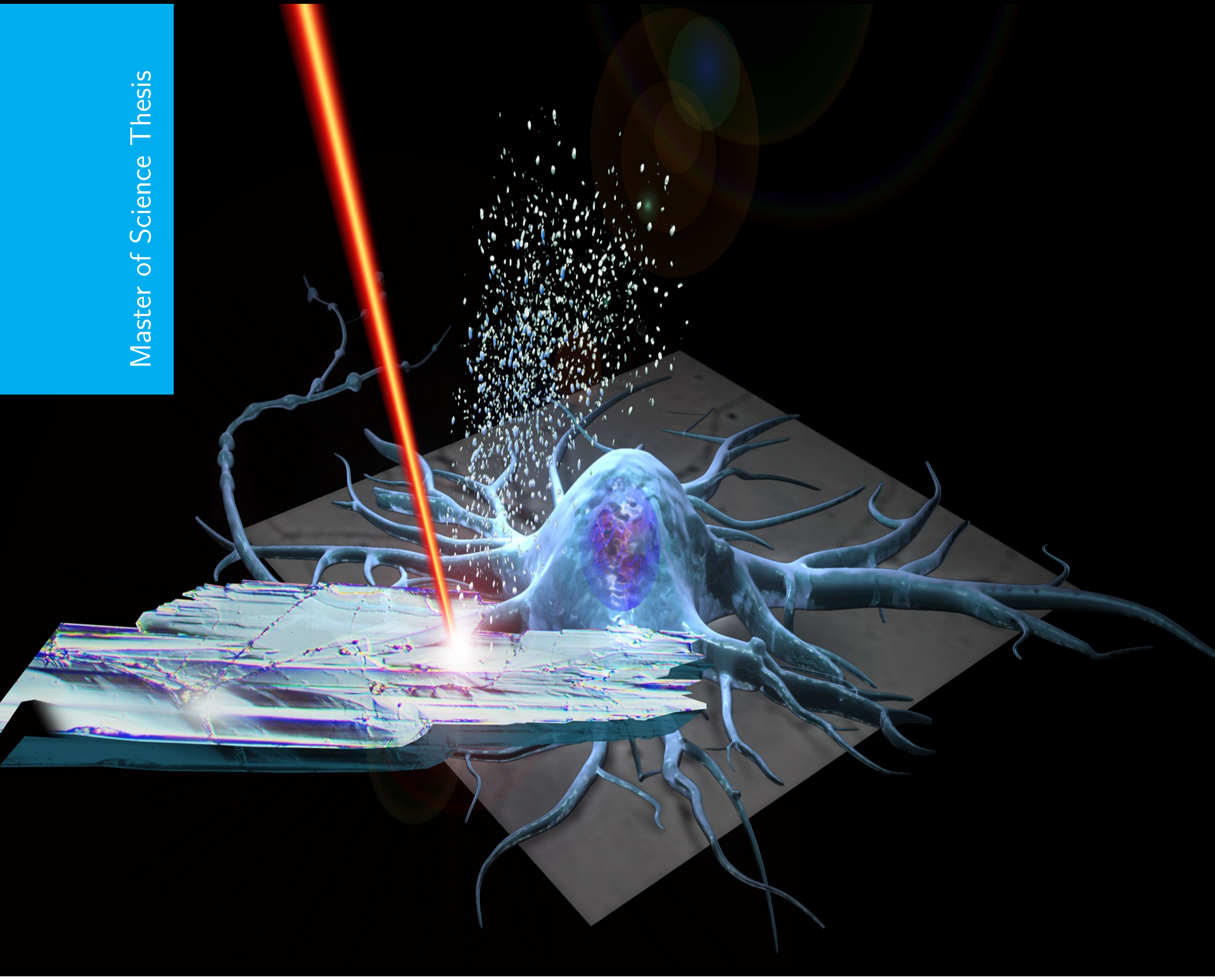
Data-Driven Soft Discriminant Maps

Class-aware Linear Feature Extraction in Imaging Mass Spectrometry

Thomas Cornelis Booij

Master of Science Thesis



Delft University of Technology

Delft Center for Systems and Control

Data-Driven Soft Discriminant Maps

Class-aware Linear Feature Extraction in Imaging Mass Spectrometry

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Thomas Cornelis Booij

January 29, 2021

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Copyright © Delft Center for Systems and Control (DCSC)
All rights reserved.

Delft University of Technology
Department of
Delft Center for Systems and Control (dcsc)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis entitled

Data-Driven Soft Discriminant Maps

by

Thomas Cornelis Booij

in partial fulfillment of the requirements for the degree of

Master of Science Systems and Control

Dated: January 29, 2021

Supervisor(s): ______________________

dr.ing. Raf Van de Plas

Reader(s): ______________________

prof.dr. Gleb Vdovin

____

dr. Matthias Alfeld

Abstract

Retrieving actionable information from large datasets is increasingly computationally expensive due to the current trend of ever-increasing dataset sizes. Reducing dataset sizes with dimensionality reduction techniques is often necessary for statistical analysis techniques, such as classification, to be computationally feasible. Most dimensionality reduction methods do not require any additional information to accomplish their task. However, datasets used for classification, for example, are accompanied by a set of class-labels as well. This extra information can improve dimensionality reduction techniques by explicitly preserving features that explain differences between classes.

A field where high-dimensional and large datasets are standard is Imaging Mass Spectrometry (IMS), a technique that simultaneously records the abundance and spatial location of molecules throughout biological tissue samples. Classification has been applied to IMS datasets for a wide range of scenarios, including the diagnosis of disease, distinguishing between tumour types for personalized treatment, and identifying biomarkers. A recently introduced dimensionality reduction method called Soft Discriminant Map (SDM), designed to incorporate class information and prevent overfitting when used on high-dimensional datasets, is a promising candidate to reduce the size and dimensionality of IMS datasets. However, SDM currently requires manual setting of a free parameter β that influences class separation in the newly constructed feature-space.

This thesis explores the use of SDM on IMS datasets in classification use cases and proposes a framework to set β in a data-driven way: Data-Driven Soft Discriminant Map (DD-SDM). Furthermore, the sensitivity of the classification performance to changes in β is examined. DD-SDM is compared to similar state-of-the-art dimensionality reduction methods in terms of classification performance. The performed experiments show that DD-SDM successfully finds a value for β where the classification performance is on par with, or in some scenarios better than, state-of-the-art dimensionality reduction methods while using fewer features. Setting β either too low or too high results in a suboptimal feature space and worsens classification performance. Golden section search, the search strategy used to find the optimal β in DD-SDM, succeeds in finding the optimal β in fewer iterations than more naive methods. With the use of an artificial dataset in combination with a novel evaluation metric, the Peak Conservation Score (PCS), the distinctive ability of DD-SDM to discard features that are common between classes and to actively select for discriminative features is demonstrated. The DD-SDM framework is furthermore applied to real-world IMS measurements of rat brain and mouse kidney tissue.

Table of Contents

List of Figures

List of Tables

Acknowledgements

This thesis has been quite the journey, academically but primarily personally and mentally speaking. In the past two years, I learned a lot about the ins and outs of tackling a project that is potentially unlimited in scope. This thesis taught me about dealing with rigorous scoping, insecurities, imposter-syndrome, and motivating myself to work without external pressure. Most of all, I learned about the vital importance of having friends around to get through the tough times. In this area, I would especially like to thank Marc, Mátě and Jeroen Swart. You guys helped to keep my sanity in times where I didn't see the finish line. My biggest thanks and love goes out to my family. I feel blessed by the overwhelming and unconditional love and support you always gave me, please know that I will never take this for granted.

Next, I want to thank my supervisor Raf Van de Plas for introducing me to the fascinating field of Imaging Mass Spectrometry. Thank you for being my mentor, teaching me what an oxford comma is, and for the valuable sparring sessions about life as a whole.

With this, I would also like to express my gratitude towards William Perry, Eric Skaar, Jeffrey Spraggins, and Richard Caprioli for providing the Imaging Mass Spectrometry datasets which were acquired at Vanderbilt University.

Many people have helped me in one way or another and deserve to be mentioned in this section. I would like to thank Lukasz, Rens, Freek, Jeroen Ubels, Jac, Sam, Karst, Tomas Uittenbooghaard, Thomas Janssen, Fien, Valerie and all other people that either helped me directly by providing feedback on my work or indirectly by inspiring me to keep on working towards graduation.

Delft, University of Technology
January 29, 2021

Thomas Cornelis Booij

Chapter 1

Introduction

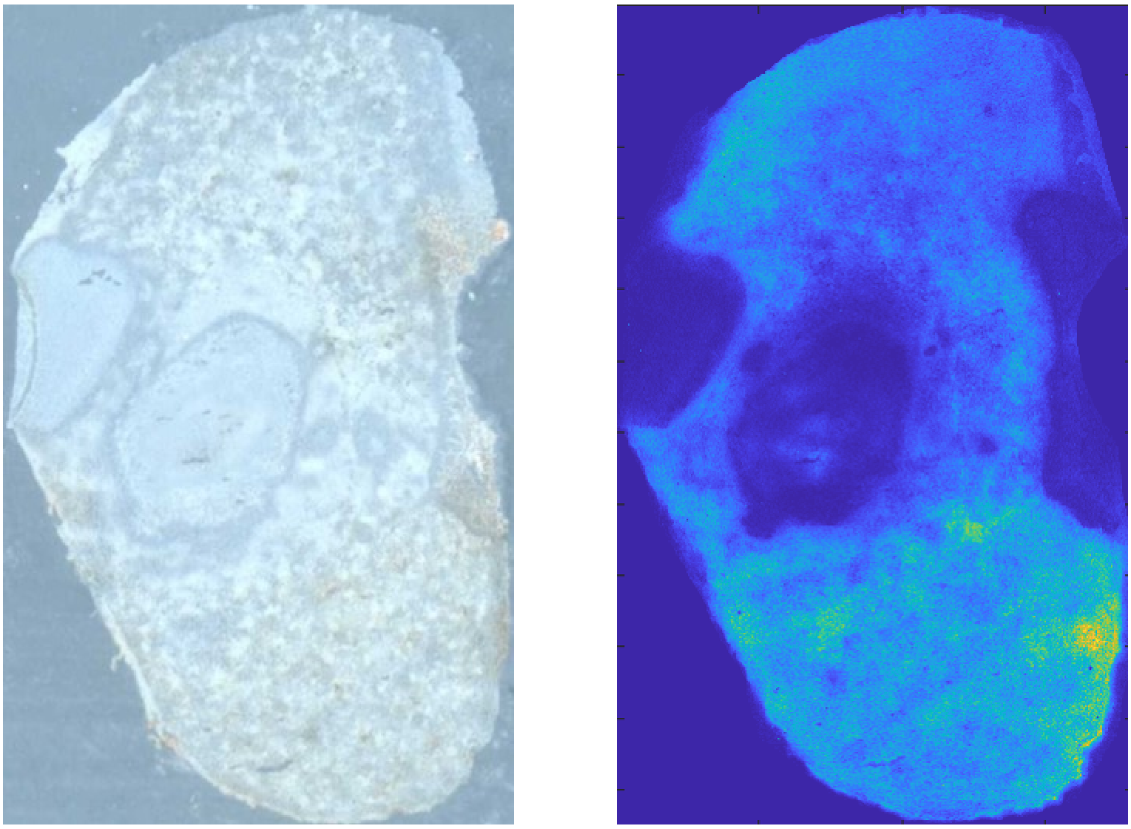
The amount of data gathered in the world is rapidly increasing. Many industries and academic fields collect enormous amounts of data in the hope of revealing insights about the world. New challenges accompany this trend. Although computational power has been improved over the years, turning big data into actionable information is still a non-trivial task. For analysis techniques to be feasible and effective, the dimensionality of these datasets often has to be reduced. The process of reducing the dimensionality of datasets while conserving the important information within is called Dimensionality Reduction (DR).

A field with an abundance of potentially information-rich data is Imaging Mass Spectrometry (IMS). This method enables the simultaneous recording of a wide range of ions with their respective locations in biological tissue. For every measured ion, an ion image can be constructed as seen in Figure 1-1-b. This visualization shows the distribution of an ion's abundance with a specific mass-over-charge value(m/z) across the tissue. An ion image gives additional information that is not present in the microscopy image shown in Figure 1-1-a.

Having the molecular contents of a tissue sample available enables research such as the molecular differentiation between common cancer types [1], detecting HER2-status, the presence of a gene that causes breast cancer [2], and finding biomarkers in tumours to measure the response to preliminary therapy in breast cancer [3]. Even outside the biological field, IMS is used to perform insightful analyses. For example, IMS has been successfully applied to predict gender, ethnicity, and age from fingerprints, advancing the tools available in forensics.

Since an IMS dataset consists of up to thousands of m/z measurements per pixel, dimensionality reduction methods are often applied to eliminate redundant and irrelevant features. An often-used method for DR with IMS data is Principal Component Analysis (PCA) [4–7]. PCA is an unsupervised DR technique that requires no additional information to perform its task and aims to extract a reduced set of features from the data that maximizes the variance present in the newly constructed features[8].

When prior information is available in the form of class information about data points in a dataset, classification can be performed to extract actionable information from the dataset. Classification is a type of analysis that aims to predict the class of data points based on a training dataset of data points for which the class is known. For example,



(a) Microscopy image

(b) A sample ion image with m/z 920

Figure 1-1: Two visualizations of a tissue sample from a mouse kidney. Using IMS, the distribution of abundance of ions within a tissue can be visualized in an ion image.

predicting which pixels in a tissue sample are part of tumour-tissue, based on pixels that we know describe the molecular spectrum of a tumour.

Often, class information first gets used in the classification step performed after dimensionality reduction. In a recent study Liu and Gillies [9] proposed a novel DR method that incorporates this class information already in the DR phase, and called the method Soft Discriminant Map (SDM). Using this information, SDM aims to extract features that represent class differences instead of characteristics of the entire dataset like PCA. This way, classification performance could be improved, and more useful features could be constructed for maximizing classification performance. SDM is based on a well-known technique called Linear Discriminant Analysis (LDA), which is also occasionally used for dimensionality reduction while employing class information[10]. However, LDA is known to easily suffer overfitting[9] when applied to high dimensional datasets. Overfitting is a phenomenon that causes the extracted features to capture the specific noise patterns in the training data instead of the general biological patterns we would like to conserve. SDM introduces a parameter β that affects the amount of overfitting. By overcoming the major shortcoming of LDA, SDM exhibits beneficial behaviour of both LDA and PCA (*i.e.* SDM uses class information to extract informative features while mitigating the risk of overfitting when applied to high dimensional training datasets).

Research objectives Leading up to this thesis, SDM has a parameter β that has to be set manually. This immediately raises a few questions: To what value should this β be set to achieve maximal classification performance? How sensitive is this performance to

β? Does using SDM with an optimal β result in a better performance than using other dimensionality reduction methods such as PCA and LDA?

These objectives can be stated as four main research questions which will be addressed in this thesis:

1. When applied to a real-world IMS dataset, how does the classification performance after applying Soft Discriminant Map (SDM) compare to after applying Principal Component Analysis (PCA) or LDA?

2. What is the sensitivity of the classification performance to the value of the parameter β in SDM?

3. How can β be set in an automated, data-driven way that maximizes classification performance for a given dataset? How efficient is golden section search compared to grid search to find the optimal β?

4. Is SDM able to conserve features that are class-specific and discard features that have similar values between classes?

Thesis outline This thesis will start by explaining the fundamentals needed to understand the remainder of the thesis. Since SDM will be evaluated on real-world datasets originating from IMS measurements, the first section will cover the characteristics of IMS-data and specifics about obtaining these datasets.

Next, the steps in a typical classification-pipeline are discussed. In the last fundamentals section, the use of DR is motivated, the general approaches of DR are described, and the specific methods PCA, LDA, and SDM are explained in detail.

