponding to block G_k. Given the partition $\boldsymbol{\pi} = \{G_1, G_2, \ldots, G_c\}$ in c blocks, the observation in different blocks are independent

$$p(\boldsymbol{Y}|\boldsymbol{\pi}) = \prod_{k=1}^{c} p_{G_k}(\boldsymbol{Y}_{G_k}).$$

where p_{G_k} is the joint density of the random vector $\boldsymbol{Y}_{G_k} = (y_{k,1}, y_{k,2}, \ldots, y_{k,n_k})$ The element p_{G_k} is called *data factor*.

A product partition distribution can be defined to more general objects, other than contiguous clusters Hartigan [1990], such as hierarchical (trees) and ordered objects (spatial, temporal).

A more general PPM [Barry and Hartigan, 1992, 1993] assumes the following parametric approach which is considered in our work. In this more general model, each observation y_i has an associated parameter θ_i, $1 \leq i \leq n$. The random partition $\boldsymbol{\pi} = \{G_1, G_2, \ldots, G_c\}$ induces the random clustering of $\boldsymbol{Y}$ by partitioning $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$ in c clusters such that $\theta_i = \theta_{G_k}, \forall i \in G_k$, that is the parameter θ_i is constant within the cluster G_k. In this model, the joint density of the observations and parameters is

expressed as the following product distribution:

$$p(\boldsymbol{\theta}, \boldsymbol{Y}|\boldsymbol{\pi}) = \prod_{k=1}^{c} p(\boldsymbol{Y}_{G_k}|\theta_{G_k})p(\theta_{G_k}),$$

where

$$p(\boldsymbol{Y}_{G_k}|\theta_{G_k}) = \prod_{i \in G_k} p(y_i|\theta_{G_k}).$$

and $p(\theta_{G_k})$, is the *prior* distribution for the cluster parameters θ_{G_k}.

For a given cluster G_k, the joint density of the data factor $\boldsymbol{Y}_{G_k}$ is

$$p(\boldsymbol{Y}_{G_k}) = \int p(\boldsymbol{Y}_{G_k}|\theta_{G_k})p(\theta_{G_k})d\theta_{G_k}.$$

Given the observations, the conditional distribution of parameters and partition, is also a product distribution with posterior density given by (2.5) and (2.6), respectively.

$$p(\theta_{G_k}|\boldsymbol{Y}_{G_k}) = \frac{p(\theta_{G_k})\Big[\prod_{i \in G_k} p(y_i|\theta_{G_k})\Big]}{p(\boldsymbol{Y}_{G_k})}, \tag{2.5}$$

and

$$p(\boldsymbol{\pi}|\boldsymbol{Y}) \propto \prod_{k=1}^{c} c(G_k)p(\boldsymbol{Y}_{G_k}). \tag{2.6}$$

The PPM provides us a convenient framework to make inference in a probabilistic way about the clustering of the parameters based on observations. In this work, we use the clustering structure of the PPM to group levels of a categorical variable. More details are given in the description of the proposed model in chapter 3.

2.5 Bayesian Linear Regression Model

A Bayesian linear regression model, consists in modelling the observed outcome $\boldsymbol{Y} \in \mathbb{R}^{n\times 1}$, assumed in this work to be continuous, as a function that depends on the data $\boldsymbol{X} \in \mathbb{R}^{n\times D}$ that can be discrete or continuous. The *normal linear model* 2.7, in which the observed outcome is normally distributed given the data is the most widely used

version of the model.

$$\boldsymbol{Y} \mid \boldsymbol{\beta}, \sigma_y^2, \Sigma_y \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{\beta}, \sigma_y^2 \Sigma_y). \tag{2.7}$$

A common choice is to consider the observation errors to be independent and have equal variance. That is, the co-variance matrix Σ_y is set to be equals to $\boldsymbol{I}$, where $\boldsymbol{I}$ is a $n \times n$ identity matrix.

The parameters $\boldsymbol{\beta} \in \mathbb{R}^{D\times 1}$ and $\sigma_y^2 \in \mathbb{R}_{>0}$ are unknown. Common prior choice for $\boldsymbol{\beta}$ and σ_y^2 are normal and inverse-gamma distribution respectively. The model is hierarchically represented as follows:

$$\begin{aligned} y_i &= \boldsymbol{X}_i\boldsymbol{\beta} + \epsilon_i \\ \epsilon_i \mid \sigma_y^2 &\sim \mathcal{N}(0, \sigma_y^2) \\ \boldsymbol{\beta}|\sigma_y^2 &\sim \mathcal{N}(\mu_{\boldsymbol{\beta}}, \sigma_y^2 \Sigma_{\boldsymbol{\beta}}) \\ \sigma_y^2 &\sim IG(\gamma, \eta) \end{aligned}$$

Where $\mu_{\boldsymbol{\beta}} \in \mathbb{R}^{D\times 1}, \Sigma_{\boldsymbol{\beta}} \in \mathbb{R}^{D\times D}$ are the prior mean and co-variance for $\boldsymbol{\beta}$ and $\gamma \in \mathbb{R}_{>0}, \eta \in \mathbb{R}_{>0}$ are the prior shape and rate for σ_y^2. Posterior distribution of the parameters is used to analyze the Bayesian linear model. Samples from the posterior distribution are typically obtained using MCMC.

A indicator variable is often used to include a categorical variable in the regression model. This allows the model to separate the effect of each level of the categorical variable. For a binary variable, a single 0/1 indicator variable suffices. For a categorical with V levels, a common choice is to use the dummy coding, that is $V - 1$ indicator variables in addition to a constant term. That consists in choosing a level as a reference, coefficients related to each indicator represents the change in the response of that level in comparison to the base base level maintaining everything else constant. Another approach, is to use the one-hot-encoding (OHE) of the categorical variable, that consists to use V indicator variables. The OHE should be used with caution, since it contains redundancy and can lead to multicollinearity problems, for example if used without removing the constant term.

We commend Gelman et al. [2004] (chapters 14 & 15) for a detailed presentation of a Bayesian linear regression.

2.6 Related Work

A popular question in blogs and question-and-answer websites is how to deal with categorical features with a large number of levels in machine learning models Jerry [2013]; Zumel [2012]; shadowtalker [2017]; Ferreira [2019], where a common suggested strategy is to encode the categorical features using a one-hot or dummy encoding of the features, since most models require features to be continuous rather than categorical. These representations make the input feature vector to be high dimensional since it increases linearly with the number of levels in the categorical feature. An intuitive approach to reduce the dimension of the encoded feature is to pre-process the data aggregating categories in *a priori* basis, however, this approach is prone to poorer learning performance.

These problem called the attention of some researchers. Target coding was suggested by Micci-Barreca [2001]. If Y is the response variable and Z is a categorical feature with V levels $z_1, \ldots, z_V$, substitute z_k by a numerical value $W(k)$, such as $W(k) = \mathbb{E}(Y|Z = z_k)$. This is implemented in Python in the scikit-learn category-encoders library [McGinnis, 2019] and in the R package Vtreat [John Mount, 2019]. However, we do not favor this approach as it is prone to overfitting (target leakage) and lacks a theoretical basis. CatBoost proposed by Prokhorenkova et al. [2018] is an extension of the Extreme Gradient Boosting model Chen and Guestrin [2016], that is specialized to categorical features and has gained popularity. To avoid the bias introduced by the usual target coding CatBoost resorts to an artificial ordering of the examples and treating them as a data stream. At each data point, previous examples in their ordering are treated as the history of the process alleviating the bias problem.

A regularization approach was proposed using the Lasso penalty. Usually, Lasso penalization is used for *feature selection*, where irrelevant regression coefficients are set to be zero Tibshirani [1996]. In Analysis of Variance (ANOVA), it is of interest to identify differences between levels of a factor. A Lasso-based penalty strategy was proposed by Bondell and Reich [2009] to force *pairs* of estimated coefficients of categorical factor to be *equal* in an ANOVA type of identification of differences. An extension to a *regression type problem* was proposed by Gertheiss and Tutz [2011], where predictive accuracy is of interest. Equally estimated coefficients corresponds to a cluster of categorical levels. The objective is to aggregate the levels with the same effect at the same time we fit the model, rather than aggregating them in an *ad-hoc* way before model fitting. In practice, to achieve consistent results, the Gertheiss and Tutz [2011] method depends on adaptive weights Zou [2006], a two-step procedure to compute the weights used for the Lasso penalization. These weights are based on the pairwise difference be-

tween the estimated coefficients from a first-step ordinary least squares regression. This approach of clustering categorical features, while also estimating other coefficients has been extended to various contexts in Oelker et al. [2014] and Tutz and Berger [2018]. Also, the Lasso penalization strategy was used by Hallac et al. [2015] to cluster nodes in a social graph.

A Bayesian approach to fuse levels of categorical predictors in a regression type model was proposed by Pauger and Wagner [2019]. To allow for sparse modelling of the categorical predictors, the authors proposed a finite Normal mixture, that shrinks of coefficients to zero or to non-zero with a clustering of the coefficients. That is, coefficients are fused if they are assigned to the same mixture component.

A product partition model was proposed by Smith and Allenby [2019] in the context of demand models. The goal is to forecast the product demand given prices for all related goods and other product specific covariates. A clustering structure is introduced to cluster the coefficients of price elasticity of related goods. To navigate the space of partitions a random walk Metropolis-Hasting was used. In our task we are interested in the effect of a level in the response and not in the cross elasticity.

Lastly, an information-theoretic approach was proposed in Bateni et al. [2019] for *binary classification*. In recommendation tasks such as click-through rate prediction, categorical variables usually contains hundreds to millions of levels. For example, the Criteo click prediction dataset CriteoLabs [2014] contains about 28 million categorical values, resulting in an embedding layer (interaction between input and first layer) with more than 1 billion parameters. The work in Bateni et al. [2019], proposed an efficient greedy algorithm to cluster levels of a categorical variable in a pre-specified number of clusters that maximizes the mutual information between the clustered feature and target binary label. The utility of the model was illustrated by using the learned clustering (compression) of the categorical features as a pre-processing step to fit a neural network with a reduced number of parameters in the embedding layer, thus using the model for *feature extraction*.

We design a Bayesian linear regression model, with a clustering of the effect of a categorical variable by including a product partition model. Our designed model is flexible to include constraints in the clustering structure. Being a Bayesian approach we can access uncertainty about the model estimated parameters.

Chapter 3

Product Partition Regression Model

In this chapter we present the Product Partition Regression Model (PPRM) for categorical variables. The Product Partition Model (PPM) was first introduced in A. Hartigan [1990]. It assumes that the partition of the data is a random quantity and partitions that are likely to hold are weighted by the data. The PPM model was first used to model spatial data in Hegarty and Barry [2008]. Recently, Teixeira et al. [2015, 2019] proposed a spatial product partition model based on spanning tree (SPPM-ST), as a tool to explore the space of possible partitions.

We start by describing the proposed model (section 3.1). Next, we describe the sampling scheme for model parameters (section 3.2). Last, we perform a complexity analysis (section 3.3) to draw from the full conditional distribution of each parameter.

3.1 Proposed model

Let $G = (V, E)$, be an undirected graph with V vertices $\{v_1, v_2, ..., v_V\}$ and edges $e = (v_i, v_j)$ connecting neighbour vertices. We use the graph G to represent a categorical variable with V levels, where each vertex in G represents a categorical variable and there exists an edge between two vertices if there exists an *an priori* knowledge that the represented categorical level should be clustered.

An illustration of a graph representing a spatial categorical variable is given in figure 3.1, where two districts are connected if they are spatially neighbours. A complete graph could be used where non spatially neighbours nodes would be allowed to be clustered. Even tough a complete graph can lead to a smaller number of estimated cluster obtained clusters would be harder to interpreter and most likely would have a lower predictive performance.

Figure 3.1. A graph representation of a spatial categorical variable.

By the definition above, two categorical levels can be clustered if there exists a path in G connecting their representative vertex. It might be the case that the graph G contains some restriction about the set of possible clustering that yields a graph with multiple connected components (e.g: a graph connecting political parties where left and right aligned parties are not allowed to be clustered). We factorize a graph with with $\mathcal{F}$ components as $G = \{G^1, G^2, ..., G^{\mathcal{F}}\}$, where there is no path between a vertex $u \in G^i$ and $v \in G^j$ if $i \neq j$. For instance, if we were interested on clustering a categorical variable called "job", we could restrict the set of jobs that can be clustered for example healthcare and IT jobs.

Consider n subjects independently observed in G. We denote $\boldsymbol{Y} = (y_1, \ldots, y_n)^T$, as the vector of dependent variables, where y_i is the response for ith subject. Assume that D covariates are measured and let $\boldsymbol{X}_i = (x_{i,1}, \ldots, x_{i,D})$ be the vector of covariates for the ith subject such that $\boldsymbol{X} = (\boldsymbol{X}_1^T, \ldots, \boldsymbol{X}_n^T)^T$ is a $n \times D$, that might include quantitative or dummy encoding of categorical variables with a small number of categories. We also denote $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_D)^T$ as the D-dimensional vector of regression coefficients that are assumed to be equals for all subjects.

To include the effect of the categorical variable that we are interested to cluster, we assume that part of the variability in $\boldsymbol{Y}$ is also due to a vertex's effect. Let $\boldsymbol{Z}_i = (z_{i,1}, \ldots, z_{i,V})$ be a one-hot encoding of a categorical variable indicating the vertex that subject i belongs to, that is, the coordinate $z_{i,r} = 1$ if subject i belongs to vertex r, and is zero otherwise. Let $\boldsymbol{Z}$ be the $n \times V$ matrix $(\boldsymbol{Z}_1^T, \ldots, \boldsymbol{Z}_n^T)^T$. Consider the V-dimensional column vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_V)^T$, where θ_v is the effect of belonging to

vertex v and is shared by all responses $Y_i, i \in v_r$ for $r = 1, ..., V$. We also assume that the vertex effect is additive and that for all subject i there is a linear relationship between y_i and $\boldsymbol{X}_i$ such that:

$$y_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{\theta} + \epsilon_i, \tag{3.1}$$

From equation (3.1), we have that for every observation in a given vertex V, $\mathbb{E}(y_i \mid \boldsymbol{X}_i, \boldsymbol{Z}_i) = \boldsymbol{X}_i\boldsymbol{\beta}+\theta_r$, $r = 1, \ldots, V$. We assume that the errors ϵ_i are independent and identically distributed (iid) as $\mathcal{N}(0, \sigma_y^2)$. The matrix representation of 3.1 is given by $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\theta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2\boldsymbol{I})$.

The clustering of the categorical variable is attained by introducing a clustering structure on $\boldsymbol{\theta}$. Consider the set of labels $I = \{1, 2, ..., V\}$, related to the graph $G = \{G^1, G^2, ..., G^{\mathcal{F}}\}$. Let $\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, ..., \boldsymbol{\pi}^{\mathcal{F}}\}$, be a random partition, $\boldsymbol{\pi}^f = \{G_1^f, G_2^f, ..., G_{c(f)}^f\}$ is a partition of the connected component G^f in $c(i)$ clusters, we have that $\boldsymbol{\pi}$ is a partition of I and G in $c = \sum_{f=1}^{\mathcal{F}} c(f)$ clusters. The number of possible partitions $\boldsymbol{\pi}$ is huge, for a special case where there are no restriction about the partitioning of G, the number of partitions is given by the Bell number B_V (e.g. for $V = 20$ we have $B_{20} \approx 5 \times 10^{13}$).

To make it feasible to explore the space of possible partitions, we use the strategy proposed in Teixeira et al. [2015], to introduce a random spanning tree $\mathcal{T}$ of G in the modelling. Since we can have a disconnected graph G, we extend $\mathcal{T} = \{\mathcal{T}^1, \mathcal{T}^2, ..., \mathcal{T}^{\mathcal{F}}\}$ to be a random spanning forest G, where $\mathcal{T}^f$ is a random spanning tree of the connected component G^f. We say that a partition $\boldsymbol{\pi}$ is compatible with a spanning forest $\mathcal{T}$, denoted by $\boldsymbol{\pi} \prec \mathcal{T}$, if $\boldsymbol{\pi}$ can be obtained by removing edges from $\mathcal{T}$. It follows that $\boldsymbol{\pi} \prec \mathcal{T} \Leftrightarrow \boldsymbol{\pi}^f \prec \mathcal{T}^f, i = 1, 2, ..., \mathcal{F}$. Given $\mathcal{T}^f$, the induced subgraph by the set of vertex in $G_i^f, i = 1, 2, ..., c(f)$ is a connected component in G^f, which we call a cluster.

Given $\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, \ldots, \boldsymbol{\pi}^{\mathcal{F}}\}$ and $\mathcal{T}$ such that $\boldsymbol{\pi} \prec \mathcal{T}$, for every component, we assume that there are common effects for all vertices in cluster G_k^f are equal to a common cluster parameter θ_{G_k}, that is

$$\theta_r = \theta_{G_k^f}, \forall r \in G_k^f, f = 1, 2, ..., \mathcal{F}$$

It is important to note that the dimension of the vector $\boldsymbol{\theta}$ is not changing. Let $\varphi^f : V^f \to \{1, 2, ..., c(f)\}$, be a function that maps the index of a vertex in G^f to its partition id in $\boldsymbol{\pi}^f$, we define $\varphi : V \to \{1, 2, ..., c\}$ as

$$\varphi(i) = \begin{cases} \varphi^1(i) & \text{if } i \in G^1 \\ \varphi^2(i) + c(1) & \text{if } i \in G^2 \\ ... & \\ \varphi^{\mathcal{F}}(i) + \sum_{i=1}^{\mathcal{F}-1} c(i) & \text{if } i \in G^f \end{cases}$$

Therefore, given a partition $\boldsymbol{\pi}$, we have that the coordinates of $\boldsymbol{\theta}$ are given by $\boldsymbol{\theta} = (\theta_{G_{\varphi(1)}}, \theta_{G_{\varphi(2)}}, \ldots, \theta_{G_{\varphi(V)}})^T$. Thus, instead of having V distinct values, $\boldsymbol{\theta}$, has (at most) c distinct values. Consequently, given a partition $\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, ..., \boldsymbol{\pi}^{\mathcal{F}}\}$ we can write that $\boldsymbol{\pi} = \{G_1, G_2, \ldots, G_c\}$. An important assumption is that given $\boldsymbol{\pi} \prec \mathcal{T}$, $\boldsymbol{\theta}_{G_k}, k = 1, ..., c$ are iid.

Let $\boldsymbol{Y}_{G_k}$, $\boldsymbol{X}_{G_k}$ and $\boldsymbol{Z}_{G_k}$ be the $n_k \times 1$ vector of responses, the $n_k \times D$ matrix of covariates and the $n_k \times V$ the vertex indicators, related to subjects belonging to the nodes in cluster G_k , where n_k is the sample size at cluster G_k. The Product Partition Regression Model (PPRM) induced by a spanning tree, denoted by $(\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\beta}, \sigma_y^2) \sim$ PPRM-ST, is the joint distribution of $(\boldsymbol{\theta}, \boldsymbol{\beta}, \sigma_y^2, \boldsymbol{\pi}, \mathcal{T}, \mathcal{D})$ that satisfies the following conditions:

(i) Given $\mathcal{T} = \{\mathcal{T}^1, \mathcal{T}^2, ..., \mathcal{T}^{\mathcal{F}}\}, \boldsymbol{\pi} = \{G_1, G_2, ..., G_c\} \prec \mathcal{T}, \boldsymbol{\theta}, \boldsymbol{\beta}, \sigma_y^2$, the set of observations $\boldsymbol{Y}_{G_k}$ and $\boldsymbol{X}_{G_k}$ are independent for each cluster $k = 1, 2, ..., c$. The joint distribution for $\boldsymbol{Y}_{G_k}$ is given by:

$$\boldsymbol{Y}_{G_k} | \boldsymbol{\pi}, \mathcal{T}, \boldsymbol{\theta}, \boldsymbol{\beta}, \sigma_y^2, \boldsymbol{X}_{G_k}, \boldsymbol{Z}_{G_k} \sim \mathcal{N}(\boldsymbol{X}_{G_k}\boldsymbol{\beta} + \boldsymbol{Z}_{G_k}\boldsymbol{\theta}, \sigma_y^2 \boldsymbol{I})$$

Where $\boldsymbol{Z}_{G_k}\boldsymbol{\theta} = \theta_{G_k}\mathbf{1}_{n_k}$ and $\mathbf{1}_{n_k}$ is a column vector with dimension n_k and its elements equals to 1.

(ii) Given $\mathcal{T}$, $\boldsymbol{\pi} = \{G_1, G_2, ..., G_c\} \prec \mathcal{T}$ and σ_y^2, the common parameters $\theta_{G_1}, \theta_{G_2}, \ldots, \theta_{G_c}$ are iid such that $\theta_{G_k} | \sigma_y^2, \boldsymbol{\pi}, \mathcal{T} \overset{iid}{\sim} \mathcal{N}(\mu_\theta, v_\theta \sigma_y^2)$ for $k = 1, 2, ..., c$, where $\mu_\theta \in \mathbb{R}$ is the prior mean effect common to all areas and v_θ is a positive constant that partially quantifies the prior degree of uncertainty about θ_{G_k}

(iii) Given $\mathcal{T}$ and a hyperparameter ρ, the prior distribution of $\boldsymbol{\pi}$ is the product distribution

$$p(\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, ..., \boldsymbol{\pi}^{\mathcal{F}}\}) = \prod_{f=1}^{\mathcal{F}} p(\boldsymbol{\pi}^f | \mathcal{T}^f)$$

$$p(\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, ..., \boldsymbol{\pi}^{\mathcal{F}}\}\}) = p(\boldsymbol{\pi} = \{G_1, G_2, ..., G_c\})$$

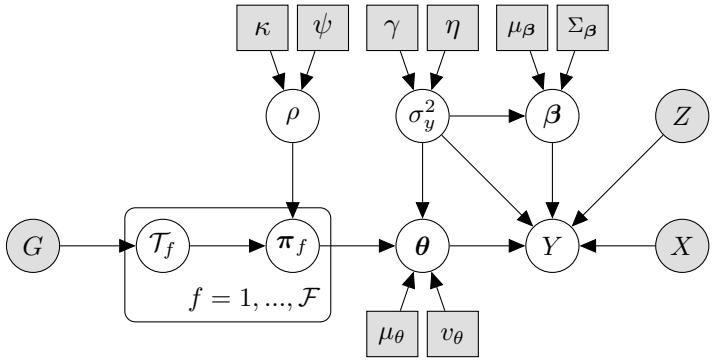


Figure 3.2. Graphical model representation of a PPRM. Hyper-parameters are in a square gray box.

$$p(\boldsymbol{\pi} = \{G_1, G_2, ..., G_c\}|\mathcal{T}) = \frac{\prod_{k=1}^{c} c_{G_k}(\rho)}{\sum_{\mathcal{C}(\mathcal{T})} \prod_{k'=1}^{c} c_{G_{k'}}(\rho)} \; I[\boldsymbol{\pi} \prec \mathcal{T}]$$

where $I[\cdot]$ is the indicator function, $c_{G_k}(\rho) \geq 0$ is the prior cohesion associated to the sub-graph G_k and measure how likely vertices in G_k are clustered *a priori* and $\mathcal{C}(\mathcal{T})$ is the set of all the partitions compatible with the specific spanning forest $\mathcal{T}$ in which we are conditioning.

