# Enhancing Interpretability in UMAP

A Methodological Approach to Understanding the Low-Dimensional Graph of UMAP

Miriam Kebernik

Thesis advisor: Dr. S. M. H. Huisman (Leiden University)

Defended on the 10th of March, 2025.

**MASTER THESIS**
**STATISTICS AND DATA SCIENCE**
**UNIVERSITEIT LEIDEN**

# Contents

# Abstract

Dimension reduction techniques are useful when working with large datasets to simplify data while preserving important information. One such technique is Uniform Manifold Approximation and Projection (UMAP), a nonlinear dimension reduction method. However, interpreting the low-dimensional embeddings produced by UMAP can be challenging, which limits its utility. This thesis addresses the interpretability challenge of UMAP by exploring the question: "How can the low-dimensional embedding of UMAP be interpreted?".

To answer this question, we developed an interpretation framework. After applying UMAP to the dataset (step 1), we apply an unsupervised clustering technique (step 2) to discretize the nonlinear structure and assign pseudo-labels to each data point. Next, we apply a supervised learning technique (step 3), where features serve as predictors and the pseudo-labels are the outcomes. By analyzing the goodness-of-fit metrics from this step, we can select features that provide insights into the global or local structure of the data (step 4), visualize their values, and provide interpretations (step 5).

We applied this interpretation framework to two different datasets. The first dataset contains numeric and categorical features about houses and apartments, while the second is a larger dataset comprising numeric gene expression values from brain samples. Applying the framework to both datasets provided valuable insights into the embeddings. Specifically, it helped identify which features are important to better understand the overall distribution of data points (i.e., the global structure) and which ones explain the layout of data points within smaller clusters or areas of the graph (i.e., the the local structure). Identifying these features enhances our understanding of the UMAP outputs.

Our interpretation framework is robust against changes in the fixed seed and demonstrates applicability across different datasets. This work enhances the practical utility of UMAP by making its embeddings more interpretable, thereby facilitating deeper data analysis. Future research could focus on developing alternative interpretation strategies or refining the proposed approach.

# 1 Introduction

Due to advances in technology and the increasing ability to collect and store large amounts of information from various sources, data analysts are often faced with big data. This abundance of data comes from multiple fields such as genomics, image analysis, document classification, astronomy, and atmospheric science. Although large datasets can help us analyze and understand complex phenomena, they can also present significant challenges for analysis and interpretation (Johnstone & Titterington, 2009; Pires & Branco, 2019).

Traditional approaches to data exploration, such as direct visualization or simple statistical analyses, are often insufficient to uncover the underlying patterns and relationships hidden within such data. Dimension reduction techniques aim to simplify the data into a more manageable form without significantly compromising its integrity (Johnstone & Titterington, 2009; Pires & Branco, 2019).

Principal Component Analysis (PCA) (Pearson, 1901), a linear dimension reduction technique, reduces the dimensionality of data by transforming it into a new space where the axes represent composite variables or dimensions that capture a significant portion of the original data's variance. Linear Multidimensional Scaling (MDS) (Torgerson, 1952), reduces dimensionality by visualizing the dissimilarity between two objects as distance in the lower-dimensional space.

While both methods can be effective for datasets where the relationships between features (also called variables, i.e. the columns of a dataset) are linear, these techniques usually do not work as well with nonlinear data structures. Important information can be lost when applying a linear dimension reduction technique to a nonlinear dataset because these techniques might not be able to preserve the internal structures of a nonlinear dataset in the low-dimensional space (Habowski et al., 2021).

Recognizing the limitations of linear dimension reduction techniques in handling complex, nonlinear data structures, researchers have developed a range of nonlinear dimension reduction methods. These methods, including techniques like Sammon Mapping (Sammon, 1969), Isomap (Tenenbaum et al., 2000), Laplacian Eigenmaps (Belkin & Niyogi, 2003), or t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten & Hinton, 2008), aim to maintain the inherent geometry of data by considering the nonlinear relationships between data points. However, despite their improved performance in certain scenarios, challenges such as computational efficiency, interpretability, and the ability to balance preservation of local versus global data structures remain.

## 1.1 UMAP

Uniform Manifold Approximation and Projection (UMAP), which is also a nonlinear dimension reduction technique, aims to reduce calculation speed and improve the preservation of both local and global structures (McInnes et al., 2018b). While a more in depth explanation of UMAP can be found in Chapter 2, the following is meant to give a general and short overview of this dimension reduction technique.

UMAP constructs a high-dimensional graph to represent the objects relationships, specifically the relationship of an object to its nearest neighbors, and then seeks to replicate this graph's structure as closely as possible in a lower-dimensional space. This process improves the preservation of both global and local data structures as compared to linear dimension reduction techniques. Global structure refers to the broad relationships among objects that are significantly separated in the dataset, capturing the overarching distribution patterns. Local structure, on the other hand, highlights the relationships among neighboring objects.

A practical demonstration of global and local structures is presented in a figure on a website by Coenen and Pearce (2022). They applied UMAP to the Fashion MNIST dataset (Xiao et al., 2017), which contains images of different types of clothing. In this context, the global structure refers to how all types of shoes are separated from other types of clothing, such as tops or bags. When referring to the local structure, we focus on a single cluster and look for a pattern. For example, within the cluster of shoes, we find a pattern, namely sandals tend to cluster with other sandals, and ankle boots with other ankle boots. This balance between preserving global and local structure makes UMAP particularly adept at maintaining the relationships between data objects across scales (McInnes et al., 2018b). Consequently, UMAP has been applied in various fields, from bioinformatics (Becht et al., 2019) to text analysis (Ordun et al., 2020), facilitating data exploration and data visualization, as seen with the Fashion MNIST dataset above.

Despite its strengths, UMAP's application also shows some challenges, most notably, the issue of interpretability. Unless the dataset contains images like in the example above, the resulting low-dimensional plots often lack clear explanations for the positioning and grouping of objects. In contrast to PCA, where the axes can be interpreted in terms of the original variables, UMAP's dimensions do not inherently carry such meanings. For analysts looking to build on the output of UMAP, this lack of interpretability can be an obstacle when determining how to proceed. Understanding why certain objects are close to each other and others are far from each other in the low-dimensional space becomes a difficult task, limiting the utility of the technique in scenarios where interpretability is important (McInnes et al., 2018b).

## 1.2 Research Aim

Addressing the interpretability challenge of UMAP's output is a critical step towards increasing the usability of nonlinear dimension reduction techniques in data science. If we can better understand the low-dimensional representation of datasets, then we can also make more informed decisions for next steps, gain deeper insights into data structures, and have more robust analyses across different disciplines.

There are different approaches in other dimension reduction techniques to help with the interpretation of the low-dimensional graph, such as a biplot in PCA and MDS. Although this method is not directly applicable to UMAP, it underscores the value of visual aids in enhancing interpretability. This thesis proposes to develop similar visual enhancements tailored to UMAP, details of which will be elaborated in the coming chapters.

Ultimately, this work seeks to establish a step-by-step framework that gives an answer to the research question: '**How can the low-dimensional embedding of UMAP be interpreted?**'. With the answer, we hope to enable users to interpret the resulting low-dimensional graphs with greater clarity and insight. By doing so, we extend the utility of UMAP beyond initial data exploration and make it a more integrated part of data analysis.

## 1.3 Terminology and Applications

In the context of this thesis, the rows are typically called ‘objects’, but can alternatively be described as ‘data points’. The columns of a dataset are generally referred to as ‘features’, although they may also be called ‘variables’. For example, if we have a dataset where each row holds information about a different person, and the columns are age, income, and nationality, then the person is considered the ‘object’ or ‘data point’, whereas age, income, and nationality are the ‘features’ or ‘variables’.

This thesis shows the application of the methods on the Melbourne Housing dataset from Kaggle (Pino, 2016), containing information about houses in Melbourne, and a Brain Samples dataset from the Allen Brain Atlas Institute (Allen Institute for Brain Science, 2024a), containing gene expression values from different brain samples. The Melbourne Housing dataset includes both categorical and numeric variables, while the Brain Samples dataset, despite consisting solely of numeric variables, is significantly larger. Showing how the framework can be used for different datasets intends to help the reader generalize and apply the framework to their own dataset.

## 1.4 Thesis Structure

The structure of this thesis is designed to systematically explore and apply the five-step interpretation framework. Therefore, a brief outline of the framework will provided here, with further details available in Chapter 2.3 and subsequent chapters.

First, UMAP is applied to the dataset (step 1), after which we use an unsupervised clustering technique (step 2) to discretize the nonlinear structure and assign pseudo-labels to each data point. Next, a supervised learning technique is applied (step 3), where features serve as predictors and the pseudo-labels are the outcomes. The goodness-of-fit metrics from this step support the feature selection process (step 4). The selected features aim to provide insights into the global or local structure of the data. Then, we visualize their values, and provide interpretations (step 5).

To delve into the five-step framework, this thesis is organized into seven chapters, complemented by appendices. The initial chapters lay the theoretical foundation by detailing existing statistical techniques used in steps 1 to 3. While these methodologies have been established by prior research, our contribution lies in the novel and previously unexplored application of these techniques to interpret the output of UMAP. The later chapters demonstrate how the five-step framework can be applied to two different datasets. To provide a clear roadmap of the thesis, a short explanation of each chapter is provided below:

- Chapter 1: Introduction — Provides a general introduction to the thesis and outlines the primary research question.
- Chapter 2: UMAP and Its Interpretation — Discusses the UMAP algorithm in detail, focusing particularly on the challenges associated with the output’s interpretability and providing a five-step interpretation framework.
- Chapter 3: Unsupervised Clustering Techniques — Explains two clustering techniques that are used in this research.
- Chapter 4: Supervised Classification — Describes two different classification methods used for feature selection.
- Chapters 5 and 6: Case Studies — Application of the discussed methodologies to two different datasets. Chapter 5 focuses on the Melbourne Housing dataset which contains a mix of numeric and categorical features, while Chapter 6 explores the Brain Samples dataset with over 50,000 numeric features.

- Chapter 7: Discussion — Concludes the thesis with a discussion on the effectiveness of the methods used, acknowledging potential limitations and suggesting areas for future research.
- Appendix: Includes additional resources such as explanations of the programming code, supplementary graphs, and other relevant documentation to support the research findings.

# 2 UMAP and Its Interpretation

This chapter provides an in-depth examination of the UMAP algorithm to enhance the understanding of interpretability issues associated with it. First, the three primary steps of UMAP are outlined. Then its application is shown on an example dataset. Lastly, the framework used to interpret the low-dimensional embedding of UMAP is shown, with the detailed implementation to be covered in later chapters.

## 2.1 The Three Primary Steps of UMAP

UMAP was developed by McInnes et al. (2018b) and can be broken down into the following 3 steps: 1. Finding the $k$-nearest neighbors; 2. Creating the fuzzy simplicial set for the high-dimensional data; and 3. Optimization of the low-dimensional topological representation. This chapter is mainly based on UMAP's original paper by McInnes et al. (2018b) and the website also written by McInnes et al. (2018a). If the reader is interested in a more in-depth and mathematical explanation than what is given below, then these two sources can be consulted.

### 0) Data Preparation

Before applying UMAP to any dataset, the data should be pre-processed. This includes handling any missing values, scaling variables if appropriate, and dummy-encoding categorical variables. However, since this is generally applicable to any analysis performed, this step is not considered to be part of the 3 steps of UMAP.

### 1) Finding the k-Nearest Neighbors: The Local Manifold Structure

After providing the pre-processed data to UMAP, the algorithm first identifies the $k$-nearest neighbors of each point using a distance metric. The default value for $k$ (the number of nearest neighbors) is 15 and the default distance metric is Euclidean distance, but both of these choices can be changed. Generally, a smaller value for $k$ means that more emphasis is put on the local structure and thus finer details can be captured, while a larger value for $k$ shows more of the global structure of the data, giving the 'bigger picture'. Moreover, different algorithms for finding the approximate nearest neighbors can be implemented. The default algorithm is usually NNDescent (Dong et al., 2011), which is faster than the conventional and exact way of finding k-nearest neighbors (Cover & Hart, 1967). The goal of finding the k-nearest neighbors is to build a graph which captures the local manifold structure. Hence, the result of this step is the approximation of the local manifold structure of the data.

### 2) From Local Fuzzy Simplicial Set to Global Topological Representation

Next, based on the local manifold structure and the identified k-nearest neighbors, a fuzzy simplicial set is built, which in our case corresponds to building a weighted graph. Instead of having a binary yes or no, weights between 0 and 1 are used as connection between two points. The weight between two points represents the probability that these two points are connected. Thus, the farther apart two points are, the lower the weight, and the closer two points are, the higher the weight. The weight of point $x_i$ to its neighbor $x_{i,j}$ is defined as:

$$w(x_i, x_{i,j}) = \exp\left(-\frac{\max(0, d(x_i, x_{i,j}) - \rho_i)}{\sigma_i}\right). \tag{2.1}$$

where:

- $x_{i,j}$ is one of the $k$-nearest neighbors of $x_i$, as identified in step 1;
- $d(x_i, x_{i,j})$ indicates the distance between these two points;
- $\rho_i$ indicates the distance from $x_i$ to its nearest neighbor;
- $\sigma_i$ is a scaling parameter, see equation 2.2.

The weight for the connection between $x_i$ and its nearest neighbor always equals 1, since $d(x_i,\ x_i\text{'s nearest neighbor}) - \rho_i = 0$. This is because $\rho_i$ is the distance between $x_i$ and its nearest neighbor. Plugging that into the equation (2.1), we can see that $e^0 = 1$.

The value of the scaling parameter $\sigma_i$ is specific to each point and set to be such that:

$$\sum_{j=1}^{k} \exp\left(-\frac{\max(0, d(x_i, x_{i,j}) - \rho_i)}{\sigma_i}\right) = \log_2(k). \tag{2.2}$$

Hence, each $x_i$ has its own scaling parameter $\sigma_i$, which varies based on the local density and distribution of its nearest neighbors. It should be noted that the weight between two points can differ, depending on which 'direction' you go. If, for example, point A has a lot of close neighbors, it might happen that a particular point B is not part of its $k$-nearest neighbors. However, if point B is not in a dense area and does not have a lot of close neighbors (compared to point A), then point A might fall into one of its k-nearest neighbors. Therefore, the weight from point A to point B will be 0, and therefore lower than the weight from point B to point A. To reconcile these differing weights, they are symmetrized in the following way:

$$w(x_i, x_j) = (w_i(x_i, x_j) + w_j(x_j, x_i)) - w_i(x_i, x_j) \times w_j(x_j, x_i), \tag{2.3}$$

where:

- $w(x_i, x_j)$ represents the symmetrized weight between $x_i$ and $x_j$;
- $w_i(x_i, x_j)$ represents the weight from $x_i$ to $x_j$;
- $w_j(x_j, x_i)$ represents the weight from $x_j$ to $x_i$.

The fuzzy simplicial sets discussed so far are based on the k-nearest neighbors from step 1 and thus, give us a local representation. To get a global representation of the data, the local fuzzy simplicial sets are combined to create a global topological representation of the high-dimensional data.

### 3) Optimization of the Low-Dimensional Topological Representation

In this step, the low-dimensional representation of the data is initialized through spectral embedding. This sets the initial layout of data points in the low-dimensional space, upon which the fuzzy simplicial set representing the topology is then constructed. The layout of the low-dimensional fuzzy simplicial set is optimized by minimizing the cross-entropy between the high-dimensional and the low-dimensional representation.

The equation for the cross-entropy is as follows:

$$\sum_{e \in E} \left( w_h(e) \log \left( \frac{w_h(e)}{w_l(e)} \right) + (1 - w_h(e)) \log \left( \frac{1 - w_h(e)}{1 - w_l(e)} \right) \right), \tag{2.4}$$

where $w_h(e)$ corresponds to the weight of the 1-simplex $e$ (i.e. the weight of the connection between two points) in the high-dimensional layout and $w_l(e)$ corresponds to the weight of the 1-simplex $e$ in the low-dimensional layout. The high-dimensional weights $w_h(e)$ are fixed and were computed in step 2, whereas the low-dimensional weights $w_l(e)$ are changing in this step to minimize the cross-entropy.

The cross-entropy equation (2.4) can be regarded as a kind of force-directed graph layout algorithm. The first term $w_h(e) \log \left( \frac{w_h(e)}{w_l(e)} \right)$ is minimized if $w_l(e)$ is as large as possible, which is the case when the points in the low-dimensional space are as close as possible to each other. On the other hand, the second term $(1 - w_h(e)) \log \left( \frac{1-w_h(e)}{1-w_l(e)} \right)$ will be minimized when $w_l(e)$ is as small as possible, meaning the points in the low-dimensional space must be far away from each other.

The low-dimensional weight is calculated as follows:

$$w_l(x_i, x_j) = \left(1 + a(\|x_i - x_j\|_2^2)^b\right)^{-1}, \tag{2.5}$$

where $a$ and $b$ are hyperparameters that are dependent on the chosen minimum distance between points. Of the equation 2.5, a gradient is calculated, which indicates the direction in which the points should move. As a result of adjusting the positions of these points, the squared Euclidean distance, $\|x_i - x_j\|_2^2$, between any two points $x_i$ and $x_j$ changes. Consequently, this modifies the values of $w_l$ of equation 2.5 and therefore also of equation 2.4.

Through this push and pull behaviour, which is balanced by the weights from the high-dimensional representation, the low-dimensional layout will settle in a local optimum, which represents the high-dimensional data as best as it can. The optimized layout gives the low-dimensional embedding of UMAP.

Since UMAP converges to a local optimum, rather than a global one, one important consideration is that UMAP includes a stochastic component and only achieves a local optimum of the low-dimensional embedding. Consequently, the exact coordinate values of each point can differ from one application to the next, with only the general relative positions remaining approximately consistent, though not identical, across different runs. This variability presents a challenge in interpreting the output, as UMAP can produce different embeddings for the same dataset. However, our approach primarily focuses on interpreting the output from a specific run of UMAP and does not attempt to resolve this issue of variability. Therefore, in our implementation of UMAP, the seed was fixed to ensure replicability.

## 2.2 Application of UMAP

To underscore the interpretive challenges of low-dimensional embeddings from UMAP, we present an initial visualization using the Melbourne Housing dataset in Figure 2.1. This dataset, detailed further in chapter 5.1, includes various parameters like price and the number of parking spaces for over 8,000 houses and other types of residences.

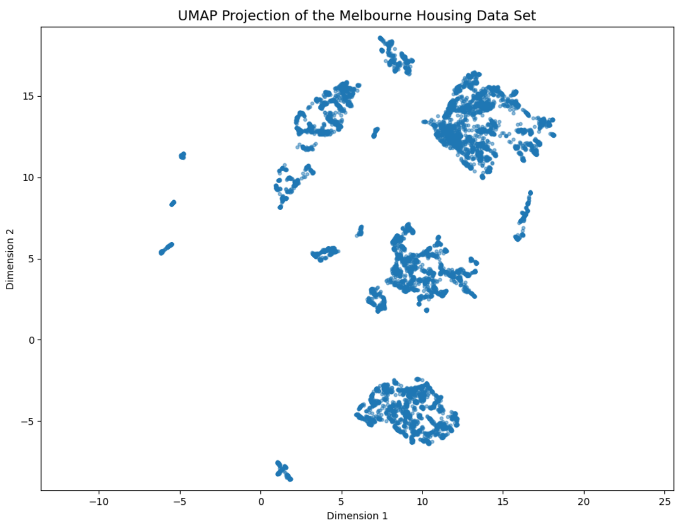

***Figure 2.1:*** *UMAP output illustrating the low-dimensional embedding of the Melbourne Housing dataset. Each point represents a residence.*

Figure 2.1 provides an early look at the type of visualization UMAP produces. It reveals cluster-like formations; however, the reasons behind these groupings remain unclear at this stage. As mentioned earlier, UMAP's transformation is based on a nonlinear process where data points are pushed or pulled according to their similarities, calculated from distance metrics. This nonlinear nature means that the original variables do not maintain a direct, linear relationship with the coordinates in the resulting map. Unlike PCA, where the axes can be interpreted as directions of increasing or decreasing values for specific features, UMAP's coordinates cannot be understood in this way. The main goal of this thesis is to provide an approach to solving the challenges associated with interpreting the low-dimensional output of UMAP. This approach will be laid out through detailed analysis and discussion in the following chapters.

## 2.3 Interpretation Framework of the Low-Dimensional Embedding

To address the interpretability challenge, we have developed a five-step interpretation framework to explore and better understand the low-dimensional embedding generated by UMAP.

The goal is to visualise those features that provide insights into the low-dimensional configuration and interpret the UMAP output based on these visualizations. Given that datasets analysed with UMAP typically contain many features, one step in the framework is feature selection. This step helps in deciding which features to visualize. In the following, the steps involved in our interpretation framework will be outlined:

1. **UMAP**: As a first step in the interpretation framework, UMAP needs to be applied to a dataset. The goal is to have the low-dimensional embedding of the high-dimensional data. The low-dimensional embedding will be used in the following steps.

2. **Unsupervised Clustering**: Perform unsupervised clustering on the objects within the low-dimensional embedding to assign provisional labels. This step generates a pseudo-labeled dataset that aids in subsequent supervised analysis. These pseudo-labels are necessary given that datasets processed with UMAP usually do not have labels. Additionally, even in cases where labels exist, our aim is to understand the structure revealed by the UMAP output rather than analyse pre-existing labels in the raw data.

3. **Supervised Classification**: Implement classification analyses, using each feature as a predictor and the pseudo-labels as the outcomes. This method assesses how effectively each feature predicts the cluster formations.

4. **Feature Selection**: Perform feature selection based on the outcome of the classification step. Different selection criteria might be applied to find features that are important for the global structure and those that are important for the local structure.

5. **Visualization and Interpretation**: Create visualizations for influential features identified in step 4. These visualizations map feature values onto the UMAP graph. Interpret the output based on the visualizations.