After the fundamentals, the methods chapter will explain our proposed extension to SDM which we have named Data-Driven Soft Discriminant Map (DD-SDM). This chapter will also state the evaluation method used to compare DD-SDM to PCA and LDA.

In the experiments chapter, the datasets used will be described, including the construction of an artificial dataset and two real-world IMS datasets. The artificial dataset is constructed in a way to represent a simple IMS dataset. In the second half of this chapter, the research questions are stated again, along with the experiments' descriptions that aim to answer these questions.

The experiments' results are then presented and discussed. The thesis ends with stating the main conclusions and making recommendations for further research.

Chapter 2

Fundamentals

2-1 Imaging mass spectrometry

Imaging Mass Spectrometry (IMS) is a technique capable of analyzing the molecular contents of tissue samples. By performing mass spectrometry experiments on a predefined grid's locations on the tissue, molecular and spatial information is simultaneously acquired.

Combining these two information sources enables IMS to visualize the spatial distribution of a wide range of molecules, ranging from small molecules such as cell metabolites and lipids to bigger molecules such as peptides and proteins. Up to thousands of distinct molecular masses can be measured at the same time. The visualization of the distribution of a particular ion within a tissue sample is called an ion image and is shown in Figure 2-3b.

IMS links the heavily studied field of pathology and the deep molecular mass spectrometric analysis to the analysis of tissues [11]. IMS has become famous for analyzing biological tissues without prior labelling target analytes by combining spatial and spectral information. Other techniques as immunohistochemistry do require prior labelling. This fact makes IMS especially suitable for exploratory research.

This chapter gives an overview of the different steps in the IMS process. Different variants of IMS are covered in section 2-1-1. An explanation of the characteristics and interpretation of the dataset that results from the IMS experiments are stated in section 2-1-2 and some applications of IMS are given in section 2-1-3.

2-1-1 Process of IMS

This section will outline the different steps in the IMS pipeline. A rough overview of the workflow of Matrix-Assisted Laser Desorption Ionization (MALDI) IMS is shown in Figure 2-1. First, the sample is usually prepared for analysis by slicing a thin slice of tissue and placing it on a plate. An ionization step is then performed to release the ions from the tissue. An electromagnetic field transports the ions into a mass analyzer. This device measures the intensity of many distinct mass-over-charge(m/z) values of the released ions. Many technical choices are to be made in each step of the process that all affect the resulting dataset's quality. This section will highlight some of these considerations and discusses common techniques used in each step.

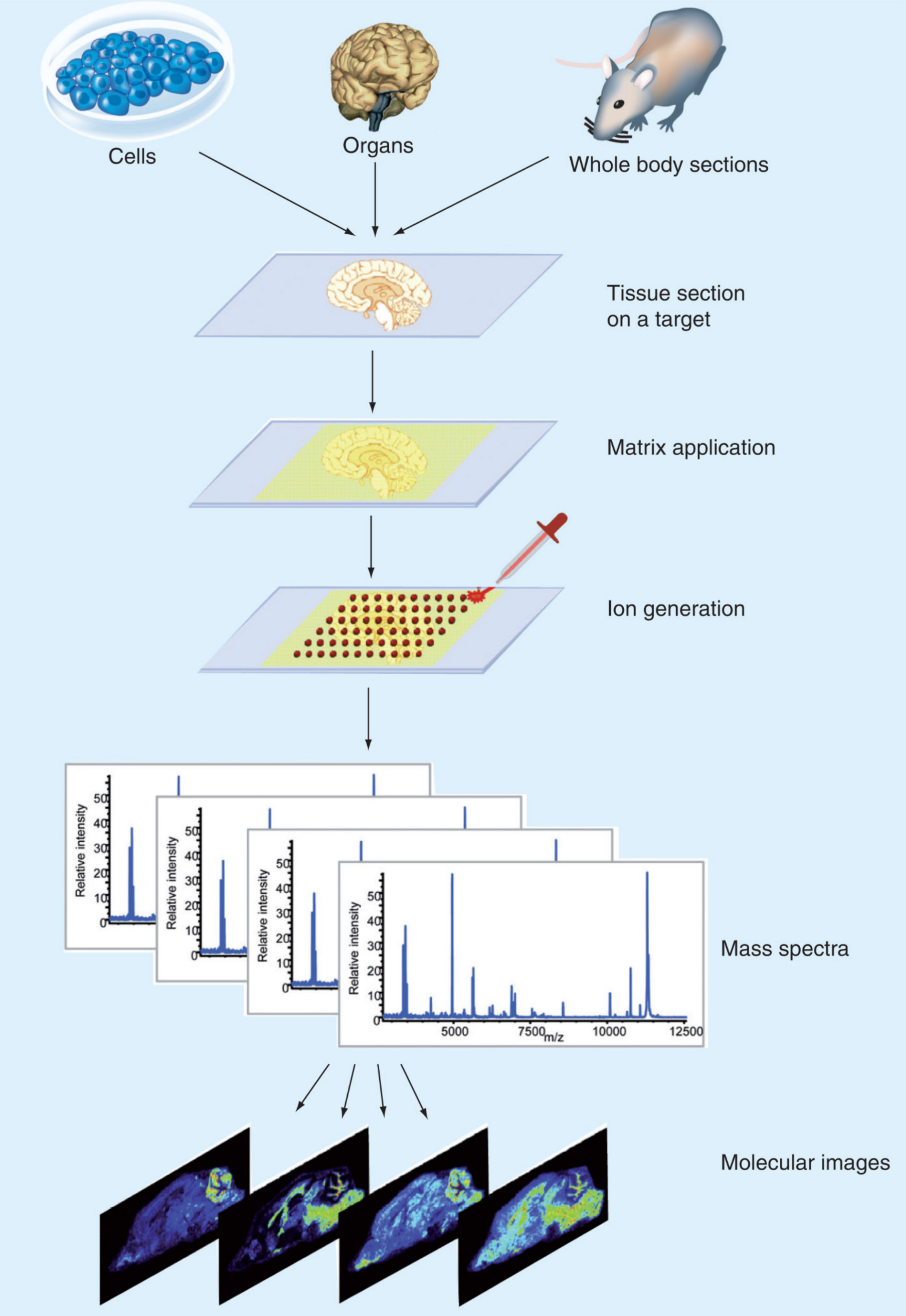


Figure 2-1: Schematic representation of the process of MALDI IMS The tissue is placed onto a glass slide. A matrix is applied to extract ions from the tissue. A mass analyzer records the abundances over a range of m/z-values. These abundances can be plotted for a pixel in a mass spectrum. Molecular images can be constructed to show the distribution of a specific m/z value throughout the tissue. Source of image: Boggio et al. [12]

2-1-1-1 Sample Preparation

After a researcher has selected tissue to investigate its molecular contents, the sample needs to be prepared for IMS measurement. It is essential to have a standardized sample handling and sample preparation workflow to achieve good spectral quality and reproducibility of the measurements. Small deviations in some parts of the preparation process could result in a large variation in obtained measurements. The goal of a successful sample preparation protocol is to maintain the true spatial distribution and abundance of molecules in the tissue by minimizing degradation or chemical modification. Even though the three major ionization methods described in the upcoming section are very different, they share some of the same steps in the sample preparation process [13].

The preparation pipeline can be divided into four steps: collection, sample processing, post-sectioning processing, and ionization-aiding treatments. The following sections describe these steps and are based on the work of Goodwin [13].

Collection In literature, many classes of samples have been analyzed ranging from unicellular algae [14] and surgically dissected organs [15] to whole animals [16]. For all classes, it is vital to perform a rapid collection of the tissue to prevent degradation and delocalization of the target compounds. When animals are studied, ethical guidelines have to be followed.

After the tissue has been collected, it has to be stabilized to prevent biological processes from degrading the specimen. Most often, this stabilization is performed by snap-freezing the tissue to temperatures below $-70°$C.

Sample Processing Since most IMS techniques are best performed on tissue that is flat, thin slices (typically 5-10 μm thick) are cut using a cryostat microtome after tissue collection. The slices will be mounted on a target plate, either with adhesive tape applied before slicing or using thaw-mounting, where the frozen tissue-slice is transferred by carefully placing a warmer plate on the slice. The latter method is not advised when multiple samples are analyzed in parallel since the difference in exposure-time to the warm plate could result in variation between samples.

Post-sectioning processing After the slice has been mounted on a plate, the tissue can be stored or analyzed immediately. Several processing steps are available to significantly improve the measurement results and ensure the stability of ion abundances during the experiment. Drying the tissue to mitigate sample instability after getting the sample out of the freezer can be performed by freeze-drying [17], dehydration by solvent washing [18], and air-drying under a stream of nitrogen [19]. Next to dehydration, another processing step is to wash the tissue with organic solvents to remove ionization-suppressing small molecules and lipids. However, when small molecules are of interest in the analysis, washing should be avoided since these small molecules could be removed by the washing process.

Another post-sectioning processing option is on-sample enzymatic digestion, which will divide heavy protein molecules that are normally out of the range of the mass analyzer into smaller peptide fragments. This process can be performed by *e.g.* spraying a homogeneous coating [20] or discrete spots [21] of specific enzymes.

An optical image is often acquired from the same tissue to compare IMS results to traditional histological knowledge. The tissue is stained with a staining agent that does not interfere with the Mass Spectrometry (MS) measurement to get a representative image. Specific protocols have to be followed to not alter the target analyte abundance [22].

IMS measurements capture the relative abundance of ions, allowing the visualization of analytes in the tissue. In some cases, it is even possible to quantify the measurements. In a study by Franck et al. [23], they used an analyte with a known concentration that they added to the target plate as a benchmark to determine the abundance of the drug tiotropium in rat lung tissue sections.

Ionization-aiding treatments In Matrix-Assisted Laser Desorption Ionization Imaging Mass Spectrometry (MALDI[1] IMS [2]) a substance, called a chemical matrix, is applied to the tissue. The matrix aims to extract the maximum amount of analyte from the tissue to the surface, after which the matrix crystallizes. The quality of the IMS measurements depends heavily on the selection of the most effective ionization matrix for the analysis task but also on the process of applying the matrix to the tissue. Matrix application is an important step in the preparation pipeline since differences in matrix thickness, application rate, and drying times have a large impact on the generation of high-quality, reproducible data. The choice of matrix is largely determined by the mass-range of interest and the wavelength of the laser used to extract the ions. An extensive review of matrix selection for IMS has been published by MacAleese, Stauber, and Heeren [26].

2-1-1-2 Ionization

As discussed in the previous section, the matrix is applied to the tissue to extract ions in MALDI IMS. Over the years, many compounds have been used as matrix, but a small subset is primarily used in the literature. Three commonly used compounds are α-Cyano-4-hydroxycinnamic acid (HCCA), sinapinic (or sinapic) acid (SA), and 2,5- dihydroxybenzoic acid (DHB) [27].

After the matrix has been applied evenly, a laser beam, *e.g.* with a footprint of 10 μm, is fired at the tissue. The laser will eject ions from the matrix, controlled by the pulse length of the laser [28]. The goal of the ionization step is to be able to accelerate the particles with an electromagnetic field to measure their mass using a mass analyzer.

Ionization technique selection Next to MALDI IMS, several other ionization variants are present in the field. The choice of a particular ionization method is heavily determined by the requirements of the study. Buchberger et al. [29] state that the different ionization methods have there own characteristics, ranging from the m/z ranges that can be measured to the spatial resolution possible. Cole and Clench [30] give a comparison of the commercially available IMS methods while commenting on the advances to be made in the field for effective clinical application of IMS to identify and diagnose cancer.

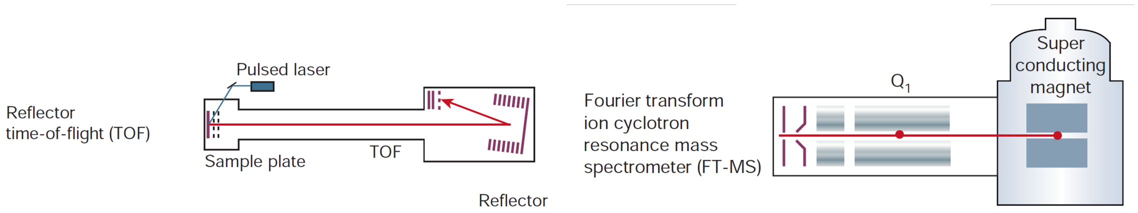
A different ionization method is Secondary Ion Mass Spectrometry (SIMS). SIMS fires an ion-beam at the substrate to achieve ionization, having the advantage of obtaining spatial resolutions down to a sub-cellular level ($\sim$500nm). Because of this resolution, SIMS is able to characterize individual organelles. The characterization is, however, limited to small molecules and metabolites and does not generally detect heavier molecules such as peptides, proteins, and most intact lipids [12].

In Figure 2-1, the process of MALDI IMS is visualized. In the step from ion generation to the acquisition of mass spectra (covered in section 2-1-2), an important device is omitted which will determine the mass of the measured ions; the mass analyzer.

[1] The MALDI technique for conventional mass spectrometry was invented in 1987 by M. Karas and F. Hillenkamp [24].

[2] The MALDI IMS process found broad introduction through Caprioli *et al.* [25] in 1997

2-1-1-3 Mass analyzer



(a) Schematic representation of the reflector time-of-flight mass analyzer

(b) Schematic representation of the Fourier transform ion cyclotron resonance mass analyzer

Figure 2-2: Schematic representations of two mass analyzers

After ejection of the ions from the matrix *e.g.* by using a laser pulse, a mass analyzer determines their mass-over-charge values. The key parameters of mass analyzers are sensitivity (the ability to detect small ion-abundances), resolution, and mass accuracy [31]. There exist multiple types of mass analyzers, two of which will be considered in the following sections.

Time-of-flight One common mass analyzer type uses the principle of Time-of-Flight (TOF) [32] (Figure 2-2a). With this technique, using an electromagnetic field with known potential, ions are accelerated through a field-free region where the time is measured to cross this region and hit an ion detector.

To calculate the mass-over-charge value from the time-of-flight, the energy equilibrium of the ion is taken into account:

$$
\begin{aligned}
E_p &= E_k && \text{(2-1)}\\
qU &= \frac{1}{2}mv^2 && \text{(2-2)}\\
qU &= \frac{1}{2}m\left(\frac{d}{t}\right)^2 && \text{(2-3)}\\
\frac{m}{q} &= \frac{2U}{\left(\frac{d}{t}\right)^2} && \text{(2-4)}\\
\frac{m}{q} &= \frac{2U}{d^2}t^2 && \text{(2-5)}
\end{aligned}
$$

Where U is the electric potential of the electro-magnetic field in volts, $\frac{m}{q}$ is the mass-over-charge ratio, d is the distance travelled by the ion, and t is the time measured for the ion to travel this distance. The mass-over-charge value is quadratically dependent on the time of flight. It is important to note that these calculations provide a simplified evaluation. In practice other factors such as initial velocity of the ions are taken into account [32].

Fourier transform ion cyclotron resonance mass spectrometer Another mass analyzer type is based on the Fourier transform ion cyclotron resonance (FT-ICR) technique. A schematic overview of this mass analyzer type is shown in Figure 2-2. The ejected ions are trapped inside an electromagnetic field, which forces the ions to spin in a circular orbit. With a simplified relationship, the mass-over-charge ratio can be obtained from these angular velocities:

$$\omega = B_0 \frac{q}{m} \tag{2-6}$$

Where ω is the cyclotron frequency of the ion, B_0 is the strength of the electro-magnetic field, and q/m is the inverse mass-over-charge ratio of the ion[33]. The ions will rotate with a radius that among other things depend on the mass of the ion:

$$r = \frac{\sqrt{2mkT}}{qB_0}, \tag{2-7}$$

where q is the ion's charge, m is the ion's mass, k is the Boltzman constant, T is the environmental temperature, and B_0 is the strength of the applied electromagnetic field. The radii of the ions can be controlled by changing the strength of the electromagnetic field. The detector plates in the ion trap measure the current induced by the orbiting ions. By gradually increasing the excitation field to increase the kinetic energy of the ions, a sweep is made across all orbital frequencies. The time-domain measurements from the detected current by the detector plates are Fourier transformed to retrieve the angular velocities of the trapped ions. These velocities can be mapped back to m/z values by using Equation 2-6 [33].