(iv) Given σ_y^2, the regression coefficients have prior $\boldsymbol{\beta} \mid \sigma_y^2 \sim \mathcal{N}(\mu_{\boldsymbol{\beta}}, \sigma_y^2 \Sigma_{\boldsymbol{\beta}})$, where $\mu_{\boldsymbol{\beta}} \in \mathbb{R}^D$ is the known vector of prior means and $\Sigma_{\boldsymbol{\beta}}$ is a $D \times D$ symmetric, positive definite matrix.

(v) The prior distribution of the model variance σ_y^2 is an inverse-gamma distribution with parameters $\gamma > 0$ and $\eta > 0$, denoted by $\sigma_y^2 \sim \text{IG}(\gamma, \eta)$, with mean $\eta(\gamma - 1)^{-1}$, if $\gamma > 1$, and variance $\eta[(\gamma - 1)^2(\gamma - 2)]^{-1}$, if $\gamma > 2$;

(vi) The prior distribution of a spanning tree $\mathcal{T}$ is uniform on the space of spanning trees of the original graph G, that is $p(\mathcal{T}) \propto 1$.

A graphical representation of the proposed model is given on Figure 3.2.

The clustering effect on the categorical variable $\boldsymbol{\theta}$ is by conditioning its distribution on $\boldsymbol{\pi}$. Note that, if the edge set of the input graph G is empty, a clustering structure is not used, we reduce the model to a standard Bayesian linear regression.

To generate a partition of a graph G we remove its edges such that a set of disjoint connected sub-graphs are obtained. By conditioning on the structure of a spanning forest, this task is substantially simplified. To generate a partition of G into c clusters, we only have to remove $c - \mathcal{F}$ edges among the $V - \mathcal{F}$ edges in the spanning forest. Thus, a natural approach is to set the prior cohesion to be a function of the

edges weights, such as its probability of being removed. With this in mind we assume that edges are independently removed with probability $\rho \in (0,1)$, common to all edges, the prior cohesion of sub-graph G_k is a function of ρ and is given by

$$c(G_k) = \begin{cases} (1-\rho)^{|E(G_k)|-1}\rho & \text{if } k < c \\ (1-\rho)^{|E(G_k)|-1} & \text{if } k = 1 \end{cases} \tag{3.2}$$

where $|E(G_k)|$ is the number of edges not removed in G_k. Hence, the prior expectation for the number of clusters is given by:

$$\mathbb{E}[C] = (V - \mathcal{F})E(\rho) + \mathcal{F}.$$

Thus, if the prior distribution for ρ is centered in high values or is non-informative, *a priori* we are expecting a high number of clusters in G. To complete the model specification, we assume that ρ has a conjugate beta prior distribution with parameters $\kappa > 0$ and $\psi > 0$, denoted by

$$\rho \sim \text{Beta}(\kappa, \psi).$$

Considering the likelihood (i), the prior distributions (ii)-(vi), and the prior cohesion (3.2), the joint distribution $p(\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma_y^2, \boldsymbol{\pi}, \rho, \mathcal{T})$ is proportional to:

$$\prod_{k=1}^{c} \left[p(\boldsymbol{Y}_{G_k} \mid \boldsymbol{\beta}, \theta_{G_k}, \sigma_y^2, \boldsymbol{\pi}, \mathcal{T}) p(\theta_{G_k} \mid \sigma_y^2, \boldsymbol{\pi})\right] p(\boldsymbol{\beta} \mid \sigma_y^2) p(\boldsymbol{\pi} \mid \rho, \mathcal{T}) p(\sigma_y^2) p(\rho). \tag{3.3}$$

Next we describe the MCMC scheme to sample from the posterior distribution.

3.2 Gibbs sampler for the PPRM

The goal is to sample from the joint distribution of the model unknown parameter $\boldsymbol{\Omega} = (\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma_y^2, \boldsymbol{\pi}, \rho, \mathcal{T})$ conditioned on $\mathcal{D} = (\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X}, G)$. The posterior distribution 3.3 does not have a closed formula. To sample from the posterior distribution we use the Gibbs Sampler. Thus, we need the full conditional distribution (fcd) for each model parameter.

Sample parameters with known fcd: The posterior fcd for $\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma_y^2$ and ρ has a closed formula. Thus, samples from their posterior distribution are obtained by Gibbs sampling. Let $\tilde{\boldsymbol{S}} = (\boldsymbol{X}^T\boldsymbol{X} + \Sigma_{\boldsymbol{\beta}}^{-1})^{-1}$, $r_{G_k} = \sum_{i \in G_k}(y_i - \boldsymbol{x}_i\boldsymbol{\beta})$, $\tilde{Y} = (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{\theta})^T(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{\theta})$, $\tilde{\boldsymbol{\beta}} = (\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})^T\Sigma_{\boldsymbol{\beta}}^{-1}(\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})$ and $\tilde{\boldsymbol{\theta}} = \frac{1}{v_\theta}\sum_{k=1}^{c}(\theta_{G_k} - \mu_\theta)^2$ we

have the following fcd:

$$\boldsymbol{\beta} \mid \boldsymbol{\theta}, \sigma_y^2, \mathcal{D} \sim \mathcal{N}(\tilde{\boldsymbol{S}}(\boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\theta}) + \Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}}), \sigma_y^2\tilde{\boldsymbol{S}})$$

$$\theta_{G_k} \mid \boldsymbol{\beta}, \sigma_y^2, \boldsymbol{\pi}, \mathcal{D} \sim \mathcal{N}\Big(\frac{v_\theta r_{G_k} + \mu_\theta}{v_\theta n_k + 1}, \frac{v_\theta \sigma_y^2}{v_\theta n_k + 1}\Big), k = 1, 2, \ldots, c$$

$$\rho \mid \boldsymbol{\pi}, \mathcal{T} \sim \text{Beta}(\kappa + (c - \mathcal{F}), \psi + (V - c))$$

$$\sigma_y^2 \mid \boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}, \mathcal{D} \sim \text{Inv-Gamma}\Big(\gamma + \frac{1}{2}(n + D + c), \eta + \frac{1}{2}(\tilde{Y} + \widetilde{\boldsymbol{\beta}} + \widetilde{\boldsymbol{\theta}})\Big)$$

Sample the tree: To sample $(\mathcal{T} \mid \boldsymbol{\pi}), \boldsymbol{\pi} \prec \mathcal{T}$, we use an extension of the approach used in Teixeira et al. [2015]. Provided that the input graph G can have multiple connected components, instead of sampling a minimum spanning tree, we sample a minimum spanning forest. The weighing scheme for the edges is the same as it was defined in Teixeira et al. [2015]. Assuming the prior cohesion in (3.2), the posterior full conditional distribution of these valid trees is a uniform distribution over the set of trees compatible with the current partition. Let w be the weight of each edge of the input graph G. A pseudo-code to sample a spanning tree (or forest) is given in algorithm 2. For edges connecting vertices in the same cluster a weight is draw from a uniform distribution with lower values (eg: (0, 1)), otherwise a weight is draw from a uniform distribution with higher values (eg: (10, 20)). The idea is that edges connecting vertices in the same component will be selected before edges connect vertices in distinct components.

Algorithm 2: Algorithm to sample a minimum spanning forest

```
Initialize(w)
for (u, v) ∈ E do
    if φ(u) equals to φ(v) then
        w_{u,v} ~ U(0, 1)
    else
        w_{u,v} ~ U(10, 20)
T ← Prim(G, w)
return (T = {T^1, T^2, ..., T^F})
```

Sampling the partition: A random partition $\boldsymbol{\pi} \prec \mathcal{T}$ is represented by an $(V - \mathcal{F})$-dimensional random vector $\mathbf{U} = (U_1, \ldots, U_{V-\mathcal{F}})$, where $U_l = 0$ if the l-th edge is removed from the spanning tree and $U_l = 1$, otherwise. Thus, following Teixeira et al. [2015], samples from the posterior of $\boldsymbol{\pi}$ are obtained sampling from the posterior of $\boldsymbol{U}$ by using a Partially Collapsed Gibbs sampler as follows. Initialize the partition $\boldsymbol{\pi}$ at step 0 by creating a $(V - \mathcal{F})$-dimensional vector assigning 0 for each coordinate.

For example, we can assume that, at step zero, the original graph is not partitioned by setting $\mathbf{U}^{(0)} = (0, \ldots, 0)$. To generate from the posterior of U_l of $\mathbf{U}$, at the step s of the algorithm, we need to calculate the ratio:

$$R_l = \frac{p(U_l = 1|U_1, \ldots, U_{l-1}, U_{l+1}^{(s-1)}, \ldots, U_{V-1}^{(s-1)}, \sigma_y^2, \boldsymbol{\beta}, \boldsymbol{\theta}, \mathcal{T}, \rho, \mathcal{D})}{p(U_l = 0|U_1, \ldots, U_{l-1}, U_{l+1}^{(s-1)}, \ldots, U_{V-1}^{(s-1)}, \sigma_y^2, \boldsymbol{\beta}, \boldsymbol{\theta}, \mathcal{T}, \rho, \mathcal{D})} \tag{3.4}$$

The partitions in the numerator and denominator of (3.4) have a single difference. Assume that, given the partition in the numerator, $U_1, \ldots, U_{l-1}, U_l = 1, U_{l+1}^{(s-1)}, \ldots, U_{V-1}^{(s-1)}$, the cluster G_k contains the nodes connected by edge l. The partition in the denominator assumes that cluster G_k is split in two by disconnecting edge l, creating new clusters G_k^* and G_k^{**}. All other clusters are shared by both partitions. These two partitions induced different configurations for $\boldsymbol{\theta}$. Thus, to make possible the sampling of $\boldsymbol{\pi}$ via Gibbs sampler, we must integrate $\boldsymbol{\theta}$ out in expression (3.4). Numerical stability and faster convergence ratio is attained if we also integrate out ρ.

$$R_l = \frac{\psi + V - c - 1}{\kappa + c - \mathcal{F}} \times \left[\frac{(v_\theta n_{k^*} + 1)(v_\theta n_{k^{**}} + 1)}{(v_\theta n_k + 1)}\right]^{\frac{1}{2}} \times \exp\left\{\frac{-1}{2\sigma_y^2}(-\psi - \phi(G_k) + \phi(G_k^*) + \phi(G_k^{**}))\right\}$$

where $\phi(G_k) = \left(\frac{\mu_\theta}{v_\theta} + r_{G_k}\right)^2 \left(n_k + \frac{1}{v_\theta}\right)^{-1}$, $n_k = n_{k^*} + n_{k^{**}}$, $r_{G_k} = \sum_{i \in G_k}(y_i - \boldsymbol{x}_i\boldsymbol{\beta})$ and c is the number of clusters in the partition in the numerator when computing the ratio R_l. Details derivation of R_l is available in the appendix Section A.

The Ratio R_l is a balance between the graph clustering tendency (first term) and the predictive of the components (second and third terms). The first term depends on how many clusters are in the graph in proportion to the number of edges, it does not depend on the nodes (components) that the edge is connecting. In the other hand, the second term strongly depends on the model predictive of the component that is obtained by keeping the edge versus the model predictive by the components obtained by removing the $l - th$ edge.

We sample from the distribution of U_l by sampling a value $u \sim \mathcal{U}(0, 1)$ with the following accept-reject criterion:

$$U_i = \begin{cases} 1 & \text{if } R_i \geq \frac{u}{1-u} \\ 0 & \text{otherwise} \end{cases}$$

Lastly, at a given iteration s, the binary vector $\boldsymbol{U}$ is initialized, with values

compatible with the previous partition. That is, we assign $U_l = 1$, if the vertices that the edge i connects were in the same cluster in the previous iteration and $U_l = 0$ otherwise.

Algorithm 3: MCMC scheme to draw S samples from the posterior distribution

Input: $\mathcal{D}, S$
Initialize $(\sigma_y^{2(0)}, \boldsymbol{\beta}^{(0)}, \rho^{(0)}, \mathcal{T}^{(0)}, \boldsymbol{\pi}^{(0)}, \boldsymbol{\theta}^{(0)})$;
for *s = 1 to S* **do**
  $\boldsymbol{\beta}^{(s)} \sim p(\boldsymbol{\beta}|\boldsymbol{\theta}^{(s-1)}, \sigma_y^{2(s-1)}, \mathcal{D})$;
  $\mathcal{T}^{(s)} \sim p(\mathcal{T}|\boldsymbol{\pi}^{(s-1)}, \mathcal{D})$;
  $\boldsymbol{\pi}^{(s)} \sim p(\boldsymbol{\pi}|\boldsymbol{\beta}^{(s)}, \sigma_y^{2(s-1)}, \mathcal{T}^{(s)}, \mathcal{D})$;
  $\rho^{(s)} \sim p(\rho|\boldsymbol{\pi}^{(s)}, \mathcal{T}^{(s)}, \mathcal{D})$;
  for *k = 1 to* $c^{(s)}$ **do**
    $\theta_{G_k} \sim p(\theta_{G_k}|\sigma_y^{2(s-1)}, \boldsymbol{\beta}^{(s)}, \boldsymbol{\pi}^{(s)}, \mathcal{D})$;
    for $v \in G_k$ **do**
      $\theta_v^{(s)} = \theta_{G_k}$;
  $\sigma_y^{2(s)} \sim p(\sigma_y^2|\boldsymbol{\beta}^{(s)}, \boldsymbol{\theta}^{(s)}, \mathcal{D})$;

3.3 Computational complexity analysis

In this section, we provide a complexity analysis to draw from the fcd of each model parameter along with some implementation details. Algorithm 3, shows the order we sample from the posterior distribution. The computational complexity for each operation is at table 3.1.

Table 3.1. Computational complexity for each iteration

Method	Draw $\boldsymbol{\beta}$	Draw $\mathcal{T}$	Draw $\boldsymbol{\pi}$	Draw ρ	Draw $\boldsymbol{\theta}$	Draw σ_y^2
Complexity for iteration s	$O(n(D+V)+D^3)$	$O(\|E\|\log(V))$	$O(V^2)$	$O(1)$	$O(V^2)$	$O(n(D+V)+D^2)$

3.3.1 Draw $\boldsymbol{\beta}$

To draw from the fcd of $\boldsymbol{\beta}$ at iteration s, we have to compute its fcd mean and covariance. Since $\tilde{S}$ and $\Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}}$ are constant during the MCMC, we cache their values before starting the MCMC, if we do not cache them, it would add an additional $O(nD^2+D^3)$ operations per iteration. To compute the mean $\tilde{S}(X^T(Y-Z\theta)+\Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}})$,

we have 3 matrix multiplications with a total cost of $O(n(V+D)+D^2)$. To compute the co-variance $\sigma_y^2\tilde{S}$, we need $O(D^2)$ operations. We use the R package MASS, [1] to sample from a D-variate normal distribution. The package uses the eigendecomposition method, which has a cost of $O(D^3)$ operations. Thus, the total cost to draw from the posterior of $\boldsymbol{\beta}$ is $O(n(V+D)+D^3)$.

We can cache the partial residuals for each level of the categorical variable we are interested to cluster, this caching significantly improves the model performance. Before starting the MCMC chain, we can pre-process the input data to create a list $L_x = [\boldsymbol{X}_{r_1}, \boldsymbol{X}_{r_2}, ..., \boldsymbol{X}_{r_R}]$, where $\boldsymbol{X}_{r_i}$ contains the design matrix of the elements that belongs to the i-th level, similarly we can create L_y and L_n that contains the response variable and number of samples for each level. Next we create a list of partial residuals $L_v = [(\boldsymbol{Y}_{r_1} - \boldsymbol{X}_{r_1}\boldsymbol{\beta}), (\boldsymbol{Y}_{r_2} - \boldsymbol{X}_{r_2}\boldsymbol{\beta}), ..., (\boldsymbol{Y}_{r_V} - \boldsymbol{X}_{r_V}\boldsymbol{\beta})]$ for each level, the total cost to pre-process the data is $O(nRD)$. The cached partial residuals are invalidated on every iteration after sampling $\boldsymbol{\beta}$, thus we have to update their values, the update is done in $\sum_{i=1}^{V}(n_{r_i}D + n_{r_i}) = O(nD)$ operations. Note that it does not increases the order of complexity to sample the model betas.

3.3.2 Draw $\mathcal{T}$

To draw a spanning tree $\mathcal{T}$ at iteration s, we assign a weight for each edge in G following the rule in Algorithm 3.2, which is performed in $O(|E|)$ operations. Next, to find a MST we use R package igraph. [2] The library implements the Prim's algorithm with a binary heap, which has a time complexity of $O(|E|\log(V))$.

3.3.3 Draw $\boldsymbol{\pi}$

To draw a partition $\boldsymbol{\pi}$ at iteration s, we start by initializing the partition as the current spanning tree, which is done in $O(V)$ operations, since the spanning tree has at most $V-1$ edges. Next, we initialize the partition by keeping or deleting its edges. This is accomplished in at most V^2 operations because we can remove at most V edges from a partition that is represented by an adjacent list graph.

After initializing the partition at the given iteration we need to compute the ratio R_i for each edge in the spanning tree. For this we have to find the set of vertices that belongs to G_k^* and G_k^{**}, which is done by performing two DFS in the spanning tree, thus it is achieved in $O(V)$ operations. Next we need to compute the residuals $r_{G_k^*}$ and $r_{G_k^{**}}$. If we use memoization to cache the partial residuals, we can find $r_{G_k^*}$ in linear

[1] `https://www.rdocumentation.org/packages/MASS/versions/7.3-51.4/topics/mvrnorm`
[2] `https://www.rdocumentation.org/packages/igraph/versions/1.2.4/topics/mst`

time in V. If we do not use the memoization we have to subset the data that belongs to G_k^* and G_k^{**} and compute its residuals, which is done in $O(nDR)$.

After drawing the partition, we update the number of clusters and clustering id of each vertex in G, that is achieved in $O(V)$ operations by performing a DFS in the partition. Thus, the total cost to sample a partition is $O(V^2)$ with memoization, if we do not use memoization the cost should be substituted by $O(V^2nD)$ per iteration.

3.3.4 Draw ρ

To sample ρ at iteration s, we have to lookup the values of n, V and c, that is a constant operation. We use the r-base function rbeta to sample a single value from a beta distribution. That function uses an accept rejection criterion, [3] thus we can assume is it is done in a constant number of operations. Thus, the time complexity to sample ρ at iteration s is $O(1)$.

3.3.5 Draw $\boldsymbol{\theta}$

To draw $\boldsymbol{\theta}$ at iteration s, we need to compute the fcd mean and variance for each cluster. The posterior mean and variance can be computed in $O(V)$ operations using the cache or in $O(nDR)$ without the cache. Since the number of clusters is always smaller or equal to the number of vertices in G, we have a time complexity of $O(V^2)$ to compute the fcd mean for all clusters using cache or $O(V^2nD)$ without cache. Next, in a linear time in V we assign $\theta_v = \theta_{G_k}, \forall v \in G_k$ for $k = 1, ...c$. Thus, the total cost to sample $\boldsymbol{\theta}$ is $O(V^2)$.

3.3.6 Draw σ_y^2

To draw σ_y^2 at iteration s, we need to compute the sum of squared of the residuals $\tilde{Y} = (\boldsymbol{Y} - \boldsymbol{X\beta} - \boldsymbol{Z\theta})^T(\boldsymbol{Y} - \boldsymbol{X\beta} - \boldsymbol{Z\theta})$ which is done in $O(n(D+V))$ operations, $\tilde{\boldsymbol{\beta}} = (\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})^T\Sigma_{\boldsymbol{\beta}}^{-1}(\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})$ and $\tilde{\boldsymbol{\theta}} = v_\theta^{-1}\sum_{i=1}^{C}(\boldsymbol{\theta}_{G_i} - \mu_\theta)^2$ which are computed in $O(D^2+V)$ operations. To sample from a single value from a inverse gamma distribution we use an wrapper function to an r-base function to sample from a gamma distribution. It uses an accept rejection criterion method that we can assume to be done in a constant number of operations. Thus, the total complexity per iteration is $O(n(D+V)+D^2)$ if $\Sigma_{\boldsymbol{\beta}}^{-1}$ is cached or $O(n(D+V)+D^3)$ if it is not cached.

[3] `https://www.rdocumentation.org/packages/stats/versions/3.5.3/topics/Beta`

Chapter 4

Experimental Evaluation

We evaluate our model using the Munich house rent dataset used in Gertheiss and Tutz [2011] and Inside Airbnb datasets for different locations around the globe. Their examples are addresses and the response variable indicates its monetary market value. The datasets have a spatial categorical variable indicating the neighbourhood/district/region where a sample is located. It is widespread knowledge among real estate agents that the three most important factors determining a house price are location, location, and location. For all datasets, we analyze the clustering of this spatial predictor by restricting the set of possible clustering to spatially contiguous regions. This is achieved by creating an input graph G, where each level of the categorical variable is mapped to a node and an edge between two nodes is created if they are spatially adjacent. An example of a input graph is given in figure 4.1.

4.1 Munich house rent dataset

The Munich rent dataset is available in the V package catdata Gunther Schauberger [2014] and it was used previously by Gertheiss and Tutz [2011]. It contains 2053 observations collected in sampled personal interviews from residents in Munich in 2003. There is a spatial variable indicating in which of the 25 districts the home is located in addition to 9 predictor variables. We transformed the predictors to categorical variables as described in Gertheiss and Tutz [2011]. The response variable is the monthly rent in Euros.