For smaller datasets, directly visualizing the values of each feature in the low-dimensional embedding is possible, simplifying the visual interpretation of the low-dimensional embedding. However, for larger datasets, typically processed by UMAP, where visual methods become impractical due to their complexity and scale, the outlined five-step interpretation framework is necessary.

In chapter 6.4 we extend our methodology beyond the selection of individual features in the following way: Before performing the supervised classification step, we first group the features and then perform classification. This means that the predictors are now the feature groups, rather than individual features as outlined above. The mean values of the selected feature groups are then visualized in the final step. The details and reasons behind this extension will become clearer in chapter 6.4.

# 3 Unsupervised Clustering

In this chapter, we introduce unsupervised clustering techniques, which are used in the second step of our framework. The goal of this clustering step is to provide pseudo-labels for subsequent analysis involving supervised classification techniques. The two clustering techniques described in the following are Gaussian Mixture Models and Hierarchical Clustering. However, it is important to note that the suitability of these techniques can vary, and other unsupervised clustering methods may also be appropriate depending on the specific characteristics of the user's dataset.

## 3.1 Gaussian Mixture Models

Gaussian Mixture Models (GMMs) are probabilistic models, typically used for clustering and density estimation. When applying a Gaussian Mixture Model, the user first has to specify the number of clusters the algorithm should identify. The algorithm then computes the probability of each object belonging to each of these clusters. Lastly, each object is assigned to the cluster with the highest probability of membership, thus providing a cluster label for each object (Hastie et al., 2009).

An underlying assumption is that the data is generated from a mixture of several Gaussian distributions, each representing a different cluster. Each Gaussian distribution has a mean and a covariance matrix, both of which are unknown and need to be estimated. Moreover, each cluster has its own mixing proportion, e.g., one cluster might make up 30% of all objects, while the remaining 70% comes from one or more other clusters. These mixing proportions are also unknown and need to be estimated. The three unknown sets of parameters, namely i) the mean vector of each Gaussian distribution, ii) the covariance matrix of each Gaussian distribution, and iii) the mixing proportions, are estimated through the expectation-maximisation (EM) algorithm. The EM algorithm iteratively refines the parameters to maximize the likelihood of the data under the GMM. Initially, guesses are made for these three sets of parameters. The algorithm then alternates between the following steps until convergence:

- **E-Step (Expectation)**: Compute the expected cluster probabilities for each object, i.e. the probability that a certain object belongs to a cluster, based on the values of the three parameters (mixing proportions, mean and covariance matrix of each Gaussian distribution).
- **M-Step (Maximization)**: In the E-Step, each object is assigned a cluster probability, indicating the likelihood of the object belonging to each cluster. Using these probabilities, the estimated values for the mixing proportions, the mean and the covariance matrix of each Gaussian distribution are now updated.

Once convergence (or another stopping criterion) is reached, the estimated values for the three parameters are used to determine the final cluster labels of all objects.

Applying restrictions to the covariance matrix can simplify the model and enhance stability. Conversely, using fewer or no restrictions increases the complexity and flexibility of the model. These restrictions affect the assumed volume, shape, and orientation of each cluster. The most restrictive scenario occurs when the covariance matrix of each model (i.e. each cluster) is a (scaled) identity matrix. This assumption leads to clusters that are spherical in shape and have the same volume. In the least restrictive case, each cluster has its own unique covariance matrix, allowing for ellipsoidal clusters with varying volumes, shapes, and orientations.

Another hyperparameter to be chosen is the number of assumed underlying clusters. Although there is no perfect method that specifies the number of clusters, one solution is to choose the number of clusters based on the lowest value for the Bayesian Information Criterion (BIC). BIC can support the model selection process by balancing model fit and model complexity, favoring models that achieve a high likelihood while penalizing those with too many parameters, which in this case means too many clusters. A lower score indicates better model fit.

In summary, the Gaussian Mixture Model algorithm is a versatile and useful tool for clustering, providing a choice between flexibility and simplicity based on the properties of the dataset.

## 3.2 Hierarchical Clustering

Hierarchical clustering is another unsupervised clustering technique. It recursively partitions a dataset into clusters, with one of the primary approaches being agglomerative, i.e., bottom-up approach (Cohen-Addad et al., 2019). In agglomerative clustering, each data point initially represents its own cluster. Clusters are then merged based on their proximity, with the closest pairs of clusters (or data points) being combined in each iteration. This merging process continues until all data points are consolidated into a single cluster (Cohen-Addad et al., 2019). One can visualize this process using an upside-down tree diagram, known as a dendrogram, where individual data points are represented at the bottom and progressively merge into larger clusters as one moves up the diagram.

The main two hyperparameters in hierarchical clustering include the distance metric and the linkage method. The distance metric determines how to calculate the distance between two objects or clusters. Common metrics include Euclidean distance, which measures the shortest path in geometric space; Manhattan distance, ideal for grid-like paths; and correlation distance, which assesses the degree to which two profiles are similar.

The linkage method specifies the rule for calculating the distance between clusters by selecting specific reference points within each cluster. The options include:

- Single Linkage: The distance between two clusters is determined by the shortest distance between any two points, one from each cluster.
- Complete Linkage: The distance is based on the furthest distance between any two points, one from each cluster.
- Average Linkage: The distance is the average of all distances between all pairs of points, one from each cluster.

To determine the final number of clusters, a ‘cut-off’ height on the dendrogram is selected. The number of ‘branches’ or lines that are cut through determines the number of clusters you obtain at this height. Any data points connected to a branch below the cut-off belong to the same cluster.

Each linkage method has a different effect on the clustering process and the shape of the dendrogram. This in turn affects the clustering outcome. Choosing the appropriate metric and method depends on the characteristics of the dataset and the desired clustering result.

## 3.3 Conclusion

An important difference between hierarchical clustering and GMM is that hierarchical clustering does not require you to prespecify the number of clusters. Instead, you can examine the dendrogram to identify significant jumps in height between the cluster-merging steps. These jumps often indicate natural divisions within the data, allowing for an intuitive determination of the number of clusters based on the structure of the data itself. This flexibility makes hierarchical clustering particularly valuable for exploratory data analysis where the optimal number of clusters is not known beforehand.

However, as mentioned before, the two clustering methods described here represent only a small selection of all available techniques. Many other unsupervised learning methods, which cluster data points based on similarities or distances, may also be appropriate for this clustering task.

# 4 Supervised Classification

In the previous step, the continuous values of the embedding were discretized, meaning each object now belongs to a class rather than having two numeric coordinates assigned to it. This allows for the use of classification techniques, which is the third step in our five-step interpretation framework. The supervised classification step provides a quantitative measure of how well each feature predicts the pseudo-labels. This quantitative measure will then be used to support the feature selection step, which is the next step in the interpretation framework. In the following, two categorical classifiers will be discussed, namely Multinomial Logistic Regression and Random Forest.

Before discussing the chosen classification methods, we acknowledge that our approach is unconventional. Specifically, it is not common to apply unsupervised clustering to a dataset and then use supervised classification on the same data using the labels generated from the unsupervised method. Consequently, the model fit values obtained from the supervised classification should not be taken at face value. Instead, they serve as a guide to determine which features are most relevant for visualization.

## 4.1 Multinomial Logistic Regression

Multinomial logistic regression is an extension of logistic regression used when the outcome variable is categorical and consists of more than two unordered categories (Goeman & Le Cessie, 2006). Unlike binary logistic regression, which predicts the probability of belonging to one of two categories, multinomial logistic regression is used to predict probabilities for multiple outcome categories, making it suitable for more complex classification tasks.

In multinomial logistic regression, numeric features can directly be used in the model, whereas categorical features must be dummy-coded before inclusion. This transformation is necessary because multinomial logistic regression relies on linear combinations of all predictors, which presupposes that these predictors are numerical.

In the context of this study, multinomial logistic regression is applied to predict the cluster labels assigned in the previous step. The predictors are the features. To identify the features that best predict the pseudo-labels, we construct separate multinomial logistic regression models for each feature and compare their goodness-of-fit.

While some may argue that fitting a single MLR model with all features and examining the p-values can also determine feature importance, this approach has limitations when dealing with both numeric and categorical features. Our goal is to compare the predictive power on a feature-level, rather than a variable-level. For example, the previously mentioned Melbourne Housing dataset contains the features 'price' and 'region name'. 'Region name' is categorical and contains 8 different regions. Therefore, this feature needs to be dummy-coded, resulting in each level (i.e., each region) becoming its own variable. If we include 'region name' and 'price' in the same model and look at the p-values, then the comparison is made between price and a single region, rather than the predictive power of the feature ‘Region name’. If instead we create separate models for these two different features, we can compare the predictive power of features by examining the goodness-of-fit of each model as a whole. Certain limitations for this comparison apply, which are discussed later in this chapter.

Another important difference between a univariate MLR model (i.e. one feature per model) and a multivariate MLR model (i.e. multiple features in a model) is that if two informative variables are highly correlated, in a multivariate model only one of them might have a low p-value, but in univariate models they will both have low p-values. Consequently, building separate MLR models for each feature and comparing their goodness-of-fit is a more suitable method in this context.

One performance metric to evaluate these models is the F1-Score, particularly useful when dealing with imbalanced classes (Chase Lipton et al., 2014). The F1-Score, the harmonic mean of precision and recall, offers a balanced measure of a model’s accuracy. Precision is the ratio of correctly predicted positive observations to the total predicted positives, while recall is the ratio of correctly predicted positive observations to all observations in the actual class.

However, since categorical predictors with a lot of levels have the advantage of being able to fit more coefficients - since categorical features are dummy-coded - a more appropriate performance metric is BIC. As described in chapter 3.1, BIC accounts for model complexity, penalizing models with more parameters. Therefore, BIC is suitable for comparing the fit of models with categorical predictors of varying complexity and those with numeric predictors.

As previously discussed, the goodness-of-fit metrics should not be interpreted at face value due to the simultaneous application of unsupervised and supervised methodologies on the same dataset. Nonetheless, the BIC score can provide valuable guidance regarding which features may be most beneficial to visualize.

## 4.2 Random Forest

Random Forests are an ensemble learning method primarily used for classification and regression tasks (Breiman, 2001). In the case of classification, this technique builds multiple decision trees during training and outputs the mode of the classes of the individual trees.

A Random Forest algorithm operates by constructing a multitude of decision trees at training time. Each tree is built from a different subset of the training data, created through bootstrapping. Additionally, during the construction of these trees, the algorithm selects random subsets of features at each split, ensuring that the trees are diverse and reducing overfitting.

In this study, Random Forests are used to determine the importance of each feature for label prediction. Feature importance, often quantified by the ‘Mean Decrease in Impurity’ or ‘Gini Importance’, indicates how influential a feature is in determining the target outcome (Breiman, 2001). This metric is derived by assessing the reduction in Gini impurity that each feature contributes across all trees within the forest. The calculation process is outlined as follows:

- During the construction of each tree, the algorithm considers different features and thresholds to split the data at each node. The goal is to choose the feature and threshold that most effectively reduce impurity.
- The impurity reduction is calculated as the difference between the impurity of the parent node and the weighted sum of the impurities of the child nodes. This reduction is attributed to the feature used for the split.
- This process is repeated for every node in every tree. The total impurity reduction attributed to a feature across all splits and all trees is accumulated.

- To provide a relative importance score, the accumulated reduction in impurity for each feature is normalized. The resulting scores indicate how much each feature contributes to reducing impurity across the entire forest.

Random Forests can handle a large number of input variables without overfitting and are robust to noise. However, there are two main disadvantages: First, the transparency of variable importance rankings is limited. Given the study's goal to make the low-dimensional embedding of UMAP more interpretable, it is crucial that each step is understandable and traceable. Second, categorical features must be dummy-coded, and there is no method to combine the importance of each level of a categorical feature into a comprehensive importance score comparable to a numeric feature. Nonetheless, Random Forests are included as an alternative feature selection method to multinomial logistic regression, especially since it can be used for datasets with only numeric features.

## 4.3 Conclusion

The two feature selection techniques discussed in this chapter represent only a subset of the many methods available for predictive modeling. Each technique offers distinct advantages depending on the specific requirements of the analysis and the nature of the data. Multinomial logistic regression is particularly advantageous for datasets with categorical or correlated predictors. It not only assesses the goodness-of-fit in predicting labels but also incorporates model complexity through metrics like the Bayesian Information Criterion (BIC).

Conversely, Random Forest is robust against outliers and noise, and adept at handling many features, which makes it well-suited for large datasets with numeric features.

Both methods are effective at indicating how well a feature can predict the label of an object, which can give us insights into how important this feature is for the global structure of the UMAP embedding. It is important to note that features that predict the assigned labels well (i.e. have a high goodness of fit) might only be relevant in explaining the global structure and not necessarily the local structure. How feature importance can be used to explain the local structure is explained in chapter 6.6.

# 5 Application on the Melbourne Housing Dataset

This chapter demonstrates how we apply the five-step interpretation framework to better understand UMAP's low-dimensional embedding of the Melbourne Housing dataset. All analyses were performed with PyCharm version 2023.3.5, using Python version 3.12.

This chapter starts with an overview of the Melbourne Housing dataset. Next, we explain how UMAP is applied to this dataset and the resulting low-dimensional embedding is visualized (step 1 of interpretation framework). Hierarchical clustering is performed on this low-dimensional embedding in order to generate pseudo-labels (step 2). The next step, performing multinomial logistic regression, uses theses pseudo-labels as outcome variables (step 3). Multinomial logistic regression provides us with BIC scores, which inform the subsequent feature selection step (step 4). The chapter concludes by presenting and discussing visualizations of selected features, offering interpretations of the low-dimensional embedding produced by UMAP for the Melbourne Housing dataset (step 5).

## 5.1 Description of the Melbourne Housing Dataset

The Melbourne housing dataset, downloaded from Kaggle (Pino, 2016), originally consists of 34,857 objects, which in this case are residences, and 21 features, including both categorical and numeric features, which describe each residence. Although UMAP was not originally intended to be used with categorical features, we took into account that users might still want to use UMAP on a dataset that has both categorical and numeric features. Therefore, we decided to work with this dataset with some adjustments. Categorical features will have to be dummy encoded, meaning that features that have many unique values will take up much more space in the dataset than those with less unique values or numeric features. Therefore, we deleted the following categorical features: Address (34,009 unique values), Suburb (315 unique values), Real Estate Agent (250 unique values), Postcode (194 unique values) and Date (78 unique values). Furthermore, the dataset had missing values. Deleting all objects with missing values is not always advisable. However, since this dataset is only used to showcase the developed method, rather than to guide decision-making, we chose to delete all rows with missing values. After deleting the above-mentioned categorical features and all rows with missing values, this dataset now has 8887 objects and 16 features, which are:

- Building Area: building size in square metres
- Council Area: governing council for the area (33 unique values)
- Distance to CBD: distance of the residence from the Central Business District in kilometers
- Land Size: land size in square metres
- Latitude: latitude of the residence's location

- Longitude: longitude of the residence's location
- Number of Bathrooms
- Number of Bedrooms
- Number of Rooms
- Parking Spaces: number of parking spaces
- Price: residence sold for this price in Australian Dollars
- Region Name: general region the residence is located in (8 unique values)
- Residences in Suburb: the number of residences that exist in the suburb
- Residence Type:
    - House - includes: house, cottage, villa, semi, terrace
    - Unit - includes: unit, duplex
    - Townhouse
- Selling Method: how the residence was sold:
    - S - residence sold
    - SP - residence sold prior
    - PI - residence passed in
    - VB - vendor bid
    - SA - sold after auction
- Year Built: the year the residence was built.

## 5.2 UMAP (Step 1)

The preprocessing of the Melbourne Housing dataset involved dummy coding the categorical features and standardizing the numeric features to have a mean of 0 and a standard deviation of 1. Subsequently, UMAP was applied to the processed Melbourne Housing dataset. As mentioned before, UMAP's embedding is the result of a local optimum, not a global one. This introduces randomness as another run can result in a different local optimum, and thereby a different low-dimensional embedding. Therefore, we fixed the seed to ensure replicability. The result is a 2-dimensional embedding visualized in Figure 5.1.

## 5.3 Unsupervised Clustering: Hierarchical Clustering (Step 2)

Hierarchical clustering was applied to categorize each object within the dataset using the 'complete' linkage method. This approach was selected because it tends to create more balanced and well-separated clusters by considering the maximum distance between observations in different clusters. The dendrogram depicted in Figure 5.2 illustrates the heights at which various clusters merge. Selecting an appropriate number of clusters can be somewhat subjective; however, an effective strategy is to cut the dendrogram at a point where it avoids excessive splitting, yet allows for sufficient cluster formation. For this analysis, a cut-off at a height of approximately 13 was selected, resulting in five clusters. To make the cutoff point clearer, a line was added to the graph by the author. As shown in Figure 5.3, objects are color-coded according to the cluster they belong to. This visualization helps in understanding the grouping and separation of the objects based on hierarchical clustering.

In Appendix A.1 the application of Gaussian Mixture Models on this dataset is shown. The results from both clustering methods are rather similar. For the sake of clarity and focus, only the results from hierarchical clustering will be used as pseudo-labels for the next step.

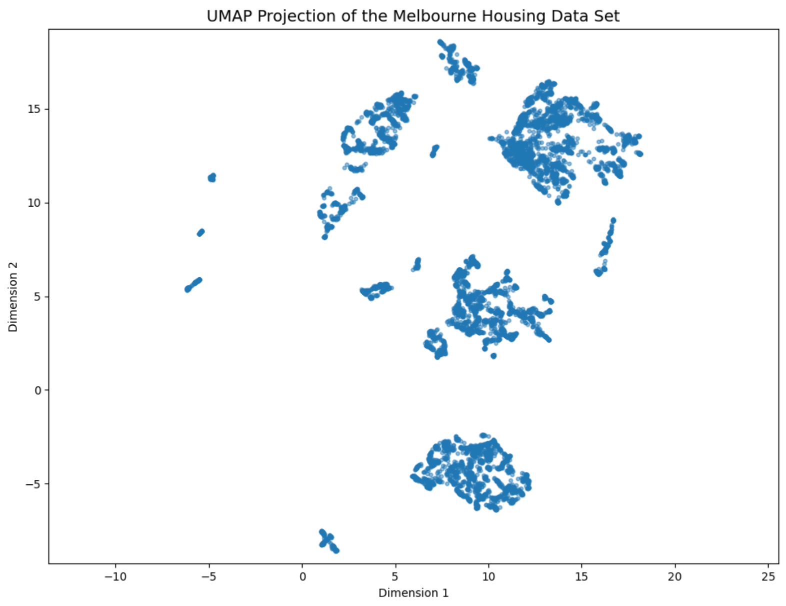

***Figure 5.1:*** *UMAP output showing the low-dimensional embedding of the Melbourne Housing dataset. Points represent residences, such as houses or apartments.*

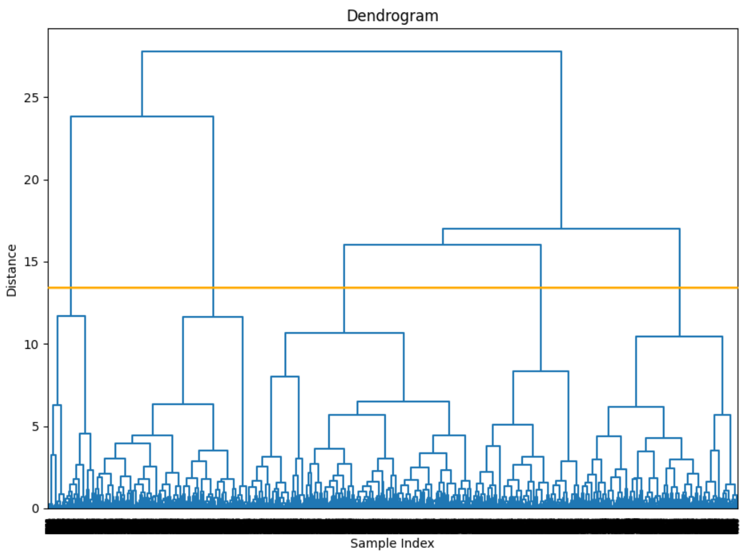

***Figure 5.2:*** *Dendrogram of Hierarchical Clustering for the Melbourne Housing dataset. It displays the clustering process, with cluster merges shown at various dissimilarity levels on the y-axis, and individual samples indexed on the x-axis.*

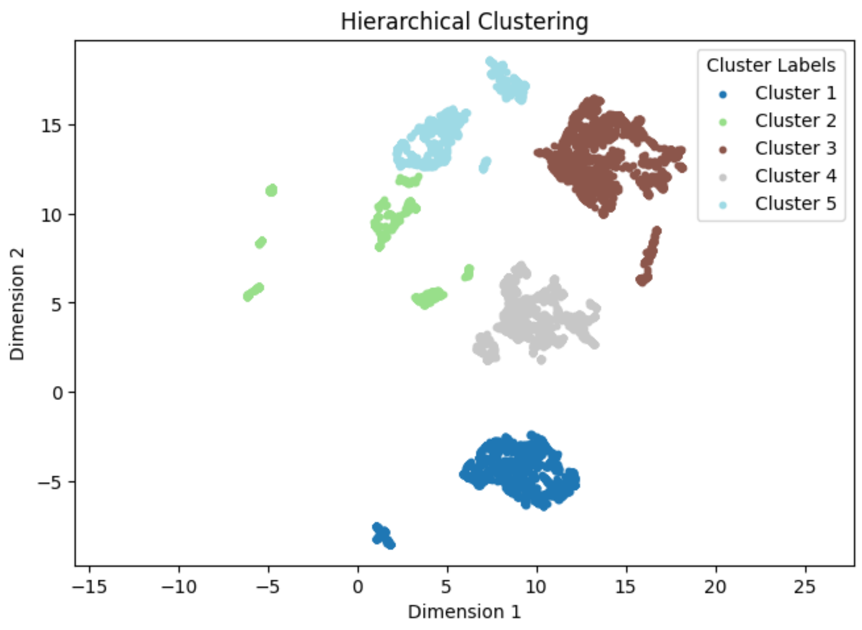

***Figure 5.3:*** *This plot visualizes the five clusters identified at a cut-off height of 13 using complete linkage, with each cluster represented by a different color.*

## 5.4 Supervised Classification: Multinomial Logistic Regression (Step 3)

In chapter 4, two classification methods were explored: multinomial logistic regression and random forests. However, due to the limitations of random forests in handling datasets that include categorical features, we will only apply multinomial logistic regression to the Melbourne Housing dataset.