2-1-1-4 Further literature

The approach taken in every step of the MALDI IMS process has an influence on the quality of the data gathered. Kriegsmann and Casadonte [34] sum up a handful of reviews that describe the experimental considerations of MALDI IMS. In particular regarding: sample collection, sample preparation, matrix deposition, ionization process, instrumentation and data-processing [35–44].

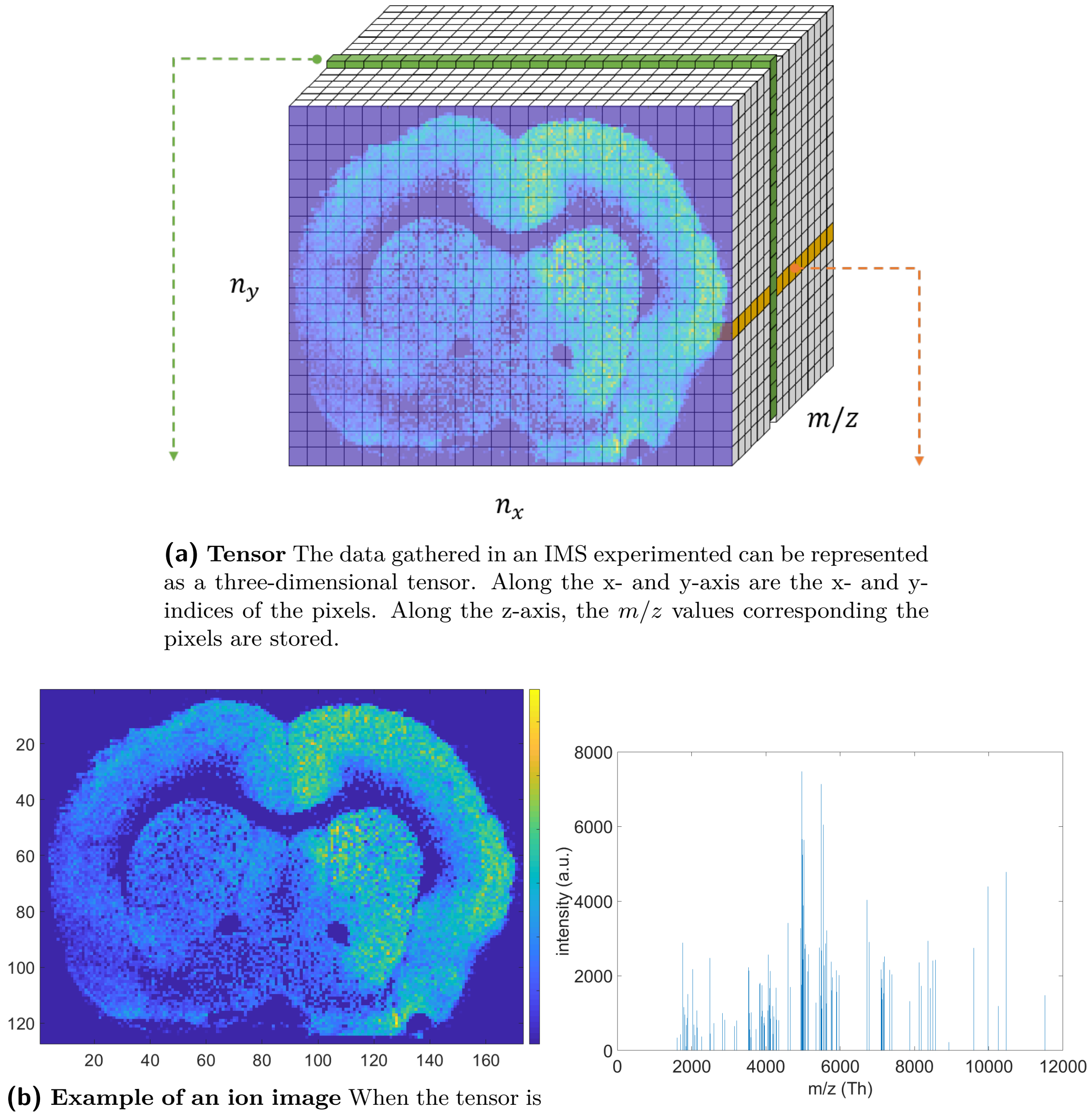
2-1-2 Data structure

All local MS measurements on the predefined grid are combined to create an IMS dataset. This dataset can be considered as a three-dimensional tensor, as shown in Figure 2-3a. Each measurement, taken at position (x, y), produces M values, one for each measured mass-over-charge value. The number of values M depends on the considered mass range and the mass analyzer's spectral resolution.

Mass spectrum The measurements acquired by the mass analyzer are mass-over-charge (m/z) values. The unit of these values is Thompson (Th), defined as Da/e or u/e. Da stands for Dalton and is interchangeably used with the symbol u to denote the unified atomic mass unit. One Da or u is defined to be 1/12 of the mass of an unbound neutral atom of carbon-12. The elementary charge e is defined as the electric charge carried by a single proton.

The recorded intensities for all m/z values are usually visualized in a mass spectrum, as shown in Figure 2-3c. The intensities are in arbitrary units, making the relative comparison of the intensity between different m/z values possible. A mass spectrum can be produced for every pixel, showing the intensity of the m/z values at that pixel's tissue location.

Ion image When the data tensor is sliced along the (x,y)-plane, all intensity values of the matrix correspond to the same m/z value. A false-colour image, referred to as an ion image, can be created by colouring the pixels according to their intensity. An example of an ion-image is shown in Figure 2-3b.



(a) Tensor The data gathered in an IMS experimented can be represented as a three-dimensional tensor. Along the x- and y-axis are the x- and y-indices of the pixels. Along the z-axis, the m/z values corresponding the pixels are stored.

(b) Example of an ion image When the tensor is sliced in the x-y plane, a false-colour image, referred to as an ion image, corresponding to a particular m/z value can be constructed. A pixel is coloured according to the intensity measured of that particular m/z value.

(c) Example mass spectrum The intensities of all m/z values belonging to a pixel are visualized using a mass spectrum. The x-axis represents the range of m/z values. The y-axis indicates the corresponding measured intensity in arbitrary units.

Figure 2-3: IMS Data structure When all pixels of a particular m/z value are considered of the three-dimensional tensor shown in Figure (a), an ion image can be constructed as shown in Figure (b). The m/z values belonging to a pixel are visualized in a mass spectrum, as shown in Figure (c).

Tensor unfolding For conventional processing of the dataset, the three-dimensional tensor $D_{Tensor} \in \mathbb{R}^{P \times Q \times M}$ with P pixels in the x-direction, Q pixels in the y-direction, and M m/z bins, is often unfolded into a matrix $D_{Matrix} \in \mathbb{R}^{M \times P \cdot Q}$ as shown in Figure 2-4. The rows and columns of D_{Matrix} denote the pixels and the m/z values, respectively. As seen in Figure 2-3b, the borders of the image are not covered with brain tissue. Since these pixels contain no information, their indices are stored elsewhere, and the corresponding rows are often removed from D_{matrix}. The indices can later be used to reconstruct the tensor after analysis on the 2D dataset.

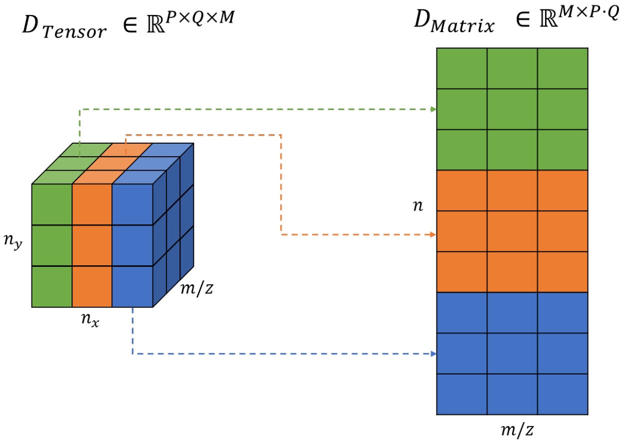

$D_{Tensor} \in \mathbb{R}^{P \times Q \times M}$
$D_{Matrix} \in \mathbb{R}^{M \times P \cdot Q}$
n_y
n_x
m/z
n
m/z

Figure 2-4: Tensor unfolding The three-dimensional tensor $D_{Tensor} \in \mathbb{R}^{P \times Q \times M}$ with P pixels in the x-direction, Q pixels in the y-direction and M m/z bins is unfolded into a matrix $D_{Matrix} \in \mathbb{R}^{M \times P \cdot Q}$.

With increasing resolution in the spatial and spectral domain, IMS datasets tend to become large. When a small tissue of 1 × 1 cm is measured with a pixel size of 10 μm, the dataset can consist of up to 10^8 pixels. The number of analyzed peaks per pixel could become arbitrary large depending on the mass analyzer's spectral range and resolution, ranging from hundreds to even thousands of distinct m/z values in concrete experiments. This combination results in large datasets that are impractical for humans to interpret manually. Hence, sensible computational methods are needed to process, analyze and interpret the data successfully.

2-1-3 Applications of IMS

The combination of spatial and molecular information enables IMS to be applied in a wide range of fields. This section will denote some of these applications. The described applications are categorized as spatial exploration, biomarker discovery and pattern recognition.

2-1-3-1 Spatial exploration

IMS is a label-free method, enabling exploratory research of molecules without prior labelling of targets. As discussed in the previous section, the dataset consists of an ion image for each measured m/z bin. These ion images are a great tool for visualizing the spatial distribution when the ion of interest is known in advance. However, these ion images are not so informative when no *a priori* knowledge is available. Since the number of ion images per experiment can run into the thousands, human interpretation of all ion images is unfeasible.

For this reason, multivariate analysis methods such as Principal Component Analysis (PCA) are used in literature to gain insights into the correlation between m/z bins. PCA is extensively described in Section 2-3-3-1. PCA is applied to the unfolded tensor, after which the mask indices are used to reconstruct the image.

IMS measurements contain a vast amount of molecular information. However, the anatomical interpretation is nontrivial and still has to be done by experts. An attempt to automate this interpretation is made by Verbeeck et al. [45] by linking the IMS measurements to atlas databases.

2-1-3-2 Biomarker discovery

By comparing the mass spectra of healthy and diseased cells, a profile for the characteristic differences in m/z values can be made to gain an insight into which molecules are present or absent in the cells affected by the investigated disease. These profiles are called biomarkers. Biomarkers can be used to gain a deeper understanding of a disease's molecular mechanics, help diagnose a patient, and provide the information necessary to improve personalized therapies.

Biomarker discovery studies Biomarker discovery studies often apply IMS on Tissue Microarrays (TMA). With the TMA technique, a hollow needle is used to obtain small tissue samples from a tissue of interest, such as a biopsy or a tumour. Mascini et al. [46] performed such an experiment. The obtained tissue cores are placed on a plate in an array-like fashion, as seen in Figure 2-5-b. An IMS experiment can be performed for dozens of patients at the same time using this technique. This approach enables the comparison of a vast number of different tissue samples simultaneously. All samples undergo the same preparation steps, increasing the comparability of the samples. An example ion image is shown in Figure 2-5-c, showing the intensity of m/z 1105.6 of all tissue cores at once. Mascini et al. [46] analyzed the resulting mass spectra (Figure 2-5a) to obtain characteristic biomarkers.

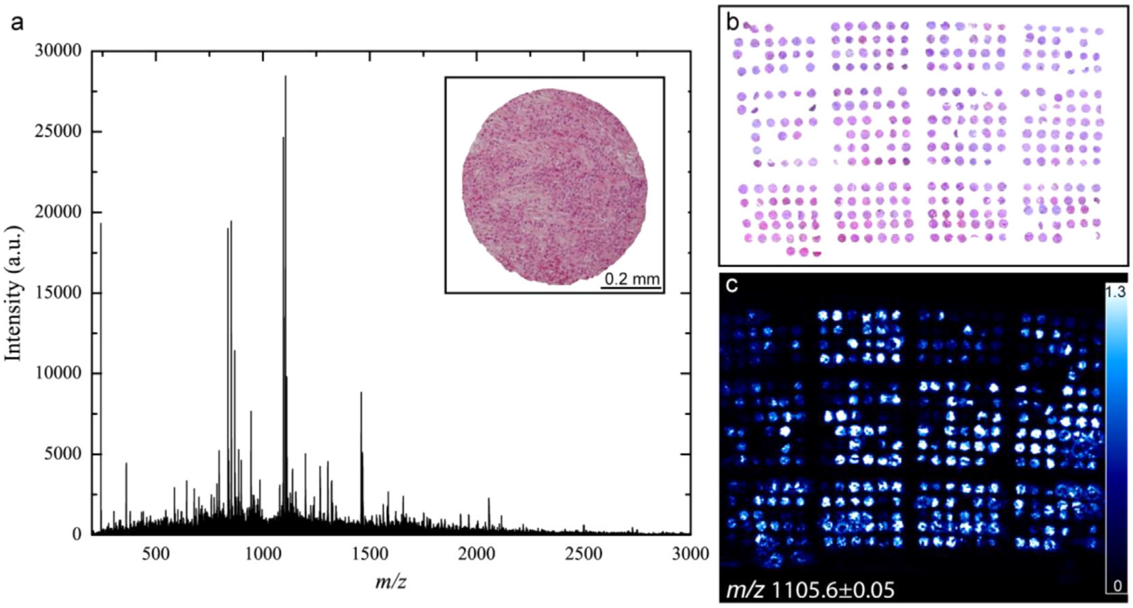


Figure 2-5: Example experiment for biomarker discovery using a TMA (a) Mass spectrum from a tissue core. In the top right a hematoxylin and eosin (H&E) stained tissue core with 80% tumor cells is shown. (b) H&E stained TMA of 120 patients. (c) The ion image corresponding to m/z 1105.6. Source of image: Mascini et al. [46]

According to Aichler and Walch [11], one of the first biomarker discovery studies using MALDI Imaging mass spectrometry was carried out in a patient group in which proteomic markers in tumor tissue were identified that were associated with the response to a preliminary therapy in breast cancer [3]. Another early study from Kurabe et al. [47] found a new biomarker called phosphatidylcholin (16:0/16:1) for the diagnosis of colorectal cancer. They successfully applied MALDI-IMS to analyze the different mass spectra obtained to find this biomarker.

2-1-3-3 Pattern Recognition

With the use of IMS data, algorithms can be trained to predict the class of mass-spectra. For example, Meding et al. [1] performed classification of six common cancer types based on proteomic profiling using MALDI IMS. Balluff et al. [2] performed classification on IMS datasets to predict HER2-status, the presence of a gene that causes breast cancer. They added valuable knowledge in oncology by proving that different cancer type tumours have overlapping molecular profiles. In a later work Balluff et al. [4] used IMS profiling to identify clonal and phenotypic heterogeneity within a tumor, gaining deeper insights into the biological processes of cancer. These findings could finally lead to new targeted therapies for the treatment of cancer.