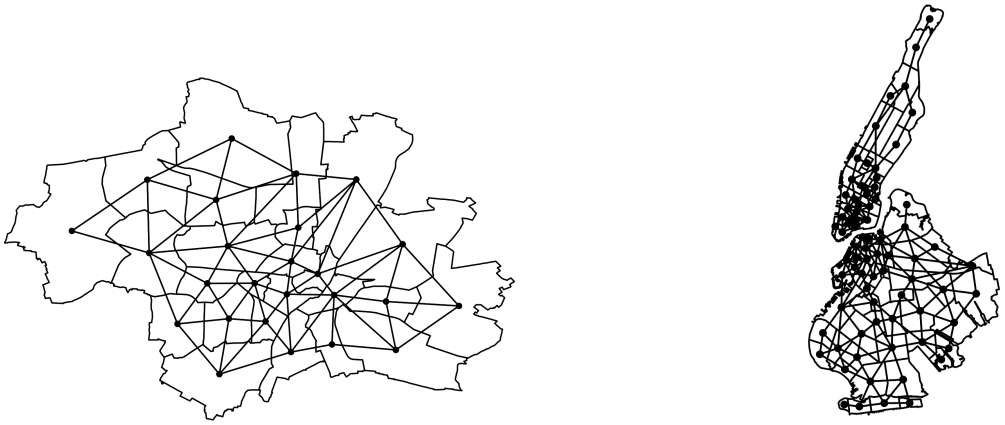


Figure 4.1. A neighbourhood graph for Munich (left) and New York (right) dataset. Two districts/neighbourhood are connected if they are spatially neighbours. Each district corresponds to a level of the categorical variable we are interested in clustering. For the New York dataset we have 2 connected components, neighbourhoods in the Manhattan island do not connect to neighbourhoods in Brooklyn.

4.2 Inside Airbnb Datasets

Airbnb is an online accommodation platform, where people can list their property or spare room for renting. We use data from Inside Airbnb,[1] an independent platform that compiles Airbnb listings by geographic areas. The data contains publicly available information such as the price per night of stay, number of reviews, amenities and neighbourhood of the places available for rent. The exact latitude-longitude coordinates of a listing is masked by Airbnb. The reported location in the dataset is a region containing the address. The difference is within 450 feet (150 meters) between the actual address and the region centroid.[2]

We selected 10 popular locations on Airbnb (Amsterdam, Barcelona, Chicago, Hawaii, London, Montreal, New York, San Francisco, Seattle, Toronto). We preprocessed the dataset by selecting only listings with at least one review and rentals of a full house or apartment. We removed outliers using a conservative approach: farther than 3 Median Absolute Deviation (MAD) score criterion as suggested in Leys et al. [2013]. We transformed all the features into categorical ones (a detailed description is available in the appendix section A.1.1). We also used the spatial neighbourhood that is provided by Inside Airbnb to create the input graph G, connecting adjacent regions. The response variable is logarithm of the price per night of stay.

[1] http://insideairbnb.com/

[2] `http://insideairbnb.com/about.html`

4.3 Posterior Predictive Inference

To evaluate the model fit, we consider the model predictive performance. Let $\tilde{\boldsymbol{X}}_i$ and $\tilde{\boldsymbol{Z}}_i$ be, respectively, the covariates and the vertex indicator related to a future response $\tilde{y}_i$. Assume the the future observation is independent of the past ones, given the parameters and covariates. To estimate $\tilde{y}_i$, we consider $\mathbb{E}(\tilde{y}_i \mid, \tilde{\boldsymbol{X}}_i, \tilde{\boldsymbol{Z}}_i)$, the mean of the posterior predictive distribution, which is given by:

$$\mathbb{E}(\tilde{y}_i \mid, \tilde{\boldsymbol{X}}_i, \tilde{\boldsymbol{Z}}_i) = \int \mathbb{E}(\tilde{y}_i \mid \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Z}}, \boldsymbol{\Omega}) P(\boldsymbol{\Omega} \mid \mathcal{D}) d\boldsymbol{\Omega}. \tag{4.1}$$

It is intractable to compute 4.1, thus this is approximated by using samples collected in the MCMC chain:

$$\widehat{y}_i = \frac{1}{S} \sum_{s=1}^{S} \widehat{y}_i^{(s)},$$

where

$$\widehat{y}_i^{(s)} \sim \mathcal{N}(\tilde{\boldsymbol{X}}_i \boldsymbol{\beta}^{(s)} + \tilde{\boldsymbol{Z}}_i \boldsymbol{\theta}^{(s)}, \sigma_y^{2(s)}).$$

4.4 Experimental set-up

We use a 5-fold cross-validation to evaluate the models predictive accuracy. Some of cities of the Airbnb dataset have neighbourhoods with less than 5 samples. To avoid to discarding these neighbourhoods and its samples, we perform a 5-fold-cv with a modification.

We denote de dataset $D = D_{fixed} \bigcup D_{rotation}$, where D_{fixed} contains samples that belong to a region with less than 5 samples and $D_{rotation}$, contains all the remaining samples. The rotation dataset is split in five $D^1_{rotation}, D^2_{rotation}, ..., D^5_{rotation}$ (sampling is stratified by the categorical variable we are interested to cluster). We create five training and validation datasets as follows $D^i_{train} = D_{fixed} \bigcup (\bigcup_{i \neq j} D^j_{rotation})$ and $D^i_{val} = D^i_{rotation}$, for $1 \leq i \leq 5$.

Table 4.1 shows the number of samples, regions and connected components for each dataset.

The model predictive performance is estimated using the mean squared error (MSE). Let n_i be the number of samples in the validation folder D^i_{val}, the $MSE_i, i = 1, \ldots, 5$ is computed as defined in 4.2, where the model parameters are estimated using

Table 4.1. Summary for each dataset. N is the number of samples, N_{fixed} number of fixed samples, V number of levels in the categorical variable of interest, $\mathcal{F}$ number of components in the input graph and Date compiled is the data in which the inside Airbnb was compiled.

Dataset	N	N_{fixed}	V	$\mathcal{F}$	Date compiled
Munich	2053	0	25	1	NA
Amster.	10741	0	22	2	08 April, 2019
Barcelona	5043	27	64	1	10 April, 2019
Chicago	2484	46	64	1	12 March, 2019
Hawaii	4518	4	30	6	03 May, 2019
London	24573	0	33	1	09 April, 2019
Montreal	7103	15	30	1	13 April, 2019
New York	11695	12	77	2	03 May, 2019
San Fran.	1920	1	35	1	03 May, 2019
Seattle	2878	24	88	2	15 April, 2019
Toronto	4340	83	136	1	08 April, 2019

D^i_{train}. We report the average MSE and standard deviation of mean squared obtained across the five folds.

$$MSE_i = \frac{1}{n_i} \sum_{y_i \in D^i_{val}} (y_i - \hat{y}_i)^2 \tag{4.2}$$

4.5 Baselines

We compare the predictive accuracy of our model against four different baseline competitors: (a) the usual Bayesian Linear Regression (*Bayes*), with dummy coding of the categorical variable using one level for each region (without any clustering); (b) the Lasso penalization strategy from Gertheiss and Tutz [2011]; (c) a finite mixture regression model (Fusion) from Pauger and Wagner [2019] to cluster effects of a categorical variable; (d) the CatBoost state of the art gradient boosting model from Prokhorenkova et al. [2018], which proposed a novel approach to compute the target statistics. We also compare the estimated number of clusters against the Lasso penalization method to fuse effects of a categorical variable.

Except for the Bayes model that we implemented, we used openly available packages in *R* for Lasso (Oelker [2015]), Fusion (Pauger et al. [2019]) and CatBoost (Prokhorenkova et al. [2017]). Details of fine tuning for each model are available in the appendix material (A.1.2).

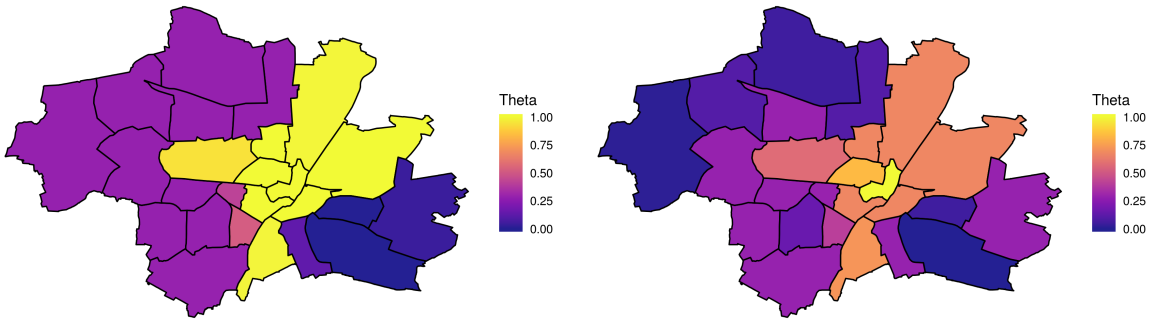


Figure 4.2. Posterior means for the vertices effects in each Munich district using PPRM (left) and Lasso (right). Effects are normalized using the min-max normalization.

4.6 Hyperparameters and Model Initialization

To analyze the data we consider the prior distributions previously mentioned with weakly informative prior hyperparameters $\mu_{\boldsymbol{\beta}} = \mathbf{0}$, $\mu_\theta = 0$, $v_\beta = 1 \times 10^4$ $v_\theta = 1 \times 10^4$, $\gamma = 0.1$, $\eta = 0.1$, $\kappa = 4$ and $\psi = 6$.

We initialize the MCMC chain for each parameter with valid values as follows: $\sigma_y^{2(0)} = 1$, $\boldsymbol{\beta}^{(0)} \sim \mathcal{N}(\mathbf{0}, \sigma_y^{2(0)}\boldsymbol{I})$, $\rho^{(0)} \sim Beta(\kappa, \psi)$. To initialize the spanning tree $\mathcal{T}^{(0)}$, for each edge in the graph G we sample a weight from a uniform distribution $\mathcal{U}(0, 1)$ and we run the Prim's algorithm to find the minimum spanning tree. We start the partition $\boldsymbol{\pi}^{(0)}$ with V components, that is, letting $U_l = 0$, $l = 1, \ldots, V - \mathcal{F}$. We initialize ${\theta_i}^{(0)} \sim \mathcal{N}(0, \sigma_y^{2(0)})$ for $i = 1, 2, ..., V$.

We run chains of size $20,000$, discarding the first $10,000$ iterations as the burn-in period and taking lag 5 sampled values to avoid strong auto-correlation, thus obtaining an sample size of $2,000$ samples.

4.7 Results

Figure 4.2 (left) shows the estimated posterior mean coefficients of the categorical clustering for each district provided by our method in the Munich dataset. We learn that the location effect can be summarized into 3 spatial clusters, with the 95% highest posterior density interval (HPD) for the number of cluster collapsed at [3, 3]. The model is uncertain about the clustering of some districts (e.g: large district in the center), thus their vertex effect is a mix of the vertex effect of its possible clustering). This is in sharp contrast with the 13 clusters estimated by the Lasso method. (see Figure 4.2

(right)). It is important to note that even though, our model is indicating a smaller number of parameters (3) in comparison to the frequentist approach Lasso (13) there is no substantial difference in its MSE, despite Lasso being tuned for best predictive accuracy in the Munich dataset.

Table 4.2 shows the results of the MSE (with their standard errors) for all methods. PPRM presents the best results in the linear models (Bayes, Lasso, Fusion). In most datasets CatBoost had smaller MSE, but the difference between the methods are not significant at 95% level using the non-parametric Mann-Whitney U-test. Therefore, in practice, there is no empirical evidence of real difference between the algorithms in terms of predictive power. Our intuition for the MSE to be similar for PPRM, Lasso and Fusion is that all of them are linear models and the obtained residuals were normally distributed indicating that a linear model is fitting the data set well. The only difference between the linear models are in the number of parameters.

Table 4.2. Mean Square error and standard deviation (MSE/SD), 95% highest posterior density interval for the number of clusters under PPRM (HPD-C) and the Lasso estimated number of clusters (LaC) for all different methods and Munich house rent (Row 1) and Inside Airbnb (Rows 2-7) datasets. Relative MSE difference against PPRM model in bracket.

Dataset	PPRM	Bayes	Lasso	Fusion	CatBoost	HPD-C	La-C
Munich	4.044/0.21	4.053/0.198 (0.22%)	4.06/0.21 (0.4%)	4.229/0.254 (4.57%)	4.052/0.224 (0.2%)	[3, 3]	13
Amster.	0.076/0.001	0.076/0.001 (0%)	0.076/0.001(0%)	0.076/0.003 (0%)	0.075/0.002 (-1.32%)	[7,8]	7
Barcel.	0.14/0.006	0.138/0.005 (-1.43%)	0.141/0.006(0.71%)	0.14/0.004 (0%)	0.129/0.005 (-7.86%)	[3,5]	17
Chicago	0.139/0.005	0.137/0.006 (-1.44%)	0.146/0.008(5.04%)	0.142/0.01 (2.16%)	0.135/0.008 (-2.88%)	[4,6]	7
Hawaii	0.142/0.008	0.141/0.008 (-0.7%)	0.142/0.008(0%)	0.142/0.003 (0%)	0.137/0.004 (-3.52%)	[9,11]	17
London	0.081/0.001	0.081/0.001 (0%)	0.081/0.001(0%)	0.081/0.002 (0%)	0.08/0.001 (-1.23%)	[11,13]	13
Montreal	0.117/0.002	0.116/0.002 (-0.85%)	0.117/0.005(0%)	0.118/0.006 (0.85%)	0.115/0.001 (-1.71%)	[4,6]	11
New York	0.079/0.003	0.079/0.003 (0%)	0.082/0.003(3.8%)	0.079/0.003 (0%)	0.08/0.003 (1.27%)	[9,12]	15
San Fran.	0.103/0.008	0.103/0.008 (0%)	0.111/0.01(7.77%)	0.105/0.009 (1.94%)	0.108/0.01 (4.85%)	[3,5]	4
Seattle	0.109/0.006	0.108/0.004 (-0.92%)	0.124/0.01(13.76%)	0.113/0.004 (3.67%)	0.11/0.006 (0.92%)	[6,7]	5
Toronto	0.1/0.009	0.101/0.01 (1%)	0.11/0.005(10%)	0.102/0.004 (2%)	0.103/0.01 (3%)	[4,6]	3

Table 4.3 shows the effect of including the categorical predictor for each dataset. The first row (1) displays the MSE/SD for a Bayesian Linear Regression with the categorical predictor and the second row (2) displays results without the predictor. Errors were estimated using a paired 5-fold-cv with same set of hyperparameters. The feature impact on MSE has distinct impacts in each dataset, the lowest being the Munich (1.6%) and the highest New York (45%).

It is intuitive to cluster the effects of the categorical predictor using the marginal distribution as in Bateni et al. [2019]. In figure 4.3 (right), is not clear that there is a spatial clustering in the effect of the Seattle neighbourhood. On the other hand, in 4.3 (upper/lower left), when controlling for effects such a number of bathrooms, bedrooms and obtaining the partial residuals (that is $\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}$) a spatial clustering

Table 4.3. Predictive accuracy of a Bayesian Linear Regression (1) with the categorical feature (2) without the categorical feature and relative change of not including the feature Δ.

	Munich	Amsterdam	Barcelona	Chicago	Hawaii	London	Montreal	New York	San Fran.	Seattle	Toronto
(1)	4.053/0.198	0.076/0.001	0.138/0.005	0.137/0.006	0.141/0.008	0.081/0.001	0.116/0.002	0.079/0.003	0.103/0.008	0.108/0.004	0.101/0.01
(2)	4.118/0.195	0.084/0.002	0.147/0.007	0.157/0.009	0.178/0.009	0.098/0.001	0.126/0.005	0.115/0.001	0.113/0.01	0.126/0.009	0.113/0.006
Δ	1.6%	10.53%	6.52%	14.6%	26.24%	20.99%	8.62%	45.57%	9.71%	16.67%	11.88%

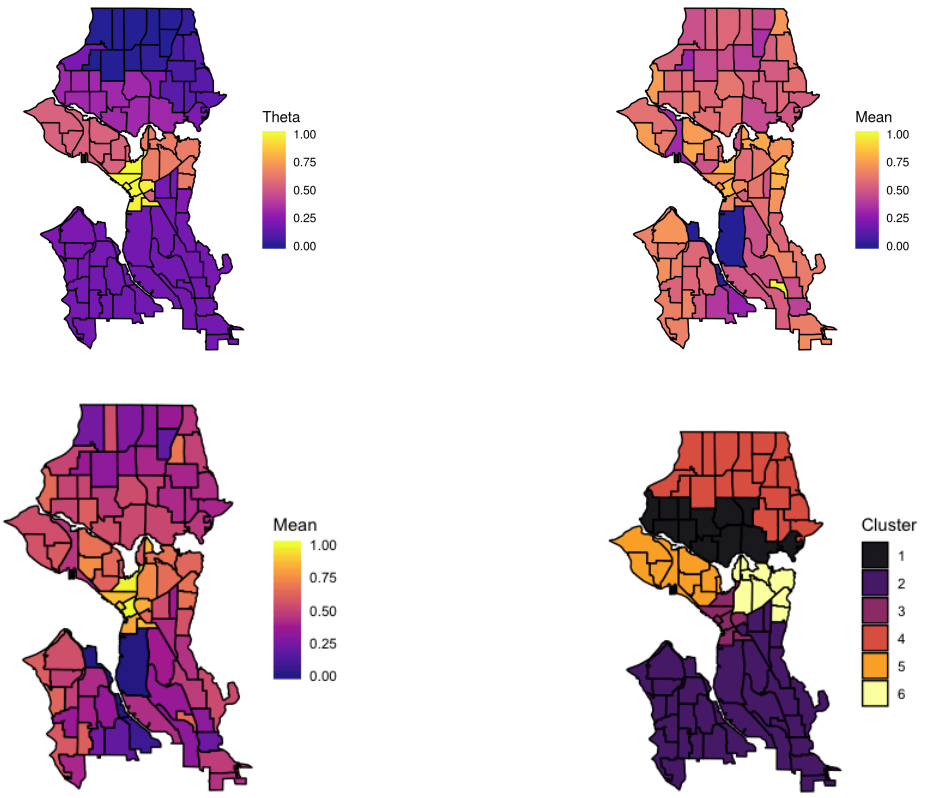


Figure 4.3. Posterior means for the vertices effects in each Seattle neighbourhood using PPRM (upper left) and mean log price per night of stay (upper right). Mean partial residual ($Y - \boldsymbol{X}\boldsymbol{\beta}$) for each region (lower left) and its respective clustering (lower right) obtained in a MCMC iteration.

pattern appears. From the learned model clustering 4.3 (lower right), we observe that there is a distinct effect for the most touristic area (city centre) and its surround areas.

Examples in which there was a significant difference between our model and Lasso are show in 4.4. We have our model estimated coefficients (left) and Lasso (right). In the first row, we have the Lasso coefficients for the Toronto dataset. The method had numerical instability because of the adaptive weights and it is indicating 3 clusters having a predictive performance similar as not including the categorical predictor.

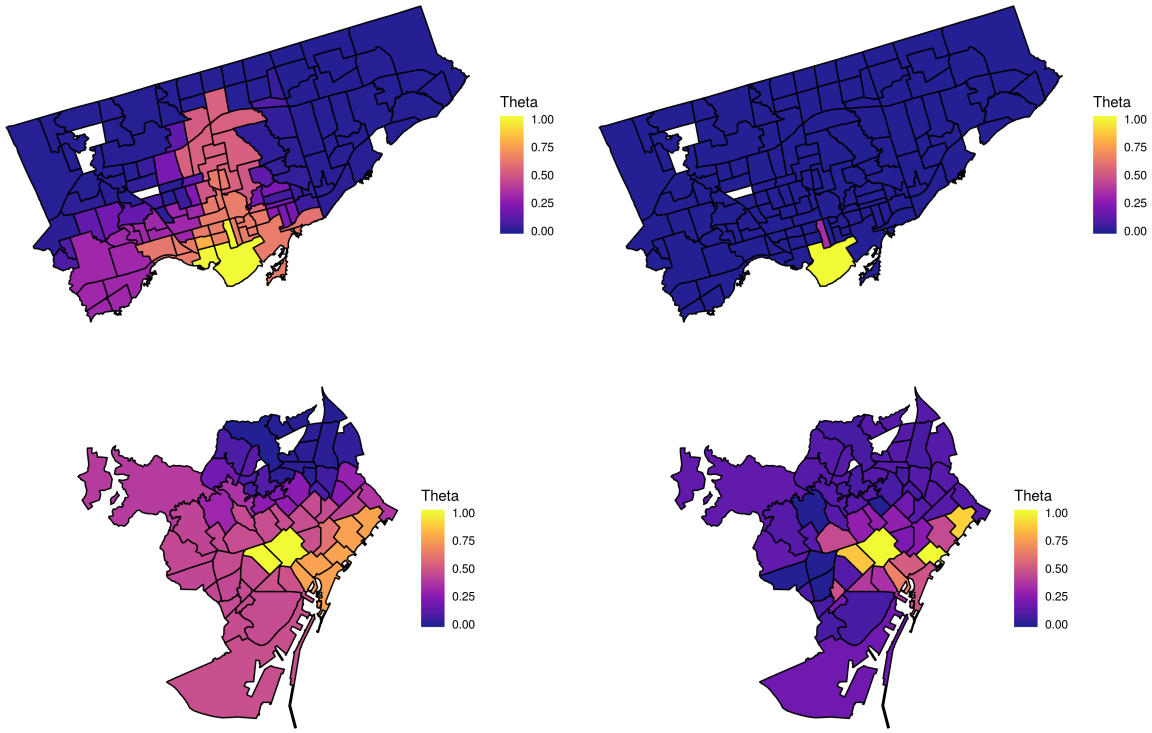


Figure 4.4. Posterior means for the vertices effects in each Toronto using PPRM (left) and Lasso coefficients (right).

In the second row, we have the coefficients obtained in the Barcelona dataset. The coefficients obtained by our model are spatially contiguous and easy to interpret (yellow color coefficients are Barcelona downtown, orange close to the beach, blue are residential area farther from touristic areas), whereas the coefficients obtained by Lasso are not spatially contiguous and it is hard to interpret.

Overall, our model was able to reduce the dimensionality of the categorical variable, by clustering levels with similar impact in the response variable, while carrying uncertainty about others coefficients. From the experimental results, it was able to maintain the predictive performance even when indicating a reduction in the number of variables (e.g: NY, Seattle, Toronto). A priori clustering knowledge was introduced by considering to cluster only spatially neighbour regions. This achieve our goals to reduce the dimensionality of the model in a data driven way without losing the predictive performance and including a prior knowledge about the clustering structure. More examples of clustering obtained is available in the appendix A.2.