Multinomial logistic regression is applied to determine which features best predict the class labels obtained from hierarchical clustering. As discussed in 4.1, this method is suitable even with a mixture of categorical and numerical data, due to the use of the Bayesian Information Criterion (BIC), which accounts for model complexity. The analysis is done on the standardized numeric features and the dummy-coded categorical features. To ensure consistency in how the data is split each time the analysis is run and ensure replicability of the results, the seed was fixed. Having a test set mitigates the effect that overfitting has on the goodness-of-fit metric. Therefore, splitting the dataset is crucial for obtaining accurate BIC and F1 scores. The logistic regression model was set to iterate up to 1000 times to ensure convergence. Each feature's influence is evaluated by fitting a model to the training set, making predictions on the test set, and subsequently calculating both the BIC and F1 score for these predictions.

Figure 5.4 displays these scores, with features arranged on the x-axis from lowest to highest BIC, indicating that features on the far left have a better model fit than those on the far right. The F1 scores are plotted in reverse to make the BIC and F1 scores visually more aligned. It is noticeable, that the F1 scores generally decrease alongside the BIC scores, which corroborates their predictive performance. However, an anomaly is observed with the feature 'Council Area', a categorical variable with 33 unique values. While its F1 score is based on the test set and thus not influenced by overfitting, ‘Council Area’ utilizes 33 different coefficients for its 33 unique values to fit the model to the data. This is unlike other categorical variables with fewer unique values or numeric variables that use only one. This disparity highlights why F1 score alone may not always reliably indicate model fit, reinforcing our reliance on BIC scores for assessing fit within this dataset.

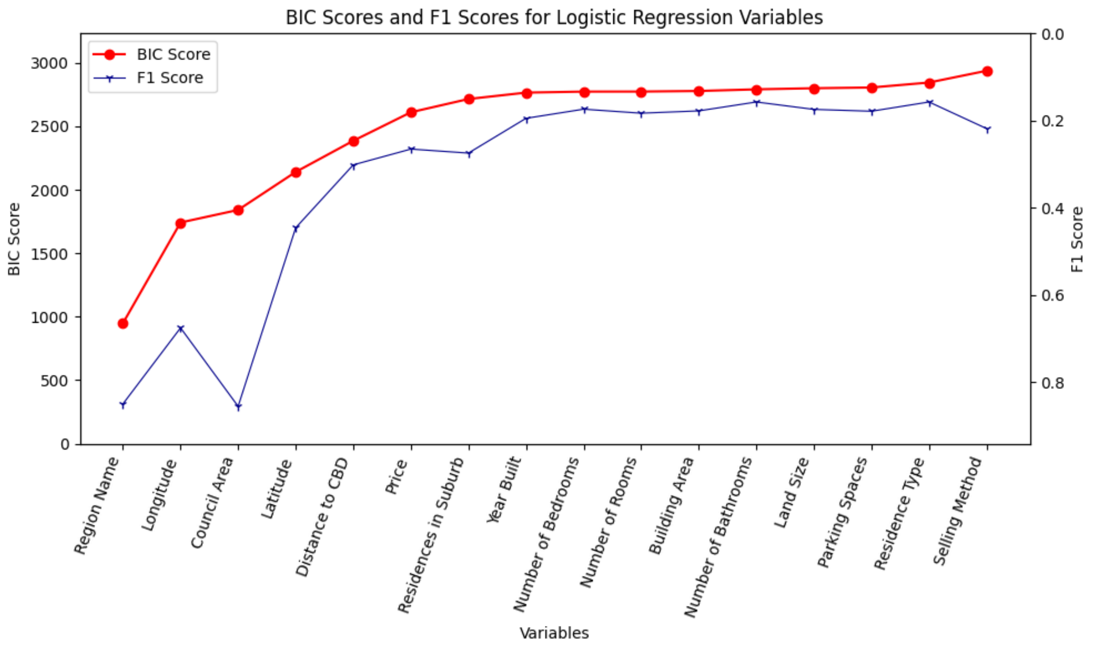

***Figure 5.4:*** *This graph presents a visualization of BIC scores and F1 scores of a multinomial logistic regression analysis applied to the Melbourne Housing dataset. Each feature’s impact is shown both in terms of their BIC and their F1 score, displayed as line graphs. A lower BIC indicates a better model fit, a higher F1 score generally suggests higher predictive accuracy. It should be noted that the F1 axis shows 0 at the top and 1 at the bottom.*

## 5.5 Feature Selection (Step 4)

In step 2 of our interpretation framework, we chose a relatively small number of clusters (five) during the discretization process. This decision emphasizes the global structure of the data because fewer clusters capture broader patterns across the entire dataset rather than detailed, localized variations that would emerge with a larger number of clusters.

As a result, in step 3 (classification), our analysis focused on identifying which features best predict this global structure. Therefore, to understand the global structure, we select to visualize features with the lowest BIC (best model fit). The top five features with the lowest BIC - ‘Region Name’, ‘Longitude’, ‘Council Area’, ‘Latitude’, and ‘Distance to CBD’ - primarily describe the geographical or spatial characteristics of the residence’s location, such as its positioning relative to the central business district and regional boundaries. This implies that certain attributes, such as the region a residence is located in, can effectively predict the label assigned to the residence during hierarchical clustering. These findings suggest a pattern where the global structure of UMAP’s embedding reflects geographical characteristics of the residences.

In contrast, the last 11 features, which show lower model fit, relate directly to attributes of the individual residences, such as 'Number of Rooms', 'Price', and 'Year Built'. Since this dataset contains a limited number of features, we will visualize these 11 features to assess the extent to which they explain the local structure. For datasets with a larger number of features, this approach may not be feasible, and an alternative method is presented in chapter 6.6.

## 5.6 Visualization and Interpretation (Step 5)

This section addresses the central question of this thesis: How can the low-dimensional embedding produced by UMAP be interpreted? The preceding analysis evaluated the importance of each feature in capturing the global structure of the embedding. The current step focuses on visualizing the feature values. More specifically, we will focus on the following five features: Region Name, Longitude, Latitude, Number of Bedrooms, and Residence Type.

The first three features are among the top five in terms of model fit, as indicated by their BIC scores, suggesting their importance for the global structure. The features 'Number of Bedrooms' and 'Type of Residence' are among the last 11 features and show higher BIC scores, indicating a poorer model fit. Nonetheless, they still offer valuable insights into the low-dimensional embedding, as we will explore.

### Global Structure

To confirm whether a feature explains the global structure, we are looking for broad patterns when visualizing feature values in the low-dimensional embedding, such as bigger clusters or overall gradients.

Figure 5.5 shows the regions in which the residences (displayed as dots) are located. Broad clusters can be seen. More specifically, residences that are close in the low-dimensional embedding typically fall within the same geographical region. Consequently, a residence's region effectively informs the global structure revealed by UMAP, providing clear insights into commonalities and differences among residences.

Moving on to the examination of two numerical features, longitude and latitude. Figure 5.6 clearly demonstrates a distinct gradient in longitude values, which progressively increase from the bottom to the top of the figure. Similarly, Figure 5.7 illustrates a gradual increase in latitude values from the left to the right side of the graph, although this pattern is less distinct than that of longitude.

The geographical characteristics of residences, specifically their locations within Melbourne, largely explain the global structure of UMAP's low-dimensional embedding, as evidenced by the patterns observed in the three figures. This structure corresponds with the larger clusters identified through hierarchical clustering. While some residences appear as outliers, deviating from the values of their neighboring residences, such anomalies are consistent with the non-linear nature of UMAP and are anticipated. Despite these exceptions, the overarching global structure of the embedding is clearly influenced by the geographical attributes of the residences.

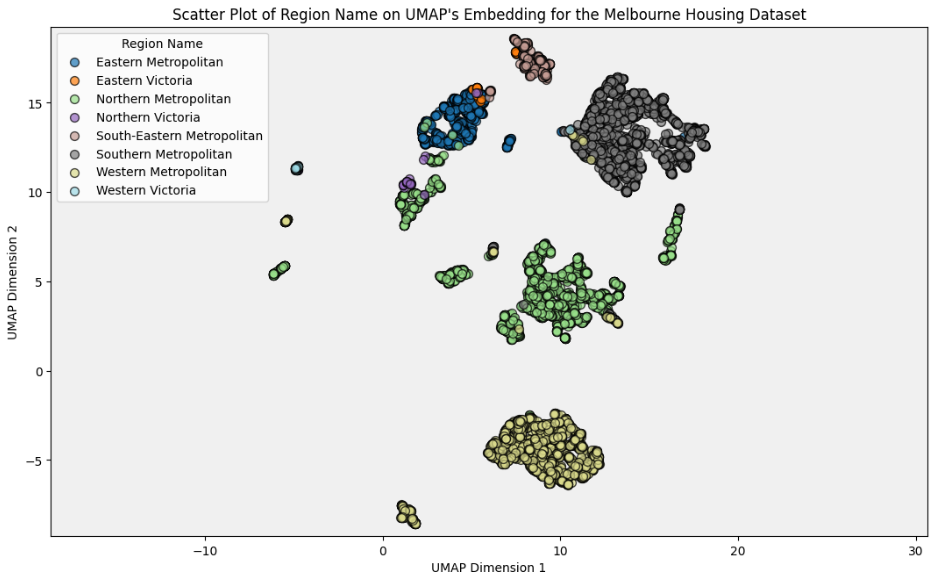

***Figure 5.5:*** *UMAP's low-dimensional embedding of the Melbourne Housing dataset, with residences color-coded according to their region.*

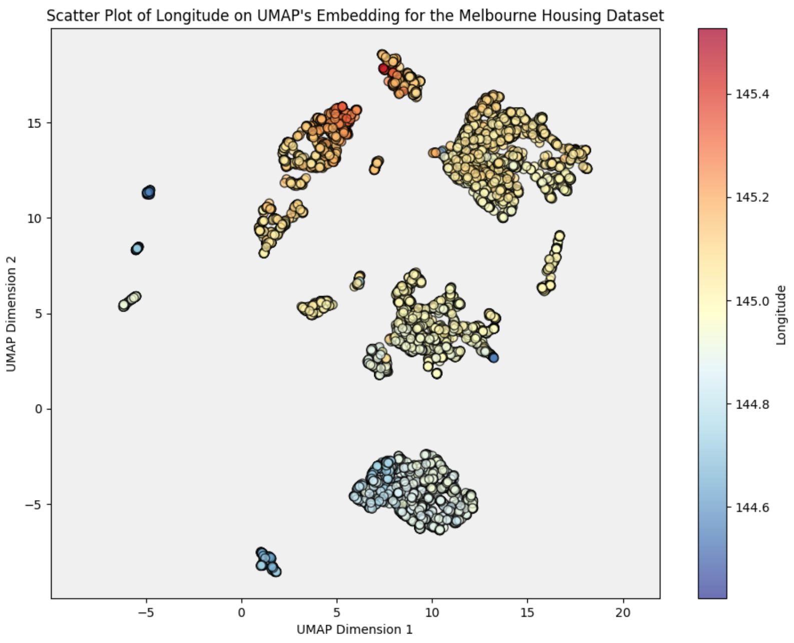

***Figure 5.6:*** *Low-dimensional embedding of UMAP, where residences are colored according to their longitude value.*

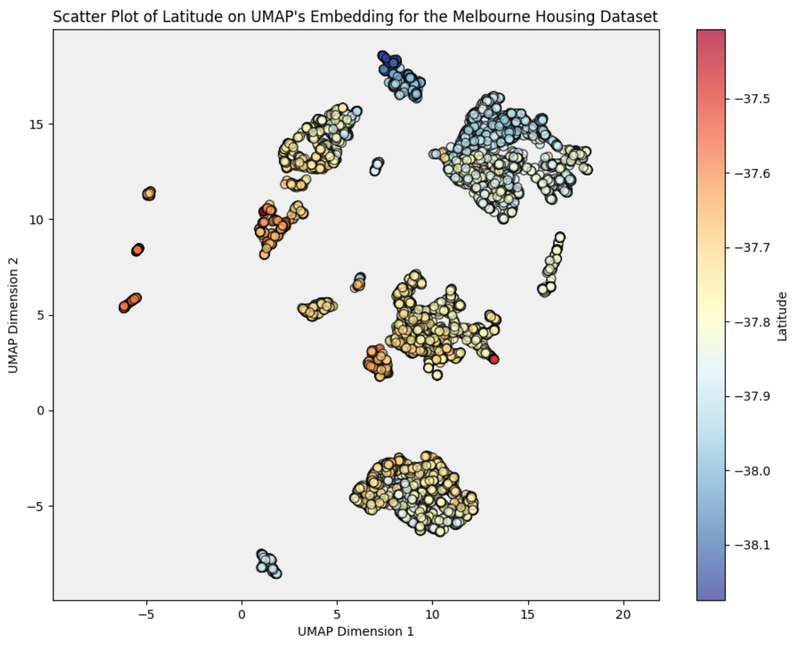

***Figure 5.7:*** *Low-dimensional embedding of UMAP, where residences are colored according to their latitude value.*

## Local Structure

To identify which features are meaningful for understanding the local structure, we need to look for patterns within clusters. This includes gradients within a cluster in case of numerical features or observing clustered regions in case of categorical features, which means seeing distinct subclusters within a larger cluster. Generally, features with a high model fit tend to explain the global structure better. Therefore, we will examine the visualizations of features with a lower model fit, as described before.

In Figure 5.8, the number of bedrooms per residence is visualized. Larger clusters, identified by hierarchical clustering, display gradients which indicates that although these residences belong to the same broader category (sharing similar geographical locations as indicated by region names, latitude, and longitude), they differ in other features and are organized accordingly. For example, within the upper right cluster at approximately coordinates (13, 12), residences positioned further to the left tend to have more bedrooms compared to those further to the right of the cluster. Please note, to improve clarity and interpretability of the visualization, the number of bedrooms have been capped at 7. This means that residences with 7 or more bedrooms are all given the same color. The reason is that only a small number of residences had an unusually high number of bedrooms which then skewed the color distribution. For example, residences with 1 bedroom had a similar shade of color (blue) to residences with 4 bedrooms. Hence, it was very difficult to visually distinguish them, which made the visualization ineffective. By capping values at 7 bedrooms the visual differentiation becomes much clearer.

In Figure 5.9 we see the visualization of residence type in the low-dimensional embedding. Subclusters within bigger clusters can be seen, an indication that residence type can give us insights into the local structure. More specifically, we observe a similar distribution pattern

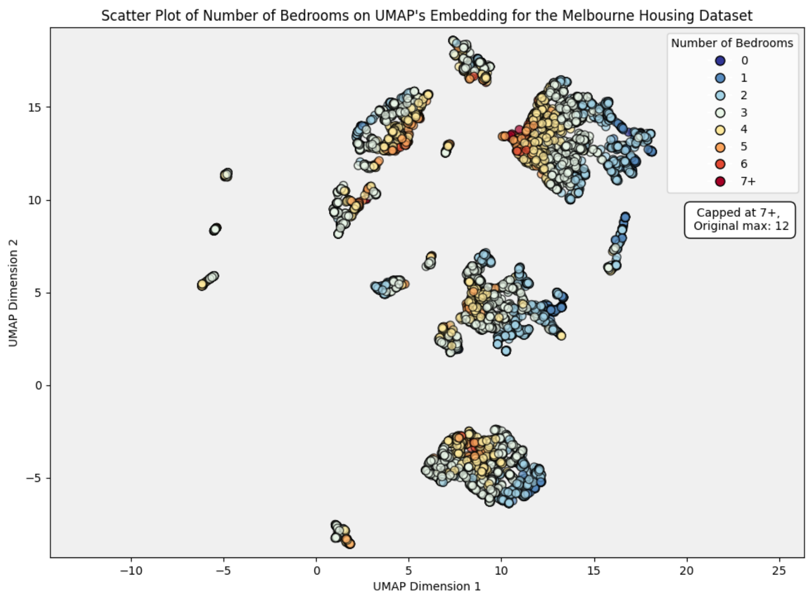

***Figure 5.8:*** *Low-dimensional embedding of UMAP, showing the number of bedrooms per residence.*

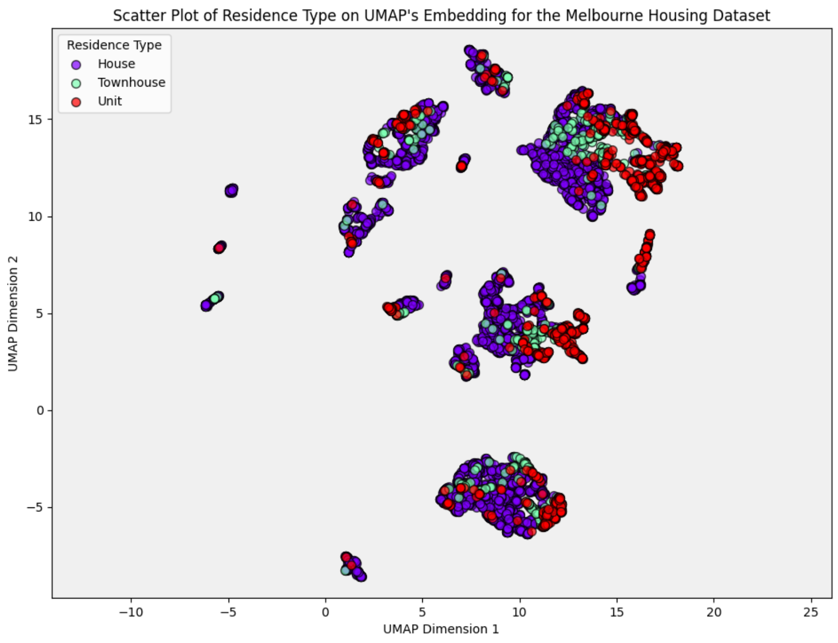

***Figure 5.9:*** *Low-dimensional embedding of UMAP, showing the residence type. Houses also include cottages, villas, semi-detached houses, and terraced houses. Units also include duplexes.*

to the number of bedrooms: The residence type ‘House’ is typically found in the same area of the cluster that shows a higher number of bedrooms, whereas ‘Townhouses’ and ‘Units’ are more common in areas with fewer bedrooms. This similarity is intuitive, as houses are generally larger than townhouses and duplexes and thus they typically have more bedrooms. Similar local patterns, showing gradients and subclusters, can also be observed with many of the other features that had a lower model fit in step 3, such as ‘price’ or ‘number of bathrooms’. Visualizations of these two features, along with all other features not yet presented here, are provided in Appendix A.2.

## Neither Global Nor Local Structure

The feature ‘Selling Method’, which describes how a residence was sold, exhibits the lowest model fit among all variables analyzed. As illustrated in Figure 5.10, the UMAP visualization of this feature does not reveal any discernible pattern; the distribution appears random.

The underlying reason is that the global structure is predominantly shaped by features related to geographical location, which are inherently interconnected. Similarly, the local structure is associated with features intrinsic to the residence itself, also closely related. These groups form the majority of the features and thus have a significant impact on the UMAP outcome. However, ‘Selling Method’ is uncorrelated to these features and therefore shows neither a global nor a local pattern in the UMAP embedding.

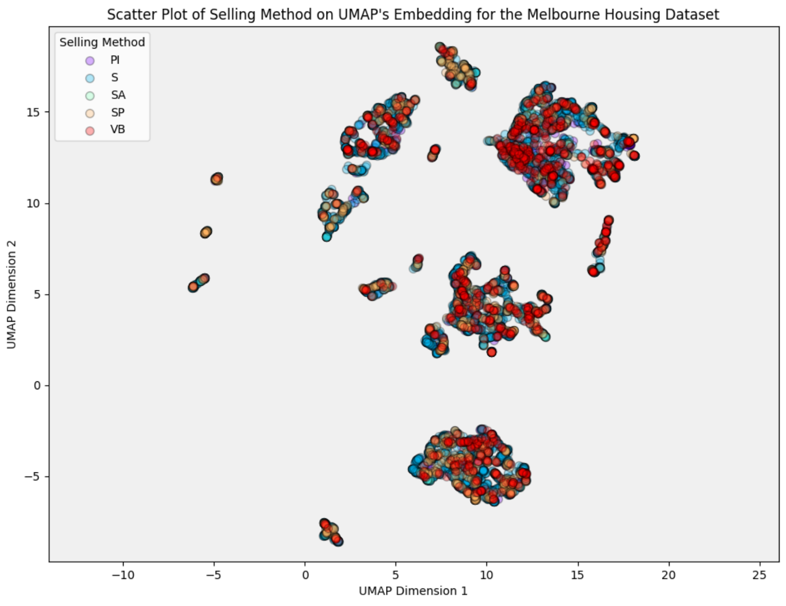

***Figure 5.10:*** *UMAP output showing the selling method. PI = residence passed in; S = residence sold; SA = sold after auction; SP = residence sold prior; VB = vendor bid.*

## 5.7 Robustness

In this analysis, robustness refers to the stability and reliability of our results across different runs or when using varying fixed seeds. Our goal is to provide a reliable framework that enables users to consistently interpret their UMAP results without relying on chance or the need to select the 'correct' fixed seed.

To examine the robustness of our interpretation framework, we distinguish between two primary sources of randomness: the randomness introduced by our interpretation framework and the randomness inherent in the UMAP algorithm. This section first focuses on the randomness stemming from our interpretation methods.

The sole source of randomness within our previous analyses is the splitting of the dataset into training and test sets for multinomial logistic regression. To analyze the effect of this randomness, we altered the fixed seed in the logistic regression function and compared the Bayesian Information Criterion (BIC) and F1 scores with those obtained using a the initial fixed seed, as shown in Figure 5.4.