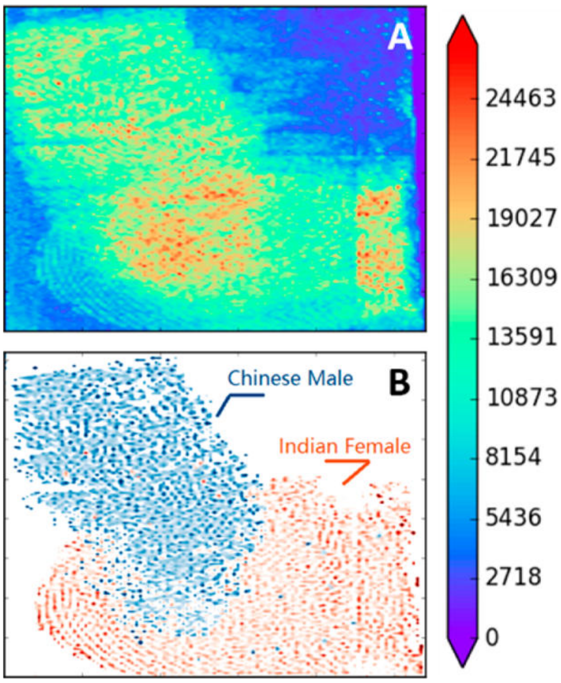


Figure 2-6: Recognition of overlapping fingerprints (a) Ion image at $m/z = 253$ (b) The classification result of the trained classifier. The pixels predicted to be of a Chinese male and Indian female are shown in blue and red respectively. Source of image: Zhou and Zare [48]

IMS is also used in non-biological fields, such as forensics. For example, Zhou and Zare [48] used IMS to predict gender, ethnicity and age from fingerprints. Figure 2-6 shows the result of an experiment where two overlapping fingerprints are analyzed. The classification model (classification will be broadly discussed in Section 2-2) correctly predicts that the fingerprints belong to a Chinese male and an Indian female. These predictions could help forensic professionals get more information about a suspect based on their fingerprints found at the crime scene, showing that IMS has a wide range of application possibilities.

2-2 Classification

As stated in Chapter 2-1, there are many applications of IMS. One such application is to learn about cancer tumours by studying the molecular profiles of different cancer types. Meding et al. [1] used IMS data to study the molecular content of six different tumour types. These learnings can then be applied to help diagnose tissue in one of the categories. The analysis often makes use of a technique called classification. In this section, the theoretical background is presented to understand how such an analysis is performed.

First, mathematical definitions are presented that will be used in the remainder of this thesis. After that, the different steps of a typical classification pipeline are discussed: reducing the size of the original dataset, training a classification algorithm, validation of the classification performance and the application of the algorithm to make predictions about new data.

2-2-1 Definitions

Throughout this thesis, several technical terms are used consistently. In this section, we will define some of these terms. When measurements are taken, *e.g.* from an IMS experiment, they are stored in a dataset D_{full} which could be represented as a table shown in Figure 2-7. For example, with IMS, a mass spectrum is recorded and stored in a row of the table for every pixel. The set of measurements corresponding to a pixel will be referred to as an observation or a data point. Each measurement of an observation is called a feature. All feature values belonging to an observation are stored in a feature vector $\boldsymbol{x} = [x_1, x_2, x_3, x_{...}, x_M]$. The jth feature value of the nth observation is referred to as $x_{n,j}$. The total number of features recorded per observation is referred to as the dimensionality M of a dataset. The total number of observations is denoted with N.

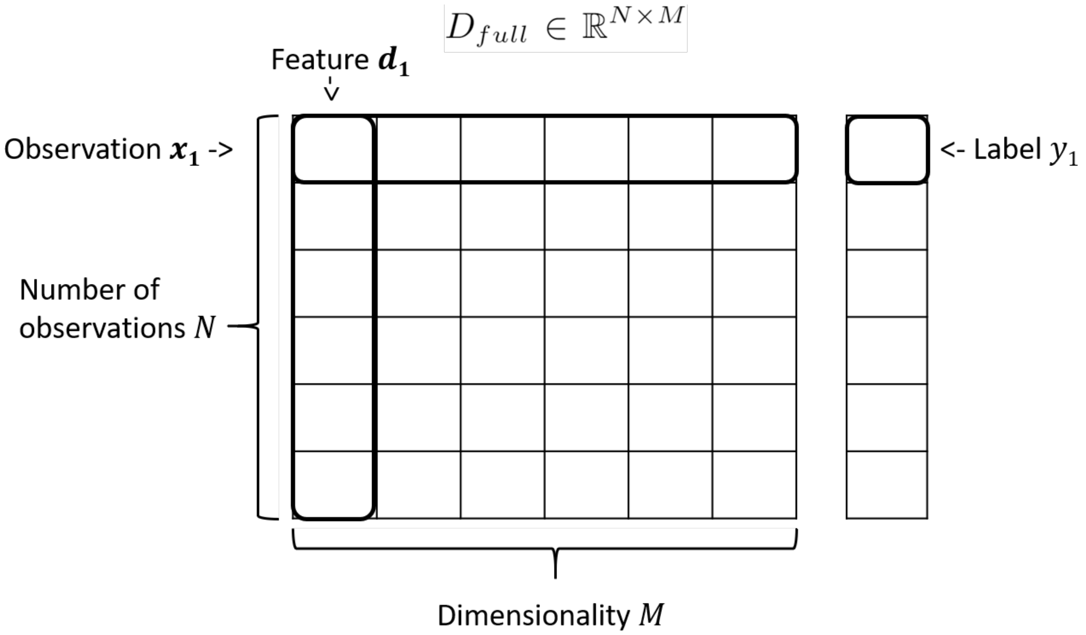

$D_{full} \in \mathbb{R}^{N \times M}$
Feature $\boldsymbol{d_1}$
Observation $\boldsymbol{x_1}$ ->
<- Label y_1
Number of observations N
Dimensionality M

Figure 2-7: Schematic representation of a dataset Data is typically stored in a table. Each row is called an observation or data point described by a feature vector $\boldsymbol{x}$. Each observation also has a corresponding label y that indicates to which class the observation belongs. The number of features is referred to as the dimensionality M, and the number of observations is denoted with N.

In classification, each observation has a corresponding class label, indicating to which class it belongs. For example, when an IMS dataset is analyzed, a researcher could assign sections of the examined tissue as healthy or diseased. Using the tumour example again, the label indicates to which cancer type a mass spectrum belongs. These different groups are referred to as *classes.* The label y is stored for each observation in addition to the feature vector $\boldsymbol{x}$ to indicate to which class each pixel belongs.

2-2-2 Dimensionality reduction

When data has been acquired and stored in a dataset, not all features are informative for determining the class of an observation. Especially when handling big datasets with a high number of features, it becomes increasingly essential to only analyze informative features for many reasons.

Since this procedure called Dimensionality Reduction (DR) will be the main focus of this thesis, it will be explored in full in section 2-3. For the sake of the current section, we will, for now, assume that a method exists that reduces the full dataset $D_{full} \in \mathbb{R}^{N \times M}$ to a reduced dataset $D_{red} \in \mathbb{R}^{N \times K}$ with $K < M$.

2-2-3 Classification

When we consider the dataset from Figure 2-7, each observation corresponds to a class. When new observations are made, the classification goal is to assign a class to these new, unlabeled observations. In the example of IMS, this means that the trained classifier should predict which cancer type an unlabelled mass spectrum belongs to. In other words: the goal of classification is to find a model called a *classifier* that has a feature vector $\boldsymbol{x}$ as an input and predicts a class-label as an output.

In the remainder of this section, the several steps involved with classification are discussed: building the classifier model with training data, validating the classifier's performance, and using the classifier to predict new samples.

2-2-3-1 Training

To make accurate predictions of the class of unlabelled observations, the observations of which the label is known are analyzed to find patterns that explain how the features determine the class. There is assumed to be a function

$$f(\boldsymbol{x}) = y \tag{2-8}$$

that describes the mapping from feature vector to the class label.

This function's learning is done in the training-phase of classification by studying observations for which the corresponding class-label is known. This set of observations will be referred to as the *training dataset.* Since the training data *supervises* the building of the model, classification is defined to be in the field of supervised learning. Many approaches recognize patterns in labelled data called *classifiers.* As an example, the following section describes one such classifier.

Example classifier: Gaussian Naïve Bayes Assume that we have a two-dimensional two-class dataset as shown in Figure 2-8. The blue crosses and the red stars correspond to class one (ω_1) and class two (ω_2) respectively. The shown training dataset consists of a number of observations $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_N\}$ with corresponding class labels $\{y_1, y_2, y_3, \ldots, y_N,\} \in \{\omega_1, \omega_2\}$. On the left in Figure 2-8, the two elements of the feature vectors are plotted against each other.

The classifier called Gaussian Naïve Bays will start by assuming that the two classes are generated by a Gaussian distribution with different mean vectors $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and diagonal covariance matrices Σ_1 and $\Sigma_2 \in \mathcal{R}^{2\times 2}$. The parameters of these distributions can now be estimated based on the values of these feature vectors.

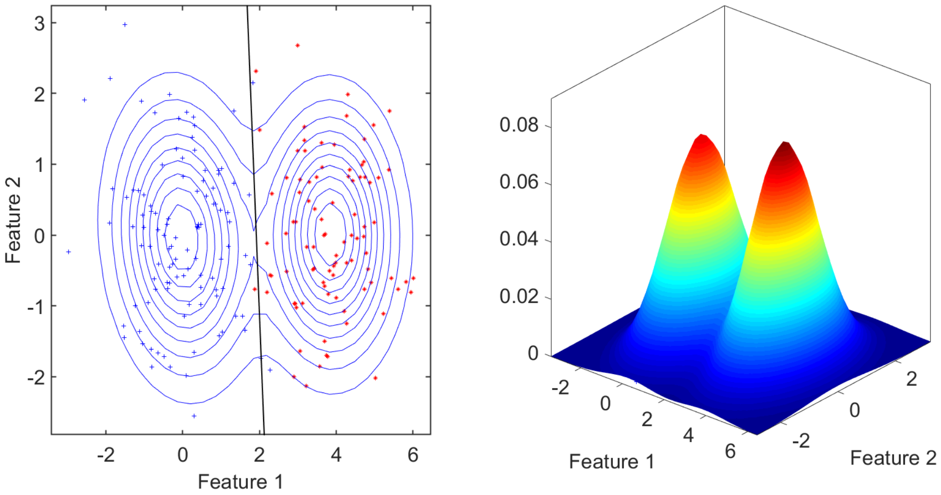


Figure 2-8: Plots illustrating Gaussian naive Bayes The left figure shows 200 observations belonging to class one (blue crosses) and class two (red stars). The contour lines represent the estimated Gaussians, shown in three dimensions in the right figure. The vertical black line in the left figure represents the decision boundary where the class-conditional probabilities are equal.

The means are estimated by calculating the sample mean vector. The jth element is calculated (with $K = 2$ in our example) as follows:

$$\bar{x}_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}, \quad j = 1, \ldots, K. \tag{2-9}$$

Thus, the sample mean vector contains the averages of each feature of a class:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_j \\ \vdots \\ \bar{x}_K \end{bmatrix} \tag{2-10}$$

Naïve Bayes owes its name to the assumption that the features are uncorrelated. Consequently, the sample covariance is estimated for every feature independently. The sample covariance matrix is, therefore, a diagonal K-by-K matrix $\mathbf{Q}$ with entries

$$q_{jk} = \frac{1}{N-1}\sum_{i=1}^{N}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k) \tag{2-11}$$

Since we now have estimates for the parameters defining the Gaussian distributions, it is possible to plot them as shown in the right plot in Figure 2-8. The Gaussians serve as an estimate for the class-conditional probability density functions (pdf) $p(\boldsymbol{x}|blue)$ and $p(\boldsymbol{x}|red)$.

To arrive at the class-conditional probabilities needed to make a prediction about a class-label we will refer to the Bayes' Rule:

$$P(\omega_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\omega_i)P(\omega_i)}{p(\boldsymbol{x})} \tag{2-12}$$

where $p(\boldsymbol{x})$ is the pdf of $\boldsymbol{x}$ and for which we have in the two class case:

$$p(x) = \sum_{i=1}^{2} p(\boldsymbol{x}|\omega_i)P(\omega_i) \tag{2-13}$$

$P(\omega_i)$ is used to denote the *a priori probabilities* which are estimated from the available training feature vectors. When there are N observations of which N_1 belong to class one and N_2 to class two, then $P(\omega_1) = \frac{N_1}{N}$ and $P(\omega_2) = \frac{N_2}{N}$.

The mapping learned from the data as in Equation (2-8), will now be based on the calculated class conditional probabilities from Equation (2-12):

$$\begin{aligned} &\text{if} \quad P(\omega_1|\boldsymbol{x}) > P(\omega_2|\boldsymbol{x}), \quad \boldsymbol{x} \text{ is classified to } \omega_1, \\ &\text{if} \quad P(\omega_1|\boldsymbol{x}) < P(\omega_2|\boldsymbol{x}), \quad \boldsymbol{x} \text{ is classified to } \omega_2 \end{aligned} \tag{2-14}$$

Such a classification rule is also called an *hypothesis.* The line corresponding to the case when $P(\omega_1|\boldsymbol{x}) = P(\omega_2|\boldsymbol{x})$ is drawn in the feature space and is referred to as the *decision boundary.* For the example shown in Figure 2-8 the vertical line in the left plot represents the generated *decision boundary.* New observations are plotted in the same figure and given a label based on its side of the decision boundary.

2-2-3-2 Model validation

When a classifier is trained and a decision boundary is constructed, the trained classifier is used to predict the labels of unseen observations.

Among other reasons, since there are many classifiers available, it is necessary to have a performance measure that shows how different classifiers perform compared to each other. This measure helps to select a classifier that adequately models the characteristics of the dataset.

Accuracy An often-used performance metric in classification is accuracy. Accuracy is defined as:

$$\text{accuracy} = \frac{\#\text{ of correctly classified points}}{\text{total }\#\text{ of observations}} \tag{2-15}$$

If we go back to the example of Figure 2-8, the total number of observations is 200. Furthermore, there are two blue crosses and one red star on the wrong side of the decision boundary, making the number of correctly classified points equal to 197. The accuracy would be calculated as $197/200 = 0.985$.

One could be tempted to conclude that 98,5 % of correctly classified observations directly indicates that the classifier is performing well. However, this accuracy is based on the correctly classified observations used to generate the decision boundary. The accuracy based on the training dataset is therefor called the *apparent accuracy* or *training accuracy.* The performance we are interested in is how well the classifier performs when applied to unseen data.