Chapter 5

Conclusion

A generative model for the common practical problem of dealing with categorical variables with a large number of levels in predictive models was proposed. We illustrated our approach in the specific context of regression models, but it can be adapted to other contexts such as supervised classification. We represent the possible clustering of the covariate levels as a graph partitioning problem. The graph can be a complete graph if no structure is assumed or a more constrained one, as the spatial case in used in this project. A complete graph might lead to a smaller number of estimated clusters, but obtained clusters would be harder to interpret. Our PPRM model is especially suited when we have prior knowledge of how the levels might be clustered, as in the real estate examples we used. This prior knowledge is based on domain knowledge. PPRM showed to perform well, even without parameter tuning. The posterior density for the number of clusters were concentrated ex: Amsterdam $[7, 8]$. In comparison to Lasso and Fusion, our model achieved a better MSE in all datasets. Most importantly, it produces interpretable results in terms of the effect of each cluster of covariate levels. Additionally, it provides an uncertainty measure in terms of the posterior distribution for those effects as well as the clustering.

In contrast with methods such as [Prokhorenkova et al., 2018], we avoid the encoding of the categorical data using a target coding approach, which may overfit due to the target leakage problem or underfit due to prediction shift. The Lasso methods proposed by Gertheiss and Tutz [2011]; Oelker et al. [2014] do not provide inference about the clustering structure, in contrast with PPRM. Additionally, its performance strongly depends on adaptive weights, which uses a w_{ij} weight equals to inverse of the absolute difference between the ordinary least squares coefficients. When the difference is very small, numerical instabilities plague the calculation, a problem we faced in our Airbnb datasets.

For a spatial categorical variable, which is well known to have a clustering tendency, our model showed to be able to obtain the same predictive power than other competing methods but it provided much more interpretable results (in terms of the resulting spatial clusters) and it significantly reduced the number of parameters (from 77 to [9, 12] parameters in New York considering the 95% HPD. That is a reduction of about 80% without compromising the predictive performance.

There are several directions for future work. The most promising are to extend the model for binomial and multinomial type of response. These extensions can be included in the MCMC scheme by using data augmentation strategy [Albert and Chib, 1993; Holmes and Held, 2006]. Also, a possible extension is to handle outliers in the data, which leads coefficients to be biased. This extension can be achieved by allowing the distribution of the residuals to have a more flexible distribution other than the normal. For example it is possible to use data augmentation to extend the residuals to a normal independent class of distributions [Lange and Sinsheimer, 1993], that includes t-distribution, which has wider tails. Last, an analysis using the proposed model for feature extraction to be used for more complex model would aggregate value to the model by showing an end-to-end machine learning task. For example it would be useful for the Kaggle community. To achieve this, we need to select a strategy to post- process the model to find a clustering from the MCMC chain.

Bibliography

A. Hartigan, J. (1990). Partition models. *Communications in Statistics-theory and Methods - COMMUN STATIST-THEOR METHOD*, 19:2745–2756.

Albert, J. H. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679.

Barry, D. and Hartigan, J. A. (1992). Product partition models for change point problems. *Ann. Statist.*, 20(1):260--279.

Barry, D. and Hartigan, J. A. (1993). A bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309--319. ISSN 01621459.

Bateni, M., Chen, L., Esfandiari, H., Fu, T., Mirrokni, V. S., and Rostamizadeh, A. (2019). Categorical feature compression via submodular optimization. *CoRR*, abs/1904.13389.

Bondell, H. D. and Reich, B. J. (2009). Simultaneous factor selection and collapsing levels in anova. *Biometrics*, 65(1):169--177.

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov Chain Monte Carlo*. CRC press.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785--794, New York, NY, USA. ACM.

CriteoLabs (2014). Display advertising challenge. `https://www.kaggle.com/c/criteo-display-ad-challenge`.

Dyk, D. A. V. and Park, T. (2011). Partially collapsed gibbs sampling and path-adaptive metropolis–hastings in high-energy astrophysics.

Ferreira, H. (2019). Dealing with categorical features in machine learning. https://www.kdnuggets.com/2019/07/categorical-features-machine-learning.html. Accessed: 2019-09-14.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition.

Geman, S. and Geman, D. (1984). Geman, d.: Stochastic relaxation, gibbs distribution, and the bayesian restoration of images. ieee trans. pattern anal. mach. intell. pami-6(6), 721-741. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741.

Gertheiss, J. and Tutz, G. (2011). Sparse modeling of categorial explanatory variables. *Annals of Applied Statistics - ANN APPL STAT*, 4.

Gunther Schauberger, G. T. (2014). *catdata: categorical data*.

Hallac, D., Leskovec, J., and Boyd, S. (2015). Network lasso: Clustering and optimization in large graphs. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 387--396. ACM.

Hartigan, J. (1990). Partition models. *Communications in Statistics - Theory and Methods*, 19(8):2745–2756.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Hegarty, A. C. and Barry, D. (2008). Bayesian disease mapping using product partition models. *Statistics in medicine*, 27 19:3868–93.

Holmes, C. C. and Held, L. (2006). Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Anal.*, 1(1):145--168.

Jerry (2013). How to handle categorical predictors with too many levels? https://stats.stackexchange.com/questions/67938/how-to-handle-categorical-predictors-with-too-many-levels. Accessed: 2019-05-12.

John Mount, N. Z. (2019). vtreat: A statistically sound 'data.frame' processor/conditioner. https://cran.r-project.org/web/packages/vtreat/index.html. Accessed: 2019-05-12.

Lange, K. and Sinsheimer, J. S. (1993). Normal/independent distributions and their applications in robust regression. *Journal of Computational and Graphical Statistics*, 2(2):175--198. ISSN 10618600.

Leys, C., Ley, C., Klein, O., Bernard, P., and Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49:764â766.

McGinnis, W. (2019). Category encoders. `http://contrib.scikit-learn.org/categorical-encoding/index.html`. Accessed: 2019-05-12.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Micci-Barreca, D. (2001). A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter*, 3(1):27--32.

Oelker, M.-R. (2015). *gvcm.cat: Regularized Categorical Effects/Categorical Effect Modifiers/Continuous/Smooth Effects in GLMs*. R package version 1.9.

Oelker, M.-R., Gertheiss, J., and Tutz, G. (2014). Regularization and model selection with categorical predictors and effect modifiers in generalized linear models. *Statistical Modelling*, 14(2):157–177.

Pauger, D., Leitner, M., Wagner, H., and Malsiner-Walli, G. (2019). *effectFusion: Bayesian Effect Fusion for Categorical Predictors*. R package version 1.1.1.

Pauger, D. and Wagner, H. (2019). Bayesian effect fusion for categorical predictors. *Bayesian Anal.*, 14(2):341--369.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2017). Catboost: unbiased boosting with categorical features. R package version 0.14.2.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2018). Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, pages 6638--6648.

Roberts, G. and Smith, A. (1994). Simple conditions for the convergence of the gibbs sampler and metropolis-hastings algorithms. *Stochastic Processes and their Applications*, 49(2):207 – 216. ISSN 0304-4149.

shadowtalker (2017). Principled way of collapsing categorical variables with many levels? `https://stats.stackexchange.com/questions/146907/principled-way-of-collapsing-categorical-variables-with-many-levels`. Accessed: 2019-05-12.

Smith, A. N. and Allenby, G. M. (2019). Demand models with random partitions. *Journal of the American Statistical Association*, 0(0):1–35.

Teixeira, L. V., Assunção, R. M., and Loschi, R. H. (2019). Bayesian space-time partitioning by sampling and pruning spanning trees. *Journal of Machine Learning Research*, 20(85):1–35.

Teixeira, L. V., Assuncao, R. M., and Loschi, R. H. (2015). A generative spatial clustering model for random data through spanning trees. In *2015 IEEE International Conference on Data Mining*, pages 997–1002. ISSN 1550-4786.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267--288. ISSN 00359246.

Tutz, G. and Berger, M. (2018). Tree-structured modelling of categorical predictors in generalized additive regression. *Adv. Data Anal. Classif.*, 12(3):737--758. ISSN 1862-5347.

Wagstaff, K., Cardie, C., Rogers, S., and Schrödl, S. (2001). Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 577--584, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Xu, R. and Wunsch, II, D. (2005). Survey of clustering algorithms. *Trans. Neur. Netw.*, 16(3):645--678. ISSN 1045-9227.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429.

Zumel, N. (2012). Modeling trick: Impact coding of categorical variables with many levels. `http://www.win-vector.com/blog/2012/07/modeling-trick-impact-coding-of-categorical-variables-with-many-levels/`. Accessed: 2019-05-12.

Appendix A

Posterior full conditional distributions

In this chapter, we provide details on the calculations of the posterior full conditional distributions (fcd) that are required to sample from the posterior distributions.

Posterior fcd of $\boldsymbol{\beta}$

$$
\begin{aligned}
p(\boldsymbol{\beta}|\boldsymbol{\theta}, \sigma_y^2, \boldsymbol{\pi}, \mathcal{T}, \rho, \mathcal{D}) &\propto p(\boldsymbol{Y}|\boldsymbol{\beta}, \boldsymbol{\theta}, \sigma_y^2, \boldsymbol{\pi}, \mathcal{T}, \rho, \boldsymbol{X}, \boldsymbol{Z}) p(\boldsymbol{\beta}|\sigma_y^2) \\
&\propto \exp\Big\{\frac{-1}{2\sigma_y^2}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{\theta})^T(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\boldsymbol{\theta}) + (\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})^T\Sigma_{\boldsymbol{\beta}}^{-1}(\boldsymbol{\beta} - \mu_{\boldsymbol{\beta}})\Big\} \\
&\propto \exp\Big\{\frac{-1}{2\sigma_y^2}(\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\theta}) + \boldsymbol{\beta}^T\Sigma_{\boldsymbol{\beta}}^{-1}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}})\Big\} \\
&\propto \exp\Big\{\frac{-1}{2\sigma_y^2}(\boldsymbol{\beta}^T(\boldsymbol{X}^T\boldsymbol{X} + \Sigma_{\boldsymbol{\beta}}^{-1})\boldsymbol{\beta} - 2\boldsymbol{\beta}^T(\boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\theta}) + \Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}})\Big\} \\
&\propto \mathcal{N}\Big((\boldsymbol{X}^T\boldsymbol{X} + \Sigma_{\boldsymbol{\beta}}^{-1})^{-1}(\boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\theta}) + \Sigma_{\boldsymbol{\beta}}^{-1}\mu_{\boldsymbol{\beta}}), \sigma_y^2(\boldsymbol{X}^T\boldsymbol{X} + \Sigma_{\boldsymbol{\beta}}^{-1})^{-1}\Big)
\end{aligned}
$$

Posterior fcd of θ related to cluster G_k

Let $\boldsymbol{\theta}_{\boldsymbol{G_k}}$ be a vector of dimension $n_{G_k} \times 1$ with all its values equals to θ_{G_k}, in other words $\boldsymbol{\theta}_{\boldsymbol{G_k}} = \theta_{G_k}\mathbf{1}_{n_{G_k}\times 1}$

$$
\begin{aligned}
p(\boldsymbol{\theta}_{G_k}|\boldsymbol{\beta},\mathcal{T},\boldsymbol{\pi},\rho,\sigma_y^2,\mathcal{D}) &\propto p(\boldsymbol{Y}|\boldsymbol{\beta},\boldsymbol{\theta},\sigma_y^2,\boldsymbol{\pi},\mathcal{T},\rho,\boldsymbol{X},\boldsymbol{Z})p(\boldsymbol{\theta}_{G_k}|\boldsymbol{\pi},\sigma_y^2)\\
&\propto \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\sum_{i\in G_k}(\boldsymbol{y}_i - x_i\boldsymbol{\beta} - \boldsymbol{\theta}_{G_k})^2 + \frac{1}{v_\theta}(\boldsymbol{\theta}_{G_k}-\mu_\theta)^2\Big)\Big\}\\
\propto \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\sum_{i\in G_k}(\boldsymbol{\theta}_{G_k}^2 - 2\boldsymbol{\theta}_{G_k}\boldsymbol{y}_i + 2\boldsymbol{\theta}_{G_k}x_i\boldsymbol{\beta}) + \frac{1}{v_\theta}(\boldsymbol{\theta}_{G_k}^2 - 2\boldsymbol{\theta}_{G_k}\mu_\theta)\Big)\Big\}\\
\propto \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(n_{G_k}\boldsymbol{\theta}_{G_k}^2 - 2\boldsymbol{\theta}_{G_k}\sum_{i\in G_k}(\boldsymbol{y}_i - x_i\boldsymbol{\beta}) + \frac{\boldsymbol{\theta}_{G_k}^2}{v_\theta} - \frac{2\boldsymbol{\theta}_{G_k}\mu_\theta}{v_\theta}\Big)\Big\}\\
\propto \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\boldsymbol{\theta}_{G_k}^2(n_{G_k}+\frac{1}{v_\theta}) - 2\boldsymbol{\theta}_{G_k}(\sum_{i\in G_k}(\boldsymbol{y}_i - x_i\boldsymbol{\beta}) + \frac{\mu_\theta}{v_\theta})\Big)\Big\}\\
\sim \mathcal{N}\left(\frac{\sum_{i\in G_k}(\boldsymbol{y}_i - x_i\boldsymbol{\beta}) + \frac{\mu_\theta}{v_\theta}}{n_{G_k}+\frac{1}{v_\theta}}, \frac{\sigma_y^2}{n_{G_k}+\frac{1}{v_\theta}}\right)
\end{aligned}
$$

Posterior fcd of ρ

$$
\begin{aligned}
p(\rho|\boldsymbol{\pi},\mathcal{T},\boldsymbol{\theta},\boldsymbol{\beta},\sigma_y^2,\mathcal{D}) &\propto p(\rho)p(\mathcal{T})p(\boldsymbol{\pi}|\rho,\mathcal{T})\\
&\propto \rho^{\kappa-1}(1-\rho)^{\psi-1}\rho^{c-\mathcal{F}}(1-\rho)^{V-c}\\
&\propto \rho^{(\kappa+c-\mathcal{F})-1}(1-\rho)^{(\psi+V-c)-1}\\
&\sim Beta(\kappa+(c-\mathcal{F}),\psi+V-c)
\end{aligned}
$$

Posterior fcd of σ_y^2

$$
\begin{aligned}
p(\sigma_y^2|\boldsymbol{\beta},\boldsymbol{\theta},\mathcal{T},\rho,\boldsymbol{\pi},\mathcal{D}) &\propto p(\boldsymbol{Y}|\boldsymbol{\beta},\boldsymbol{\theta},\sigma_y^2,\boldsymbol{\pi},\mathcal{T},\rho,\boldsymbol{X},\boldsymbol{Z})p(\sigma_y^2)p(\boldsymbol{\theta}|\sigma_y^2)p(\boldsymbol{\beta}|\sigma_y^2)\\
&\propto (\sigma_y^2)^{-a-\frac{1}{2}(n+D+c)-1}\exp\Big\{\frac{-1}{\sigma_y^2}\Big(b+\frac{1}{2}(\tilde{Y}+\tilde{\boldsymbol{\beta}}+\tilde{\boldsymbol{\theta}})\Big)\Big\}\\
&\sim IG\left(a+\frac{1}{2}(n+D+c), b+\frac{1}{2}(\tilde{Y}+\tilde{\boldsymbol{\beta}}+\tilde{\boldsymbol{\theta}})\right)
\end{aligned}
$$

Where $\tilde{Y} = (\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\boldsymbol{\theta})^T(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\boldsymbol{\theta})$, $\tilde{\boldsymbol{\beta}} = (\boldsymbol{\beta}-\mu_{\boldsymbol{\beta}})^T\Sigma_{\boldsymbol{\beta}}^{-1}(\boldsymbol{\beta}-\mu_{\boldsymbol{\beta}})$ and $\tilde{\boldsymbol{\theta}} = \frac{1}{v_\theta}\sum_{k=1}^{c}(\boldsymbol{\theta}_{G_k}-\mu_\theta)^2$

Posterior fcd of the partition $\boldsymbol{\pi}$

Under the proposed model assumptions, given a partition, the likelihood based on the complete data is decomposed as the product of likelihoods related to each cluster.

Denote the likelihood by $L(\boldsymbol{Y}|\boldsymbol{U}) = \prod_{k=1}^{c} L(\boldsymbol{Y}_{G_k})$. Assuming that the edge U_l disconnects the cluster G_k generating two clusters G_k^* and G_k^{**} if $U_l = 0$. To generate a new partition via Gibbs sampler, we must calculate the ratio $R_l = \frac{p(U_l=1|U_{-l},\sigma_y^2,\boldsymbol{\theta},\boldsymbol{\beta},\rho,\mathcal{T})}{p(U_l=0|U_{-l},\sigma_y^2,\boldsymbol{\theta},\boldsymbol{\beta},\rho,\mathcal{T})}$.

$$p(U_l = t|U_{-l}, \sigma_y^2, \boldsymbol{\beta}, \boldsymbol{\theta}, \mathcal{T}, \rho, \mathcal{D}) \propto L(\boldsymbol{Y}|\boldsymbol{U})p(\sigma_y^2)p(\boldsymbol{\beta}|\sigma_y^2)p(\boldsymbol{\theta}|U_l = t, U_{-l}, \sigma_y^2)p(\rho)p(\boldsymbol{\pi}|\rho, \boldsymbol{U}), \tag{A.1}$$

where $p(\boldsymbol{\theta}|U_l = 1, U_{-l}, \sigma_y^2) = p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)\prod_{i\neq k} p(\boldsymbol{\theta}_{G_i}|\sigma_y^2)$ and $p(\boldsymbol{\theta}|U_l = 0, U_{-l}, \sigma_y^2) = p(\boldsymbol{\theta}_{G_k^*}|\sigma_y^2)p(\boldsymbol{\theta}_{G_k^{**}}|\sigma_y^2)\prod_{i\neq k} p(G_i|\sigma_y^2)$. Expanding the terms in (A.1) we have:

$$\begin{aligned} R_l = &\frac{L(\boldsymbol{Y}_{G_k})\prod_{i\neq k} L(\boldsymbol{Y}_{G_i})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)\prod_{i\neq k} p(\boldsymbol{\theta}_{G_i}|\sigma_y^2)p(\sigma_y^2)}{L(\boldsymbol{Y}_{G_k^*})L(\boldsymbol{Y}_{G_k^{**}})\prod_{i\neq k} L(\boldsymbol{Y}_{G_i})p(\boldsymbol{\theta}_{G_k^*}|\sigma_y^2)p(\boldsymbol{\theta}_{G_k^{**}}|\sigma_y^2)\prod_{i\neq k} p(\boldsymbol{\theta}_{G_i}|\sigma_y^2)p(\sigma_y^2)} \\ &\times \frac{p(\boldsymbol{\beta}|\sigma_y^2)}{p(\boldsymbol{\beta}|\sigma_y^2)} \frac{\int p(\boldsymbol{\pi}|\rho, U_l = 1, U_{-l})p(\rho)d\rho}{\int p(\boldsymbol{\pi}|\rho, U_l = 0, U_{-l})p(\rho)d\rho}. \end{aligned}$$

To achieve a numerical stable solution, we integrate ρ out of the equation (A.1).

$$\begin{aligned} \int p(\boldsymbol{\pi}|\rho, \boldsymbol{U})p(\rho)d\rho &= \frac{\Gamma(\kappa+\psi)}{\Gamma(\kappa)\Gamma(\psi)} \int \rho^{(\kappa+c-\mathcal{F})-1}(1-\rho)^{(\psi+V-c)-1}d\rho \\ &= \frac{\Gamma(\kappa+\psi)}{\Gamma(\kappa)\Gamma(\psi)} \frac{\Gamma(\kappa+c-\mathcal{F})\Gamma(\psi+V-c)}{\Gamma(\kappa+\psi+V-\mathcal{F})}, \end{aligned}$$

where c is the number of clusters in the partition spanned by the binary vector $\boldsymbol{U}$.