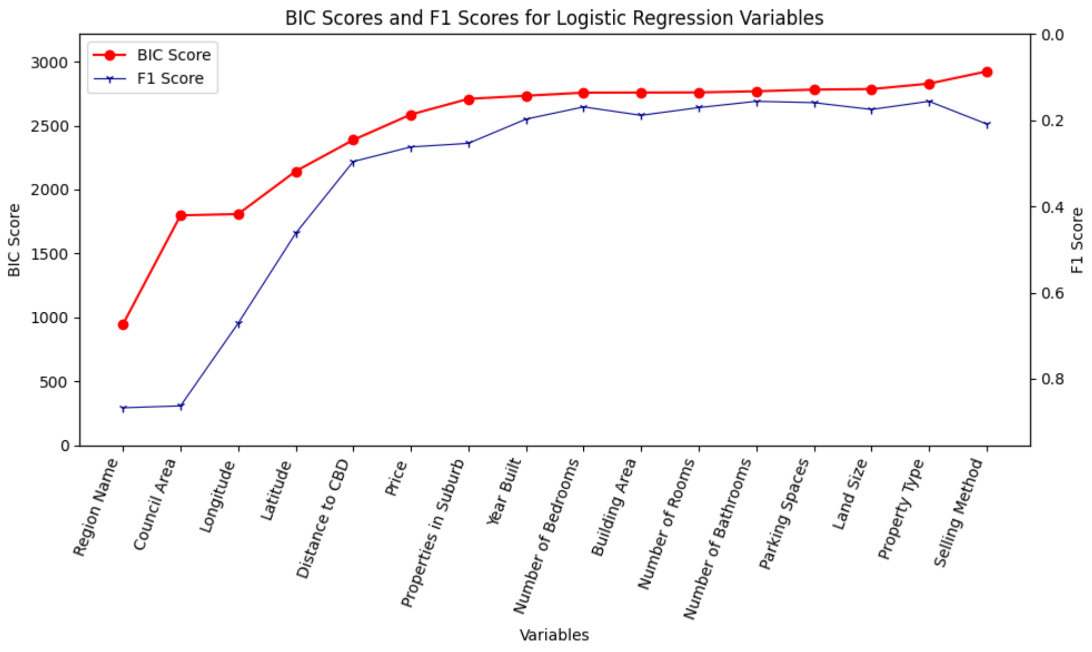

***Figure 5.11:*** *Visualization of BIC scores and F1 scores of a multinomial logistic regression analysis when using a different fixed seed. Please note that the axis for the F1 score is reversed: 0 is at the top, whereas the maximum is at the bottom.*

The results, illustrated in Figure 5.11, are broadly consistent with our previous findings. Although there was a slight shift in the ranking of certain features, such as the council area, building area or parking spaces, the overall BIC scores remained closely aligned. This consistency underscores the robustness of our primary findings: even minor fluctuations in BIC scores, which led to changes in feature order, did not affect our fundamental interpretations. The features with lower BIC scores, which are indicative of the geographical location of the residence, continue to explain the global structure, while those relating to residence characteristics explain the local structure and consistently exhibit higher BIC scores. Moreover, the feature 'Selling Method' again scored lowest in model fit which is also consistent with our previous interpretation.

For completeness, we also examined the extent to which randomness introduced by UMAP influences the final interpretations and therefore, changed the fixed seed of the UMAP algorithm. While the UMAP maps displayed minor variations, the ordering of features by BIC score and the overall patterns in the visualizations remained similar to previous findings. Since addressing the inherent issues of the UMAP algorithm is not a goal of this thesis, detailed graphs from this part of our analysis are documented in Appendix A.3.

In summary, variations in the fixed seed do not alter the final interpretations of the UMAP output, affirming the robustness of our interpretation framework.

## 5.8 Conclusion

Close examination of the low-dimensional embedding of UMAP reveals a complex interplay between global and local structures. The global structure is predominantly defined by the geographical region in which a residence is located. In contrast, the local structure is characterized by specific features of a residence, such as the number of rooms and the type of residence, among others. Lastly, a feature that neither contains information about geographical location nor inherent characteristics of the residence itself is ‘Selling Method’, which scored the lowest in model fit and showed no pattern in the UMAP embedding.

Organizing features by their BIC scores and focusing on those with the highest model fit proved effective for identifying features that are important to the global structure. For the local structure, selecting features with lower model fit offers a valuable preliminary indication of which features warrant further analysis. While this method alone does not conclusively identify features responsible for the local structure (as seen with ‘Selling Method’), it provides useful insights, especially for features with medium to high BIC scores (i.e., medium to low model fit). In our dataset with a limited number of features, this approach was sufficient to identify features important for the local structure. However, for larger datasets, this will not suffice; therefore, chapter 6.6 presents a more complex method for finding features that offer insights into the local structure.

# 6 Application on the Brain Samples Dataset

This chapter shows the application of the aforementioned interpretation framework to a dataset containing the probe expression values of human brain samples. As before, all analysis were performed with PyCharm version 2023.3.5, using Python version 3.12.

This chapter starts with an overview of the Brain Samples dataset. Then we show step 1 of the interpretation framework, namely the application of UMAP. This is followed by the application of the unsupervised clustering technique GMM to produce pseudo-labels (step 2). Before going to the third step of the interpretation framework, we show how the probes are grouped to make interpretation more insightful and analysis more manageable. After this, the supervised classification technique Random Forest is applied (step 3). The predictor variables are the different probe groups, and the pseudo-labels from step 2 are the outcome variables. This is followed by feature selection for the global and the local structure (step 4). Lastly, the visualization and interpretation of the low-dimensional embeddings are presented (step 5).

## 6.1 Description of the Brain Samples Dataset

The Allen Brain Atlas has made 6 datasets available (Allen Institute for Brain Science, 2024a), each of which shows the data for one brain. For this research, we are working with the data of a single brain and downloaded the file 'H0351.2001' (Allen Institute for Brain Science, 2024b), which contains the following 6 data files. The main dataset can be found under the 'Microarray Expression.csv' file. This dataset contains gene expression values, which refer to the activity of a gene, as measured by the number of RNA copies that are present in the cell. These values are normalized across all brains and the dataset is arranged by *probe* × *sample*. Probes in this context refer to short DNA sequences that bind with specific RNA sequences from the brain samples. Each probe on a microarray corresponds to a specific gene or part of a gene, and multiple probes can be designed to target different regions of the same gene. This allows for precise measurement of the gene's expression level. Hence, the gene expression values are given for different samples from the donor's brain, using these probes to quantify how active each gene is in each sample.

The 'Ontology.csv' file contains details on samples from all 6 brains, while the 'SampleAnnot.csv' file contains further details on the samples from donor 9861. The samples are listed in the same order as the columns in 'Microarray Expression.csv'.

The 'Probes.csv' file contains metadata for the probes in the 'MicroarrayExpression.csv' file.

The 'PACall.csv' file contains a present/absent indicator that shows whether the probe's expression is well above background.

Lastly, the 'Readme.txt' file shows the above mentioned information.

Please note that in the following, when the 'Brain Samples dataset' is mentioned, then we are referring to the dataset found in the 'Microarray Expression.csv' file. This dataset contains 946 samples and 58,692 probes. There are no missing values. The gene expression values in this dataset range from 1.47 to 18.38, with a median of 4.91, a mean of 5.22 and a standard deviation of 2.92.

The Brain Samples dataset was first prepared by transposing the original data so that the rows represent samples and the columns represent probes. This step was necessary because the goal is to reduce the information from thousands of probes (i.e. reduce genetic information) into two dimensions, making it easier to visualize and analyze the relationships among brain samples. Therefore, when the dataset is arranged with rows as samples and columns as probes, we refer to it as the '**Brain Samples dataset**'. In contrast, when the rows represent probes and the columns represent samples, we call it the '**transposed Brain Samples dataset**'.

Although the data from the Allen Brain Atlas website is already normalized, the probes (i.e., columns) still have different means and standard deviations. This could cause UMAP to focus more on these differences in scale rather than the actual differences and similarities between probes. To prevent this, we further scaled the Brain Samples dataset so that each column has a mean of 0 and a standard deviation of 1.

## 6.2 UMAP (Step 1)

The UMAP algorithm from the UMAP package was applied to the scaled dataset using a fixed seed. This produced a two-dimensional embedding, which represents the samples in a low-dimensional space. The resulting graph is displayed in Figure 6.1.

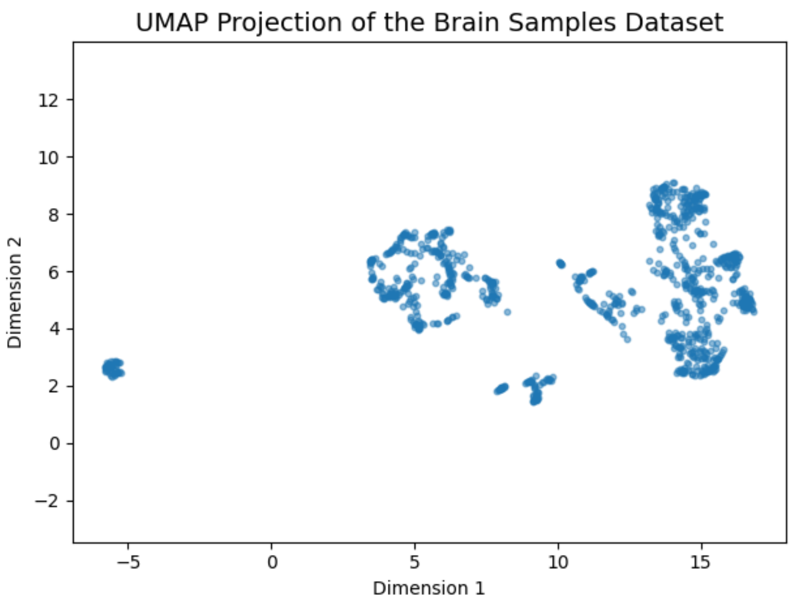

***Figure 6.1:*** *UMAP output showing the low-dimensional embedding of the Brain Samples dataset. The points show the different samples of one brain. There are 946 samples in the dataset.*

## 6.3 Clustering of Samples: GMM (Step 2)

Since hierarchical clustering was already used for the previous dataset, we will now showcase the application of GMM clustering. The results from hierarchical clustering on the Brain Samples dataset can be found in Appendix B.2.

To determine the optimal number of clusters for the Gaussian Mixture Model, the BIC score was evaluated across a range of cluster numbers. As detailed in chapter 3.1, the BIC score provides a useful metric for balancing model fit with complexity, with lower scores indicating a better model fit. Figure 6.2 illustrates the BIC scores for cluster numbers ranging from 1 to 100.

To further refine this analysis, we examined a zoomed-in graph that focusses on the BIC scores for 1 to 40 clusters, which is presented in Appendix B.1. The minimum BIC score is observed at 28 clusters. However, we would like to have a smaller number of clusters. The reasons for this choice are that we want to focus on the global structure rather than finding the optimal number of clusters and a smaller number of clusters is more manageable for analysis and interpretation which is an important factor in this thesis. Deciding on the number of clusters is inherently subjective, and there is often no definitive right or wrong choice. This dataset specifically offers many clustering opportunities, showing the inherent hierarchical nature of brain anatomy: all samples are part of the brain, subdivided into major regions like the cerebrum or cerebral cortex, and these regions in turn are broken down into further specific subregions, providing many clustering possibilities.

For this analysis, we decided to use 8 clusters, as this clustering visually aligns with how we would intuitively group the samples, providing a natural fit. The low-dimensional graph with objects colored according to their assigned GMM label can be found in 6.3

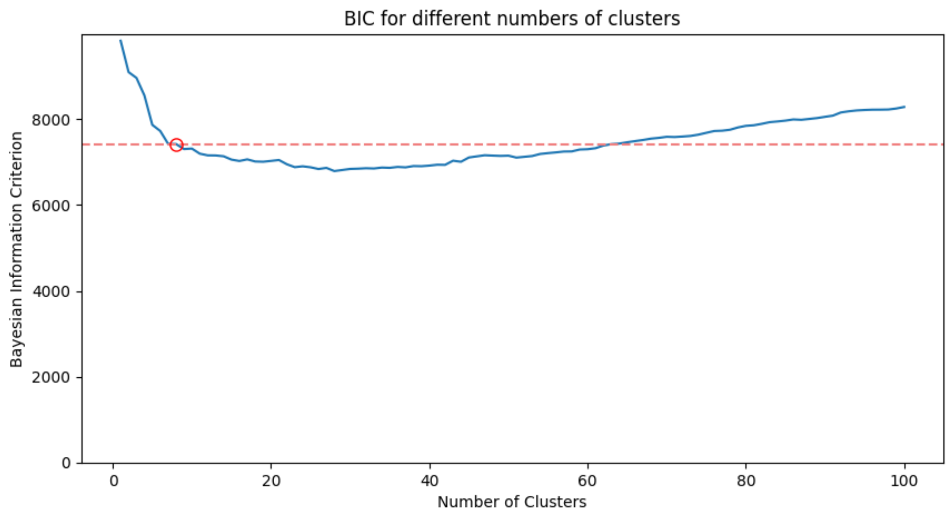

***Figure 6.2:*** *BIC scores for GMM clustering applied to the low-dimensional embedding of the Brain Samples dataset. The number of clusters range from 1 to 100. Lower BIC scores indicate a better model fit. The horizontal line shows the BIC score of the chosen number of clusters, which is 8.*

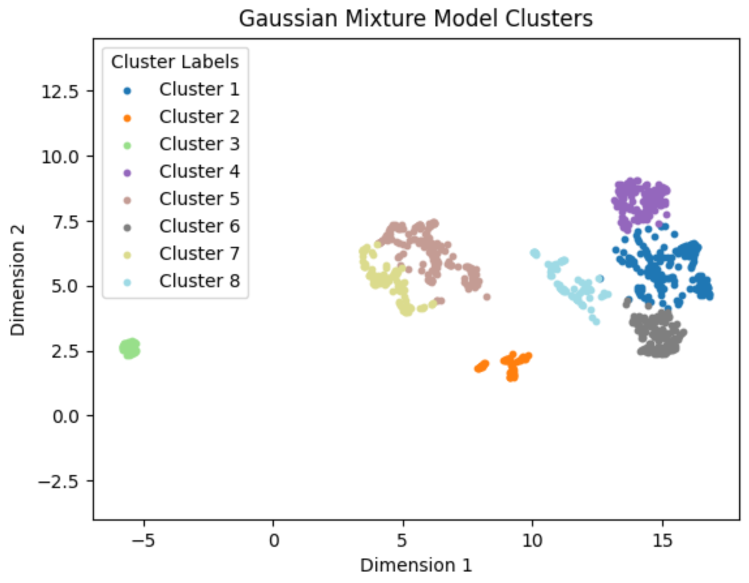

***Figure 6.3:*** *UMAP's low-dimensional embedding of the Brain Samples dataset, where each point represents a brain sample. The samples are colored based on the 8 clusters chosen by the author from the GMM clustering results.*

## 6.4 Grouping the Probes (Extra Step)

Identifying individual probes that significantly affect the global structure of the dataset may not be particularly informative because genes, which are measured by probes, are interdependent and collectively influence traits. Instead, a more meaningful approach involves grouping the probes and then determining which of these probe groups best predicts the sample clusters observed in Figure 6.3, and thus best explains the global structure of the UMAP output. We decided to categorise the probes into 100 distinct groups using a Gaussian Mixture Model. The choice of this number balances the need to discern differences between groups and maintain homogeneity within them against considerations of computational efficiency and interpretability. With a higher number of groups, interpretability is being complicated and the computational demand increases. We selected a spherical covariance type which assumes uniform variance across all dimensions. This further enhances computational efficiency, as otherwise there would be too many parameters to estimate.

The probes are grouped based on the transposed, standardised Brain Samples dataset, which is the same dataset previously utilised in the UMAP analysis, only now transposed to have 58,692 rows, each representing a probe, and 946 columns, each corresponding to a sample. Therefore, the grouping of probes was conducted in the high-dimensional space, as our aim is not to generate pseudo-labels for the UMAP output, but to create groups of probes that enhance interpretability. We also considered issues related to the curse of dimensionality. However, given the substantial number of objects (over 58,000 probes), which ensures the high-dimensional space is adequately populated, we opted to carry out the grouping process in the high-dimensional space.

Although visualizing the probe groups is not necessary, a visualization can be of interest in order to gain a better understanding and an approximate indication of what the probe space looks like. Given the high-dimensional nature of the dataset, visualizing the probe groups requires a dimension reduction technique. For this purpose, we applied UMAP to the transposed version of the scaled Brain Samples dataset. Figure 6.4 illustrates the distribution and grouping of probes within the 2D UMAP space. The probes are colored according to the 100 groups derived from the GMM clustering results. It should be noted that the visualization is presented in two dimensions only for ease of understanding, but the actual clustering was performed using the high-dimensional data. Furthermore, using 100 distinct colors introduces challenges in color differentiation, which complicates the visual clarity and distinctness of the groups. Lastly, a legend is omitted as it would only provide the group number, offering no further insight.

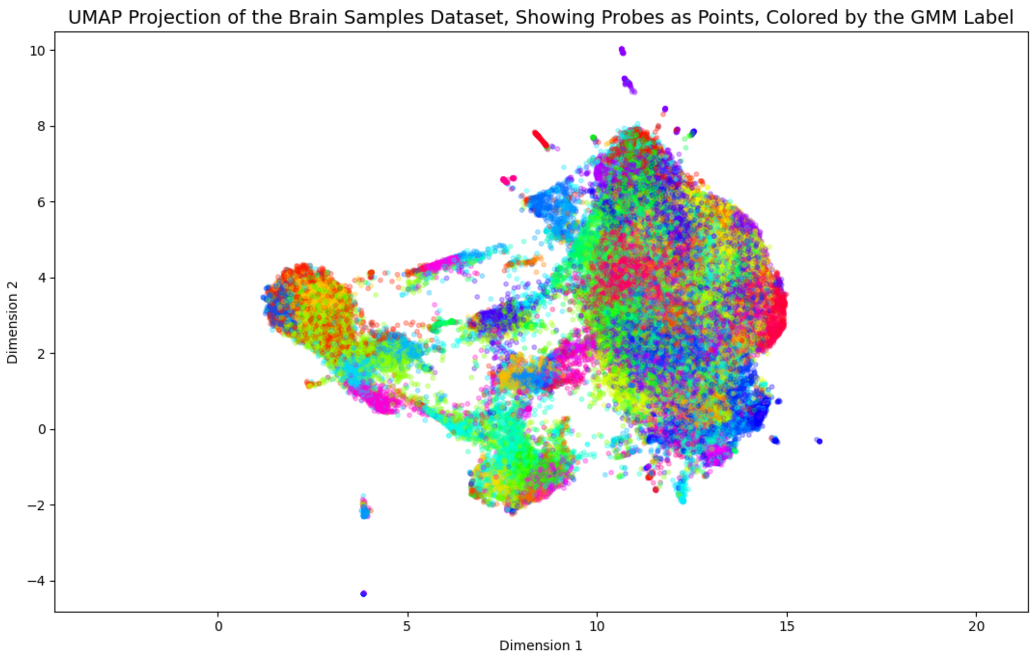

***Figure 6.4:*** *UMAP's low-dimensional embedding of the transposed Brain Samples dataset, where each point represents a probe. The coloring shows the clustering of probes into 100 groups based on the GMM results applied to high-dimensional data.*

## 6.5 Supervised Classification: Random Forest (Step 3)

Before proceeding with the supervised classification step, we first calculated the average value of all standardised probes within each group for every sample. For instance, if probes 3, 5, and 10 belong to one group (e.g., probe group 1), we calculated the mean value of these probes for sample 1, which then represented the value for probe group 1 for that sample. This process was repeated for each sample and each probe group, with the result of having the average values for all 100 probe groups accross more than 900 samples.

Having demonstrated the use of multinomial logistic regression on the previous dataset, we will now employ the random forest algorithm. The results of using multinomial logistic regression for step 3 are documented in Appendix B.3.

The random forest algorithm is applied to the standardized dataset containing the average probe group values per sample. The seed is fixed to ensure replicability. The output is a sorted list of feature importances, which are also visualized in the graph in Figure 6.5.

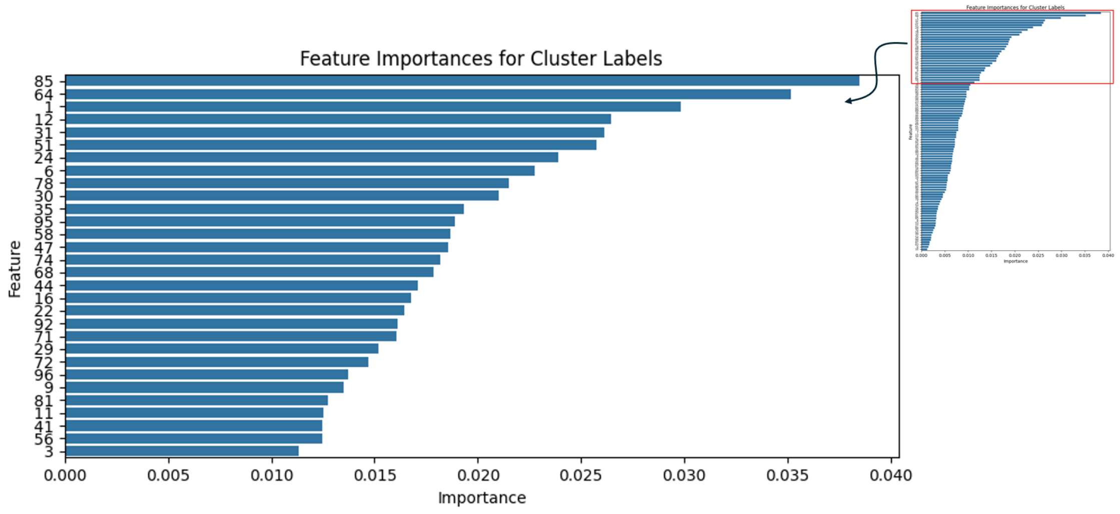

***Figure 6.5:*** *This graph shows the feature importances for the 30 most important probe groups derived from applying Random Forest to the standardized Brain Samples dataset containing average probe group values.*

## 6.6 Feature Selection (Step 4)

### Global Structure

As described in chapter 5.5, when choosing a relatively small number of clusters in step 2, in our case eight clusters, the supervised classification step can give us insights into which features best explain the global structure - namely those with the best model fit. Therefore, in the next step we will visualize probe groups 85 and 64, as they rank as the two most important probe groups based on the random forest analysis. To definitively say whether these probe groups can give us insights into the global structure, we will have to look at their visualizations and examine whether they show clear, overarching patterns. This may occur when one cluster predominantly displays high values while another cluster shows overall low values, or when a global gradient is observed across clusters.