Test dataset To test whether the generated decision boundary performs well on unseen data points, the original data is split up in two parts that are called the *training dataset* D_{train} and the *test dataset* D_{test}. As illustrated in Figure 2-9, the observations in the training dataset are used to train the classifier and to generate a decision boundary. This hypothesis is then used to predict the labels of the observations in the test dataset. Since the observations in the test dataset were not used to train the classifier, the obtained accuracy is a better estimator for the generalization performance than the apparent accuracy and is called the *test accuracy.*

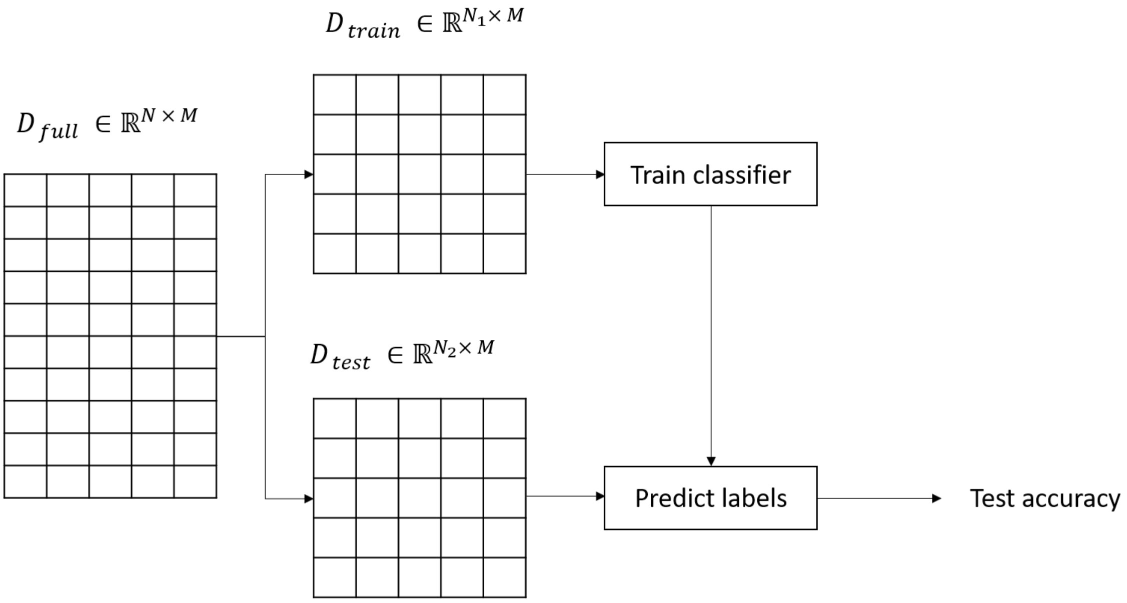

$D_{train} \in \mathbb{R}^{N_1 \times M}$
$D_{full} \in \mathbb{R}^{N \times M}$
Train classifier
$D_{test} \in \mathbb{R}^{N_2 \times M}$
Predict labels
Test accuracy

Figure 2-9: Illustration of the use of a test set to obtain the test accuracy The full dataset D_{full} is split into two datasets: D_{train} and D_{test}. D_{train} is used to train the classifier. This classifier is then used to predict the class of the observations in D_{test}. The percentage of correctly classified observations is called the test accuracy.

Cross validation A parameter that has to be set is the relative sizes of the train- and test dataset. There is a trade-off to be made here. Classifiers generally perform better when trained on more training data. However, since the test dataset is used to estimate the classifier's generalization performance, having a bigger test dataset ensures a more accurate estimate of this accuracy. As earlier mentioned, accurate test accuracy is preferred in order to make a justified choice of the classifier.

A validation-method ensuring that all observations are used for both training and testing is called *cross validation.* The dataset is split in k sections called *folds* as illustrated in Figure 2-10. A classifier is trained on $k-1$ folds, and the test accuracy is calculated using this trained classifier on the remaining fold. This process is repeated k times to obtain k test-accuracies. The total k-fold cross-validation accuracy is calculated by taking the average over the k test accuracies. The final classifier is trained on all observations.

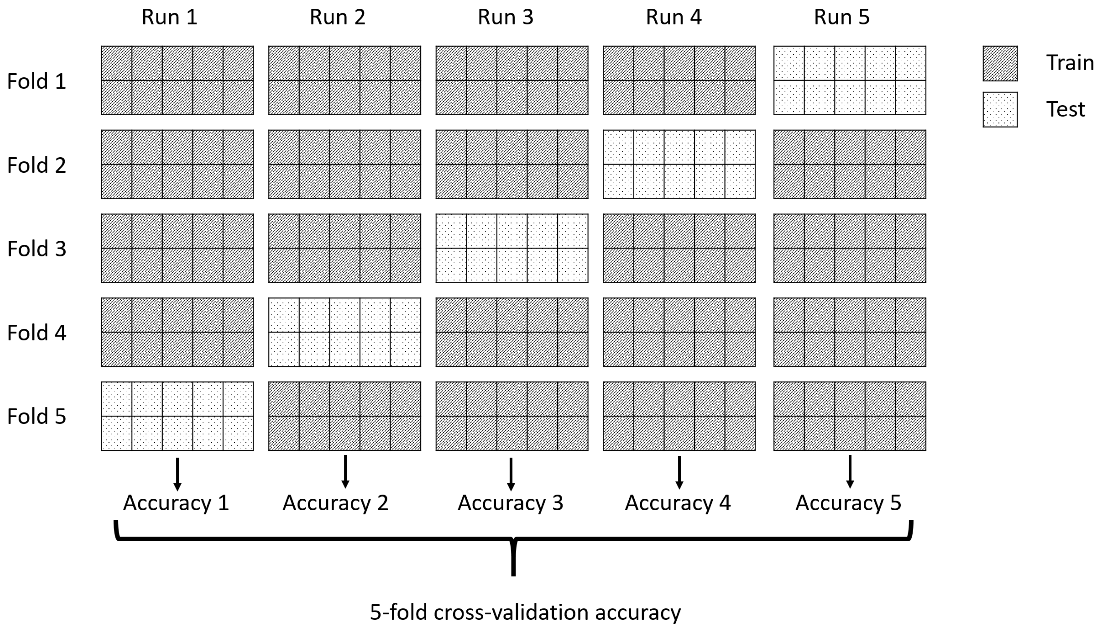


Figure 2-10: Schematic representation of 5-fold cross-validation The dataset is split into five folds. Each run, four folds are used to train the classifier, and the fifth fold is used to compute the test accuracy. The final cross-validation accuracy is computed by taking the average over the five obtained accuracies.

All observations are used in the train dataset and test dataset using this method. The number of folds k can be chosen from 1 to the number of observations N. A higher number of folds generally results in a more accurate accuracy. However, the classifier has to be trained k times, making the computational costs increase with the number of folds. The case where $k = N$ is called *leave-one-out* cross-validation, which has the highest computational load and most accurate cross-validation accuracy.

2-2-3-3 Prediction

The goal of classification is to predict the class-labels of unlabelled observations. Referring back to the example of the classification of cancer tumours at the beginning of this chapter, Meding et al. [1] obtained MALDI imaging-derived spectra data from tissue specimens from resectioned tumours. The classifier was trained on data originating from six different cancer types: Barrett's cancer, breast cancer, colon cancer, gastric cancer, hepatocellular carcinoma and papillary thyroid cancer. This makes for a six-class classification problem.

A train and test dataset were defined, and two classifier types were trained called Support Vector Machine[49] and Random Forest[50]. The test dataset was used to evaluate the performance of the classifiers.

These trained classifiers can now be used to discriminate between the different cancer types of new tissue samples.

2-3 Dimensionality reduction

This chapter will elaborate on the motivation for the application of dimensionality reduction before training a classifier. The process is illustrated in Figure 2-11. The left side of Figure 2-11 shows the full dataset $D_{full} \in \mathbb{R}^{N \times M}$ consisting of features $[\boldsymbol{d_1}, \boldsymbol{d_2}, \boldsymbol{d_3}, \ldots, \boldsymbol{d_M}]$ where $\boldsymbol{d} \in \mathbb{R}^{N \times 1}$. Dimensionality reduction aims to construct a representative, reduced dataset $D_{red} \in \mathbb{R}^{N \times K}$ with $K < M$.

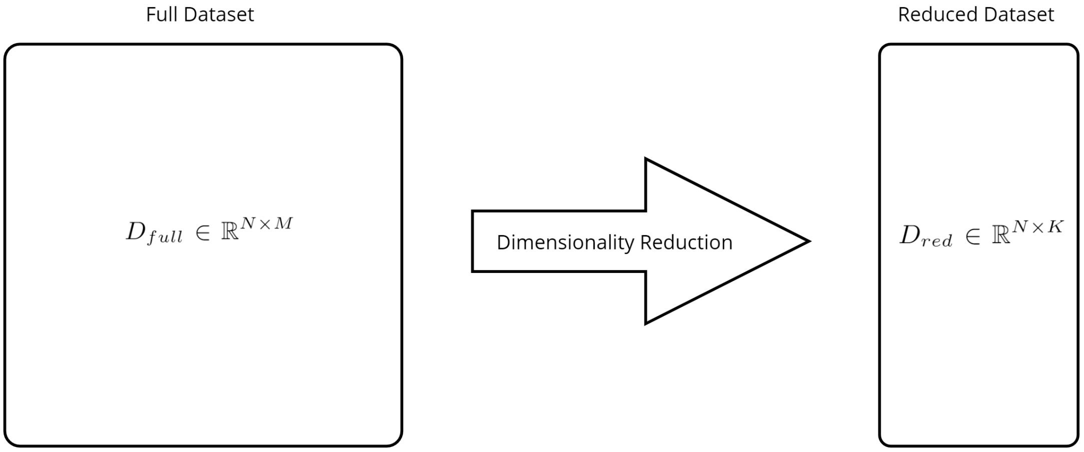

Full Dataset
$D_{full} \in \mathbb{R}^{N \times M}$
Dimensionality Reduction
Reduced Dataset
$D_{red} \in \mathbb{R}^{N \times K}$

Figure 2-11: Schematic representation of dimensionality reduction Dimensionality reduction aims to contruct a representative dataset $D_{red} \in \mathbb{R}^{N \times K}$ from the full dataset $D_{full} \in \mathbb{R}^{N \times M}$ with a reduced number of features : $K < M$.

In the rest of this chapter, the motivation for dimensionality reduction is given. The two main categories within dimensionality reduction, feature selection and feature extraction, are discussed.

2-3-1 Motivation

As discussed in chapter 2-2, the classification goal is to predict the class of an unseen observation by training a model on labelled data. One could assume that when a dataset has more features, the hypothesis created with a classifier will perform better in prediction accuracy. The rationale would be that the more features each observation has, the more information is available. With this reasoning, it will be easier to construct a decision boundary that separates the training data points. However, more observations are needed to describe the feature space spanned by the features well when more features are in a dataset. The volume of the feature space grows exponentially with the number of features. This phenomenon is illustrated in Figure 2-12. If the observations do not adequately fill the feature space, the prediction accuracy can decrease. This phenomenon is called the 'curse of dimensionality' in literature and is further discussed in section 2-3-1-1.

Even when the prediction accuracy does not decrease using all available features, there are other reasons why using a reduced feature set is preferable with model training. The size of the dataset used in the modelling phase directly correlates with the time and space needed to train the model. This effect will be discussed in section 2-3-1-4.

2-3-1-1 Curse of dimensionality

The term 'Curse of dimensionality' is used in many fields, including numerical analysis, sampling, combinatorics, machine learning, and data mining. The term is used as an umbrella term for the problems that arise with analyzing large datasets. The root of the issues that occur with an increase in dimensionality is the amount of data needed to populate the feature space adequately. For two dimensions, this problem is illustrated in the following example. A feature vector of length M of an observation can be represented by a point in a M-dimensional space. In Figure 2-12 50 observations are plotted with one and two features in a and b respectively. It is illustrated here that the average distance between points becomes larger when the dimensionality increases. In other words, space is less densely populated when the number of observations stays the same and the dimensionality increases.

In generating a decision-border by a classifier in the feature space, fewer points are available to conclude this hypothesis.

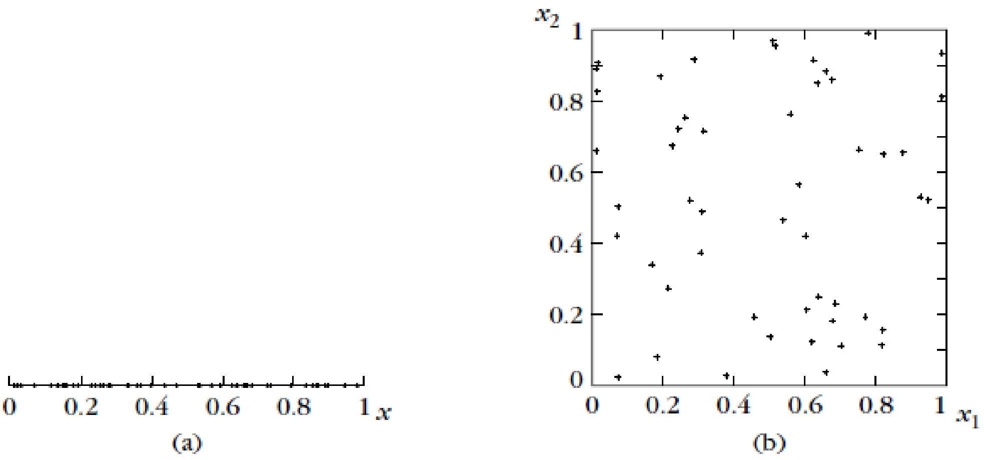


Figure 2-12: The effect of adding dimensions on the density of data points in the feature space Fifty data points drawn from a uniform distribution are shown in one dimension (a) and two dimensions (b). The points in the two-dimensional space are more spread out than in the one-dimensional space. Source of image: Theodoridis and Koutroumbas [51]

2-3-1-2 The peaking phenomenon

As a result of the less populated feature space, classifiers' performance can vastly decrease by adding more dimensions. This is a paradoxical observation, referred to as the peaking phenomenon. In theory, the probability of misclassification does not increase as the number of features increases, as long as the class-conditional densities are completely known (in other words, when there is a sufficient amount of data available that is representative for the underlying densities)[9]. This last assumption is, however, exactly the problem in practice. Simple parametric classifiers estimate the parameters of assumed class-conditional distributions and use these estimates as true values in the model. Therefore, when the dimensionality (and therefore, the number of estimated parameters) grows, the estimates' reliability and the resulting classifier's performance decreases[52]. The phenomenon is illustrated in Figure 2-13.

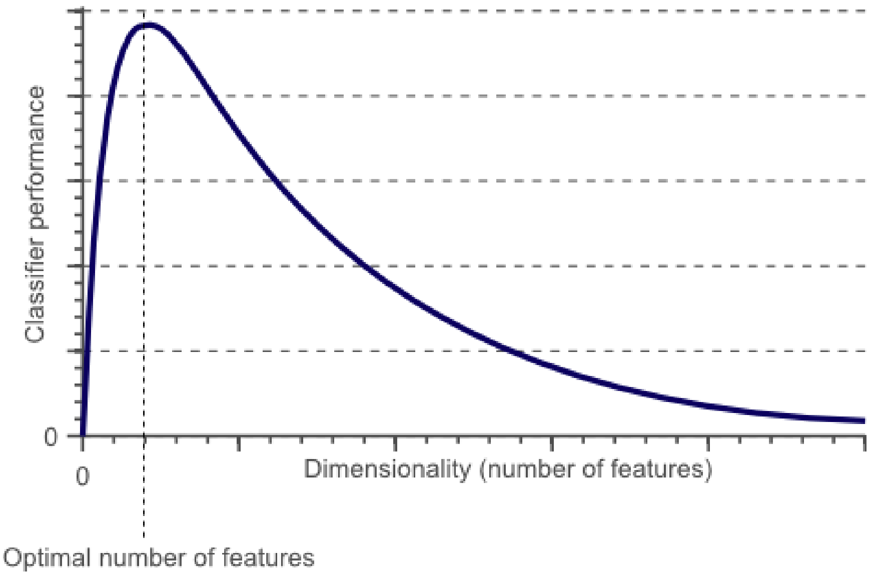


Figure 2-13: Peaking phenomenon When the dimensionality of a dataset increases, the classifier's performance increases until the optimal number of features is reached. When the dimensionality is increased any further without adding more training data points, the classifier's performance will drop. Source of image: Spruyt [53]

Many classifiers incorporate a distance measure. One of these classifiers is the Nearest Neighbour classifier that assigns the same label of the closest training points to the unseen observation. However, in high dimensions, it can be shown that the expected value of the difference of the distance from the nearest and the farthest point approaches zero:[54]:

$$\lim_{d \to \infty} E\left(\frac{dist_{max}(d) - dist_{min}(d)}{dist_{min}(d)}\right) = 0$$

However, this is true with the assumption that the features are independent and identically distributed. According to Zimek, Schubert, and Kriegel [55] there are scenarios with correlated features where Nearest Neighbour makes sense in high dimensions. In any case, applying dimensionality reduction to remove irrelevant and redundant features will improve the quality of distance measures.