By canceling out common elements on the numerator and denominator and computing the integral over ρ we obtain:

$$R_l = \frac{L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)}{L(\boldsymbol{Y}_{G_k^*})p(\boldsymbol{\theta}_{G_k^*}|\sigma_y^2)L(\boldsymbol{Y}_{G_k^{**}})p(\boldsymbol{\theta}_{G_k^{**}}|\sigma_y^2)} \frac{\psi+V-c-1}{\kappa+c-\mathcal{F}},$$

where

$$\begin{aligned} L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2) &= \Big(\frac{1}{2\pi\sigma_y^2}\Big)^{\frac{n_{G_k}}{2}} \Big(\frac{1}{2\pi\sigma_y^2 v_\theta}\Big)^{\frac{1}{2}} \exp\Big\{\frac{-1}{2v_\theta\sigma_y^2}(\boldsymbol{\theta}_{G_k} - \mu_\theta)^2\Big\} \\ &\times \exp\Big\{\frac{-1}{2\sigma_y^2}(\boldsymbol{Y}_{G_k} - \mathbf{X}_{G_k}\boldsymbol{\beta} - \boldsymbol{Z}_{G_k}\boldsymbol{\theta})^T(\boldsymbol{Y}_{G_k} - \boldsymbol{X}_{G_k}\boldsymbol{\beta} - \boldsymbol{Z}_{G_k}\boldsymbol{\theta})\Big\} \\ &= \Big(\frac{1}{2\pi\sigma_y^2}\Big)^{\frac{n_{G_k}+1}{2}} \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\boldsymbol{Y}_{G_k}^T\boldsymbol{Y}_{G_k} + \boldsymbol{\beta}^T\mathbf{X}_{G_k}^T\mathbf{X}_{G_k}\boldsymbol{\beta} \\ &\times -2\boldsymbol{\beta}^T\mathbf{X}_{G_k}^T\boldsymbol{Y}_{G_k} - 2\boldsymbol{\theta}^T\mathbf{Z}_{G_k}^T(\boldsymbol{Y}_{G_k} - \mathbf{X}_{G_k}\boldsymbol{\beta}) + \boldsymbol{\theta}^T\mathbf{Z}_{G_k}^T\mathbf{Z}_{G_k}\boldsymbol{\theta}\Big)\Big\} \\ &\times \exp\Big\{\frac{-1}{2v_\theta\sigma_y^2}(\boldsymbol{\theta}_{G_k}^2 - 2\boldsymbol{\theta}_{G_k}\mu_\theta + \mu_\theta^2)\Big\}\Big(\frac{1}{v_\theta}\Big)^{\frac{1}{2}} \end{aligned}$$

As $\boldsymbol{Z}_{G_k}\boldsymbol{\theta}$ and $\boldsymbol{Z}_{G_k}^T\boldsymbol{\theta}^T$ are, respectively, a column vector and a row vector with all of its elements equals to $\boldsymbol{\theta}_{G_k}$, we can write $L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)$ in terms of $\boldsymbol{\theta}_{G_k}$ as follows:

$$L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2) = \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\boldsymbol{Y}_{G_k}^T\boldsymbol{Y}_{G_k} + \boldsymbol{\beta}^T\boldsymbol{X}_{G_k}^T\boldsymbol{X}_{G_k}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\boldsymbol{X}_{G_k}^T\boldsymbol{Y}_{G_k} + \frac{\mu_\theta^2}{v_\theta}\Big)\Big\}$$
$$\times \exp\Big\{\frac{-1}{2\sigma_y^2}\Big(n_{G_k}\boldsymbol{\theta}_{G_k}^2 - 2\boldsymbol{\theta}_{G_k}r_{G_k} + \frac{1}{v_\theta}\boldsymbol{\theta}_{G_k}^2 - 2\frac{\mu_\theta}{v_\theta}\boldsymbol{\theta}_{G_k}\Big)\Big\}\Big(\frac{1}{2\pi\sigma_y^2}\Big)^{\frac{n_{G_k}+1}{2}}\Big(\frac{1}{v_\theta}\Big)^{\frac{1}{2}},$$

where $r_{G_k} = \sum_{i\in G_k}(y_i - x_i\boldsymbol{\beta}))$ is the sum of partial residuals of the cluster G_k, that is, it corresponds to the residuals in the linear regression fitted to data in G_k without the vertices effects $\boldsymbol{\theta}$. Working only with the elements inside the second exponential that depends on $\boldsymbol{\theta}_{G_k}$ we obtain that

$$\frac{-1}{2\sigma_y^2}\Big(\boldsymbol{\theta}_{G_k}^2\Big(n_{G_k} + \frac{1}{v_\theta}\Big) - 2\boldsymbol{\theta}_{G_k}\Big(r_{G_k} + \frac{\mu_\theta}{v_\theta}\Big)\Big) = \frac{-1}{2\sigma_y^2}\Big(n_{G_k} + \frac{1}{v_\theta}\Big)\Big(\boldsymbol{\theta}_{G_k}$$
$$-\Big(n_{G_k} + \frac{1}{v_\theta}\Big)^{-1}\Big(r_{G_k+\frac{\mu_\theta}{v_\theta}}\Big)\Big)^2 - \frac{-1}{2\sigma_y^2}\Big(\Big(n_{G_k} + \frac{1}{v_\theta}\Big)^{-1}\Big(r_{G_k+\frac{\mu_\theta}{v_\theta}}\Big)^2\Big).$$

Replacing this result in $L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)$ and integrating out $\boldsymbol{\theta}$, we have that

$$\int L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k}|\sigma_y^2)d\boldsymbol{\theta}_{G_k} = \Big(\frac{1}{2\pi\sigma_y^2}\Big)^{\frac{n_{G_k}}{2}}\Big(\frac{1}{v_\theta n_{G_k}+1}\Big)^{\frac{1}{2}}\times$$
$$\exp\Big\{\frac{-1}{2\sigma_y^2}\Big(\boldsymbol{Y}_{G_k}^T\boldsymbol{Y}_{G_k} + \boldsymbol{\beta}^T\boldsymbol{X}_{G_k}^T\boldsymbol{X}_{G_k}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\boldsymbol{X}_{G_k}^T\boldsymbol{Y}_{G_k} + \frac{\mu_\theta^2}{v_\theta} - (n_{G_k} + \frac{1}{v_\theta})^{-1}(r_{G_k+\frac{\mu_\theta}{v_\theta}})^2\Big\},$$

which can be rewritten as

$$\int L(\boldsymbol{Y}_{G_k})p(\boldsymbol{\theta}_{G_k})d\boldsymbol{\theta}_{G_k} = \Big(\frac{1}{2\pi\sigma_y^2}\Big)^{\frac{n_{G_k}}{2}}\Big(\frac{1}{v_\theta n_{G_k}+1}\Big)^{\frac{1}{2}}\times$$
$$\exp\Big\{\frac{-1}{2\sigma_y^2}\Big(SSE(G_k) + \lambda - \phi(G_k)\Big\},$$

where $SSE(G_k) = (\boldsymbol{Y}_{G_k} - \boldsymbol{X}_{G_k}\boldsymbol{\beta})^T(\boldsymbol{Y}_{G_k} - \boldsymbol{X}_{G_k}\boldsymbol{\beta})$, $\lambda = \frac{\mu_\theta^2}{v_\theta}$, $\phi(G_k) = (\frac{\mu_\theta}{v_\theta} + r_{G_k})^2(n_{G_k} + \frac{1}{v_\theta})^{-1}$. Using the fact that $G_k = G_k^* \bigcup G_k^{**}$ and $G_k^* \bigcap G_k^{**} = \emptyset$ we have that $SSE(G_k) = SSE(G_k^*) + SSE(G_k^{**})$ and $n_{G_k} = n_{G_k^*} + n_{G_k^{**}}$ we have that the ratio is given by:

$$
\begin{aligned}
R_i &= \frac{\int L(\boldsymbol{Y}_{G_k}) p(\boldsymbol{\theta}_{G_k}|\sigma_y^2) d\boldsymbol{\theta}_{G_k}}{\int L(\boldsymbol{Y}_{G_k^*}) p(\boldsymbol{\theta}_{G_k^*}|\sigma_y^2) d\boldsymbol{\theta}_{G_k^*} \int L(\boldsymbol{Y}_{G_k^{**}}) p(\boldsymbol{\theta}_{G_k^{**}}|\sigma_y^2) d\boldsymbol{\theta}_{G_k^{**}}} \frac{\psi + V - c - 1}{\kappa + c - \mathcal{F}} \\
&= \frac{\psi + V - c - 1}{\kappa + c - \mathcal{F}} \frac{(v_\theta n_{G_k^*} + 1)^{1/2} (v_\theta n_{G_k^{**}} + 1)^{1/2}}{(v_\theta n_{G_k} + 1)^{1/2}} \\
&\quad \times \exp\left\{ \frac{-1}{2\sigma_y^2} (-\lambda - \phi(G_k) + \phi(G_k^*) + \phi(G_k^{**}) \right\} \qquad \text{(A.2)}
\end{aligned}
$$

If we do not integrate over ρ, we should change the ratio $\frac{\psi+V-c-1}{\kappa+c-\mathcal{F}}$ in equation (A.2) to $\frac{(1-\rho)}{\rho}$. However, this may cause some instability in the algorithm as a division by zero is possible.

A.1 Experimental setup

All the experiments were run on the same machine with Intel Core i7-2600, 3.4GHz, 8 cores and 16 GB of RAM. Except for the CatBoost that used 8 threads, all methods were run single thread.

A.1.1 Inside Airbnb data cleaning

Firstly, we selected only samples that are rental of a full house or apartment with at least one review. The price variable was transformed from a string to a numeric value (ex: \$5.0 to 5.0). Next, we removed samples with inconsistent values such as having zero bathroom, bedrooms and rent value. Next, we transform the rent value to the log of its value.

After subsetting the data and log transform the response variable, we removed outliers from the data set. The listings prices on Airbnb showed to have an asymmetric distribution with a long tail. Thus, we removed these outliers using an extension of the Median Absolute Deviation (MAD) Leys et al. [2013], the extension consisting in computing the median of the data set and split the data set in two. The first has all the values smaller than the median and the second all the remaining values. it applies the MAD in each side. As suggested in Leys et al. [2013], we used a conservative approach to remove samples with a MAD score greater than 3.0.

Following, we selected 10 amenities that we found that are useful to add value to a listing. Those amenities are TV, air conditioning, dryer, free street parking, iron, coffeemaker, family kid-friendly, dishwasher, indoor fireplace, fire extinguisher. We created an indicator variable for each amenities. Next, we transformed the following

variables in buckets: bedrooms, bathrooms, number guests included, accommodates mean review score cleanliness, mean review score value, minimum nights and property type as described at table A.2. We did not use the number of beds as a feature because it showed to be strongly correlated with other features such as the number of bedrooms and bathrooms.

The last feature considered was an indicator if a listing host is super host. [1] In total, we considered 19 categorical predictors. A summary with the compile date and the number of samples, fixed samples, regions/districts is presented at table A.1.

Table A.1. Summary for each dataset. N is the number of samples, N_{fixed} number of fixed samples, R number of levels in the categorical variable of interest, $\mathcal{F}$ number of components in the input graph and Date compile is the data in which the inside Airbnb was compiled (available for reproducibility).

Dataset	N	N_{fixed}	R	$\mathcal{F}$	Date compiled
Munich	2053	0	25	1	NA
Amster.	10741	0	22	2	08 April, 2019
Barcelona	5043	27	64	1	10 April, 2019
Chicago	2484	46	64	1	12 March, 2019
Hawaii	4518	4	30	6	03 May, 2019
London	24573	0	33	1	09 April, 2019
Montreal	7103	15	30	1	13 April, 2019
New York	11695	12	77	2	03 May, 2019
San Fran.	1920	1	35	1	03 May, 2019
Seattle	2878	24	88	2	15 April, 2019
Toronto	4340	83	136	1	08 April, 2019

A.1.2 Experimental settings

In our experiments, we evaluate the proposed model and compare it to some alternative procedures for modeling and clustering variables: the Bayesian ordinary linear regression model (Bayes) with dummy coding of categorical variables; the gvcmcat Oelker [2015], a lasso penalization to cluster the levels of a categorical variable; the effectFusion Pauger et al. [2019], a Bayesian model to clusters levels of a categorical variable by using a finite mixture model; and the CatBoost Prokhorenkova et al. [2017], state of the art gradient boosting model with categorical feature support.

Parameter tunning

We used the gvcm.cat (Library that implements the Lasso penalization method) and CatBoost internal methods to find the best tune for key parameters.

[1] `www.airbnb.com/superhost`

Table A.2. Variables used in the Airbnb dataset. The first 10 variables are the selected amenities, they indicates if a listing has the given amenity. The "is_superhost" variable indicates if the listings host is a super host. The review scores value and review scores cleanliness are the mean value of the listings review value. The variables, bathrooms (number of bedrooms), accommodates (number of persons that can be accommodated), bedrooms (number of bedrooms) and minimum nights (minimum nights required to book the listing) were transformed in buckets. The property type variable indicates it the property is a house or an apartment. The Z variable is an indicator variable indicating the neighbourhood that a listing belongs to. Finally. Y is the response variable (log of the listing price per night of stay).

Variable	Type	Description	Values
fire_extinguisher	cat	Amenity indicator	0/1
indoor_fireplace	cat	Amenity indicator	0/1
dishwasher	cat	Amenity indicator	0/1
family_kid_friendly	cat	Amenity indicator	0/1
coffeemaker	cat	Amenity indicator	0/1
freestreet_street_parking	cat	Amenity indicator	0/1
Iron	cat	Amenity indicator	0/1
Dryer	cat	Amenity indicator	0/1
Airconditioning	cat	Amenity indicator	0/1
TV	cat	Amenity indicator	0/1
is_superhost	cat	Indicator if a host is superhost	0/1
guests_included	cat	Number of guests included	(0, 1],(1, 2], (2, ∞)
review_scores_value	cat	Mean of reviews scores value	[0,9], (9, 10]
review_scores_cleanliness	cat	Mean of reviews score cleanliness	[0, 9], (9, 10]
bathrooms	cat	Number of bathrooms	(0,1], (1, 2], (2, ∞)
accommodates	cat	Number of persons that can be accommodates	(0, 2], (2, 4], (4, ∞)
bedrooms	cat	Number of bathrooms	(0, 1], (1, 2], (2, ∞)
minimum_nights	cat	Minimum nights required to book the listing	(0, 3], (3, 7], (7, 30], (30, ∞)
property_type	cat	Type of the listed property	House/Apartment
Z	cat	Neighbourhood	Neighbourhood name
Y	num	log of the listing price per night	(0, ∞)

gvcmcat:

- 'lambda': log scale (0, 50)

The adapted weight parameter was fixed as true for all datasets. We used the model penalization to cluster only the categorical variable of interest. A method from the gvcm.cat library was used to select the lambda value with the minimum mean squared error by performing a 5-fold-cv.

catboost:

- 'depth': [2, 4, 6, 8, 10]
- 'learning_rate': log-uniform $[e^{-5}, 1]$
- 'l2_leaf_reg': log-uniform $[1, 10]$

- 'rsm': 1
- 'border': 256

We fixed the CatBoost number of iteration to 250 and sampled 10 values for the depth, learning_rate and l2_leaf_reg, creating a grid of 1000 combinations of parameter value. The model was set to select the best combination of parameters in the grid that minimized the root mean squared error by using a 5-fold-cv.

effectFusion:

We fixed the following parameters:

- 'method': "FinMix"
- 'modelSelection': NULL
- 'startsel': 500

For the prior parameters we used the library default values that are standard choice in the literature. We used 'M' as 2000 for all datasets. Before fitting the model, we dummy encoded all categorical variables that we are not interested to cluster and we treated them as a continuous variable. The categorical variable that we were interested to cluster were treated as a nominal variable.

After finding the best set of parameters for each model, we performed a 5-fold-cv as described in section 4.4.

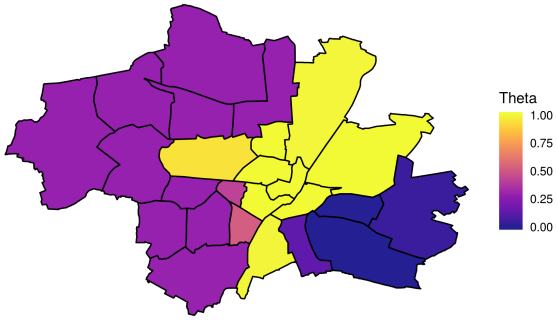
Version of the libraries

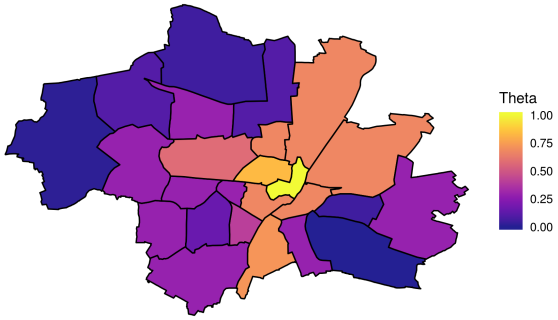
- catboost 0.14.2
- effectFusion 1.1.1
- igraph 1.2.4.1
- gvcmcat 1.9
- MASS 7.3-51.4
- R 3.5.3

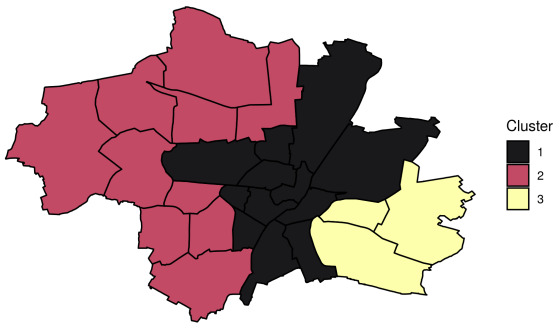
A.2 Additional plots

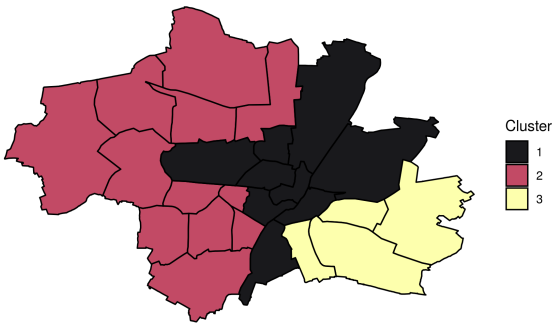
We show additional plots, for each dataset we show the posterior mean for $\boldsymbol{\theta}$ estimated by our model (left plot first row) and the estimated coefficients by the lasso penalization (right plot first row), sampled clustering from the posteriori distribution (second row), a histogram of the posteriori number of clusters (third row left plot), the neighbourhood graph used (third row right plot) and a caterpillar plot (95% highest posterior density interval) for each $\boldsymbol{\theta}$ estimated by our model. The estimated coefficient for each model were normalized using the min-max normalization.