### Local Structure

To gain insights into the local structure, we refined our approach in steps 2 and 3 of our interpretation framework. The objective is to identify specific probe groups that shed light on finer-grained patterns in the data.

1. **Adjusting the Number of Clusters in Step 2 (Unsupervised Clustering):** In step 2, we increase the number of clusters used in GMM. Previously, we chose a smaller number of clusters to emphasize the global structure of the data. By contrast, increasing the number of clusters allows us to detect more nuanced groupings, thus highlighting the

local structure. Therefore, the number of clusters for this step was increased to 28, as 28 clusters had the lowest BIC score. The resulting pseudo-labels are used for the next step.

2. **Re-running Step 3 (Supervised Clustering) with Updated Labels:**
   In step 3, we perform the Random Forest analysis using the same set of predictor variables as before — the 100 probe groups. However, the outcome variable is now the larger set of cluster labels derived from the adjusted step 2. The output of this step is a set of feature importance scores, indicating how well each probe group predicts the 28 clusters.
3. **Identifying Features Specific to the Local Structure:**
   To pinpoint the probe groups that are particularly informative about the local structure, we calculated the difference in feature importance scores between the local and global analyses. Specifically, we subtracted the feature importance scores obtained using the larger number of clusters (local structure) from those obtained using the smaller number of clusters (global structure). This difference highlights probe groups that became more important when focusing on local patterns. Figure 6.6 visualises these differences.
4. **Visualization of Probe Groups:**
   Finally, we visualize the probe groups with the highest positive differences in feature importance, which are probe group 0 and probe group 73, as marked with circles in Figure 6.6.

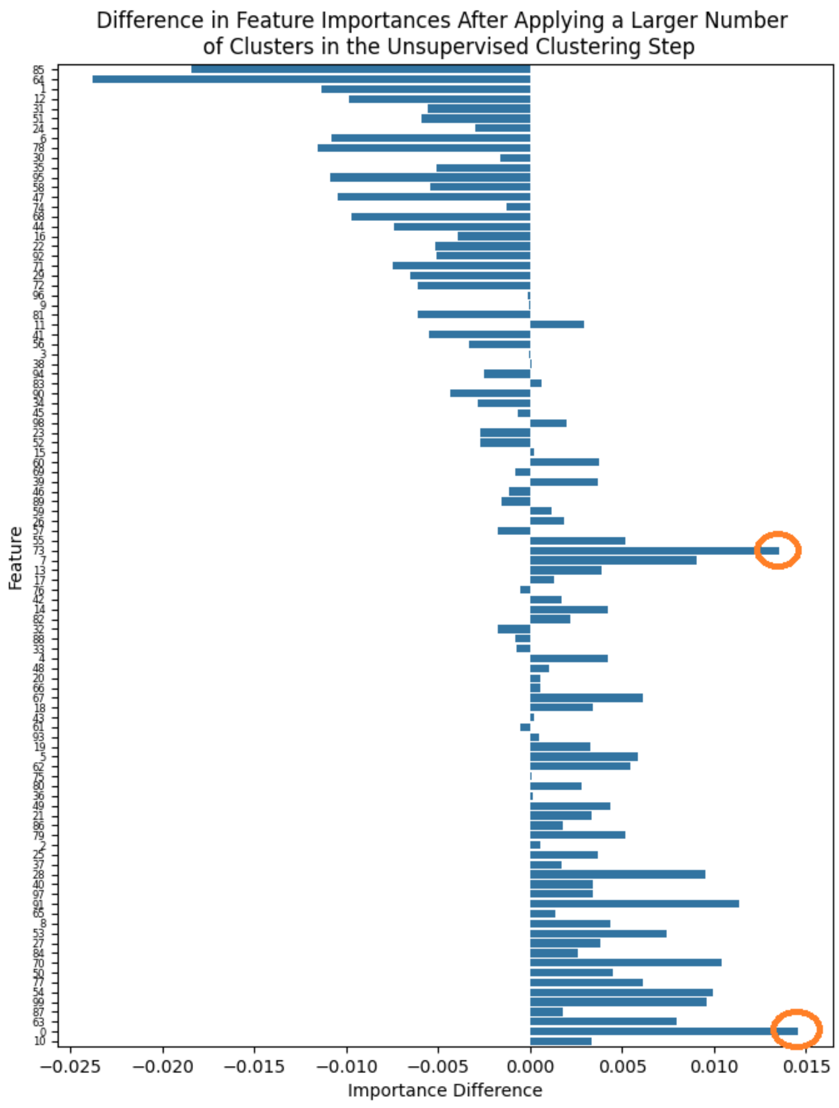

***Figure 6.6:*** *Difference between the feature importance scores obtained using 28 clusters for GMM in step 2 and those obtained using 8 clusters for GMM. A higher positive difference means that the probe group became more important when using 28 clusters in step 2 of the interpretation framework. The circles mark the two probe groups with the highest positive difference, which are probe group 0 and probe group 73. The order of the features on the y-axis is the same as in figure 6.5.*

## 6.7 Visualization and Interpretation (Step 5)

To understand the meaning of the chosen probe groups, first, the gene symbols corresponding to the probes within this group were extracted. The gene symbols were then submitted to the GOrilla website (Eden et al., 2023). GOrilla evaluates whether a specified set of genes is enriched in particular biological processes, functions, or cellular components, indicating an over-representation compared to a predefined background set. For our analysis, the background consisted of all gene symbols included in our dataset. GOrilla calculates the following for each gene set:

- **p-value**: The probability of observing the given level of enrichment under the null-hypothesis of no association, as computed under the mHG or HG statistical models. This value is not adjusted for multiple comparisons.
- **FDR q-value**: Adjusts the p-value for multiple testing using the Benjamini and Hochberg method to control the false discovery rate.
- **Enrichment score**: Quantifies the level of over-representation of genes within a specific GO term, calculated using the formula: Enrichment $= \frac{(b/n)}{(B/N)}$, where:
  - $b$ is the number of genes in the intersection of the input list and the GO term;
  - $n$ is the number of genes in the target/background set;
  - $B$ is the total number of genes associated with the GO term;
  - $N$ is the total number of genes.

These metrics collectively help to understand the statistical and biological significance of gene enrichment in our probe groups.

Although previous analyses have been implemented on standardized values, the following visualizations will show the non-standardized mean probe value. The reason is to give deeper insight into the actual mean values of each probe group.

### Global Structure: Probe Group 85

Given the complexity of interpreting gene groups, this thesis will not delve deeply into every aspect of probe group 85. However, to provide some context, the analysis conducted using the GOrilla tool identified the most significant biological process for probe group 85 as *cellular potassium ion transport*. Additionally, the predominant function is identified as *potassium channel activity*, and the most significant component as *plasma membrane part*. Corresponding p-values, FDR q-values, and enrichment scores are detailed in Table 6.1.

| **Description** | **p-value** | **FDR q-value** | **Enrichment (N, B, n, b)** |
|---|---|---|---|
| Cellular potassium ion transport | $1.81 \times 10^{-7}$ | $2.78 \times 10^{-3}$ | 6.23 (17984, 155, 242, 13) |
| Potassium channel activity | $6.17 \times 10^{-9}$ | $2.88 \times 10^{-5}$ | 8.26 (17984, 117, 242, 13) |
| Plasma membrane part | $2.65 \times 10^{-8}$ | $5.23 \times 10^{-5}$ | 1.91 (17984, 2719, 242, 70) |

***Table 6.1:*** *GOrilla results for probe group 85 showing significant enrichment among other in the biological process 'Cellular potassium ion transport', the function 'Potassium channel activity', and the component 'Plasma membrane part'.*

The visualization of the average values for probe group 85 can be found in Figure 6.7. We observe broad patterns indicating that this probe group is important to the global structure. Specifically, the right cluster exhibits high gene expression values, while the middle and left clusters display low expression values.

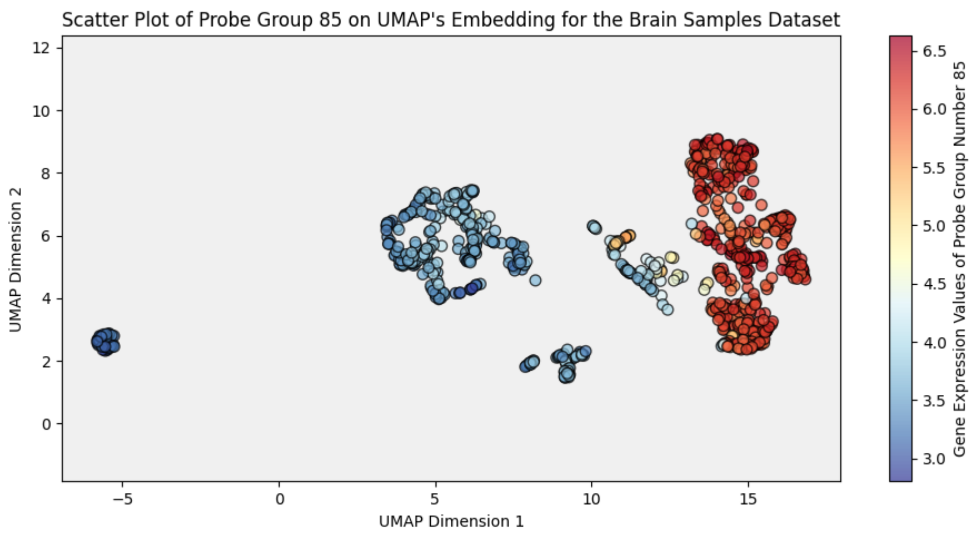

***Figure 6.7:*** *Average values of probe group 85, displayed for brain samples in the low-dimensional UMAP projection.*

## Global Structure: Probe Group 64

For probe group 64, the analysis also indicated significant enrichment in the three biological aspects. The predominant biological process is *modulation of chemical synaptic transmission*. The most significant function is *calmodulin binding*, and *neuron part* is the most significant component, with respective p-values, FDR q-values, and enrichment scores detailed in Table 6.2.

| **Description** | **p-value** | **FDR q-value** | **Enrichment (N, B, n, b)** |
|---|---|---|---|
| Modulation of chemical synaptic transmission | $1.07 \times 10^{-19}$ | $1.65 \times 10^{-15}$ | 6.70 (17984, 411, 235, 36) |
| Calmodulin binding | $1.31 \times 10^{-11}$ | $6.13 \times 10^{-8}$ | 7.38 (17984, 197, 235, 19) |
| Neuron part | $1.27 \times 10^{-25}$ | $2.50 \times 10^{-22}$ | 3.61 (17984, 1696, 235, 80) |

***Table 6.2:*** *GOrilla results for probe group 64 showing significant enrichment among other in the biological process 'Modulation of chemical synaptic transmission', the function 'Calmodulin binding', and the component 'Neuron part'.*

The visualization of the average values for probe group 64 can be found in Figure 6.8. An overall pattern can be observed, with the right cluster showing the highest gene expression values, the middle cluster showing low to medium values and the left cluster showing medium to high expression values. This overall pattern also indicates that probe group 64 indeed provides insights into the global structure.

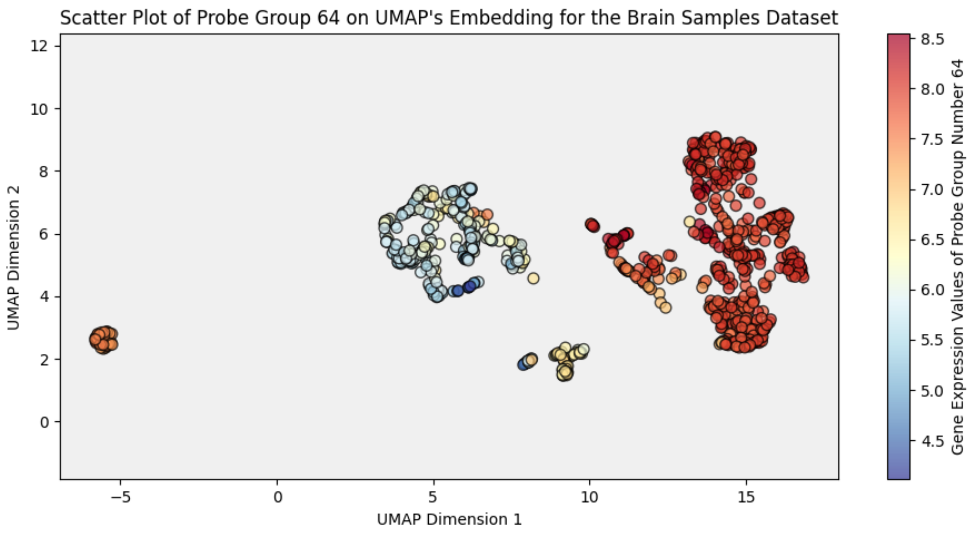

***Figure 6.8:*** *Average values of probe group 64, displayed for brain samples in the low-dimensional UMAP projection. The visualization reveals an overall gradient from right (high expression values) to left (low values).*

### Local Structure: Probe Group 0

Using the GOrilla tool, the analysis for probe group 0 indicated that the predominant biological process is *G protein-coupled receptor signaling pathway*. The most significant function is *transmembrane signaling receptor activity*, and the predominant component is *intrinsic component of membrane*. The respective p-values, FDR q-values, and enrichment scores are detailed in Table 6.3.

| **Description** | **p-value** | **FDR q-value** | **Enrichment (N, B, n, b)** |
|---|---|---|---|
| G protein-coupled receptor signaling pathway | $2.13 \times 10^{-13}$ | $3.27 \times 10^{-9}$ | 2.21 (17984, 1207, 634, 94) |
| Transmembrane signaling receptor activity | $1.16 \times 10^{-15}$ | $5.44 \times 10^{-12}$ | 2.34 (17984, 1201, 634, 99) |
| Intrinsic component of membrane | $5.97 \times 10^{-15}$ | $1.18 \times 10^{-11}$ | 1.51 (17984, 4909, 634, 262) |

***Table 6.3:*** *GOrilla results for probe group 0 showing significant enrichment among other in the biological process 'G protein-coupled receptor signaling pathway', the function 'Transmembrane signaling receptor activity', and the component 'Intrinsic component of membrane'.*

When identifying variables that define the local structure, we focus on those that reveal patterns (such as gradients or subclusters) within clusters. We can observe these patterns in the visualization of the average values for probe group 0 in Figure 6.9. Smaller gradients within the bigger clusters are visible, such as in the middle cluster: In the middle, we see higher values in a belt-like formation, and lower values on the side. Therefore, probe group 0 can be considered to be informative about the local structure.

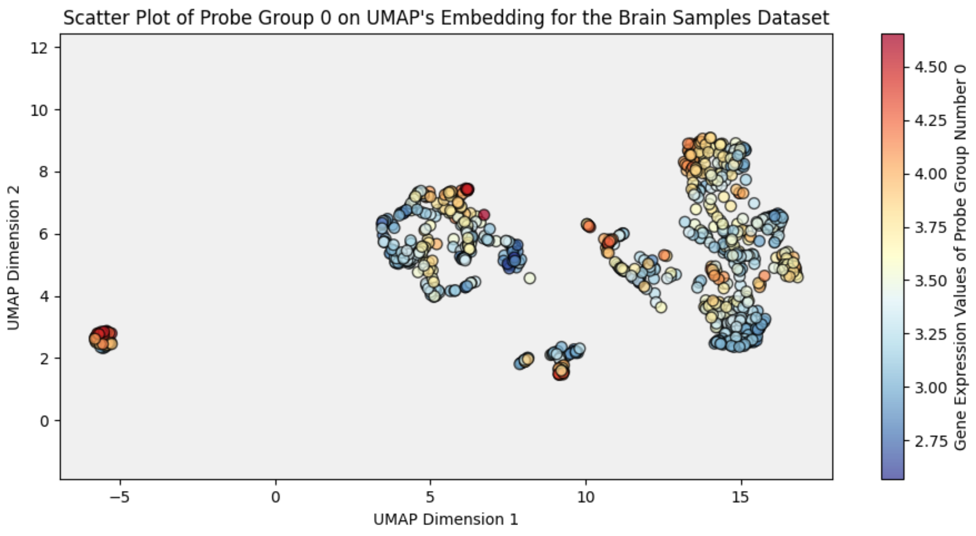

***Figure 6.9:*** *Average values of probe group 0, displayed for brain samples in the low-dimensional UMAP projection.*

## Local Structure: Probe Group 73

For probe group 73, the GOrilla analysis showed significant enrichments in the three biological aspects. The most significant biological process is *nucleic acid metabolic process*. The predominant function is *RNA binding*, and the most important component *intracellular part*. The respective p-values, FDR q-values, and enrichment scores can be found in Table 6.4.

| **Description** | **p-value** | **FDR q-value** | **Enrichment (N, B, n, b)** |
|---|---|---|---|
| Nucleic acid metabolic process | $5.30 \times 10^{-6}$ | $8.15 \times 10^{-2}$ | 1.67 (17984, 2104, 378, 74) |
| RNA binding | $2.45 \times 10^{-9}$ | $1.14 \times 10^{-5}$ | 2.08 (17984, 1599, 378, 70) |
| Intracellular part | $4.58 \times 10^{-11}$ | $9.02 \times 10^{-8}$ | 1.16 (17984, 13988, 378, 342) |

***Table 6.4:*** *GOrilla results for probe group 73 showing significant enrichment among other in the biological process 'Nucleic acid metabolic process', the function 'RNA binding', and the component 'Intracellular part'.*

The visualization of the average values for probe group 73 can be found in Figure 6.10. Gradients within clusters are visible, such as in the right cluster: The samples towards the right edge have low gene expression values, whereas the samples towards the left edge have medium to high values. As mentioned before, gradients within bigger clusters are indicative of a local structure, which is why probe group 73 can be considered to be insightful for the local structure.

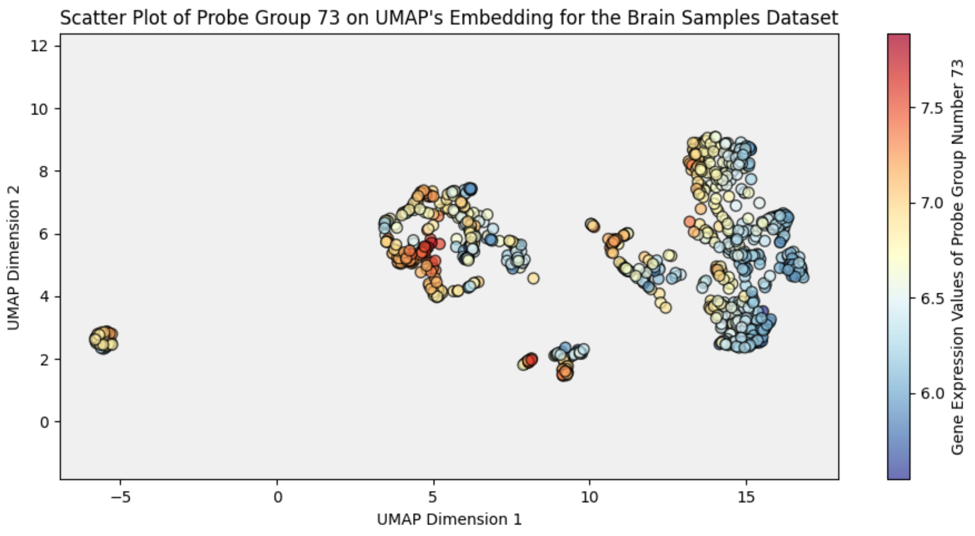

***Figure 6.10:*** *Average values of probe group 73, displayed for brain samples in the low-dimensional UMAP projection.*

## 6.8 Robustness

As previously discussed, robustness refers to the stability and reliability of our results across different runs or when using varying fixed seeds. Our goal is to provide a framework that provides consistent interpretations, regardless of the fixed seed chosen. To assess the robustness of our interpretation framework, we altered the fixed seed for both the GMM analysis in step 2 and the random forest analysis in step 3.

To determine how these changes affect the interpretation of the **global structure**, we analyzed the consistency of the top 20 probe groups (ranked by feature importance) after altering the fixed seed. The result was that 16 out of 20 probe groups were in the top 20 of both analyses. Figure 6.11 illustrates this comparison: the left graph displays the top 20 features by importance using the initial seed, and the right graph depicts them post-seed change. For clarity, features present in both graphs are highlighted in green, while those unique to one graph are in red.

Similarly, to assess the impact on the interpretation of the **local structure**, we examined the 20 probe groups with the highest feature importance difference (before changing the seed) and assessed how many of these remained in the top 20 after changing the seed. The result was that 15 out of 20 probe groups were still in the top 20. Figure 6.12 shows the findings visually, where the results with the changed seed are on the right. As before, probe groups present in both graphs are marked in green, while those unique to one graph are in red.

Although there is not a 100% overlap, the high degree of similarity in the final results for both global and local structure interpretations suggests that our method is robust.