2-3-1-3 Overfitting

A problem closely related to the curse of dimensionality is called overfitting. This phenomenon occurs when the generated model depends on the train set's specific structure rather than the underlying distribution and will not generalize for unseen data points. Overfitting happens more often in the case where more dimensions than data points are available. When we consider the case of a decision boundary consisting of a hyperplane, it can be shown that we can arbitrarily separate all data points if we use more dimensions than data points. However, this will result in an artificially complex classifier projected into a lower-dimensional space, making it perform poorly on unseen data. Hence the relationship to the curse of dimensionality.

An indication of the amount of overfitting is the difference in training and test accuracy. We consider a classifier that has a 100% accuracy on the training set (*apparent accuracy*) and an 50% classification rate on the test set (*test accuracy*). If simultaneously on the same dataset, a classifier is possible, which scores 75% on both training and test data, the first classifier was a victim of overfitting.

2-3-1-4 Time and space

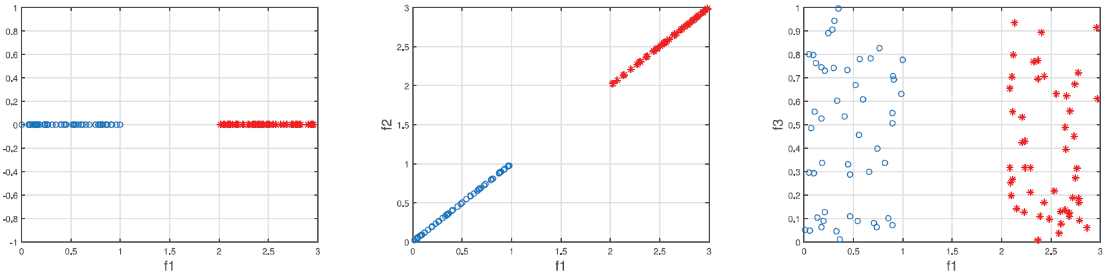
With a reduced feature-space, analyses become faster and take less space in RAM and storage capacity. In the training phase of a classifier, the time needed is positively correlated with the number of dimensions and can often increase quadratically or exponentially. Using too many dimensions makes some complex classifiers simply infeasible to train because the time and space blow up. Dimensionality reduction is needed to give access to those more sophisticated classifiers.

2-3-1-5 Effect of Redundancy

When datasets become more extensive, it becomes increasingly possible that not all the features provide information for the task at hand. For the reasons explained above, it is beneficial to remove these features from the dataset. A feature that does not contribute to the task is called an irrelevant feature. Likewise, a feature that does contribute to the task at hand is called a relevant feature.

However, not every relevant feature has to be kept to ensure an optimal dataset. For example, if we have a population dataset with two features that indicate the height of the people in the dataset in feet and meters, we would not lose information if one of the features is eliminated. Such a feature is called a redundant feature.

Figure 2-14 illustrates these three types of features. Figure 2-14a shows relevant feature f_1 on the x-axis. This feature needs to be kept in the dataset. Figure 2-14b shows two relevant but redundant features. Either f_1 or f_2 could be eliminated without loss of information. Figure 2-14c shows irrelevant feature f_3 on the y-axis. The two classes have much overlap. Dimensionality reduction methods aim to eliminate irrelevant and redundant features. A dataset with all redundant and irrelevant features removed is called an optimal subset[56].



(a) relevant feature f_1

(b) redundant feature f_2

(c) irrelevant feature f_3

Figure 2-14: Illustration of relevant, redundant and irrelevant features Relevant features are informative and should be conserved (a). Redundant features capture the same information. One of the features should be conserved (b). Irrelevant features carry no information. They should be eliminated (c). Source of image: Li et al. [57]

2-3-2 Feature selection

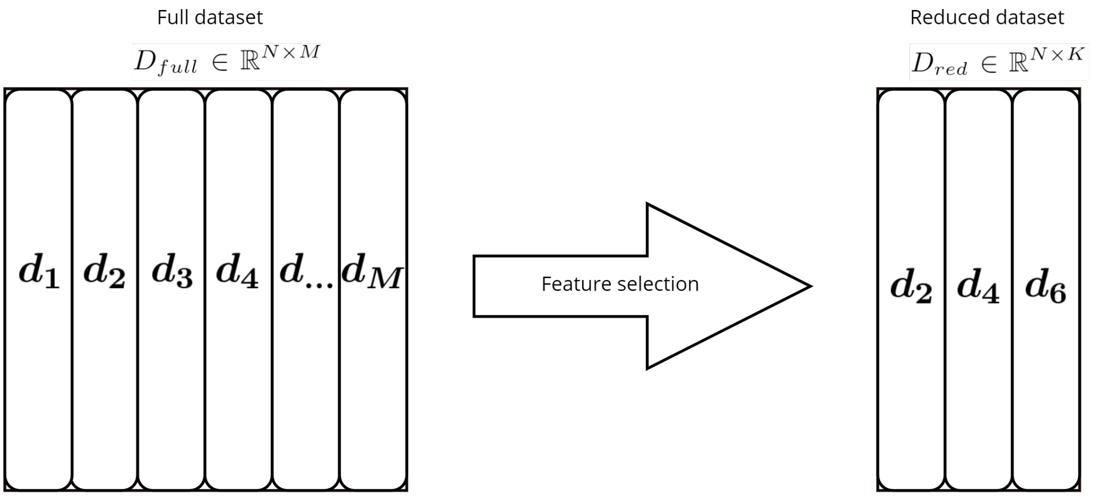


Figure 2-15: Schematic representation of feature selection Feature selection reduces the dimensionality of the full dataset D_{full} by selecting a subset of its columns and putting them in the reduced dataset $D_{red} \subset D_{full}$. In this example the algorithm selected three columns ($K = 3$): $\boldsymbol{d_2}, \boldsymbol{d_4}$ and $\boldsymbol{d_6}$.

The first approach to reduce the dimensionality of a dataset we will discuss is called feature selection. Figure 2-15 shows the full dataset D_{full} with columns $[\boldsymbol{d_1}, \boldsymbol{d_2}, \boldsymbol{d_3}, \ldots, \boldsymbol{d_M}] \in \mathbb{R}^{N \times 1}$. A feature selection algorithm constructs the reduced dataset D_{red} with a subset of these columns: $D_{red} \subset D_{full}$. In this example three columns are selected by the algorithm: $\boldsymbol{d_2}, \boldsymbol{d_4}$ and $\boldsymbol{d_6}$. Feature selection algorithms aim to find an optimal subset of the original features by eliminating irrelevant, redundant and noisy features[58]. The concepts of relevance and redundancy were explored in section 2-3-1-5. Since the remainder of this thesis mainly covers the other family of DR methods discussed in the following section: feature extraction, the full overview of feature selection methods is given in Appendix A.

2-3-3 Feature extraction

The second category within the dimensionality reduction field is called feature extraction. As opposed to feature selection where D_{red} consists of a subset of the original features, feature extraction aims to create a set of new features that are a linear or non-linear combination of the original features.

Mathematical description A schematic representation of feature extraction is shown in figure 2-16. The input to the feature extraction routine is the full dataset D_{full}, consisting of feature columns $\boldsymbol{d_{f1}}, \boldsymbol{d_{f2}}, \boldsymbol{d_{f...}}, \boldsymbol{d_{fM}}$. The routine takes these columns as input and aims to transform the columns to describe the dataset accurately. The transformed data is stored in a new dataset $D_{reduced}$ with columns $\boldsymbol{d_{r1}}, \boldsymbol{d_{r2}}, \boldsymbol{d_{r...}}, \boldsymbol{d_{rM}}$. Most of the methods aim to capture the data's characteristics in as few features as possible, sorting the new features from most to least informative. The least informative features are eliminated from the dataset to reduce the dimensionality. In the example of Figure 2-16, three feature columns are kept to reach a final dimensionality of three. Finding the optimal number of features to be kept is non-trivial and often found by trial and error (for example, choosing the number with the highest cross-validation accuracy) in practice.

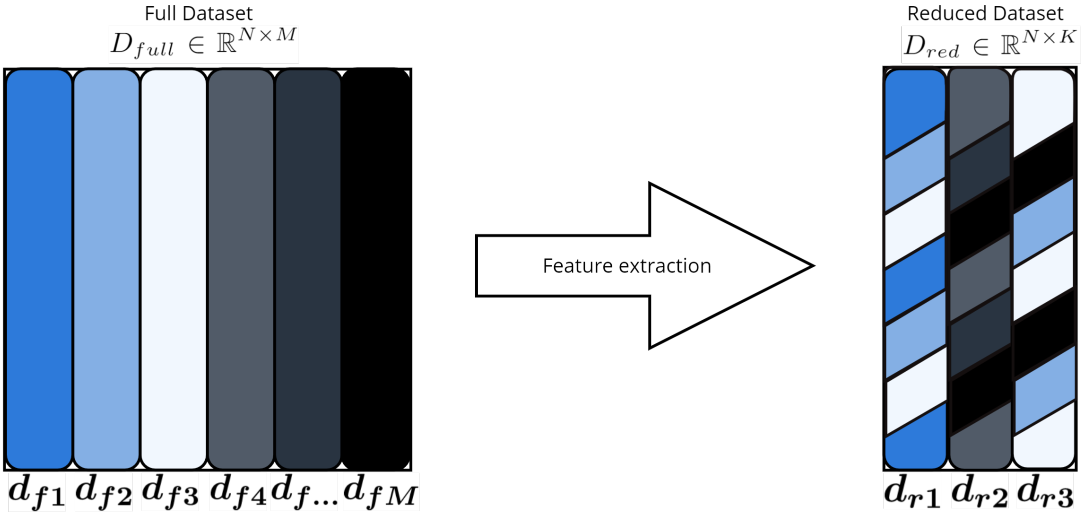

Full Dataset
$D_{full} \in \mathbb{R}^{N \times M}$
d_{f1} d_{f2} d_{f3} d_{f4} $d_{f\dots}$ d_{fM}
Feature extraction
Reduced Dataset
$D_{red} \in \mathbb{R}^{N \times K}$
d_{r1} d_{r2} d_{r3}

Figure 2-16: Schematic representation of feature extraction Feature extraction creates a combination of the features of D_{full} and puts them in D_{red}. In contrast to feature selection, D_{red} does not consist of a subset of the columns of D_{full} but a (linear) combination of them.

Examples Feature extraction methods can be largely divided into linear and non-linear techniques[59]. A commonly used linear method is PCA, an unsupervised method projecting the data in a new feature space aiming to maximize the variance in the first few created features. PCA will be extensively discussed in Section 2-3-3-1. Another linear method is Linear Discriminant Analysis (LDA). This supervised technique aims to project the data in a feature space that maximizes the class separation. LDA will be further explained in Section 2-3-3-2. A recent contribution that aims to find a trade-off between the benefits of PCA and LDA was proposed by Liu and Gillies [9]. This supervised method called Soft Discriminant Map (SDM) aims to reduce the risk of overfitting present when using LDA in high dimensional datasets by tuning a parameter that controls the separation of the classes. SDM is discussed in Section 2-3-3-3.

Examples of non-linear techniques are kernel-PCA[60], Multi-dimensional scaling (MDM)[61] and isometric feature mapping (Isomap)[62]. These non-linear techniques can detect higher-order redundancies than linear methods at the cost of higher computational complexity. This makes them costlier for high dimensional datasets and harder to interpret the relationship to the original features.

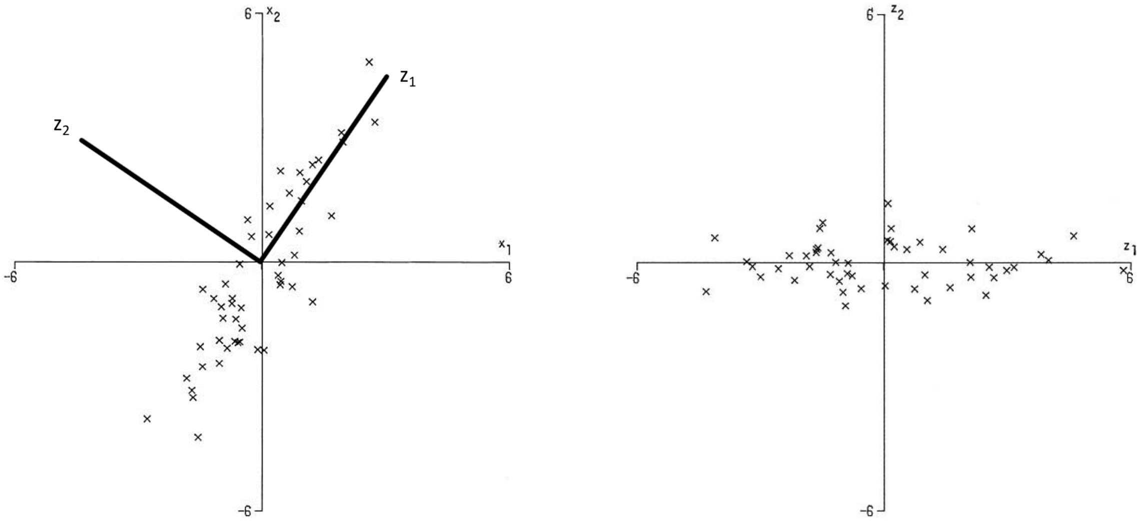
In the following sections, three linear feature extraction methods are further explored: PCA, LDA and SDM.

2-3-3-1 Principal Component Analysis

A frequently used feature extraction technique in machine learning is called PCA[8]. This unsupervised technique's main goal is to reduce a dataset's dimensionality while retaining as much variance as possible in the new feature space.

To illustrate the process of PCA, consider a dataset $D_{full} \in \mathbb{R}^{50\times 2}$ having 50 data points with 2 feature values each: x_1 and x_2. By considering this simple two-dimensional example, the dataset can be visualized in two dimensions, as seen in Figure 2-17a. The plot suggests that the features are highly correlated, which, loosely speaking, means that x_1 and x_2 move together in value. PCA aims to find the directions in this dataset with the highest variance and that are orthogonal to each other. In this simple example, the approximate directions can be drawn, as shown in Figure 2-17a. Vectors describe these directions $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ and are called the principal components. Next, the data points can be projected in the new feature space span by these principal components, as shown in Figure 2-17b.

In the context of dimensionality reduction, the next step with this simple example dataset could be to only consider z_1 in further analysis, resulting in a reduced dataset $D_{reduced} \in \mathbb{R}^{50\times 1}$. Note that in this simple example, important information is probably lost by discarding half of the features. However, in a more realistic higher dimensional correlated scenario, such as datasets from the IMS field, the last principal components have a great chance of only containing noise and can most likely safely be discarded. Note that PCA is an unsupervised technique, meaning that it does not use class information. When using PCA in a supervised setting, the assumption is that retaining most of the dataset variance is beneficial for the classification task.



(a) 50 data points plotted in the original feature space $\{\boldsymbol{x_1}, \boldsymbol{x_2}\}$. The directions of greatest variance $\{\boldsymbol{z_1}, \boldsymbol{z_2}\}$ which are orthogonal to each other are drawn. Principal component analysis aims to find these directions.

(b) The same 50 data points plotted in the new feature space $\{\boldsymbol{z_1}, \boldsymbol{z_2}\}$. In this new coordinate system most of the variance of the data points is explained by the first principal component $\boldsymbol{z_1}$.

Figure 2-17: Visualization of principal component analysis PCA aims to find the directions of most variance in the dataset and transforms the dataset in a feature space spanned by these directions called principal components. Source of images: modified from Jolliffe [8].