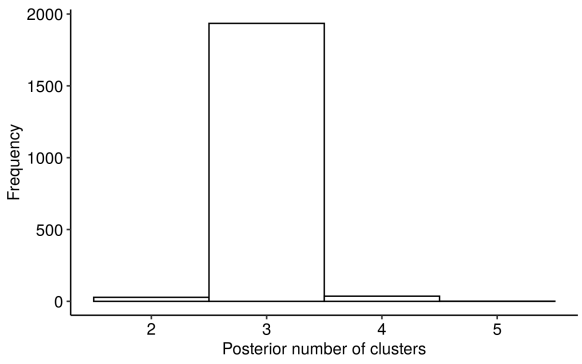
Munich house rent











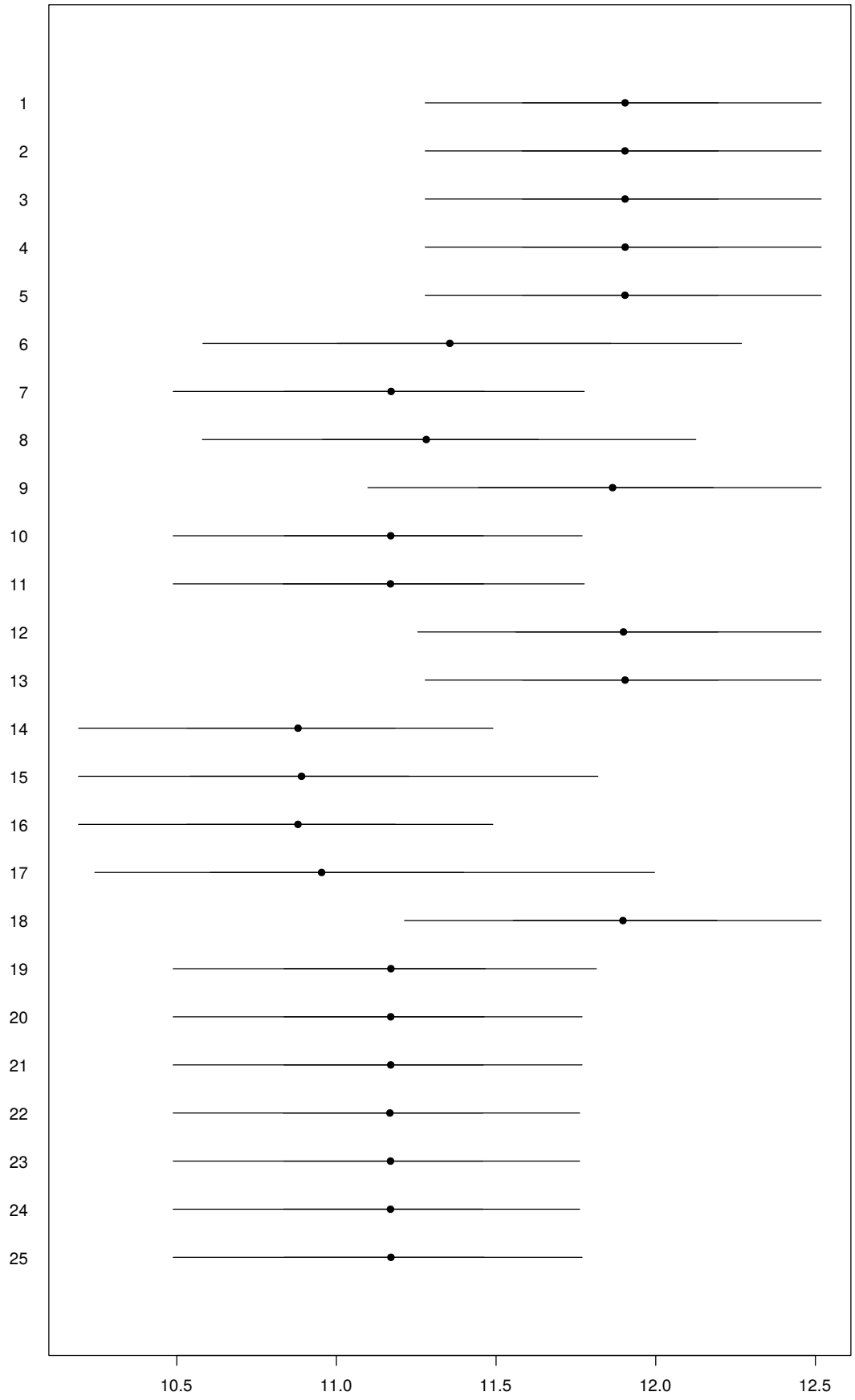


Figure A.1. Caterpillar plot

Inside Airbnb: Amsterdam

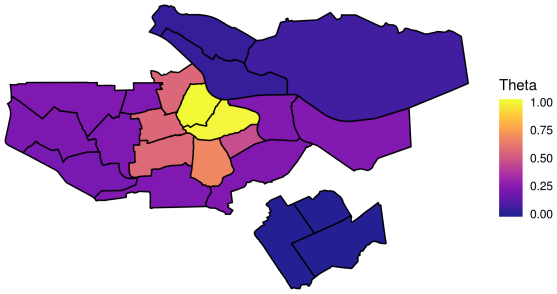


Figure A.2. PPRM posterior mean of $\boldsymbol{\theta}$

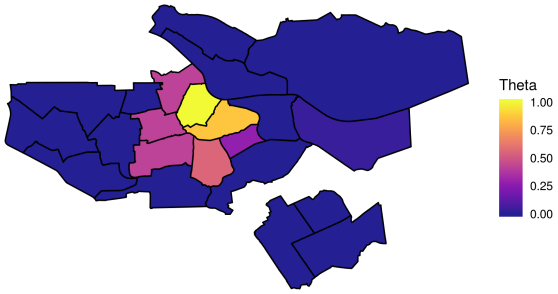


Figure A.3. Lasso estimated coefficients

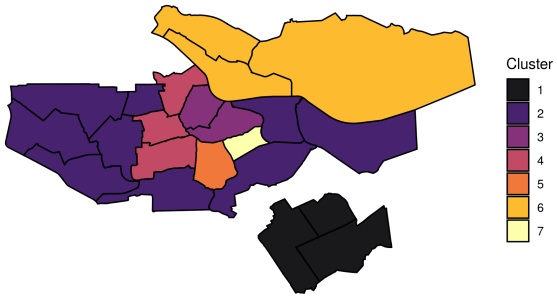


Figure A.4. PPRM sampled clustering

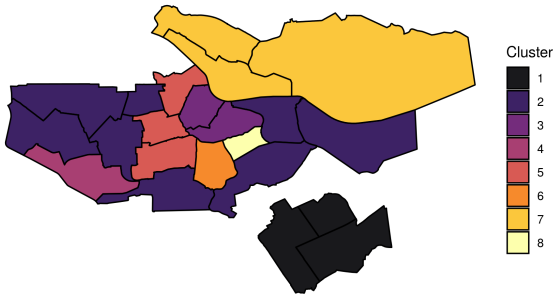


Figure A.5. PPRM sampled clustering

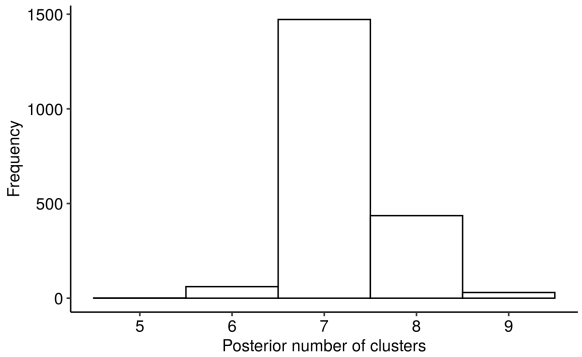


Figure A.6. PPRM posterior number of clusters

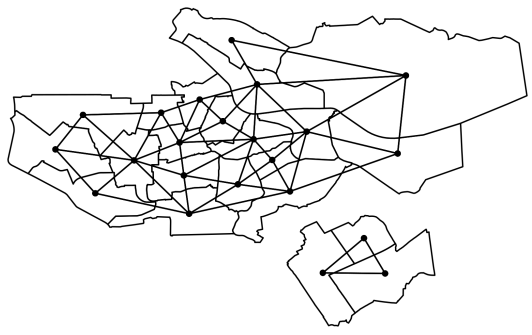


Figure A.7. Neighbourhood graph

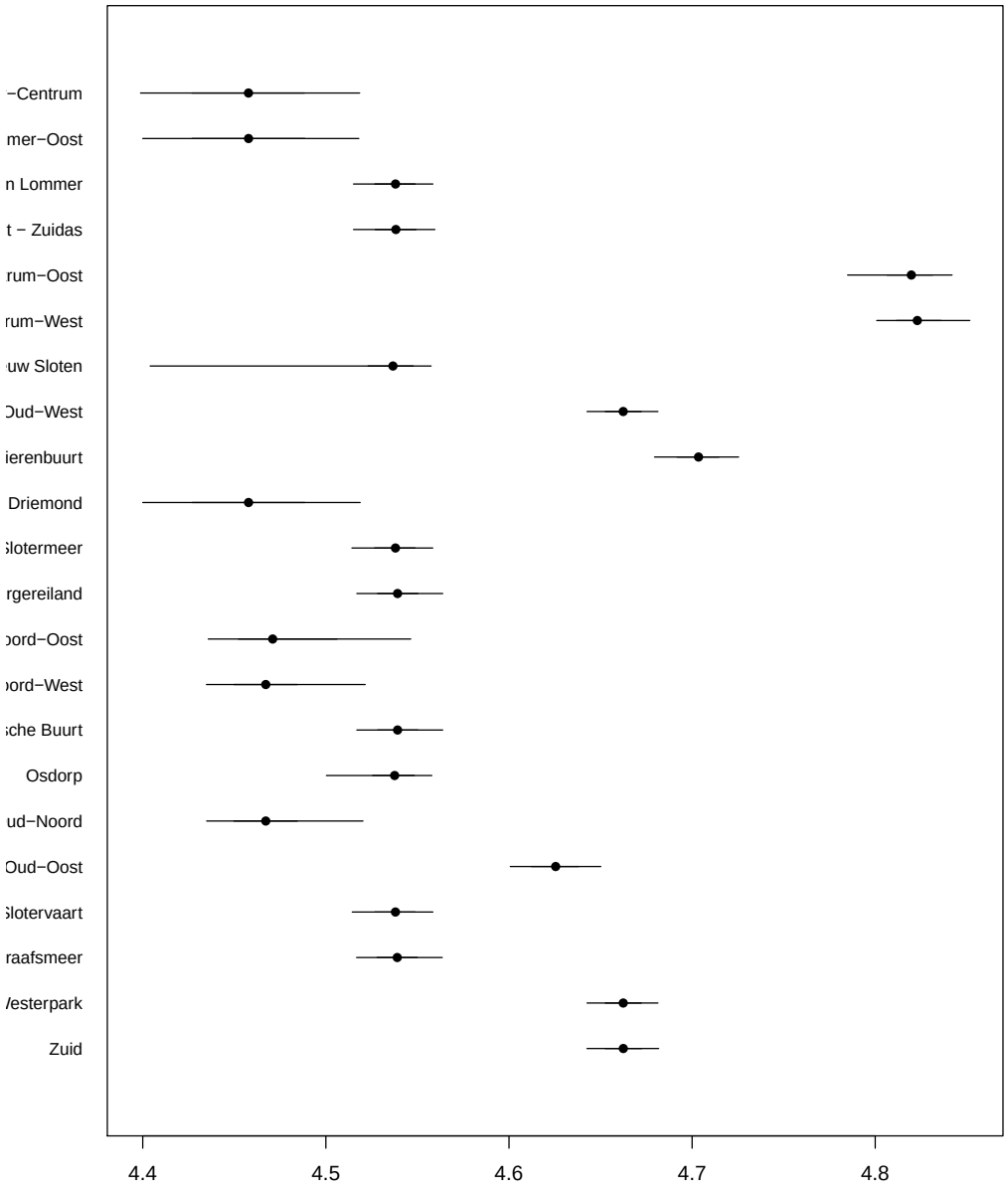


Figure A.8. Caterpillar plot

Inside Airbnb: Barcelona

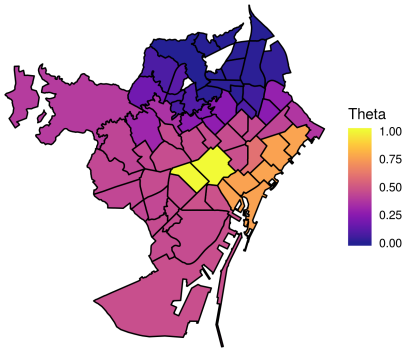


Figure A.9. PPRM posterior mean of $\boldsymbol{\theta}$

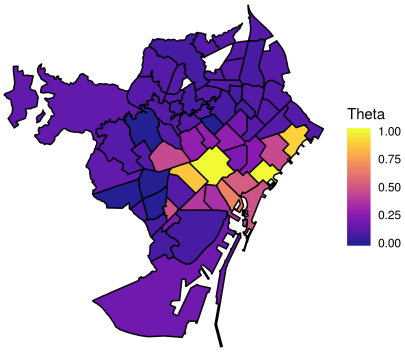


Figure A.10. Lasso estimated coefficients

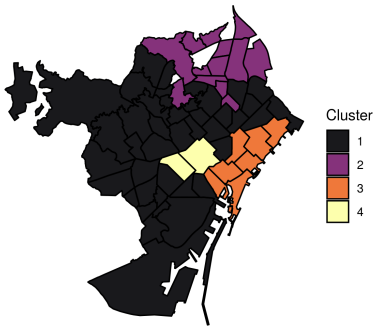


Figure A.11. PPRM sampled clustering

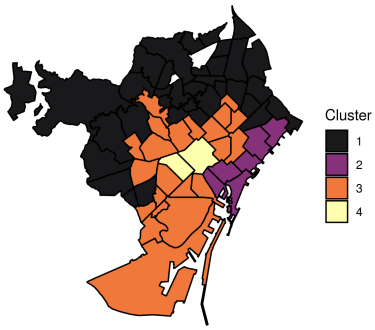


Figure A.12. PPRM sampled clustering

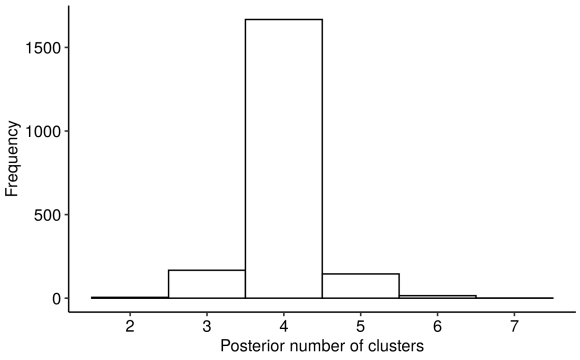


Figure A.13. PPRM posterior number of clusters

Figure A.14. Neighbourhood graph

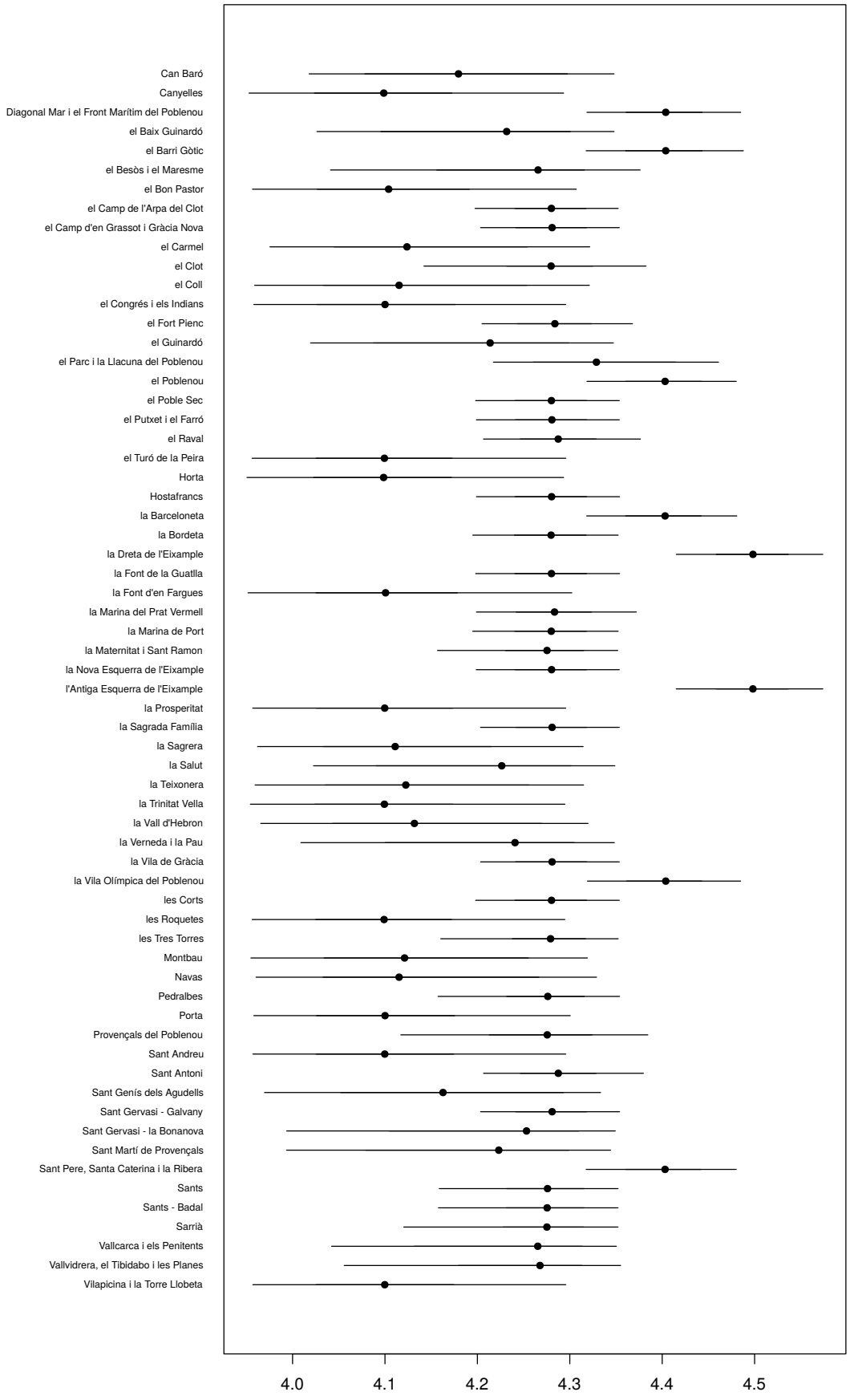


Figure A.15. Caterpillar plot

Inside Airbnb: Chicago

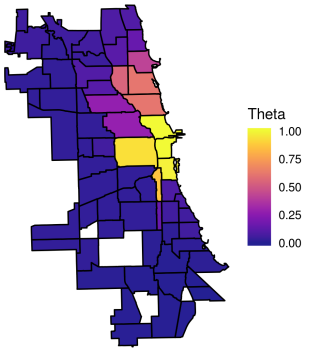


Figure A.16. PPRM posterior mean of $\boldsymbol{\theta}$

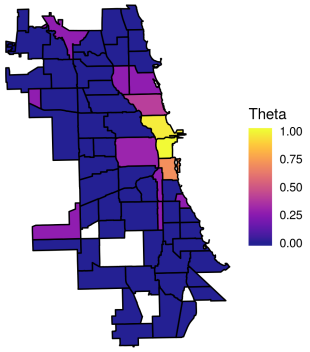


Figure A.17. Lasso estimated coefficients

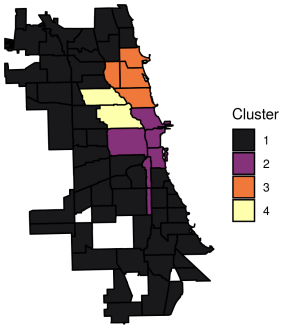


Figure A.18. PPRM sampled clustering

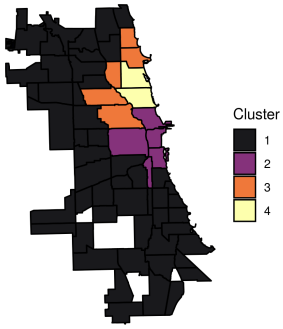


Figure A.19. PPRM sampled clustering

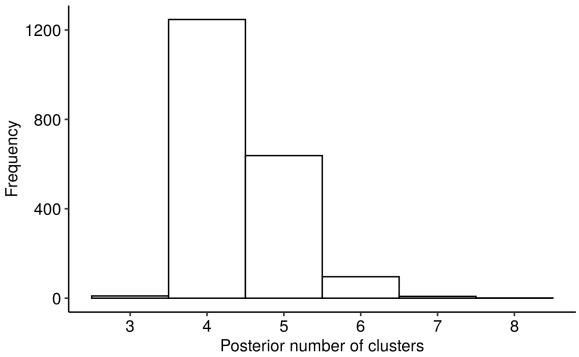


Figure A.20. PPRM posterior number of clusters

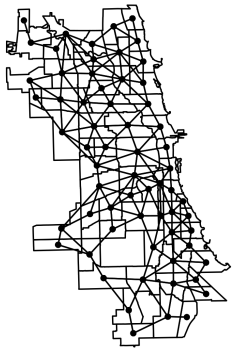


Figure A.21. Neighbourhood graph

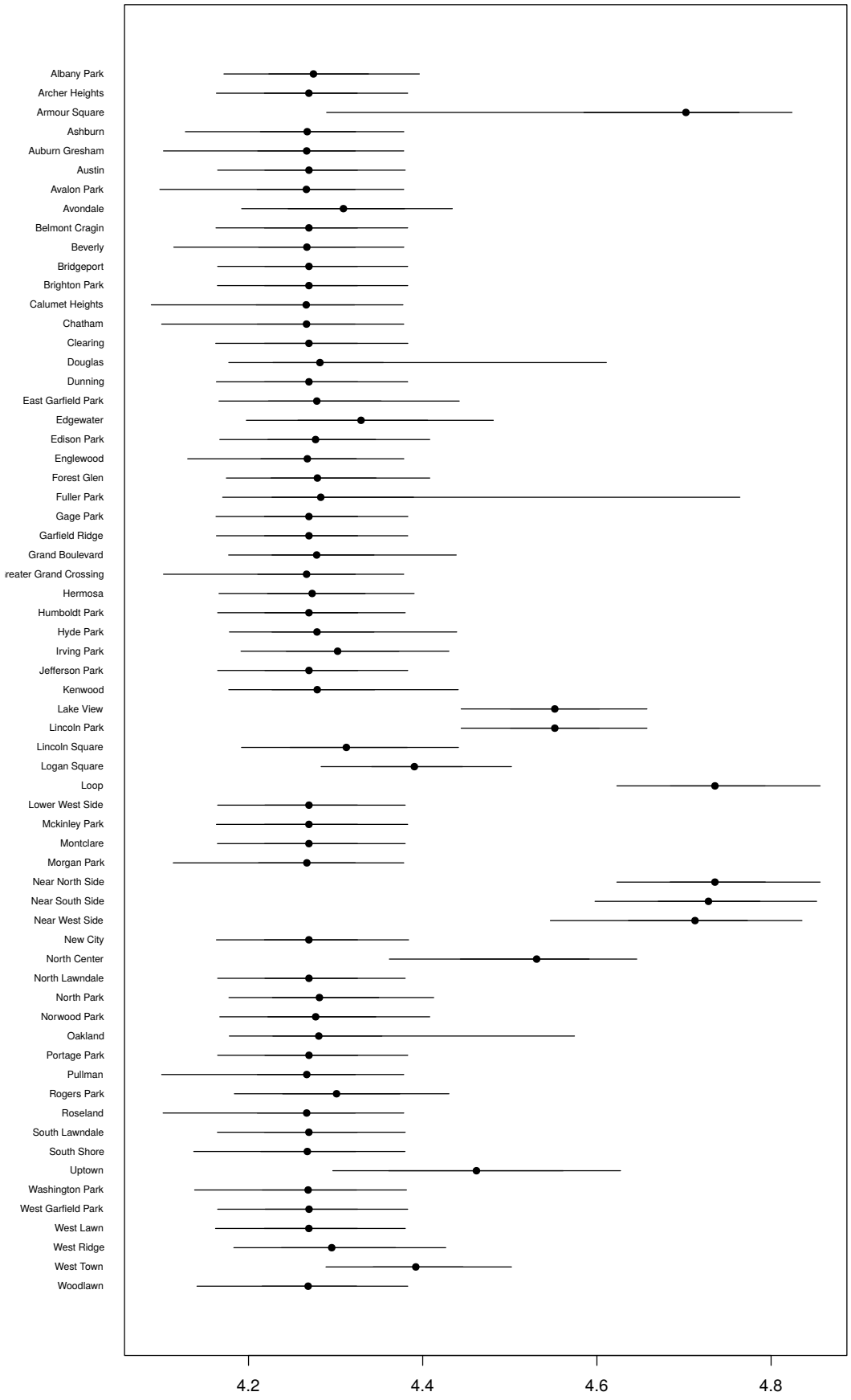


Figure A.22. Caterpillar plot

Inside Airbnb: Hawaii

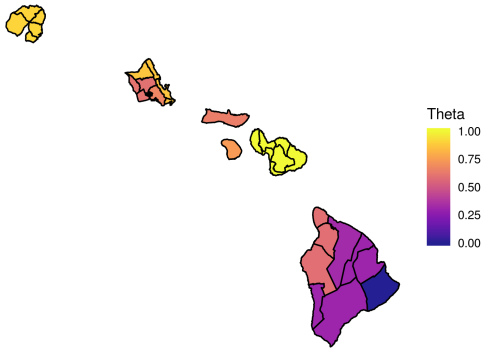


Figure A.23. PPRM posterior mean of $\boldsymbol{\theta}$

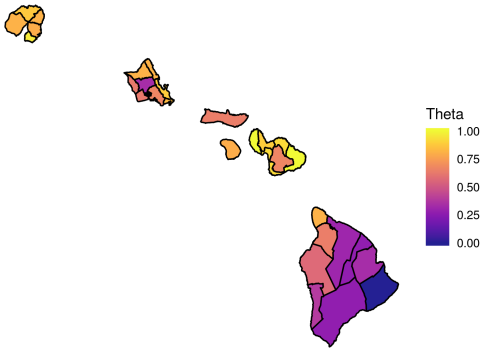


Figure A.24. Lasso estimated coefficients