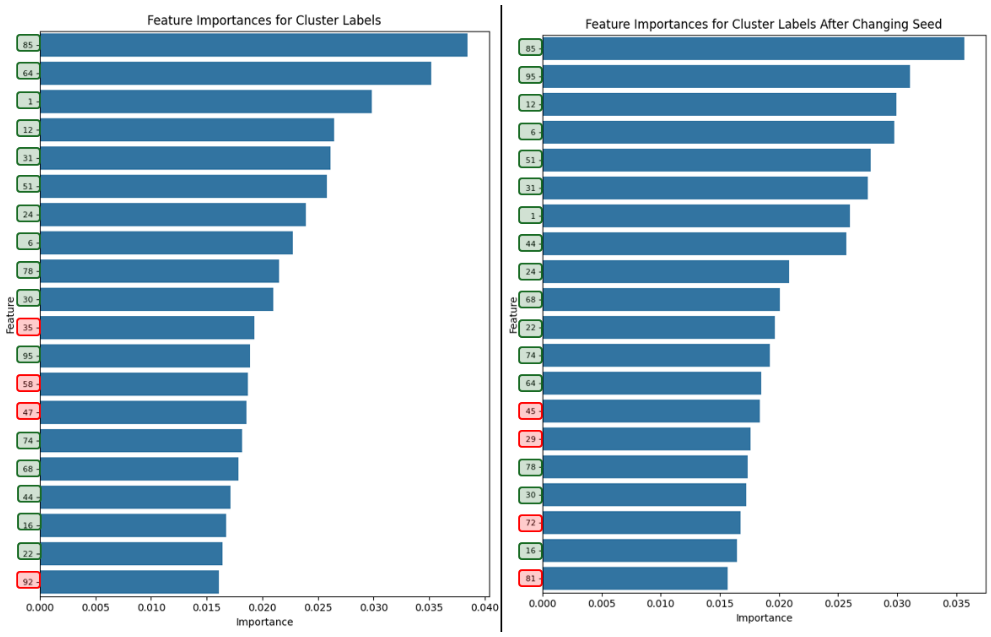

***Figure 6.11:*** *Comparing features (i.e., probe groups) with the highest 20 GMM feature importances with the initial seed (left) and after changing the fixed seed (right). Probe groups are highlighted green if they appeared in the top 20 of both iterations. They are marked red if they only appear in one of them.*

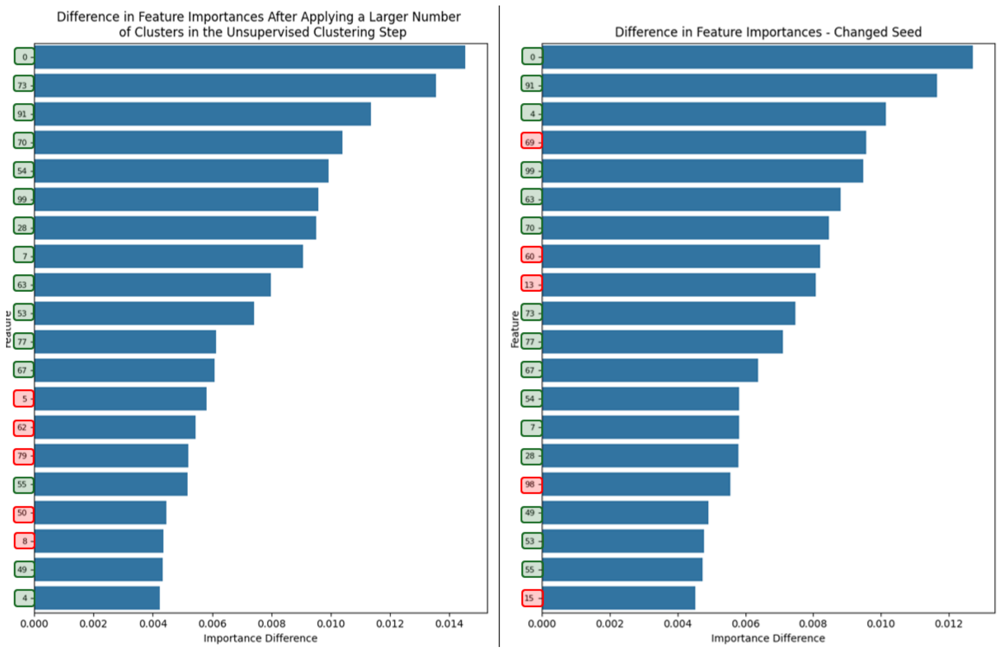

***Figure 6.12:*** *Comparing features (i.e., probe groups) with the highest 20 GMM feature importances differences before (left) and after changing the fixed seed (right). Probe groups are highlighted green if they appeared in the top 20 of both iterations. They are marked red if they only appear in one of them.*

## 6.9 Conclusion

While we have not delved deeply into the biological significance of the different probe groups, our analysis reveals that these probe groups are not random but are associated with many significant biological processes, functions, and components. It is important to note that only the most significant associations have been highlighted here, although many more were identified.

Furthermore, we have proposed a potential method for identifying features important for the local structure of the data. This method shows considerable promise and could serve as a valuable tool for future analyses.

# 7 Discussion

## 7.1 Summary of Main Findings

The primary objective of this thesis was to develop a robust and adaptable step-by-step interpretation framework for UMAP outputs, addressing the research question: ‘How can the low-dimensional embedding of UMAP be interpreted?’. The proposed five-step interpretation framework demonstrates promising results in identifying features that are important to both the global and local structures of the low-dimensional UMAP embedding. By gaining insights into these structures, we have considerably enhanced our understanding of UMAP embeddings.

Furthermore, the successful application of this interpretation framework to two different datasets, along with robustness testing through the alteration of fixed seeds, indicates that the framework is adaptable, effective and reliable. This suggests that the framework is useful and applicable to different datasets, providing consistent and meaningful interpretations of UMAP embeddings.

## 7.2 Implications

The provided interpretation framework of this thesis enhances the utility of UMAP in fields dealing with high-dimensional data, such as bioinformatics, genetics, and business analytics. By enabling more detailed interpretations of UMAP outputs, analysts can go beyond initial data explorations to more nuanced analyses. For instance, in bioinformatics, this framework could help identify subtle variations within known cancer types (Rather & Chachoo, 2022). Understanding these local structures might reveal new patterns that could inform more personalized treatment strategies, improving patient outcomes. Similarly, in business analytics, this enhanced approach allows for the identification of key factors that define the global or local structure of customer datasets. These insights can be crucial for refining marketing strategies and customer segmentation, leading to more targeted and effective business practices.

## 7.3 Limitations

One limitation of this study lies in the feature selection process for both the global and local structures. While we used BIC scores and feature importances — or differences in feature importances — to identify features relevant to the global or local structures, these approaches have only been tested on two datasets for the global structure and one dataset for the local structure. Further testing on additional datasets is necessary to ensure their effectiveness and applicability across a wider range of data.

Additionally, the feature selection methods developed in this study are based on practical effectiveness rather than grounded in statistical theory. Specifically, through trial-and-error and visual inspection, we observed that the chosen feature selection approaches appear to perform well. However, because these methods lack a rigorous statistical foundation, we cannot conclusively prove their efficacy at this point, which poses a limitation to our work.

## 7.4 Suggestions for Future Research

### Feature Selection

Future research on the feature selection process could involve testing the developed methods on additional datasets to assess their effectiveness in identifying important features.

Furthermore, providing a theoretical or mathematical foundation for the chosen feature selection methods would enhance the credibility of this interpretation framework. Such a foundation would not only strengthen confidence in the methods' efficacy but also offer deeper insights into why they work and under what conditions they may not be effective.

Lastly, developing a quantitative metric that estimates the extent to which a feature explains the global and local structures would make the feature selection process more objective. This would first require establishing a metric that quantifies the meanings of global and local structures, which presents another area for future research. Developing such an objective metric would enable more rigorous validation of our methods and facilitate the exploration of alternative approaches.

### Bypassing Clustering with Direct UMAP Prediction

In our research, we have addressed the nonlinear structures captured by UMAP by discretizing them through clustering. However, alternative strategies can bypass clustering entirely. One such method involves directly using the first two dimensions of the UMAP embedding as predictors for feature values.

Previously, we used high-dimensional features to predict low-dimensional cluster labels (see step 3 in the interpretation framework, chapter 2.3). In contrast, this approach uses the low-dimensional embedding coordinates to predict high-dimensional feature values. By comparing goodness-of-fit metrics, such as R-squared, across different features, this method offers a direct way to understand which variables best predict the position (embedding values) of an object in the reduced-dimensional space, without the need for the intermediate step of clustering to create pseudo-labels.

Exploring this method could provide insights into the inherent relationships within the data, potentially offering an alternative approach to gain insights into the global and/or local structures of the UMAP output.

The model can be expressed as follows:

$$v = \beta_1 \times d_1 + \beta_2 \times d_2 + \beta_3 \times d_1 \times d_2,$$

where

- $v$ represents the value of the feature being analyzed;
- $\beta_1$ and $\beta_2$ are coefficients for the UMAP embedding values;
- $d_1$ and $d_2$ are the embedding values from the first and second dimensions of UMAP, respectively;
- $\beta_3 \times d_1 \times d_2$ is an interaction term, introducing a nonlinear component to the model.

### Python or R Package

Moreover, the interpretation framework developed in this thesis could serve as a foundation for new programming packages, for example, in Python or R. Therefore, another goal for future research could be to develop programming packages that integrate seamlessly with existing UMAP packages, automating the analysis of outputs and making the interpretation of the UMAP output more accessible and understandable to users.

## 7.5 Conclusion

Despite the limitations noted, this thesis marks an important initial step toward enhancing the interpretability of UMAP outcomes. Prior to this work, the reasons behind the arrangement of data points in UMAP's low-dimensional output were largely unclear. This study has begun to shed light on the topology and offers valuable insights into its interpretation.

Importantly, this research has established a framework that can be customized to suit various datasets and application fields. This adaptability is crucial given UMAP's broad applicability across diverse data types, where a one-size-fits-all approach is often inadequate. Consequently, the interpretation framework developed here provides a robust foundation for future research, ensuring that this thesis makes a meaningful contribution to the field.

# 8 References

Allen Institute for Brain Science. (2024a). *Allen brain atlas* [Accessed the dataset H0351.2001 for neuroscientific research.]. Retrieved May 29, 2024, from https://human.brain-map.org/static/download

Allen Institute for Brain Science. (2024b). *Dataset h0351.2001* [Data for one brain used in neuroscientific research.]. Retrieved May 29, 2024, from https://human.brain-map.org/static/download/H0351.2001

Becht, E., McInnes, L., Healy, J., et al. (2019). Dimensionality reduction for visualizing single-cell data using umap. *Nat Biotechnol*, *37*, 38–44. https://doi.org/10.1038/nbt.4314

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*(6), 1373–1396.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Chase Lipton, Z., Elkan, C., & Narayanaswamy, B. (2014). Thresholding classifiers to maximize f1 score. *arXiv e-prints*, arXiv:1402.

Coenen, A., & Pearce, A. (2022). *Understanding umap*. PAIR-code. Retrieved August 28, 2024, from https://pair-code.github.io/understanding-umap/

Cohen-Addad, V., Kanade, V., Mallmann-Trenn, F., & Mathieu, C. (2019). Hierarchical clustering: Objective functions and algorithms. *Journal of the ACM (JACM)*, *66*(4), 1–42.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

Dong, W., Charikar, M., & Li, K. (2011). Efficient k-nearest neighbor graph construction for generic similarity measures. *Proceedings of the 20th International Conference on World Wide Web*, 577–586.

Eden, E., Navon, R., Steinfeld, I., Lipson, D., & Yakhini, Z. (2023). *GOrilla: Gene Ontology enRIchment anaLysis and visuaLizAtion tool* [Accessed: 2024-09-07]. https://cbl-gorilla.cs.technion.ac.il/

Goeman, J. J., & Le Cessie, S. (2006). A goodness-of-fit test for multinomial logistic regression. *Biometrics*, *62*(4), 980–985.

Habowski, A. N., Habowski, T. J., & Waterman, M. L. (2021). Geco: Gene expression clustering optimization app for non-linear data visualization of patterns. *BMC Bioinformatics*, *22*(29). https://doi.org/10.1186/s12859-020-03951-2

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Second). Springer. https://link.springer.com/book/10.1007/978-0-387-84858-7

Johnstone, I. M., & Titterington, D. M. (2009). Statistical challenges of high-dimensional data. *Philosophical Transactions of the Royal Society A*, *367*(1906), 4237–4253. https://doi.org/10.1098/rsta.2009.0159

McInnes, L., Healy, J., & Melville, J. (2018a). *How umap works*. Retrieved May 16, 2024, from https://umap-learn.readthedocs.io/en/latest/index.html

McInnes, L., Healy, J., & Melville, J. (2018b, February). *Umap: Uniform manifold approximation and projection for dimension reduction*. https://arxiv.org/abs/1802.03426

Ordun, C., Purushotham, S., & Raff, E. (2020, May). *Exploratory analysis of covid-19 tweets using topic modeling, umap, and digraphs*. https://doi.org/10.13016/m24rdv-a0gg

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572.

Pino, T. (2016, January). *Melbourne housing market*. Kaggle. Retrieved May 11, 2024, from https://www.kaggle.com/datasets/anthonypino/melbourne-housing-market

Pires, A. M., & Branco, J. A. (2019, February). *High dimensionality: The latest challenge to data analysis*. Cornell University. https://arxiv.org/pdf/1902.04679.pdf

Rather, A. A., & Chachoo, M. A. (2022). Umap guided topological analysis of transcriptomic data for cancer subtyping. *International Journal of Information Technology*, *14*(1), 341–348. https://doi.org/10.1007/s41870-022-00850-6

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, *100*(5), 401–409.

Tenenbaum, J. B., Silva, V. d., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, *17*(4), 401–419.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, *9*(11).

Xiao, H., Rasul, K., & Vollgraf, R. (2017). *Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms*. Zalando Research. https://github.com/zalandoresearch/fashion-mnist

# Appendix A: Application on the Melbourne Housing Dataset

This chapter contains supplementary material related to the application of the five-step framework to the Melbourne Housing dataset.

## A.1 Step 2: Unsupervised Clustering: Gaussian Mixture Model

Previously, it was shown how hierarchical clustering can be used in step 2, i.e., unsupervised clustering, of the interpretation framework. In the following, an alternative statistical method for this step is given, namely Gaussian Mixture Models. The number of clusters for the GMM model was determined by a combination of insights: Previously, hierarchical clustering provided a potential number of clusters, namely five. Moreover, inspecting the UMAP visualization suggested the presence of four prominent clusters alongside several smaller ones. Both these insights lead us to select five clusters for GMM. For the covariance matrix configuration, the GMM allows for several options: 'full', 'tied', 'diag', and 'spherical', with 'full' offering the least restrictive assumption about cluster shape and 'spherical' the most. Since the UMAP output shows that the clusters could have different sizes and shapes, we opted for the 'full' covariance matrix. This offers the most flexibility by allowing each cluster to have its own general covariance matrix, thus accommodating varying sizes, shapes, orientations, and volumes of the clusters effectively. Figure A.1 illustrates the resulting clusters.

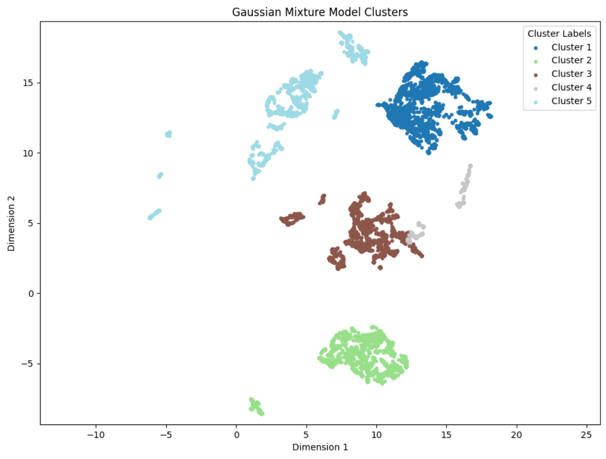

***Figure A.1:*** *This plot shows the clustering results of a Gaussian Mixture Model applied to the Melbourne Housing dataset, where five clusters were specified. A 'full' covariance matrix was used to ensure maximum flexibility in adapting to the varied shapes and orientations of these clusters.*

## A.2 Step 5: Visualizations

Below, the visualizations of the remaining features from the Melbourne Housing dataset can be found.

To enhance the clarity and interpretability of the visualizations, certain feature values have been capped at specific thresholds. The reason is that a small number of outliers — such as residences with an unusually high price — skewed the color representation and therefore made the distinction between e.g., 500,000 Australian Dollars and 3.5 million Australian Dollars unclear. By capping values at a maximum (or a minimum, such as in the case of 'Year Built') the visual differentiation becomes clearer.