Next, we consider the mathematical construction of the principal components based on the full derivation made by Jolliffe [8]. The first step of PCA is to find a linear combination $\boldsymbol{\alpha_1'x}$ of the elements of feature vector $\boldsymbol{x}$ that maximizes the variance. Where α_1 is a vector of p constants $\alpha_{11}, \alpha_{12}, \alpha_{13}, \ldots, \alpha_{1p}$ and where p is the dimensionality of the dataset such that

$$\boldsymbol{\alpha_1'x} = \alpha_{11}x_1 + \alpha_{12}x_2 + \alpha_{13}x_3 + \cdots + \alpha_{1p}x_p = \sum_{j=1}^{p} \alpha_{1j}x_j = z_1 \tag{2-16}$$

Next, we look for a similar linear function $\boldsymbol{\alpha_2'x}$ with maximum variance which is uncorrelated with $\boldsymbol{\alpha_1'x}$. This process is repeated for every principal component so that at the kth stage a linear function $\boldsymbol{\alpha_k'x}$ is found that maximizes the variance and is uncorrelated to the previously found principal components $\boldsymbol{\alpha_1'x}, \boldsymbol{\alpha_2'x}, \boldsymbol{\alpha_3'x}, \ldots, \boldsymbol{\alpha_{k-1}'x}$. Up to p principal components could be found but, by construction, most of the variation in x is accounted for in the first few components. In general, the more correlated the features of the original dataset are, the more of the variation is explained by the first few principle components.

Suppose in our example that x_1 and x_2 are random variables with a covariance matrix Σ. This matrix consists of the covariance between the ith and the jth element of $\boldsymbol{x}$ when $i \neq j$, and the variance when $i = j$. This matrix is used in the construction of the principal components. However, the real covariance matrix is unknown and estimated with the sample covariance matrix S.

It is shown by Jolliffe [8] that the coefficient vectors $\boldsymbol{\alpha}$ could be computed by calculating eigenvectors of the (sample) covariance matrix of a dataset. The principal components are computed by post-multiplying them with the original features $\boldsymbol{x}$.

In terms of computing costs, this process means that a sample covariance matrix of the dataset has to be found $S \in \mathbb{R}^{p \times p}$. When dealing with high dimensional datasets such as IMS, this matrix could become so large that it does not fit the computer's RAM. Klerk [63] proposes a variation of PCA that exploits the fact that IMS datasets are often sparse (contains a high number of zeros), which he uses to reduce the size of the covariance matrix. Race *et al.*[64] propose another variation of PCA that sequentially computes the covariance matrix, ensuring that only one data point at a time has to be kept in memory, a so-called online approach. These methods facilitate that PCA could still be used when the dimensionality of a dataset becomes large.

2-3-3-2 Linear Discriminant Analysis

LDA aims to find a linear combination of the original features that maximize class separation. Figure 2-18 illustrates LDA with a two-class, two-dimensional problem. The data points are generated from two multi-variate Gaussian distributions with equal diagonal covariance matrices and different means $\boldsymbol{\mu_1}$ $\boldsymbol{\mu_2}$ shown at the top of the figure. The line at the bottom represents LDA's direction, and the dots on this line are the projected data points. In this dimension, the two classes are still linearly separable. It can be shown that the generated dimension in this example is parallel to the vector $\boldsymbol{\mu_1} - \boldsymbol{\mu_2}$[51]. This does not hold in the Gaussians' general case with non-diagonal elements in the covariance matrix.

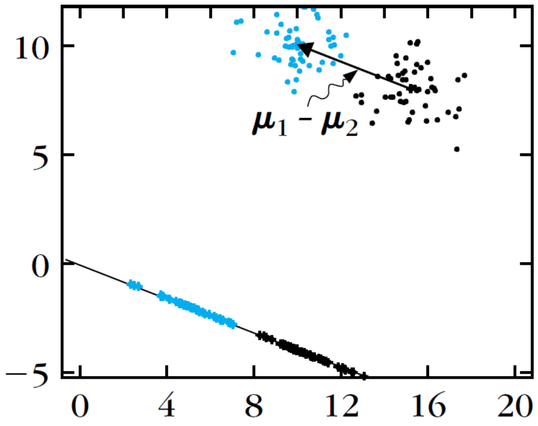


Figure 2-18: Illustration of LDA LDA aims to find a linear combination of the original features that maximally separates the classes. In this two-class problem, the line on the bottom represents the direction found by LDA with the projected data points drawn on it. In this new dimension, the two classes are fully separable, while the dimensionality has been reduced from two to one. Source of image: Theodoridis and Koutroumbas [51].

Mathematical description The objective direction is described by a linear transformation $\boldsymbol{\Omega} \in \mathbb{R}^{K \times M}$ that transforms the original dataset $D_{full} \in \mathbb{R}^{N \times M}$ to the reduced dataset $D_{red} \in \mathbb{R}^{N \times K}$. M represents the dimensionality of the full dataset D_{full} and K the dimensionality of reduced dataset D_{red} with $K < C - 1$ where C is the number of classes. $\boldsymbol{\Omega}$ corresponds to the transformation matrix $\boldsymbol{A}$ that maximizes the *Fisher Criterion*:

$$J_F(\boldsymbol{A}) = \text{trace}((\boldsymbol{A}\boldsymbol{S}_W\boldsymbol{A}^T)^{-1}(\boldsymbol{A}\boldsymbol{S}_B\boldsymbol{A}^T)), \tag{2-17}$$

where

$$S_b := \sum_{i=1}^{C} p_i(\boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}})(\boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}})^T, \tag{2-18}$$

and

$$S_w := \sum_{i=1}^{C} p_i \boldsymbol{S}_i \tag{2-19}$$

are the between-class scatter matrix and the summed within-class scatter matrix, respectively. C is the number of classes, $\boldsymbol{\mu}_i$ the mean vector of class i, p_i the *a priori* probability, estimated by $\frac{n_i}{N}$ the number of data points in class i devided by the total number of data points. The overall mean is defined as $\bar{\boldsymbol{\mu}} = \sum_{i=1}^{c} p_i \boldsymbol{m}_i$. Furthermore $\boldsymbol{S}_i$ is the sample covariance matrix of the data points in class i.

It is shown by Theodoridis and Koutroumbas [51] that the solution to this maximization problem of equation 2-17 is found by finding the eigenvectors of the objective matrix we will call Φ:

$$\Phi = S_w^{-1} S_b \tag{2-20}$$

The transformation matrix $\boldsymbol{\Omega}$ is found by taking the rows to be the eigenvectors of Φ corresponding to the K largest eigenvalues.

Maximum number of embedded dimensions LDA has the limitation that the maximum number of embedded dimensions is equal to $C - 1$ [51], where C stands for the number of classes. The derivation is shown below.
Since

$$\text{rank}(AB) \leq \min(\text{rank}(A), rank(B)) \tag{2-21}$$

hence

$$rank(S_w^{-1} S_b) \leq rank(S_b) \leq c - 1 \tag{2-22}$$

As a consequence, in a two-class classification problem, using LDA as a dimensionality reduction technique will reduce the dimensionality to one. When the original dataset has a high number of features compared to the amount of data points , $M > N$, LDA has the risk of overfitting the data as shown in [9].

To summarize, as opposed to PCA, LDA does incorporate class information to find dimensions that separate the classes well for a classification algorithm to generalize well. However, LDA has the risk of overfitting the data. The following section describes a technique that aims to mitigate the risk of overfitting while still incorporating class information.

2-3-3-3 Soft Discriminant maps

The main idea LDA is based upon is that the classification error improves by maximizing the inter-class discrimination, **i.e.** the creation of compact clusters of data points that are far away from each other. Liu and Gillies [9] show that this is untrue for high-dimensional datasets. They show that when the number of features is higher than the number of data points ($M > N$), it is arbitrary to find a feature space with LDA that minimizes the within-class variance to such an extent that the projected data points appear as a single point when plotted in a scatter plot. Namely, a dimension can always be found that ensures that all data points of a class coincide, as seen in Figure 2-19e. However, when data points of the test dataset are plotted in the same feature space, their location is, in general, not close to the training points, suggesting that the learning model suffered from overfitting.

To overcome this problem, a method called Soft Discriminant Map (SDM) is proposed by Liu and Gillies [9]. The approach modifies LDA by controlling the weight the optimization problem assigns to the minimization of the within-class variance to reduce overfitting. In Figure 2-19, the AT&T Dataset of Faces [65] is projected in two dimensions using different methods. In Figure 2-19a-d, SDM has been applied with different values of β, the coefficient controlling the amount of class separation. Subfigure e shows the result of LDA, and subfigure f shows the projection obtained by PCA. For high values of β, The data points coincide similar to LDA. For low values of β, the class separation is reduced, making the projection look more like the PCA result.

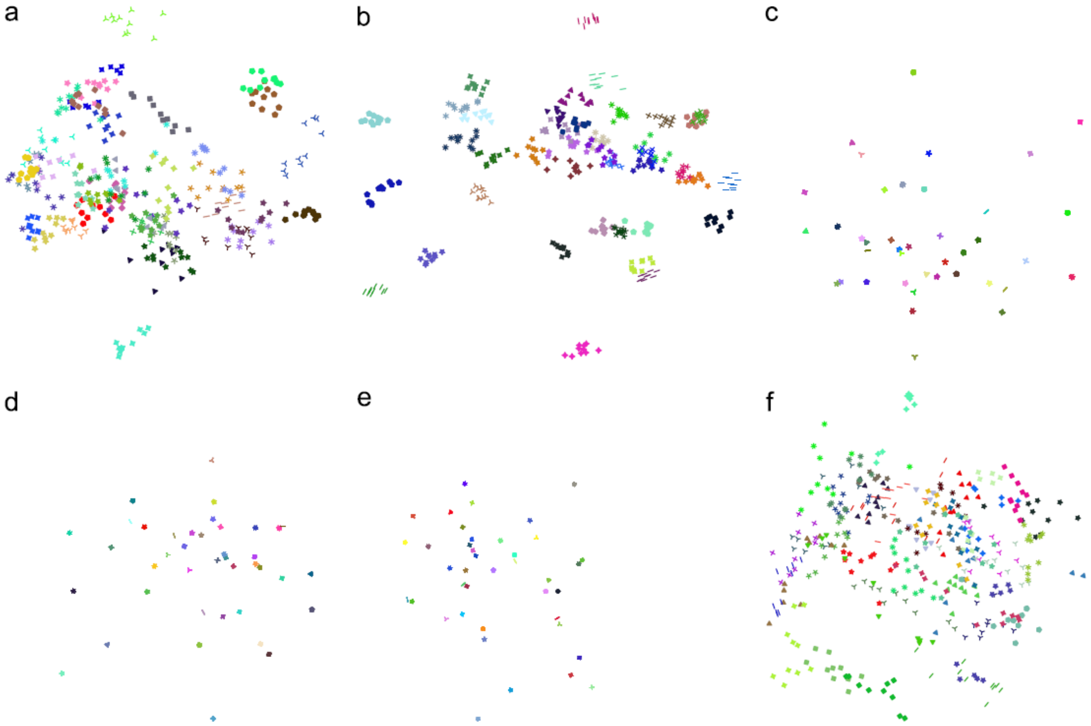


Figure 2-19: 2D projections of the AT&T Dataset of Faces computed by SDMs (a) $\beta = 10$,(b) $\beta = 50$, (c) $\beta = 10.000$,(d) $\beta = 10.000.000$, LDA in (e), and PCA in (f). Source of image: Liu and Gillies [9]

Mathematical description The derivation of the new features from SDM is similar to those of LDA. In LDA, the new directions are found by calculating the eigenvectors of the objective matrix Φ of Equation 2-20. With SDM, the directions are found by calculating the eigenvectors of a new objective matrix Φ_β:

$$\Phi_\beta := S_b - \beta S_w, \; \beta \in \mathbb{R}, \tag{2-23}$$

where S_b is the between scatter matrix, S_w the within scatter matrix and β a factor that scales the within scatter matrix. LDA maximizes the ratio between the between-class variance and the within-class variance, while SDM maximizes the weighted difference between them. The transformation matrix Ω, analogue to LDA, is found by placing the eigenvectors corresponding to the K largest eigenvalues in the rows of Ω. The optimization problem SDM aims to solve is further discussed in the following chapter.

Classification performance For classification problems using datasets with high dimensionality, SDM can use class information unlike PCA and is less likely to be victim of overfitting like LDA. Liu and Gillies [9] showed experimentally that in this scenario, SDM performs better than PCA and LDA in terms of classification error.

Chapter 3

Methods

This research proposes an extension to Soft Discriminant Map (SDM), the linear feature extraction method proposed by Liu and Gillies [9]. Before the extension, which we have named Data-Driven Soft Discriminant Map (DD-SDM), is presented in section 3-3, SDM will first be defined in section 3-1. The new feature space's classification performance is often used as a performance metric in the experiments that were performed. Section 3-2 describes the procedure used to derive this performance metric.

3-1 Soft Discriminant Maps

The essence of SDM is to influence the level of class-discrimination in the new feature space to avoid overfitting while still using class information to extract new features, called soft discriminants. SDM is used to reduce the dimensionality of the full dataset D_{Full} which has N data points and M features. In section 2-3-3-3, SDM was introduced and can also be defined with the following optimization problem given by Liu [66]:

$$\begin{aligned} \underset{\underline{w}\in\mathbb{R}^{M\times 1}}{\text{Maximise}} \quad \underline{w}^T S_b \underline{w} \quad \text{subject to} \quad & \underline{w}^T S_w \underline{w} = \alpha \\ \text{and} \quad & \underline{w}^T \underline{w} = 1, \end{aligned} \tag{3-1}$$

where $S_b \in \mathbb{R}^{M\times M}$ is the between-scatter matrix and $S_w \in \mathbb{R}^{M\times M}$ the within-scatter matrix as defined in Equations 2-18 and 2-19. These scatter-matrices are calculated based on the full dataset $D_{full} \in \mathbb{R}^{N\times M}$. $\underline{w}$ is the vector to be varied containing the weights used to linearly transform the original features to the corresponding soft discriminant, and α is a constant. The value of α is irrelevant to the construction of w since it vanishes in the derivative of the Lagrangian as seen shortly in Equation 3-3. To put Equation 3-1 into words: a unit vector needs to be found that maximizes the between-class variance (the distance between class means) as long as the within-class covariance is equal to a constant number α. To solve this optimization problem, the Lagrangian is constructed:

$$L(\underline{w}, \beta, \lambda) = \underline{w}^T S_b \underline{w} - \beta(\underline{w}^T S_w \underline{w} - \alpha) - \lambda(\underline{w}^T \underline{w} - 1) \tag{3-2}$$

By setting the derivative with respect to $\underline{w}$ to zero, we obtain

$$(S_b - \beta S_w)\underline{w} = \lambda \underline{w} \tag{3-3}$$

Repeating the definition of Equation 2-23:

$$\Phi_\beta := (S_b - \beta S_w) \tag{3-4}$$

Equation 3-3 shows that $\underline{w}$ is an eigenvector of Φ_β with eigenvalue λ.To be more specific, $\underline{w}$ is the eigenvector with the highest corresponding eigenvalue. The second soft discriminant is the eigenvector with the second-highest eigenvalue and so forth. Since both the between-scatter matrix $S_b \in \mathbb{R}^{M\times M}$ and the within-scatter matrix $S_w \in \mathbb{R}^{M\times M}$ are square and symmetric, and Φ_β is a weighted difference between them, Φ_β is also square and symmetric. A fortunate property of symmetric matrices is that their eigenvectors are guaranteed to be orthogonal. As a result, the soft discriminants are orthogonal to each other. Another way to find the subsequent soft discriminants is to repeat the optimization problem, with an added constraint that $\underline{w_2}$ should be orthogonal to $\underline{w_1}$. In general, $\underline{w_n}$ should be orthogonal to $\underline{w_i}$ with $0 < i < n$.

The first K soft discriminants are stacked horizontally to obtain a transformation matrix $\Omega \in \mathbb{R}^{M\times K}$. Matrix Ω can be used to map the original data points to the new coordinate system by post-multiplying with the full dataset $D_{full} \in \mathbb{R}^{N\times M}$:

$$D_{red} = D_{full}\Omega \tag{3-5}$$

The reduced dataset $D_{red} \in \mathbb{R}^{N\times K}$ has a reduced dimensionality K. The following section explains how this thesis evaluates whether this procedure improved the dataset for further classification analysis.