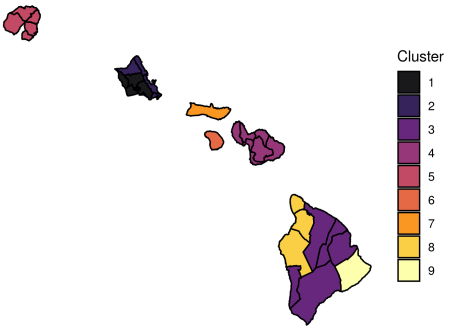


Figure A.25. PPRM sampled clustering

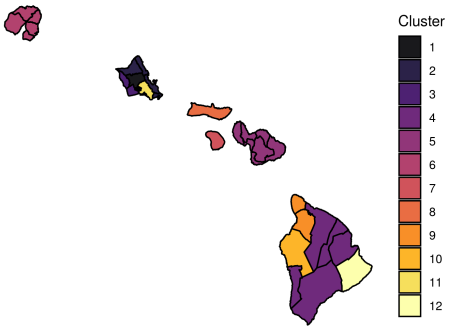


Figure A.26. PPRM sampled clustering

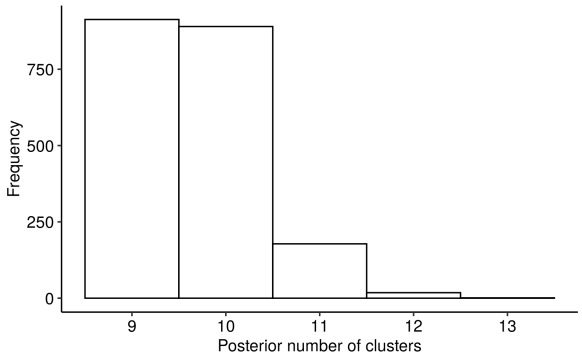


Figure A.27. PPRM posterior number of clusters

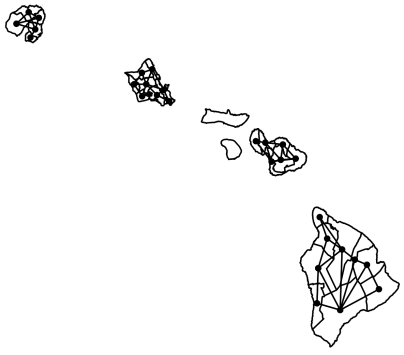


Figure A.28. Neighbourhood graph

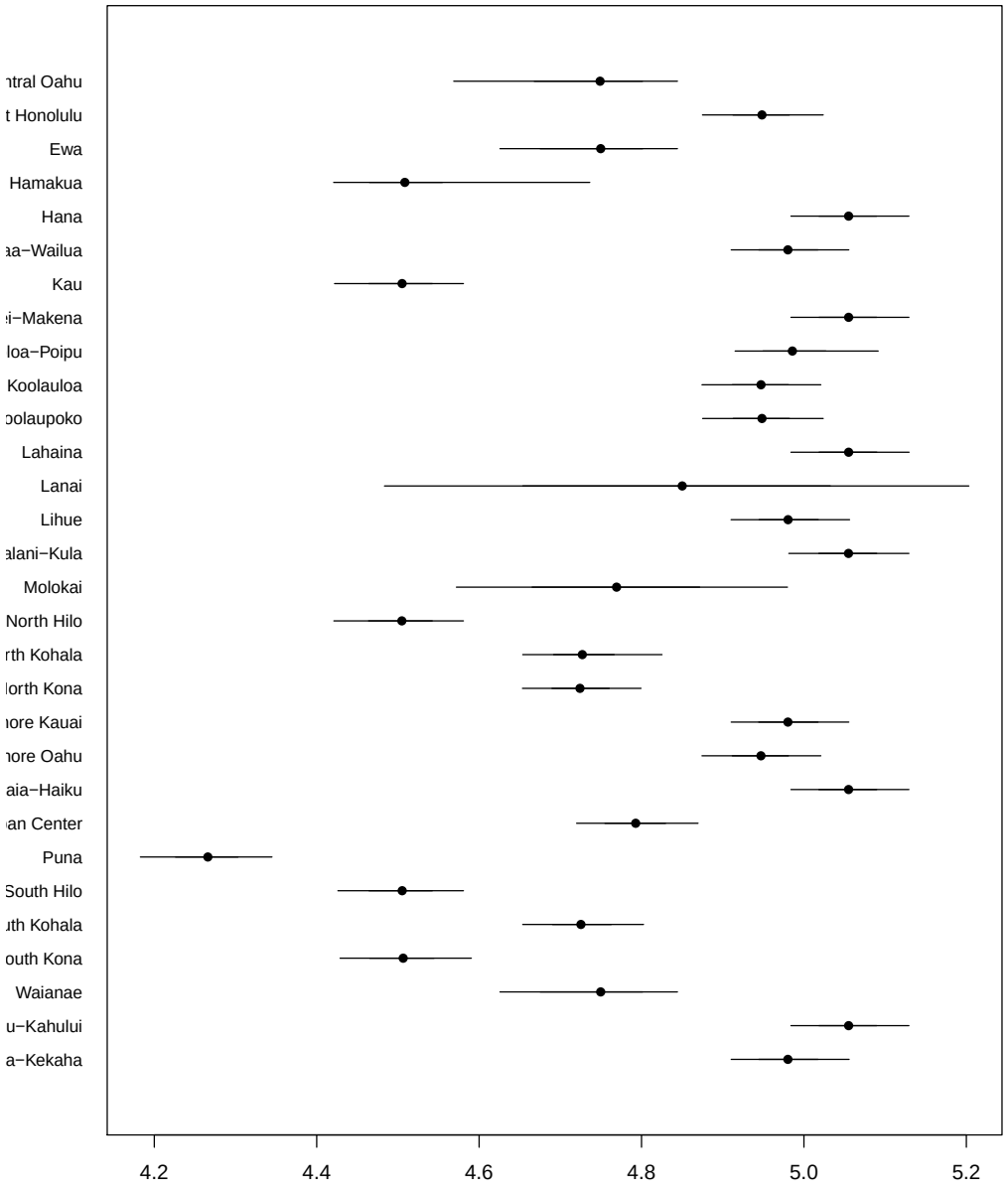


Figure A.29. Caterpillar plot

Inside Airbnb: London

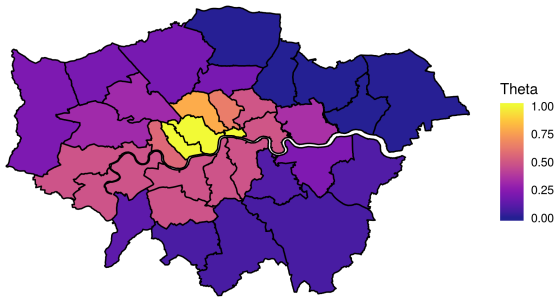


Figure A.30. PPRM posterior mean of $\boldsymbol{\theta}$

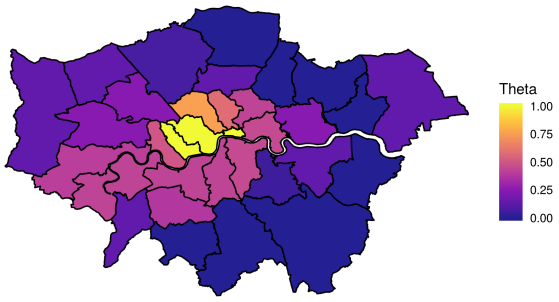


Figure A.31. Lasso estimated coefficients

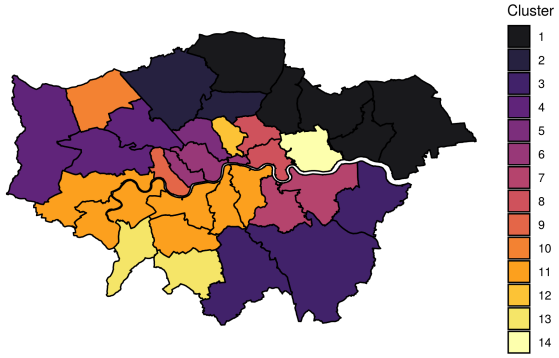


Figure A.32. PPRM sampled clustering

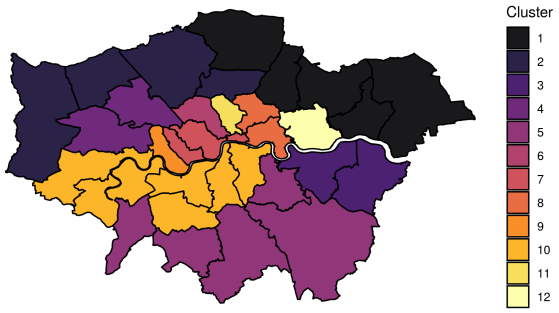


Figure A.33. PPRM sampled clustering

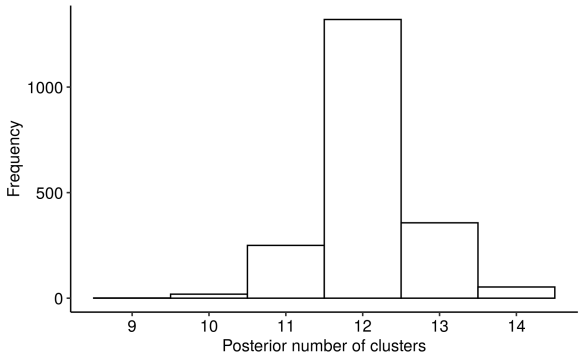


Figure A.34. PPRM posterior number of clusters

Figure A.35. Neighbourhood graph

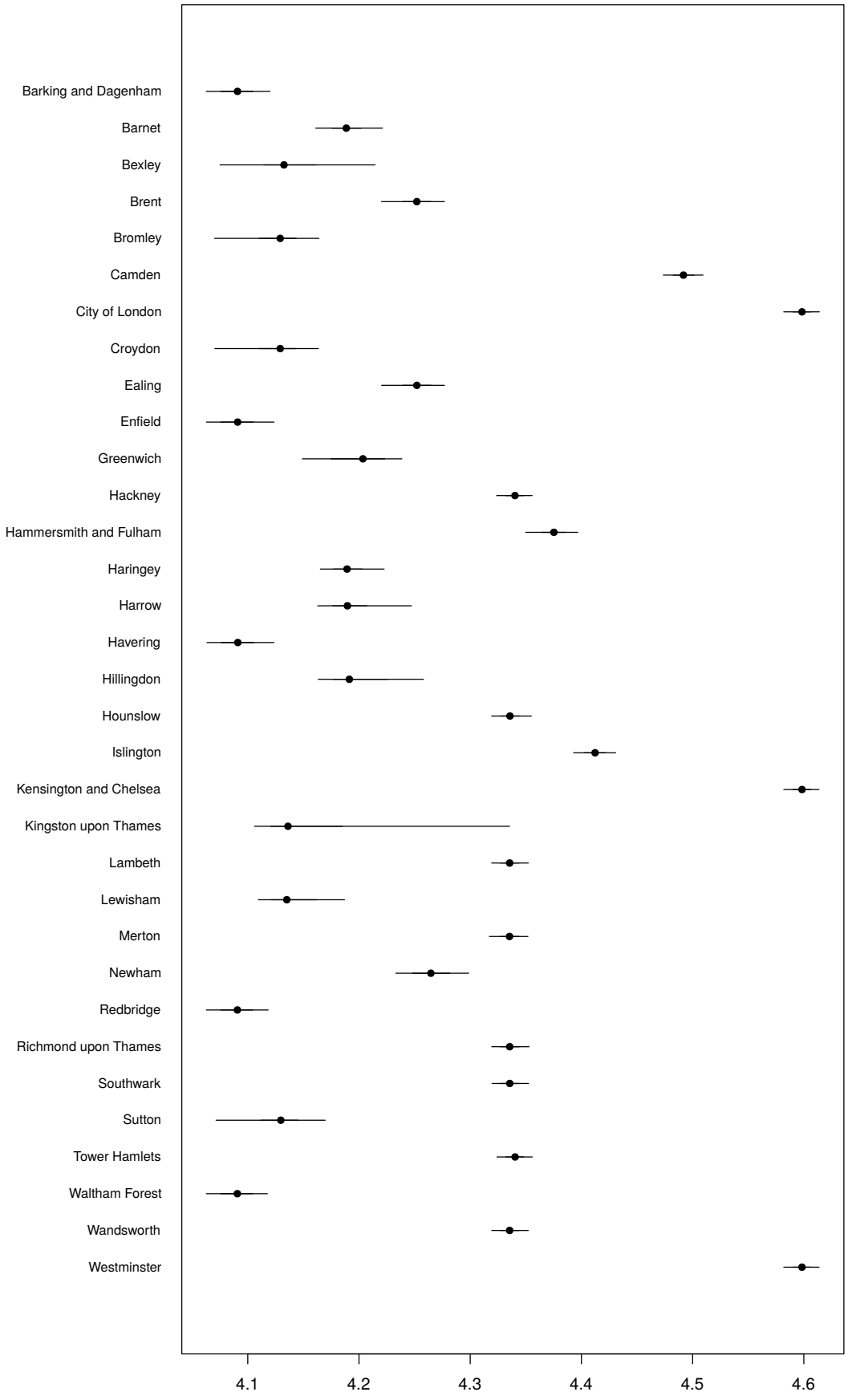


Figure A.36. Caterpillar plot

Inside Airbnb: Montreal

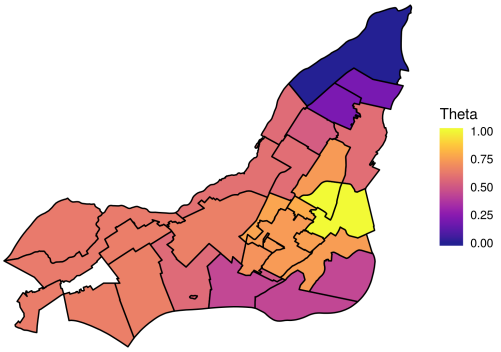


Figure A.37. PPRM posterior mean of $\boldsymbol{\theta}$

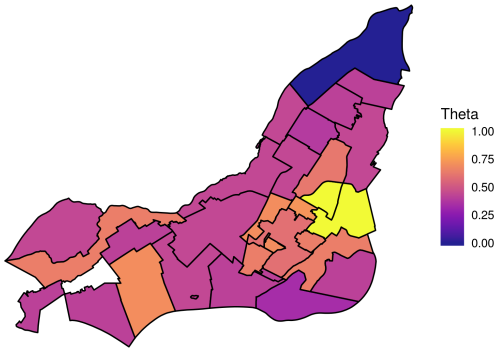


Figure A.38. Lasso estimated coefficients

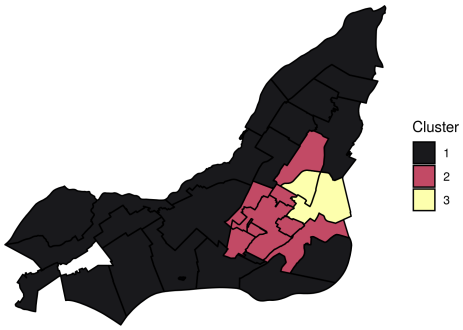


Figure A.39. PPRM sampled clustering

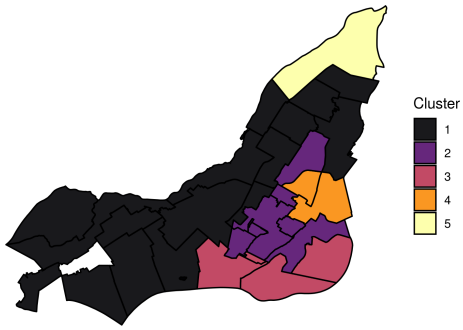


Figure A.40. PPRM sampled clustering

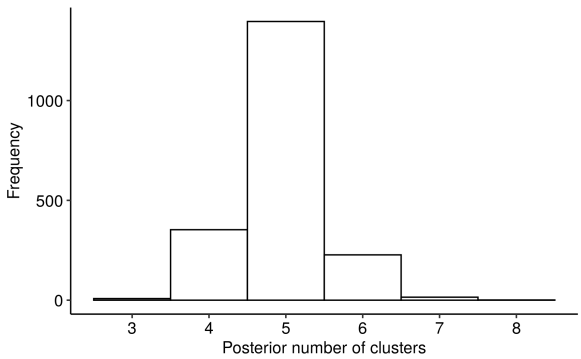


Figure A.41. PPRM posterior number of clusters

Figure A.42. Neighbourhood graph

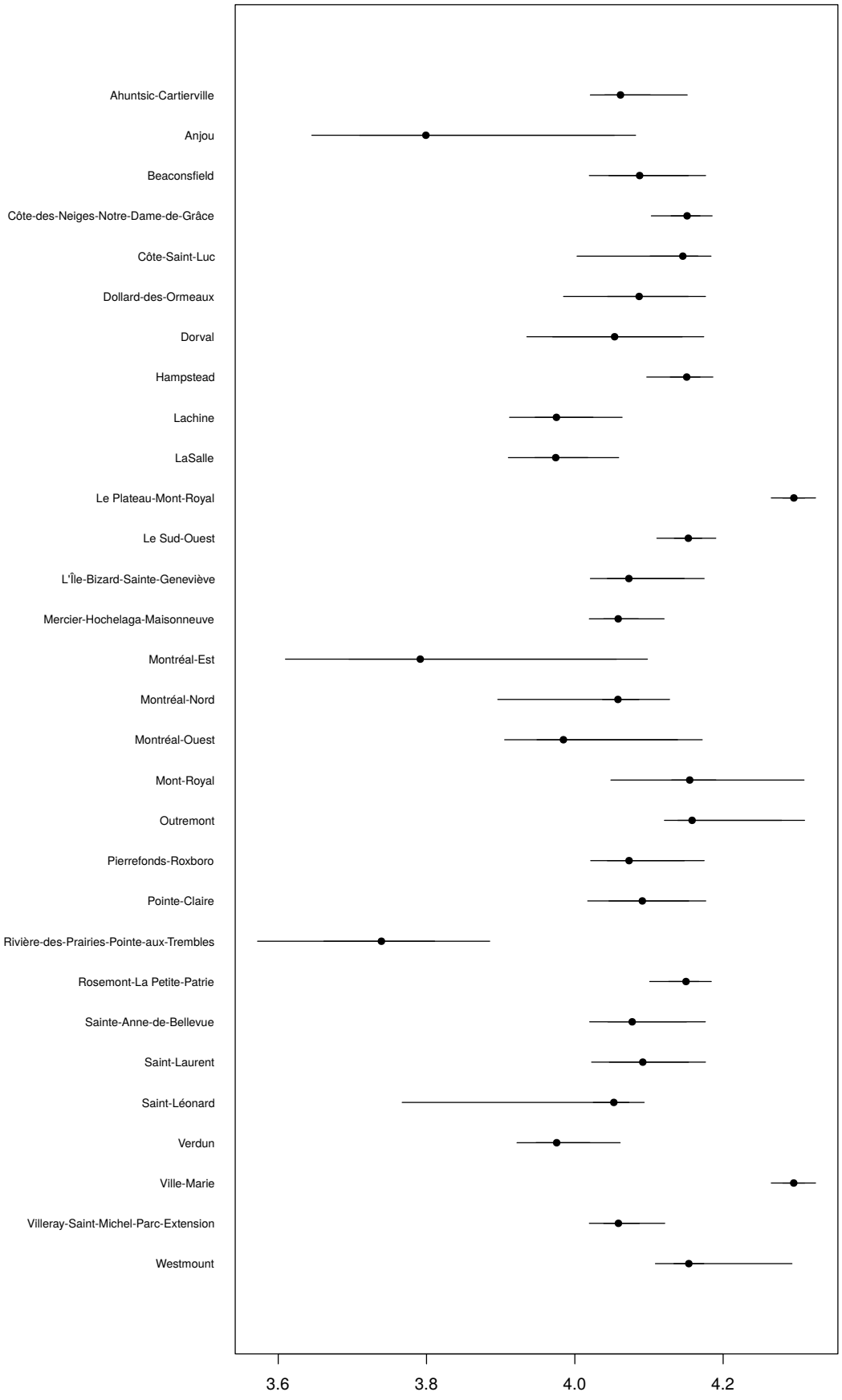


Figure A.43. Caterpillar plot

Inside Airbnb: New York

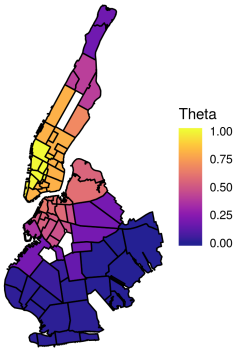


Figure A.44. PPRM posterior mean of $\boldsymbol{\theta}$

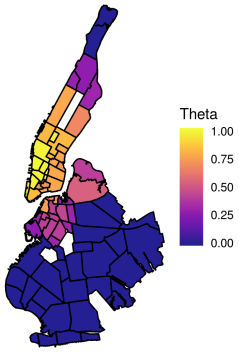


Figure A.45. Lasso estimated coefficients

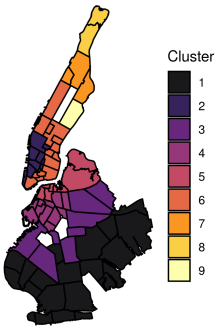


Figure A.46. PPRM sampled clustering

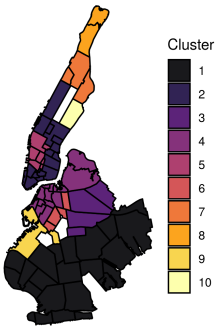


Figure A.47. PPRM sampled clustering

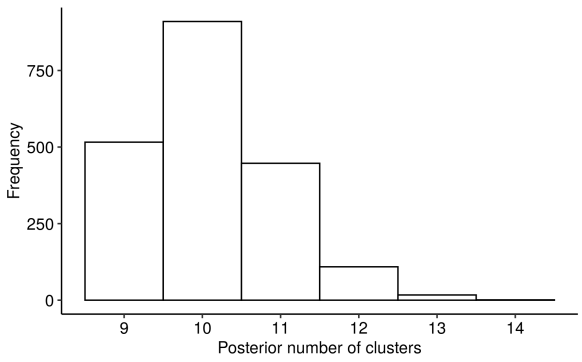


Figure A.48. PPRM posterior number of clusters

Figure A.49. Neighbourhood graph

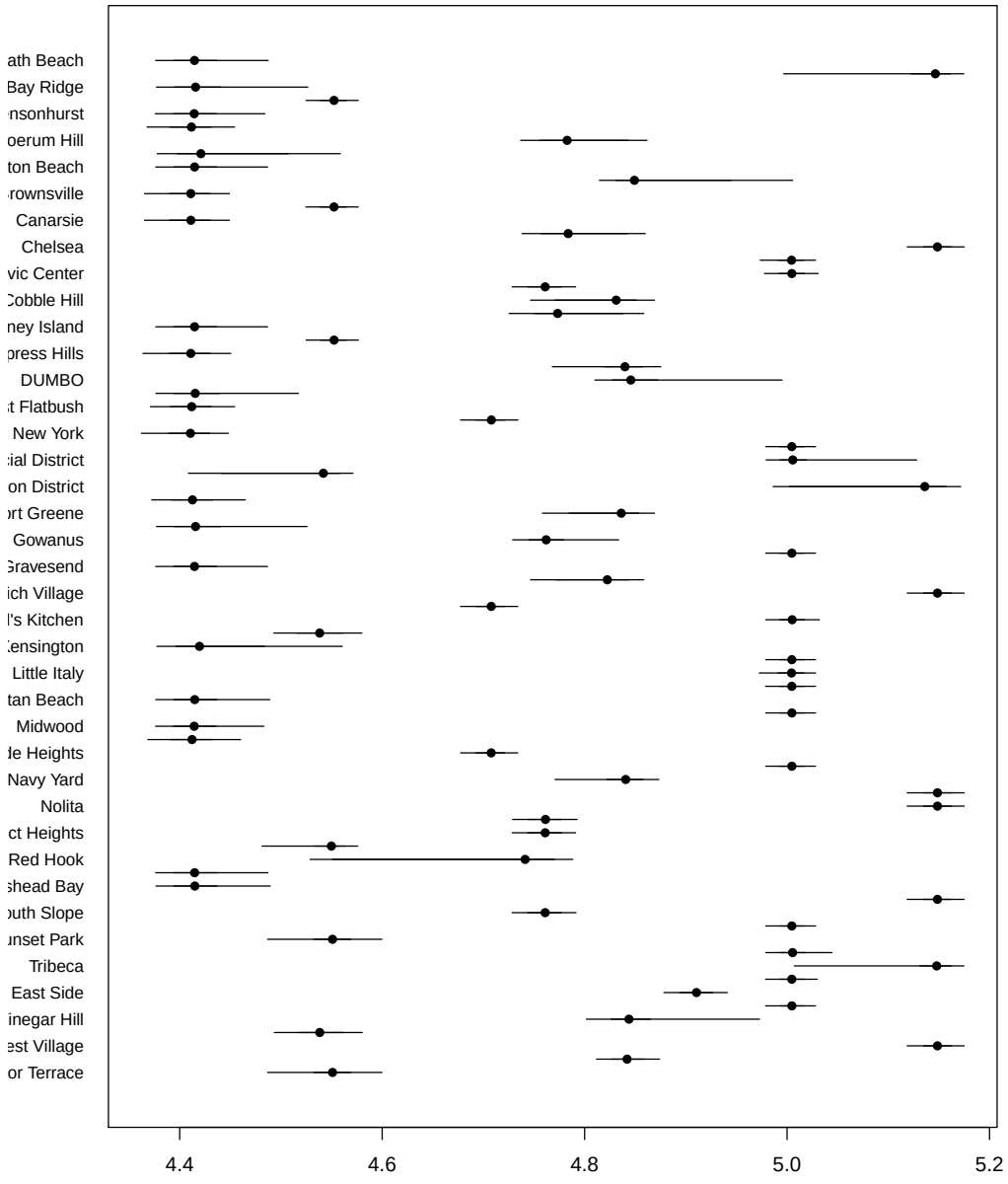
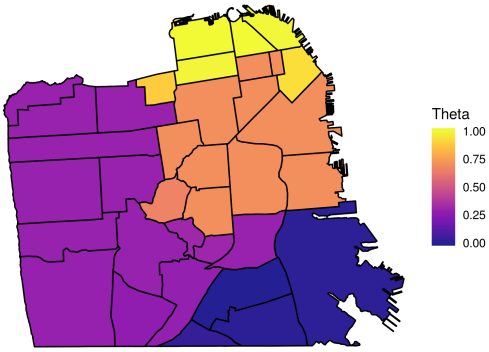