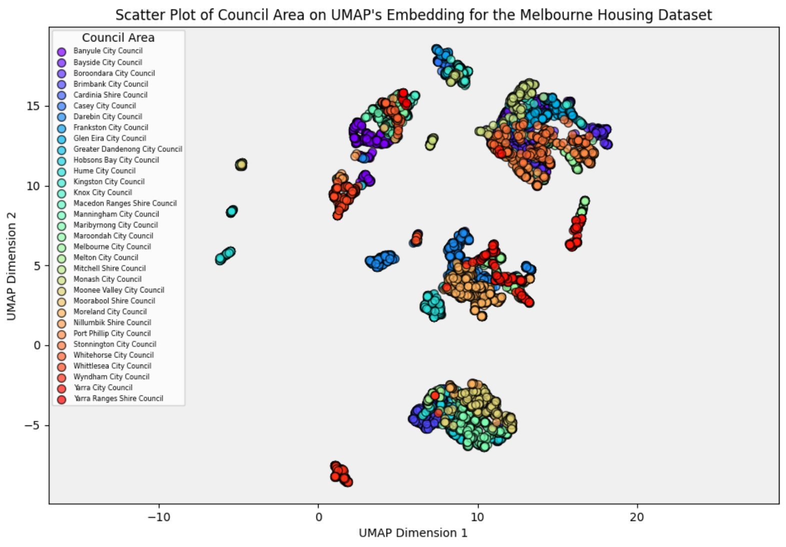

***Figure A.2:*** *UMAP output showing council area.*

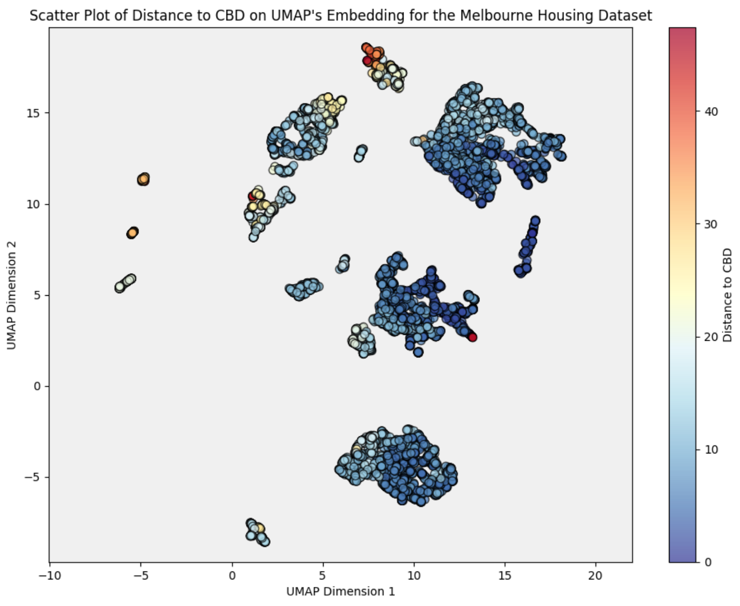

***Figure A.3:*** *UMAP output showing distance to CBD.*

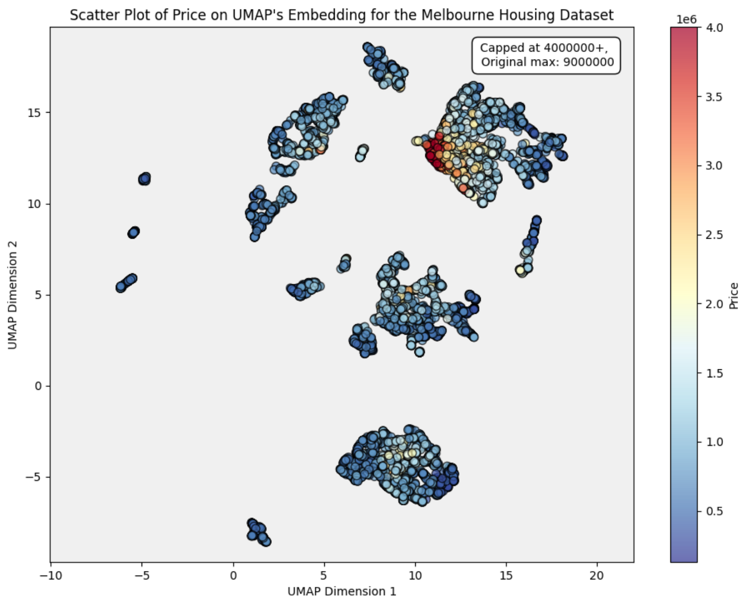

***Figure A.4:*** *UMAP output showing price of residences.*

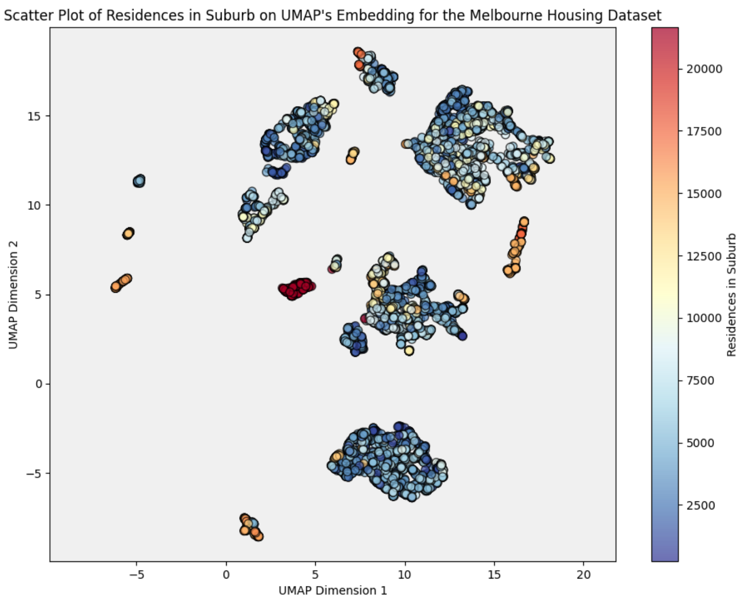

***Figure A.5:*** *UMAP output showing number of residences in suburb.*

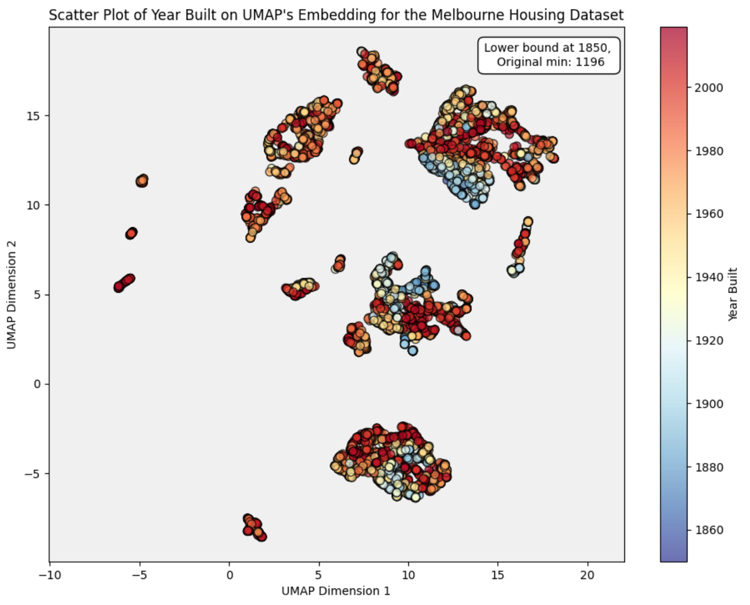

*Figure A.6:* *UMAP output showing the year the residence was built.*

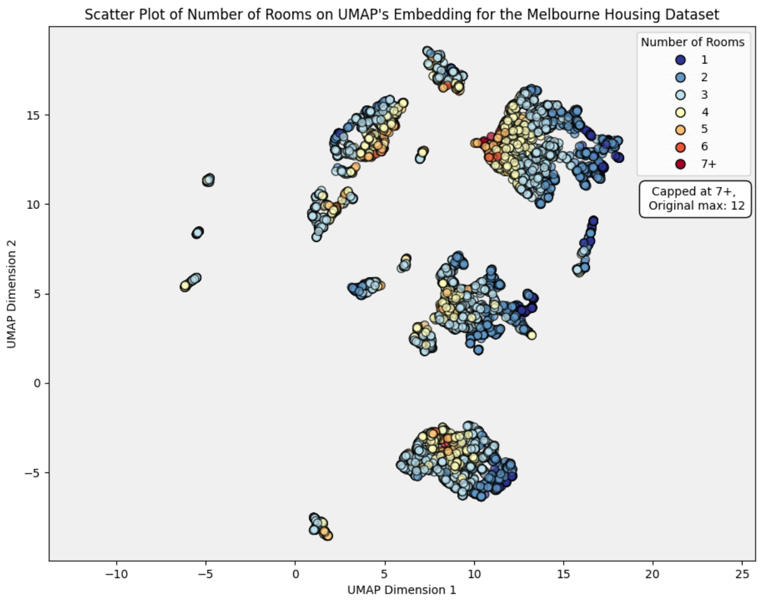

*Figure A.7:* *UMAP output showing number of rooms.*

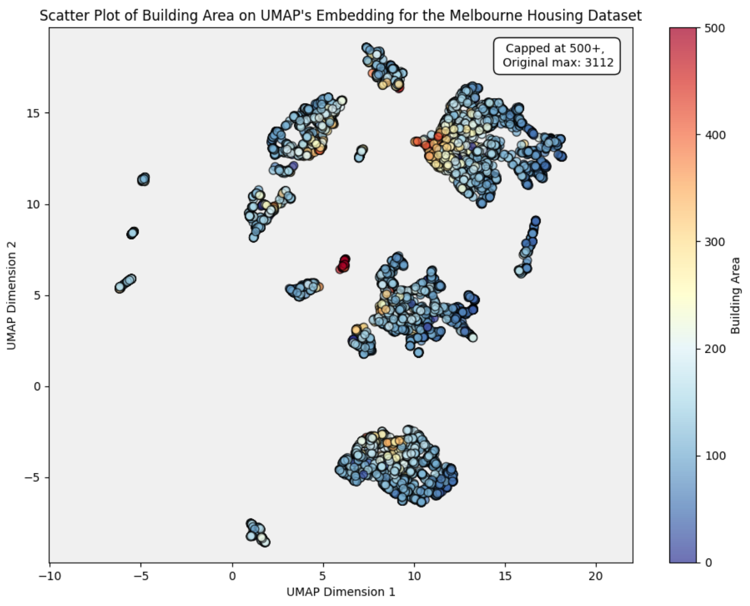

***Figure A.8:*** *UMAP output showing the building area of the residence in square meters.*

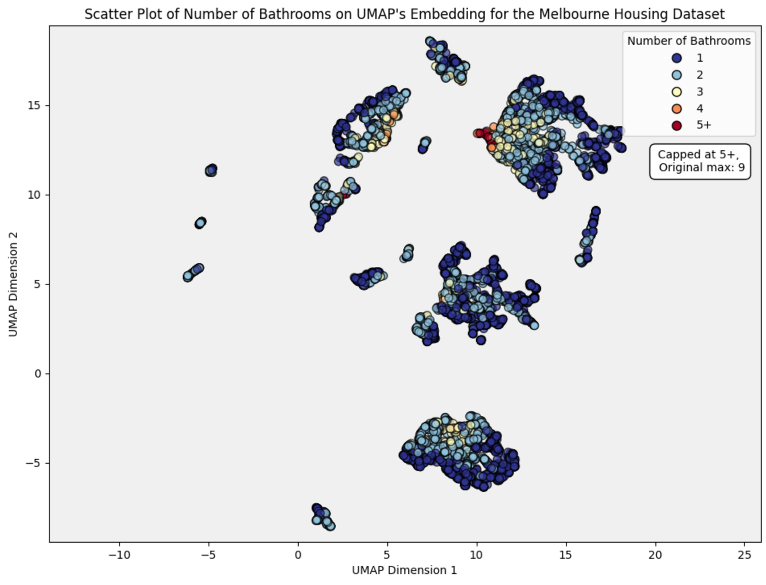

***Figure A.9:*** *UMAP output showing number of bathrooms.*

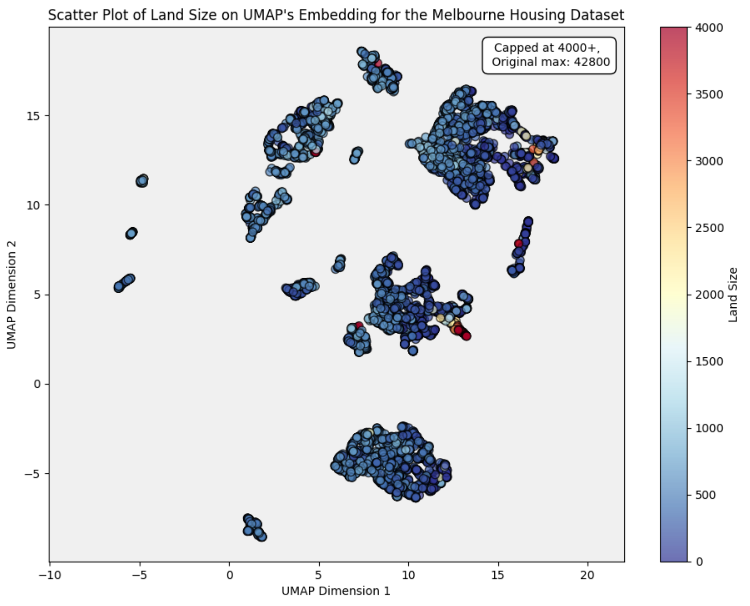

***Figure A.10:*** *UMAP output showing the land size of the residence in square meters.*

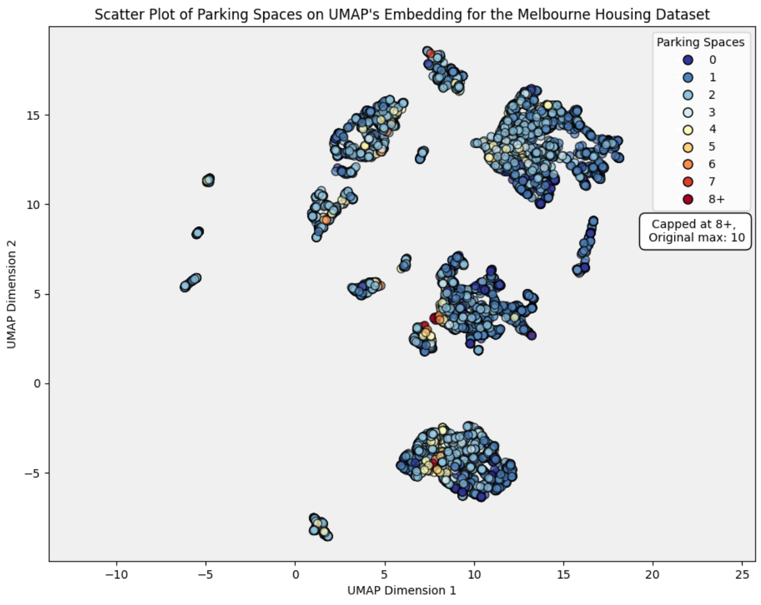

***Figure A.11:*** *UMAP output showing number of parking spaces.*

## A.3 Robustness

The results of the UMAP algorithm with a changed seed can be seen below.

### Robustness: Step 1: UMAP

The resulting UMAP outcome can be seen in figure A.12

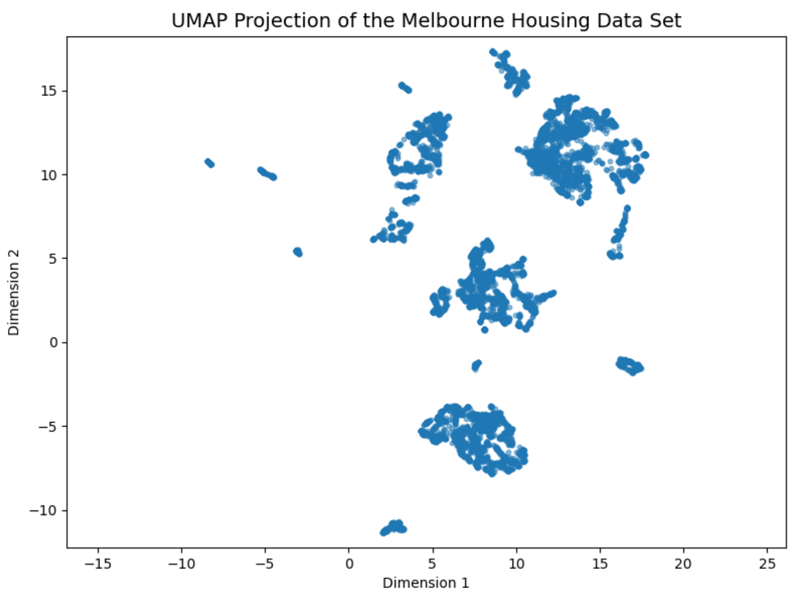

***Figure A.12:*** *UMAP output after changing its fixed seed.*

### Robustness: Step 2: Hierarchical Clustering

The dendrogram from hierarchical clustering can be seen in Figure A.13. As before, we choose 5 clusters. These clusters are visualized in Figure A.14.

### Robustness: Step 3: Multinomial Logistic Regression

The BIC scores based on multinomial logistic regression can be found in Figure A.15. The order of features are very similar to what we found previously.

### Robustness: Step 5: Visualization

We chose to visualize the same features as before, since the BIC scores and feature order is very similar to before. The visualizations tell us the same story as before and therefore, our final interpretation of the UMAP outcome does not change.

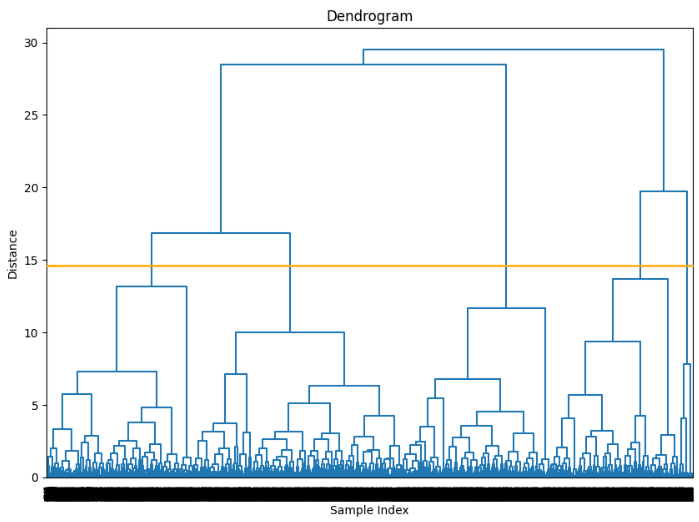

***Figure A.13:*** *Dendrogram from hierarchical clustering.*

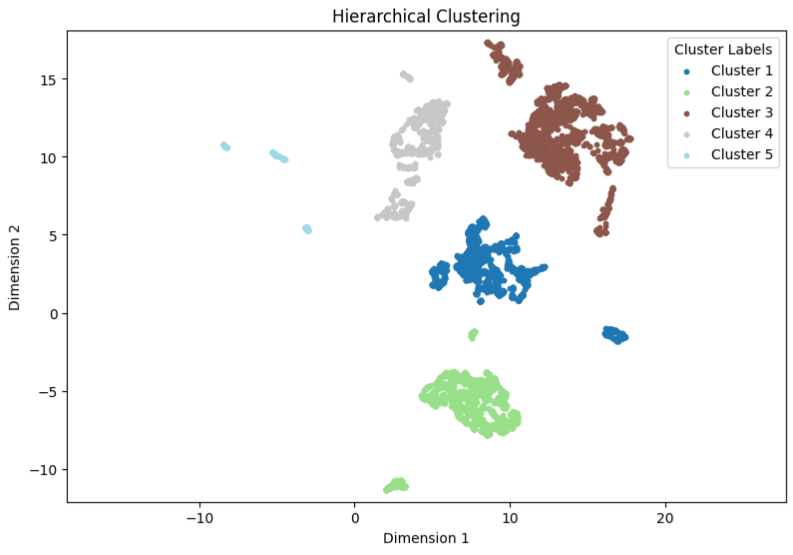

***Figure A.14:*** *Colors based on hierarchical clustering.*

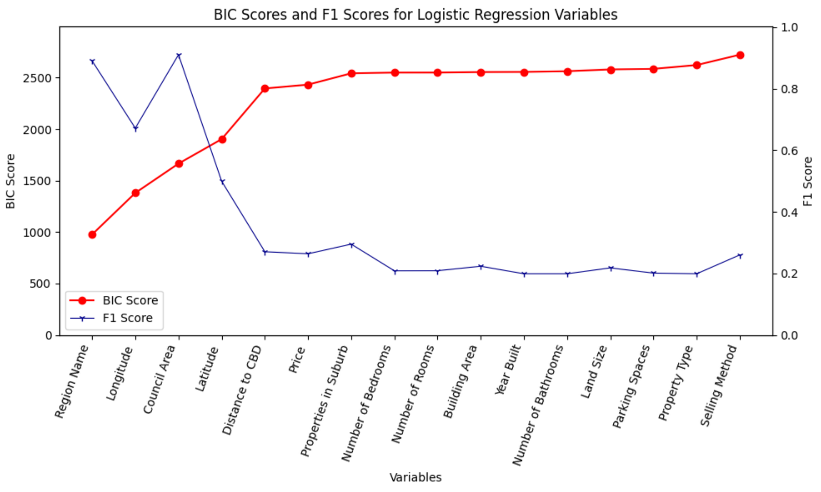

***Figure A.15:*** *BIC and F1 scores based on multinomial logistic regression.*

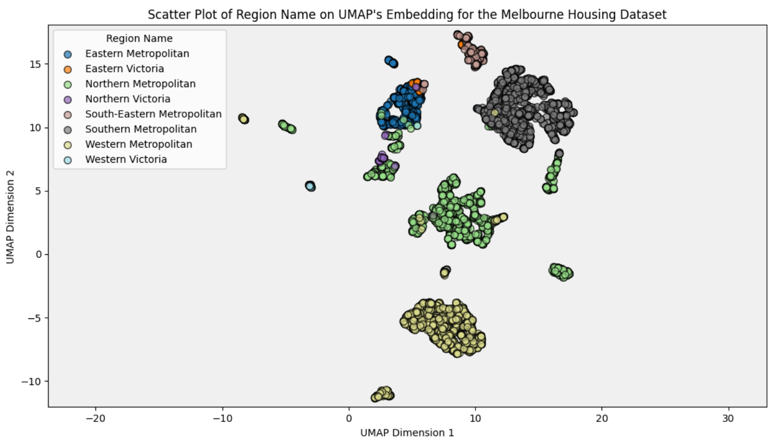

***Figure A.16:*** *Visualization of region names in the UMAP embedding.*

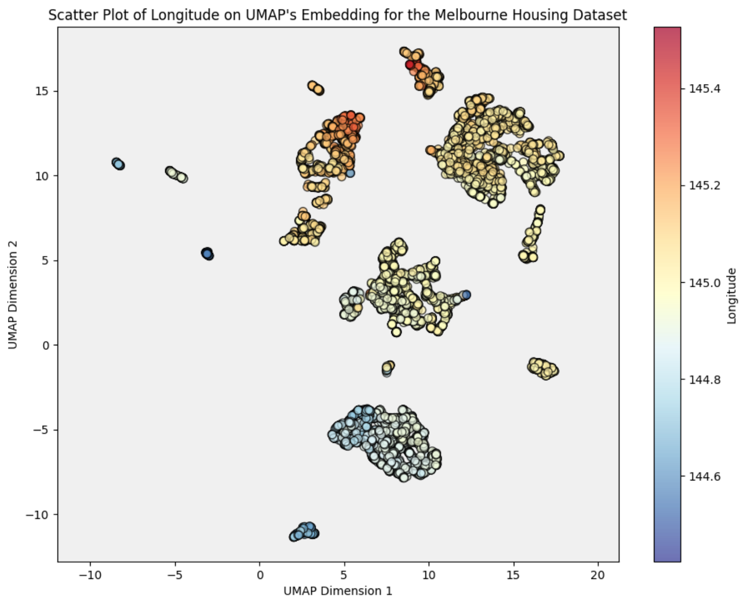

***Figure A.17:*** *Visualization of the longitude values in the UMAP embedding.*

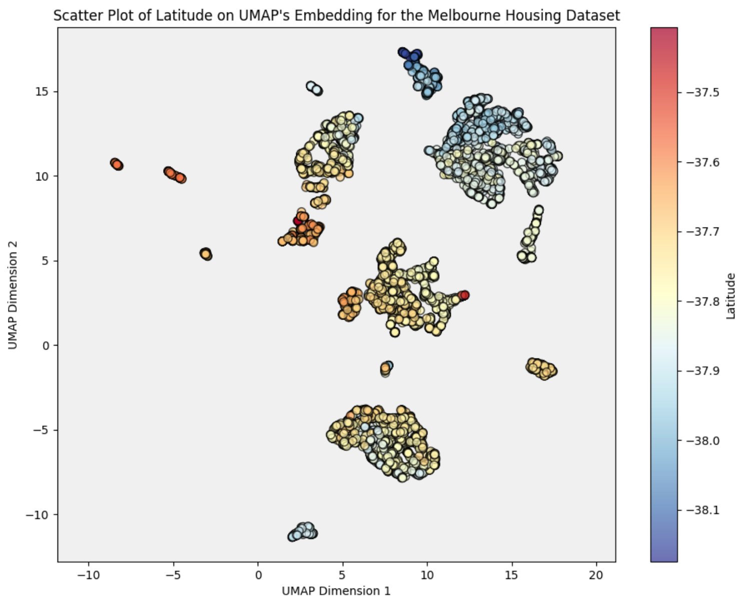

***Figure A.18:*** *Visualization of the latitude values in the UMAP embedding.*

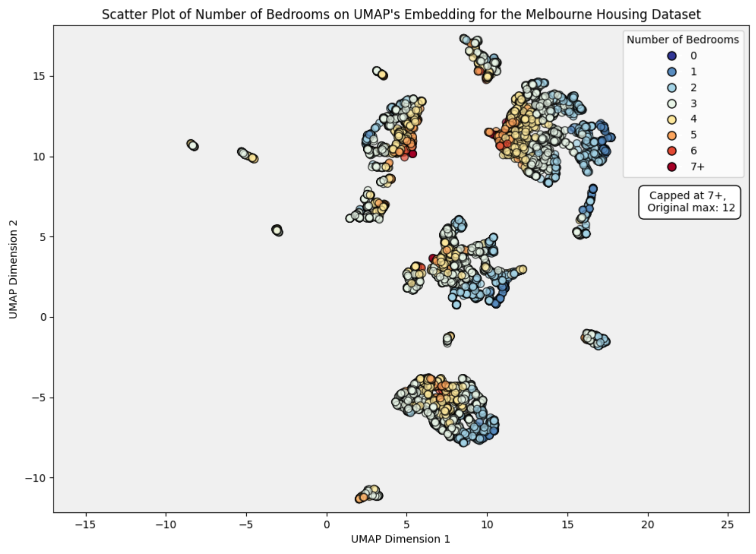

***Figure A.19:*** *Visualization of the number of bedrooms in the UMAP embedding.*

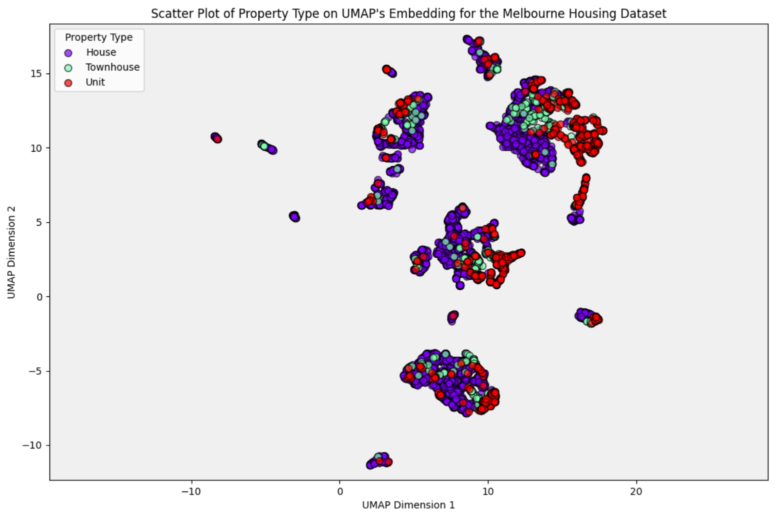

***Figure A.20:*** *Visualization of the property types in the UMAP embedding.*

# Appendix B: Application on the Brain Samples Dataset

This chapter contains supplementary material related to the application of the five-step framework to the Brain Samples dataset.

## B.1 Step 2: Unsupervised Clustering: Gaussian Mixture Models

In figure B.1, a zoomed-in graph that focusses on the BIC scores for 1 to 40 clusters can be found. The minimum BIC score is observed at 34 clusters, but the reductions in BIC scores from this point compared to smaller cluster counts, such as 8 or 14, are relatively minor.

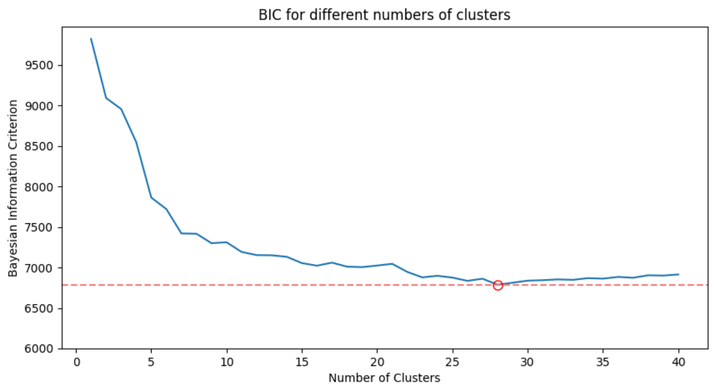

***Figure B.1:*** *BIC scores for GMM clustering applied to the low-dimensional embedding of the Brain Samples dataset. The number of clusters range from 1 to 40 clusters. The circle denotes the number of clusters with the lowest BIC score. Note that the y-axis does not start at 0.*

## B.2 Step 2: Unsupervised Clustering: Hierarchical Clustering

In this section, the results of hierarchical clustering on the Brain Samples dataset can be found.

To find balanced and well-separated clusters, the 'complete' linkage method was applied when applying the hierarchical clustering algorithm. The dendrogram in Figure B.2 shows at which heights clusters merge. As mentioned before, deciding on the number of clusters can be subjective, but a good rule of thumb is to avoid excessive splitting but have enough clusters to do well-informed analyses. For this dataset, a cut-off point shortly below 5 was chosen. To make the cutoff point clearer, a line was added to the graph. We can see that this cutoff point results in 8 clusters. Figure B.3 shows the low-dimensional UMAP embedding with samples colored according to their assigned cluster from hierarchical clustering. We can see that the clustering result is very similar to the GMM clustering result.

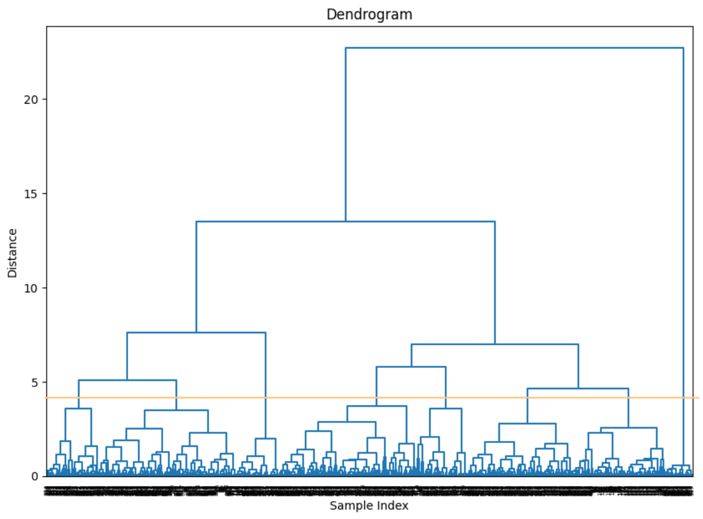

***Figure B.2:*** *Dendrogram of the hierarchical clustering from the Brain Samples dataset. The orange line indicates a subjectively chosen cutoff point at a height of approximately 5, resulting in 8 clusters.*

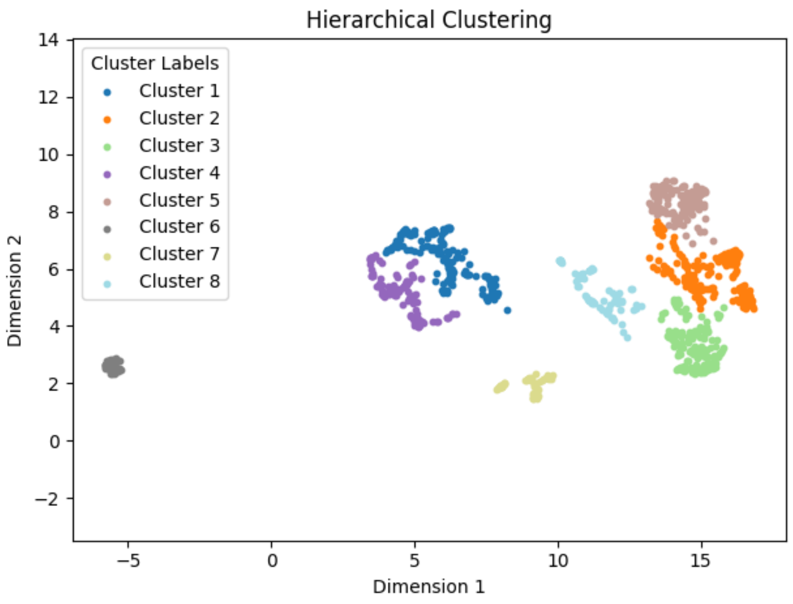

***Figure B.3:*** *UMAP low-dimensional embedding of the Brain Samples dataset, where each point represents a brain sample. The samples are colored based on the 8 clusters chosen by the author from the hierarchical clustering results.*

## B.3 Step 3: Supervised Classification: Multinomial Logistic Regression

We built a multinomial logistic regression model per feature, i.e. per probe group. We then evaluated each model's effectiveness using two metrics: the F1 score, which assesses the balance between precision and recall, and the Bayesian Information Criterion (BIC), which considers model complexity and likelihood. The results, displayed in Figure B.4, were sorted by the F1 score to facilitate the comparison of performance and complexity across models. Since we now don't have any categorical variables, we focus on F1-score as our main goodness-of-fit metric.

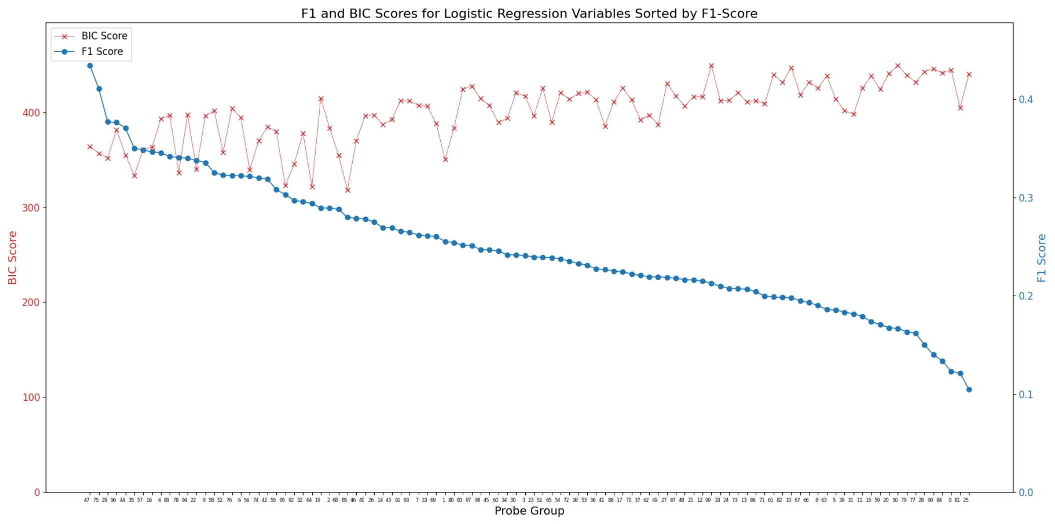

***Figure B.4:*** *This graph displays the F1 and BIC scores for 100 logistic regression models, each based on a unique probe group, sorted by F1 score to highlight those with the highest predictive performance. Each model uses the mean probe value per sample as the sole predictor, illustrating the impact of average probe values on model outcomes.*

# Appendix C: Explanation of Code

In the following, an explanation of the six different files and the functions they contain is given. The code for this analysis is available on GitHub at this repository.

## C.1 UMAP

The **A_Umap** file contains a function that utilizes the existing UMAP algorithm to run on datasets. This function is designed to reduce the dimensionality of data to two dimensions for easier visualization.

**Function `run_UMAP`**

The inputs of `run_UMAP` are:

- **standardized_dataset**: A pandas.DataFrame that should be standardized before processing.
- **random_seed** (default: 1): The seed for the UMAP algorithm's random number generator, ensuring reproducible outcomes.
- **name_dataset** (default: 'Dataset Name'): The name of the dataset, textbf for labeling the plot.
- `dataset_with_label` (optional): A pandas.DataFrame including labels in its last column, if applicable.
- **is_labelled** (default: 'No'): Indicates whether the dataset contains labels that should be reflected in the plot's coloring.
- **legend** (default: True): Whether a legend should be displayed on the plot, identifying clusters or groups based on their colors.

The outputs of `run_UMAP` are:

- **Return Object**: `embedding` - A numpy array containing the 2D coordinates of the dataset following UMAP reduction.
- **Graphical Output**: A scatter plot showing the UMAP projection. If the dataset is labeled, the plot points are color-coded according to their labels, with an optional legend to identify these colors.

## C.2 Unsupervised Clustering

The **B_UnsupervisedClustering** file contains functions that perform clustering on datasets. It focuses on hierarchical and GMM clustering, which are applied to the low-dimensional embedding resulting from applying `run_UMAP` function. The functions in this file generate visual outputs to display the clustering results and provide tools for determining the optimal number of clusters.

### Function `hierarchical_clustering`

The function `hierarchical_clustering` performs hierarchical clustering on the low-dimensional UMAP embedding and can generate both a dendrogram and a scatter plot of the clusters.

The inputs are:

- **embedding**: A 2D numpy array of the data to cluster.
- **n_clusters**: The number of clusters to form.
- **graph** (default: True): If True, both a dendrogram and a scatter plot are generated. If False, only a dendrogram is produced.
- **link** (default: ‘complete’): The linkage criterion to use (e.g., ‘single’, ‘complete’, ‘average’, ‘ward’, ‘centroid’, ‘median’).
- **alpha** (default: 1): The opacity level of the scatter plot points.

The outputs are:

- **Return Object**: The cluster labels for each point in the dataset.
- **Graphical Output**: Depending on the graph parameter, a scatter plot, a dendrogram, or both visualizing the clusters.

### Function `gmm_clustering`

The function `gmm_clustering` applies Gaussian Mixture Model clustering to the low-dimensional UMAP embedding and optionally plots the results.

The inputs are:

- **embedding**: A 2D numpy array of the data to cluster.
- **n_component**s: The number of Gaussian components (clusters) in the model.
- **cov_type** (default: ‘full’): The type of covariance parameters to use (‘full’, ‘tied’, ‘diag’, ‘spherical’).
- **alpha** (default: 1): The opacity level of the scatter plot points.
- **plot** (default: True): Whether to plot the clustering result.
- **legend_small** (default: False): If True, the legend on the plot uses a smaller font size.
- **m_iter** (default: 500): The maximum number of iterations for the GMM algorithm.
- **seed_value** (default: 111): The seed for the random number generator.

The outputs are:

- **Return Object**: **labels -** The cluster labels for each point in the dataset.
- **Graphical Output**: A scatter plot showing the clusters if **plot** is True.

### Function `find_optimal_clusters`

The function `find_optimal_clusters` evaluates the optimal number of clusters for GMM clustering using the Bayesian Information Criterion (BIC).

The inputs are:

- **data**: The dataset to cluster.
- **min_clusters**: Minimum number of clusters to consider.

- **max_clusters**: Maximum number of clusters to consider.
- **cov_type** (default: ‘full’): The covariance type for the GMM.
- **y_start**: The starting value on the y-axis for BIC plotting.
- **seed**: Seed for random state.
- **num_clus**: If specified, highlights the BIC value for this number of clusters.

The outputs are:

- **Return Objec**t: The number of clusters that minimizes the BIC.
- **Graphical Output**: A line plot showing the BIC values for each possible number of clusters. If num_clus is specified, it is highlighted on the plot for reference.

**Function `meanpergroup`**

The function `meanpergroup` computes the mean of each cluster group in the provided dataframe.

The inputs are:

- **labels**: Cluster labels as an array.
- **T_dataframe**: The dataframe containing the data grouped according to labels.

The outputs are:

- **Return Object**: A new dataframe containing the mean values for each group.

## C.3 Supervised Classification

The **C_SupervisedClassification** file includes multinomial logistic regression and random forest classification, both of which assess the influence of variables and predict outcomes based on labelled data.

**Function `logregression`**

The function `logregression` runs multinomial logistic regression on both numerical and categorical variables within a dataset. It aims to identify the most influential predictors.

The inputs are:

- **dataset**: A Dataframe containing standardized numerical variables and non-dummy coded categorical variables.
- **label**: Categorical class labels derived from clustering.
- **sort_by** (default: ‘BIC’): The criterion to sort the results by, which can be ‘AIC’, ‘BIC’, or ‘f1-score’.
- **seed** (default: 111): Seed for random state to ensure reproducibility.

The outputs are:

- **Return Object**: A Dataframe sorted by the specified criterion (sort_by), listing each variable along with its corresponding F1 score, AIC, and BIC.

#### Function `randomforest_fun`

The function `randomforest_fun` applies a Random Forest classifier to a dataset to predict labels and evaluates the importance of each feature within the model.

The inputs are:

- **standardized_dataset**: A Dataframe where numerical variables are standardized and categorical variables are one-hot encoded.
- **labels**: Integer class labels obtained from clustering.
- **seed** (default: 111): Seed for random state to ensure reproducibility.
- **max_visualized** (default: 100): The maximum number of features to visualize in the plot.
- **label_size** (default: 6): Font size for the feature labels on the plot.

The outputs are:

- **Return Object**: A tuple containing two elements:
  – A sorted list of tuples, each featuring a name and its importance.
  – A dictionary mapping each feature name to its importance, sorted in descending order.
- **Graphical Output**: A horizontal bar plot visualizing the feature importances, aiding in identifying which features most significantly impact the model predictions.

## C.4 Visualization

The **D_Visualization** file includes functions designed for visualizing data points from embeddings, suitable for both categorical and continuous data. These functions offer flexibility in displaying data, with options to adjust visual parameters like color mapping, transparency, and the placement of informational text regarding data capping.

#### Function `visualize_points`

The function `visualize_points` generates scatter plots for UMAP's low-dimensional embeddings, accommodating categorical and continuous features. It includes options for displaying capped values either as part of the colorbar label or within a text box on the plot.

The inputs are:

- **embedding**: A 2D numpy array representing the data points to be plotted.
- **feature**: The feature based on which the data points are coloured.
- **unlabelled_data_unstandardized**: DataFrame containing the original data, used to extract feature values.
- **data_set_name**: Name of the dataset for labelling purposes.
- **legendname**: Prefix for legend entries.
- **max_cap_value** (optional): Upper limit for capping feature values (continuous data).
- **min_cap_value** (optional): Lower limit for capping feature values (continuous data).
- **title** (optional): Custom title for the plot.
- **title_font**, **legendfont**: Font sizes for the title and legend.
- **colormap**: The colormap used for the plot.
- **transparency**: Opacity of the plot points.
- **note_x, note_y**: Coordinates for placing textual notes on the plot.

- **cap_info_location**: Determines where capping information is displayed ('label' or 'text_box').

The outputs are:

- **Graphical Output**: A scatter plot displaying the data points with appropriate color coding and legend. For continuous features, capping information can be included as part of the colorbar label or within a text box on the plot, based on cap_info_location.

**Function `visualize_points_discrete`**

The function `visualize_points_discrete` is tailored for visualizing discretized or binned data points. It maps numerical values to distinct colors and includes a customized legend that reflects these mappings. If the values were capped, a textual note can be included.
The inputs are:

- **embedding**: A 2D numpy array of data points.
- **feature**: The feature of interest for color coding.
- **unlabelled_data_unstandardized**: DataFrame with original data values.
- **data_set_name**: Dataset name for use in titles.
- **legendname**: Prefix for legend labels.
- **max_cap_value**: Specifies the maximum value or cap for discretizing the data.
- **title** (optional), **title_font**, **legendfont**, **transparency**, **note_x**, **note_y**: Visual customization options including the placement of notes.

The outputs are:

- **Graphical Output**: A scatter plot where discretized or binned data values are distinctly colored, with a custom legend and a note detailing the maximum value before capping and the applied cap.

## C.5 Application: Melbourne Housing Dataset

In the 'Application_Melbourne_Housing' file, a series of data preprocessing, dimensionality reduction, clustering, supervised classification, and visualization techniques was applied to analyse the Melbourne Housing dataset. The workflow is outlined as follows:

### 0. Data Preprocessing

The dataset was loaded and cleaned by renaming columns for clarity, removing unnecessary columns and deleting rows with missing values.

Numerical variables were standardized using `StandardScaler`, and categorical variables were transformed into dummy variables using `OneHotEncoder`.

### 1. UMAP

The preprocessed data was input into the UMAP function (`run_UMAP`) to reduce its dimensionality, creating a two-dimensional embedding of the housing data. A fixed seed of 111 was chosen.

When testing for robustness, as explained in chapter 5.7, this fixed seed was changed to 222.

### 2. Unsupervised Clustering

Hierarchical clustering was applied to the UMAP embedding using the previously described function `hierarchical_clustering` to categorize the data into five clusters.

As an alternative unsupervised clustering technique, GMM clustering was also performed using the function (`gmm_clustering`), creating 5 clusters based on the same embedding. The fixed seed was set to 111.

### 3. Supervised Classification

The function `logregression` was used to perform logistic regression analysis. To split the dataset into a test and a training set, a fixed seed was set to 111. When testing the method for robustness, as explained in chapter 5.7, the fixed seed was changed to 222.

This analysis identified the most influential predictors for the clusters obtained from hierarchical clustering. The results were sorted based on the Bayesian Information Criterion (BIC).

A custom dual-axis plot was applied to compare BIC scores and F1 scores across the predictors identified in the logistic regression.

### 4. Visualization

All features were visualized using the functions `visualize_points` and `visualize_points_discrete`.

For continuous features, capping was possible and this information was displayed either directly in the plot's legend or within a textbox.

Discrete visualizations were employed to show binned representations of features like the number of bedrooms and bathrooms, enhancing the interpretability of these variables in the housing context.

## C.6 Application: Brain Samples Dataset

The 'Application_Brain_Samples' file implements a the interpretation framework on the brain samples dataset. The workflow integrates data preprocessing, dimensionality reduction, clustering, and visualization to explore the structure and relationships within the data.

### 0. Data Loading and Preprocessing

The data was loaded, with one dataset having samples as rows and genes as columns, and another transposed. These datasets were scaled using `StandardScaler` to normalize gene expression levels across samples.

### 1. UMAP

The UMAP function (`run_UMAP`) was applied to the scaled dataset to produce a two-dimensional embedding, which was visualized to inspect the data distribution. A fixed seed of 123 was used.

### 2. Unsupervised Clustering

Hierarchical clustering and Gaussian Mixture Model (GMM) clustering were performed on the UMAP embeddings to identify groupings among the brain samples. GMM was performed with 8 clusters and the clustering results were used for further analysis. For GMM, the fixed seed was set to 123.

When analyzing the robustness, as outlined in chapter 6.8, this fixed seed was changed to 222.

### Extra Step: Grouping of Probes

An additional step is included: The unsupervised grouping on probes to explore patterns and relationships at the probe/gene level. The results were 100 probe groups.

### 3.1 Supervised Classification and Feature Selection for Global Structure

To extract probe groups that give insight into the global structure, first the means of the 100 probe groups was calculated using the function `meanpergroup`. Then, the function `randomforest_fun` was applied using the 100 probe group means as predictors and the 8 cluster labels from step 2 as outcome variables. The seed was fixed to be 123. The feature importances were visualized to showcase the relative importance of each probe group.

When analyzing the robustness of our interpretation framework, as outlined in chapter 6.8, the fixed seed for the random forest analysis was changed to 222.

### 3.2 Supervised Classification and Feature Selection for Local Structure

To find features relevant to the local structure, first step 2 of the interpretation framework was re-run, but with a higher number of clusters, namely 28. For this purpose, the function `gmm_clustering` was used again. Then, the function `randomforest_fun` was run with the 100 probe groups as the predictors and the 28 cluster labels as the outcome variable. Then the difference in feature importance was calculated by subtracting the feature importance based on the 8 clusters from the feature importance based on the 28 clusters.

Bar plots were used to depict the differences in feature importances derived from the 8-cluster and 28-cluster models. This visual representation highlighted which probe groups' importance increased or decreased.

### 4. Visualization and Interpretation

The most important features from both the global and local structure analysis were visualized using the `visualize_points` function.

Lastly, to gain insights into the chosen probe groups, the gene symbols from each chosen probe group were extracted as a csv file to be uploaded to the GOrilla tool.