Figure A.50. Caterpillar plot

Inside Airbnb: San Francisco



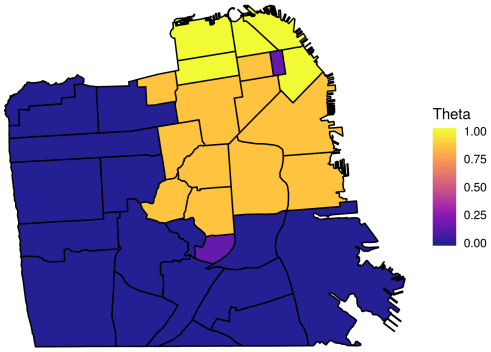
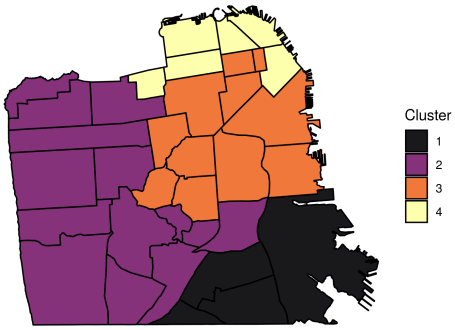


Figure A.51. PPRM posterior mean of $\boldsymbol{\theta}$

Figure A.52. Lasso estimated coefficien

0.5



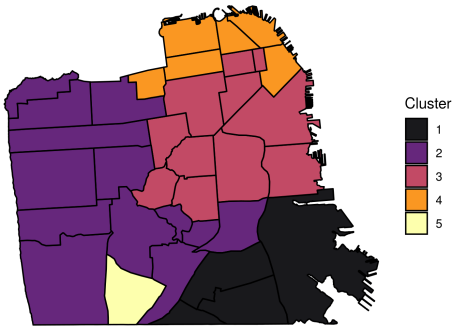
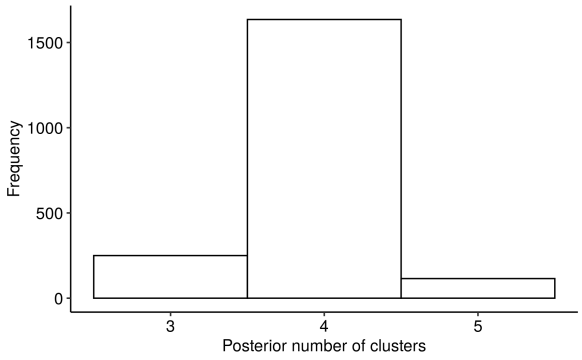


Figure A.53. PPRM sampled clustering

Figure A.54. PPRM sampled clustering



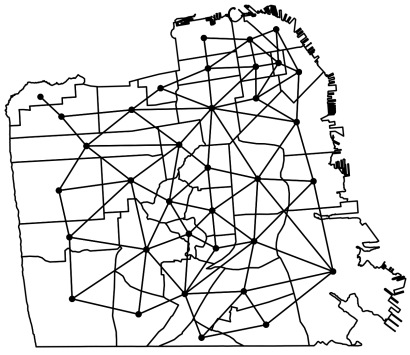


Figure A.55. PPRM posterior number of clusters

Figure A.56. Neighbourhood graph

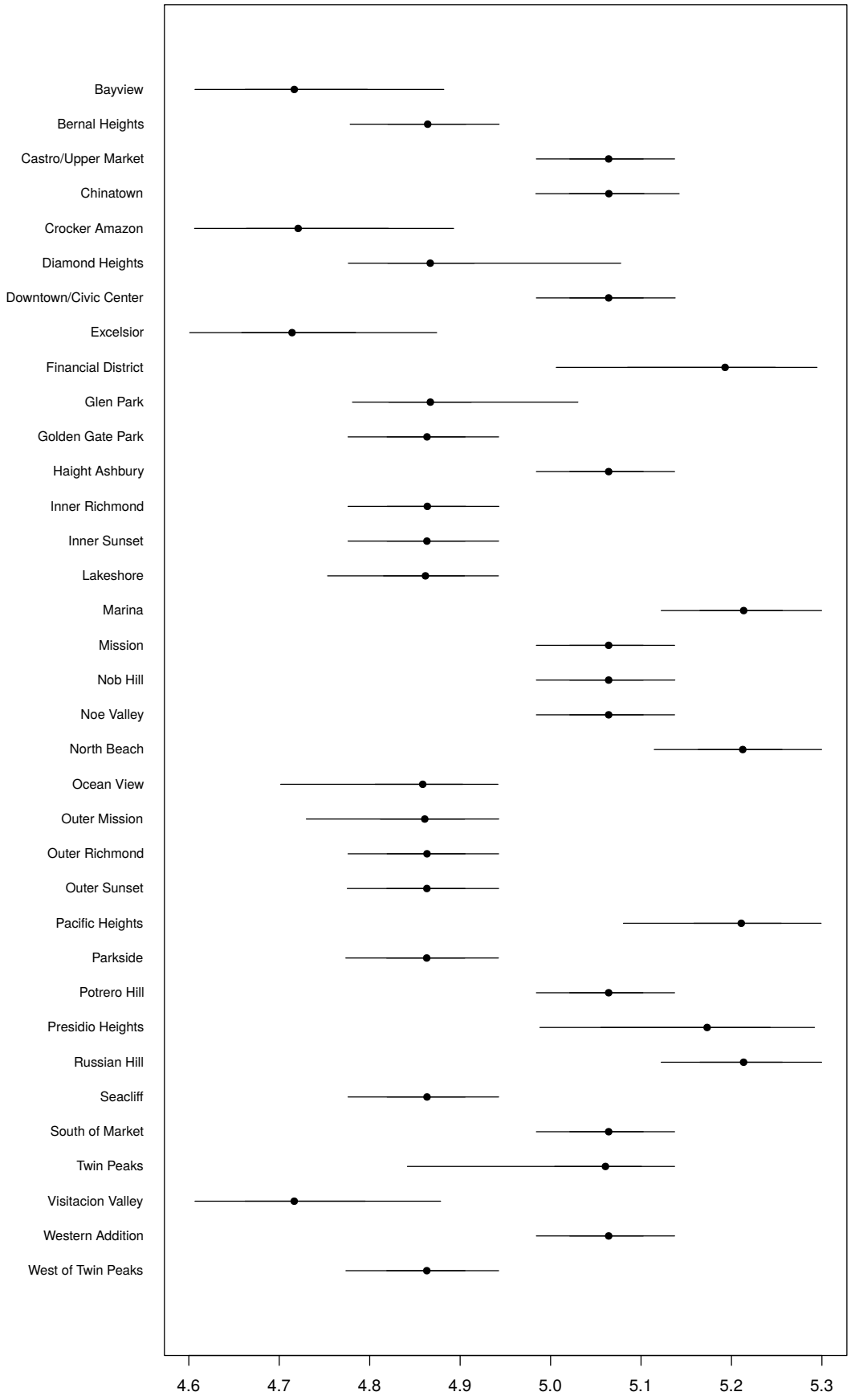


Figure A.57. Caterpillar plot

Inside Airbnb: Seattle

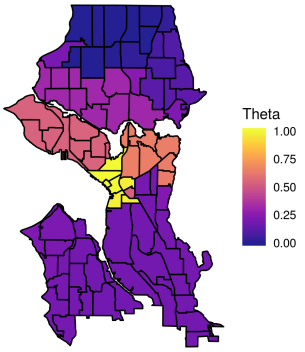


Figure A.58. PPRM posterior mean of $\boldsymbol{\theta}$

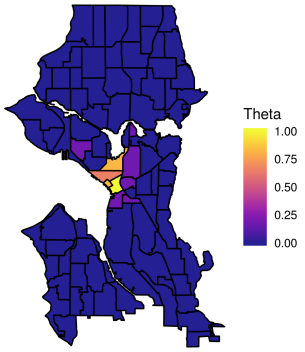


Figure A.59. Lasso estimated coefficients

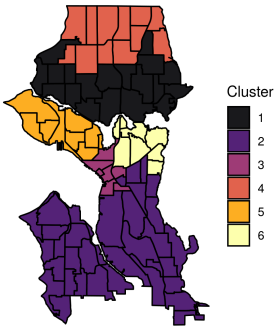


Figure A.60. PPRM sampled clustering

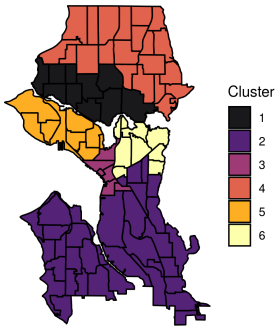


Figure A.61. PPRM sampled clustering

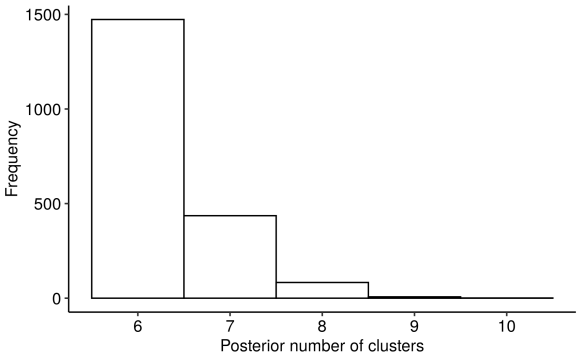


Figure A.62. PPRM posterior number of clusters

Figure A.63. Neighbourhood graph

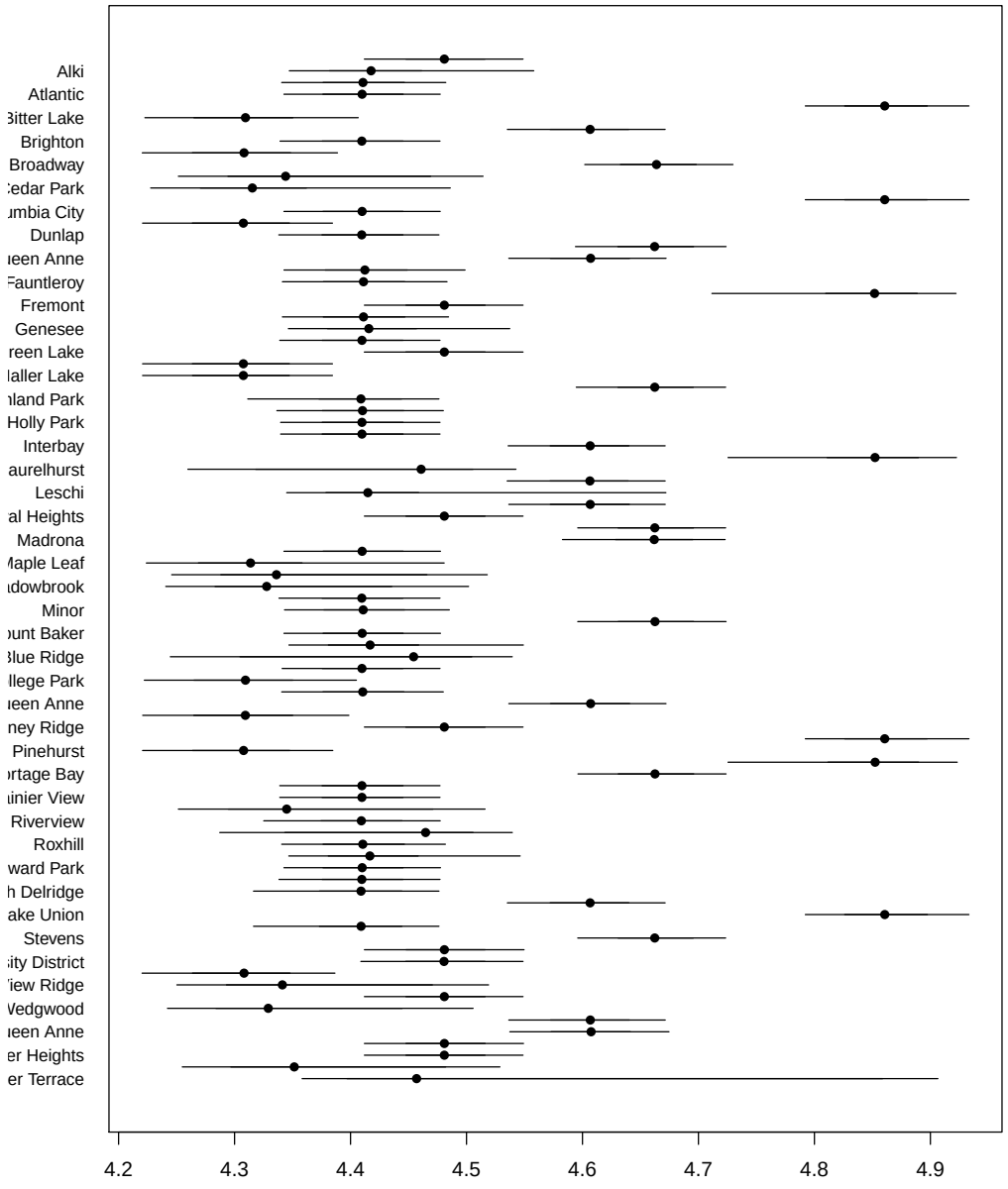


Figure A.64. Caterpillar plot

Inside Airbnb: Greater Toronto

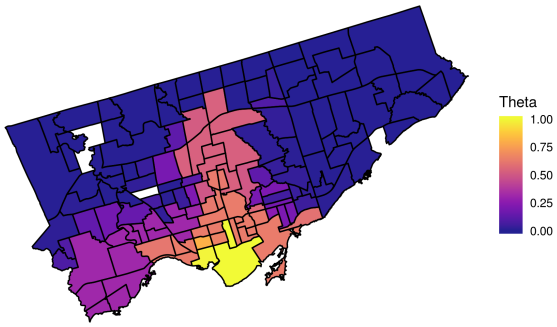


Figure A.65. PPRM posterior mean of $\boldsymbol{\theta}$

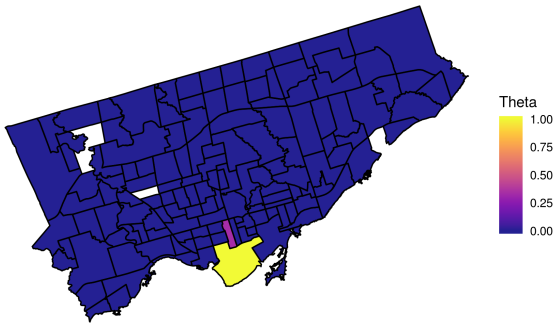


Figure A.66. Lasso estimated coefficients

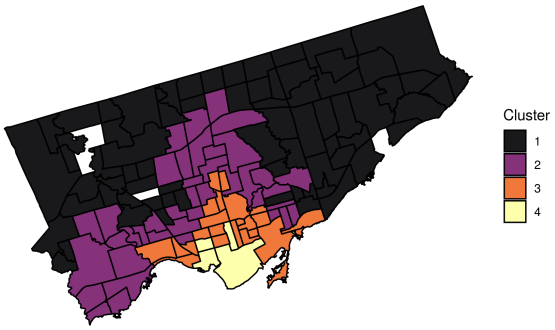


Figure A.67. PPRM sampled clustering

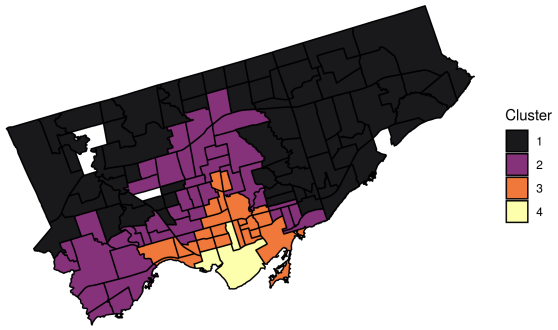


Figure A.68. PPRM sampled clustering

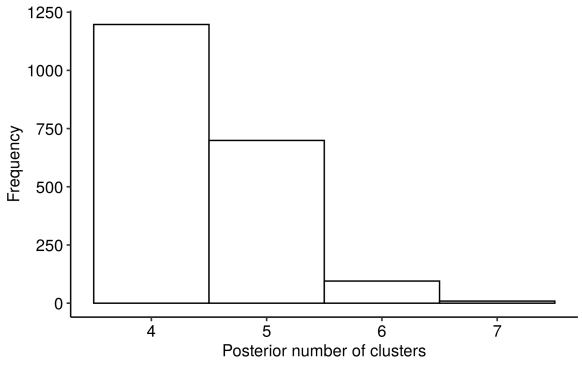


Figure A.69. PPRM posterior number of clusters

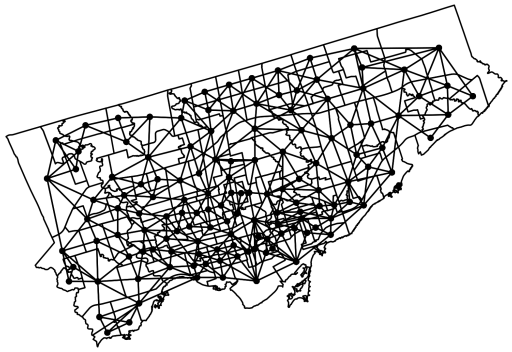


Figure A.70. Neighbourhood graph

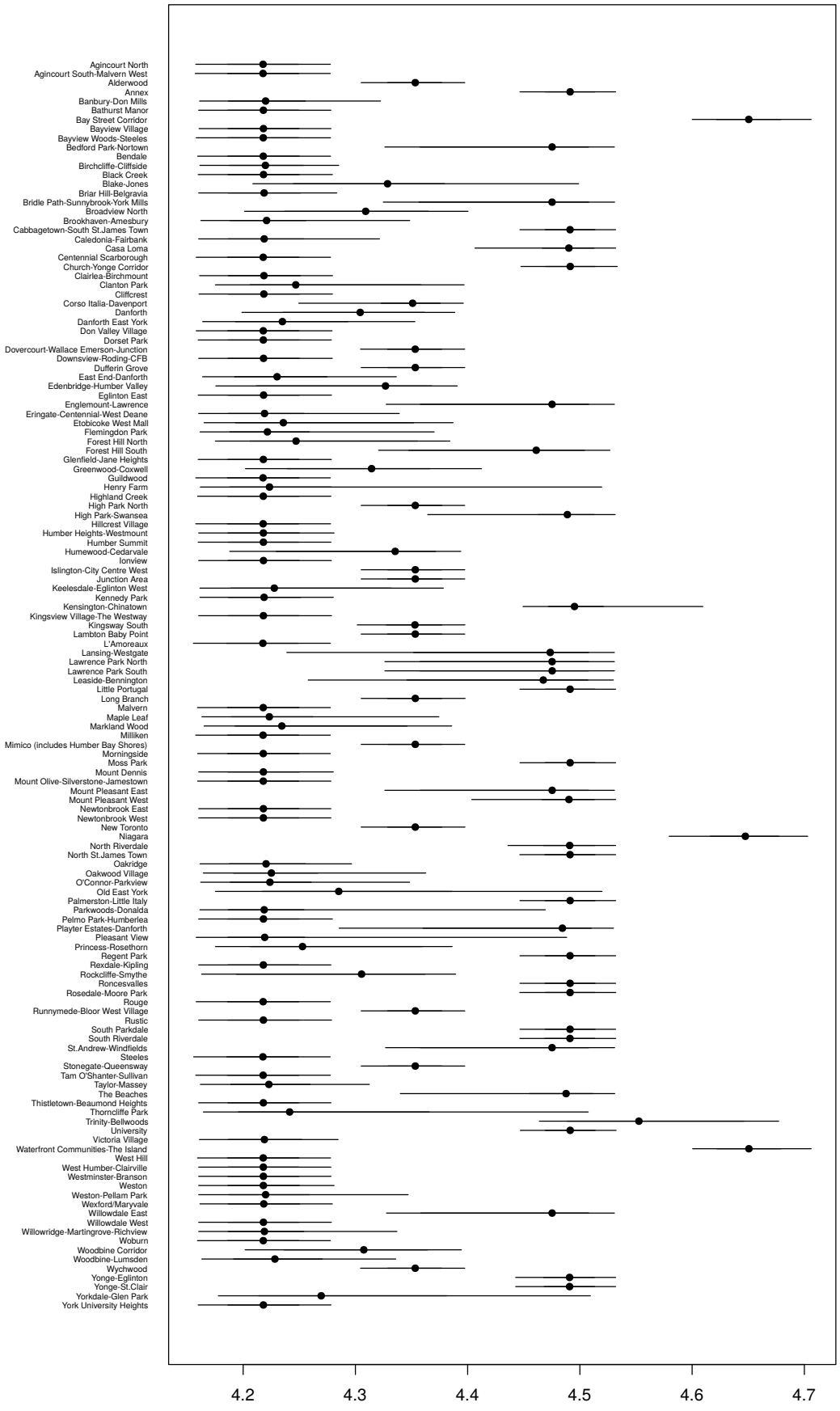


Figure A.71. Caterpillar plot