3-2 Classification performance measures

By reducing the full dataset's dimensionality, D_{full}, the problems of handling big datasets as explained in chapter 2-3 are hopefully mitigated. To evaluate whether the reduction resulted in a dataset D_{red} that is more suitable for classification, this section introduces standardized performance measures.

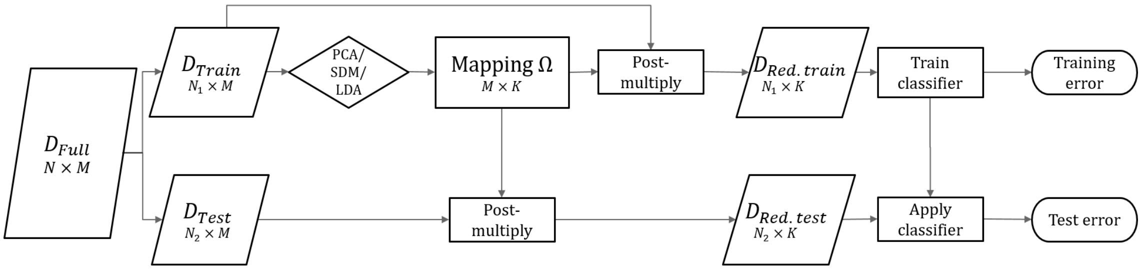


Figure 3-1: Process of applying dimensionality reduction in a classification pipeline to obtain a training- and test error The parallelograms represent datasets with their dimensions. The diamond represents the application of a dimensionality reduction method. A rectangle represents a procedure such as matrix multiplication, or the training of a classifier and the ellipses represent the calculated performance metrics.

Figure 3-1 shows the process of obtaining two performance metrics: the training error and the test error. These metrics are a measure for the classification performance of the classification pipeline and were more in-depth discussed in section 2-2-3. The full dataset D_{Full} consists of N data points with M features. Each data point is assigned to a class with a label. D_{Full} is then split into two datasets. The first being the training dataset D_{Train} with N_1 data points and M features. The second set is the test dataset with N_2 data points and M features where $N_1 + N_2 = N$.

The test dataset is used to test whether a classifier trained on this new feature space also performs well on unseen data. *E.g.*, in the case of Imaging Mass Spectrometry (IMS), this means that we want to know whether the dimensionality reduction methods capture underlying biological patterns or that they capture noise that coincidentally separates the mass spectra well. When there is a big difference between the test error and the training error, the model was a victim of overfitting.

In this thesis, SDM is compared to two other linear feature extraction methods that serve as a benchmark: Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). The dimensionality reduction method to be evaluated is applied to training dataset D_{train}, resulting in a transformation matrix $\Omega \in \mathbb{R}^{M \times K}$, where K is the number of latent variables whos value is dependent on the method used. Matrix Ω is used to project the data points of $D_{train} \in \mathbb{R}^{N_1 \times M}$ and $D_{Test} \in \mathbb{R}^{N_2 \times M}$ onto the new coordinated system resulting in the two reduced datasets $D_{Red.train} \in \mathbb{R}^{N_1 \times K}$ and $D_{Red.test} \in \mathbb{R}^{N_2 \times K}$. The projection is performed by post-multiplying Ω with the datasets:

$$D_{Red.train} = D_{Train}\Omega \tag{3-6}$$

$$D_{Red.test} = D_{Test}\Omega \tag{3-7}$$

Since the data points do not change classes during the dimensionality reduction step, the labels of the data points in the reduced datasets remain the same and are therefore still known. The labelled reduced datasets are used to train a classifier and obtain a classification model with a decision rule described in section 2-2-3. The model is used on both $D_{Red.train}$ and $D_{Red.test}$ to predict a class for each data point. Since the true labels of the data points are known, the prediction can be compared to the ground truth. The training and test error percentages are the percentages of mismatches between predicted and true classes for the training- and test datasets, respectively. A lower error means a better classification performance.

3-3 Data-Driven Soft Discriminant Maps

As discussed in section 3-1, SDM has a parameter β that has to be set manually. We can immediately raise a few questions: to what value should this β be set to achieve maximal classification performance? How sensitive is this performance to β? Does using SDM with an optimal β result in a better performance than using other dimensionality reduction methods such as PCA and LDA? The following sections will explain our proposed framework of setting β in a way that minimizes the test error on a given dataset. We have named the method DD-SDM. The experiments performed in this thesis aim to answer the other questions.

3-3-1 Bracketing of the minimum

The function to be minimized is the test error, the calculation of which was explained in section 3-2, as a function of β:

$$f(x) = Error(\beta) \tag{3-8}$$

The β needs to be found that minimizes this function, preferably while computing $f(x)$ as few times as possible. Textbook *Numerical Recipes 3rd Edition* from Press et al. [67] provides a suggestion how to approach such a one-dimensional minimization problem. The trick is to iteratively bracket the minimum while making the search interval smaller each iteration. A minimum is bracketed if there is a triplet of points $a < b < c$ where $f(b)$ is smaller than both $f(a)$ and $f(c)$. In this case, we know that the function, if it is smooth, contains a minimum in the interval $[a, c]$.

The first step is to choose an initial interval where we are guaranteed to have an optimum. Press et al. [67] proposes a method to find a suitable interval automatically. For the DD-SDM case, this interval is set manually to be from 0 to a sufficiently high number (chosen in the order of 10^2).Namely, in the experiments performed in this thesis, the optimal β was always lower than 100. However, further research could further improve DD-SDM by implementing the bracketing algorithm from Press et al. [67]. Having access to the initial interval, which contains a minimum, the next step is to find this minimum.

3-3-2 Golden-section search

The following derivation of golden section search is based on the work of Press et al. [67]. Suppose we have a starting triplet (a, b, c) that brackets the minimum. An illustration of this situation is shown in Figure 3-2. A strategy is needed for choosing the next β to evaluate. Suppose that b is a fraction y of the way between a and c:

$$y = \frac{b-a}{c-a} \tag{3-9}$$

And say that the next point to be evaluated x is an additional fraction z beyond y:

$$z = \frac{x-b}{c-a} \tag{3-10}$$

Depending on the value of $f(x)$, the new points that bracket the minimum are either (a, b, x) or (b, x, c) with length relative to the current one $y + z$ or $1 - y$ respectively. If we want to minimize the length of the worst case, these lengths should be set equal. This will result in z to be:

$$y + z = 1 - y \tag{3-11}$$

$$z = 1 - 2y \tag{3-12}$$

From this value for z it can be seen that the new point x is the symmetric point to b in the original interval, meaning that $|b - a|$ is equal to $|x - c|$. If b was chosen in the same optimal way as x, *scale similarity* would imply that x should be the same fraction of the way from b to c as b was from a to c, or:

$$\frac{z}{1-y} = y \tag{3-13}$$

When we substitute equation 3-10 into Equation 3-13, the following quadratic equation emerges:

$$y^2 - 3y + 1 = 0 \tag{3-14}$$

Which has solutions:

$$y_1 = \frac{3-\sqrt{5}}{2} = 0.3819... \tag{3-15}$$

$$y_2 = \frac{3+\sqrt{5}}{2} = 2.6180... \tag{3-16}$$

We discard the second solution since it lies outside the interval from 0 to 1 and get $y = 0.3819$. As an example, when the initial underbound $a = 0$ and the upperbound $c = 1$, b should be set at 0.3819 and x at $1 - 0.3819 = 0.6180$. These fractions are those of the so-called *golden section*, a ratio occurring many times in nature, architecture and graphic design among other areas. This is the reason why the method is referred to as *golden section search*. The next point in the iteration can always be found by going a fraction of 0.38197 into the larger of the two intervals (measuring from the central point of the current triplet).

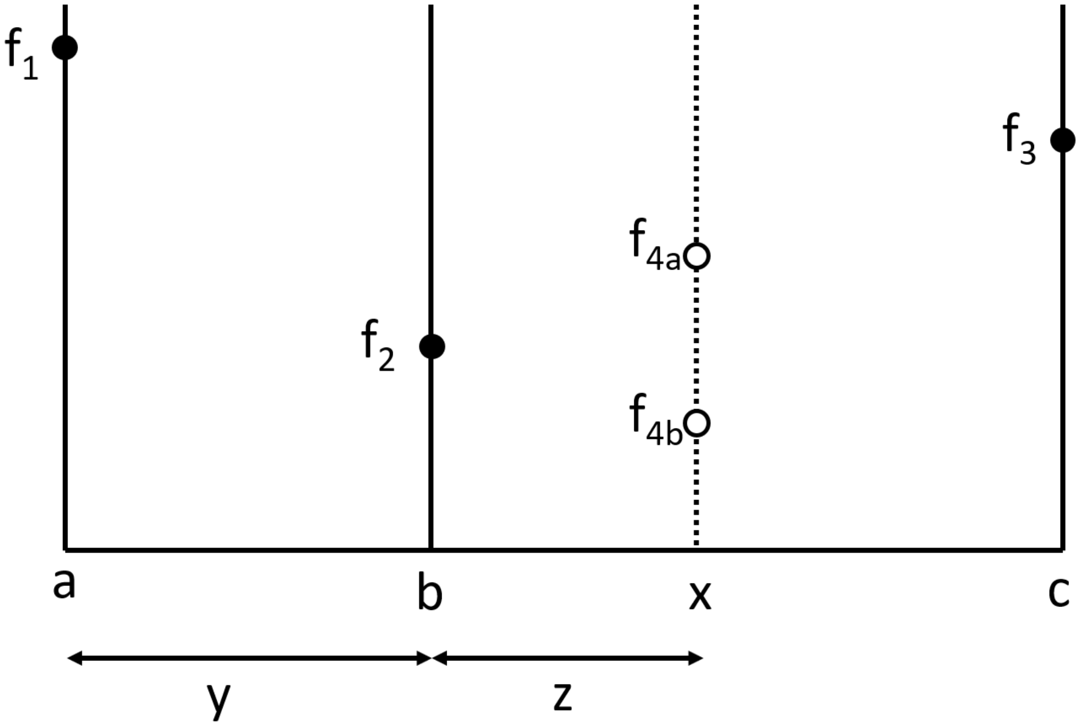


Figure 3-2: Schematic of golden-section search Schematic representation of one iteration of golden-section search. The function-value to be minimized lies on the y-axis and the argument on the x-axis. The method assumes the minimum lies between outer bounds a and c. When the function evaluated on x is higher than f_2; the new outer bounds are set to be a and x. Conversely, when f_4 is lower than f_2, the next outer bounds are set to be b and c.

Next,the value at x is evaluated. When the function value is higher than that of middle point of the current triplet (in this example point f_2), the case with point f_{4a} in Figure 3-2, the new three bracket points become (a, b, x). When the value is lower than point f_2, the new bracket points become (b, x, c). This procedure is repeated until the distance between the new point and the previous middle point is smaller than a pre-defined tolerance ϵ. Press et al. [67] show that a practical tolerance is $\sqrt{\psi}$ where ψ is the floating point precision of the computer used for the analysis. In MATLAB, the double-precision accuracy is in the order of 10^{-15}, resulting in $\epsilon = \sqrt{10^{-15}} = 3 \times 10^{-8}$. Setting ϵ smaller than this value will not achieve better results.

3-3-3 Alternative search methods

Note that golden section search is optimal for the worst case. It possible to trade off a worse worst-case for a quicker convergence to the optimal β. The method that achieves this is based on assuming that the function behaves parabolically near the minimum and is called Brent's method [67]. However, Press et al. [67] note the following:"*If your function has a discontinuous second (or lower) derivative, then the parabolic interpolations of Brents method are of no advantage,and you might wish to use the simplest form of golden section search, as described in paragraph 10.2.*". It can not be guaranteed that the test error function has a continuous first and/or second derivative since the error is highly dependent on random sampling of the full dataset. The safer choice is to use the golden section search method. Future work could apply Brent's method to validate this reasoning. Note that we have no access to the derivative of the test error function, removing the option of using derivative-based search methods.

3-3-4 Implementation

DD-SDM is, in essence, golden section search using the test error as the evaluation function. Pseudocode for DD-SDM is given in Algorithm 1. Our version of the golden section search algorithm takes the following inputs: the full labelled dataset D_{Full} , the initial lower bound a and upper bound c, and the desired accuracy ϵ as a stopping criterion. Every iteration, the considered interval $(c - a)$ becomes smaller. When this value is smaller than ϵ, the algorithm stops and returns the current β corresponding to the lowest test error. This β can be used to calculate the final SDM mapping. As earlier stated, the stopping accuracy ϵ has a theoretical limit of around 3×10^{-8}. Fortunately, experiment two of this thesis seems to indicate that such precision of β is not necessary, and an ϵ of 0.1 will suffice.

The evaluation function denoted as *getSDMTestError()* in Algorithm 1 computes a test error for a specific β. The procedure to obtain the test error is described in section 3-2 and the pseudocode is shown in Algorithm 2.

The test error value is stochastic since the error depends on the randomly chosen data points in the test and train datasets. Furthermore, the data itself can be modelled from a random distribution, making the problem non-deterministic. For this reason, the calculation of the test error is repeated several times for every β with different randomly generated partitions of the full dataset; D_{Train} and D_{Test}. The input *Folds* sets the number of repetitions. In this thesis, the number of folds was set to 5. The mean of these errors is returned as the test error. More work could be done on setting this value "optimally".

Another input to the algorithm is the relative size of the training dataset to the test dataset. This parameter's influence is analyzed in the experiment described in section 4-2-2. The input K is the number of soft discriminants SDM reduces the dataset to. An analysis of K's effect is performed in the experiment described in section 4-2-1.

The test error is also dependent on the classifier trained on $D_{red.train}$. In this thesis, the linear classifier Linear Bayes Normal is used. Since the error has to be computed multiple times, a linear method's relatively low computational cost is beneficial. A linear classifier is expected to give a good enough measure for the amount of overfitting that has occurred. Future work could perform a more rigorous analysis of the classifier's choice effect on the optimal β by weighing the classifier's computational cost against the increased quality of the returned optimal β and robustness of the algorithm to getting stuck in local minimums.

Algorithm 1: Data-Driven Soft Discriminant Map

```
Data:
$D_{Full}$ The full labeled dataset
$a$ Initial underbound
$c$ Initial upperbound
$\epsilon$ stopping criterion: accuracy
Result: $\hat{\beta}$ Optimal $\beta$
begin
    $\tau \longleftarrow \frac{\sqrt{5}-1}{2}$
    $\beta_1 \longleftarrow a + (1-\tau)(c-a)$
    $\beta_2 \longleftarrow a + \tau(c-a)$
    $Error_1 \longleftarrow$getSDMTestError($\beta_1$)
    $Error_2 \longleftarrow$getSDMTestError($\beta_2$)
    while $|c-a| > \epsilon$ do
        if $Error_1 < Error_2$ then
            $c \longleftarrow \beta_2$
            $\beta_2 \longleftarrow \beta_1$
            $Error_2 \longleftarrow Error_1$
            $\beta_1 \longleftarrow a + (1-\tau)(c-a)$
            $Error_1 \longleftarrow$getSDMTestError($\beta_1$)
        else if $Error_1 > Error_2$ then
            $a \longleftarrow \beta_1$
            $\beta_1 \longleftarrow \beta_2$
            $Error_1 \longleftarrow Error_2$
            $\beta_2 \longleftarrow a + \tau(c-a)$
            $Error_2 \longleftarrow$getSDMTestError($\beta_1$)
    if $Error_1 < Error_2$ then
        $\hat{\beta} \longleftarrow \beta_1$
    else if $Error_1 > Error_2$ then
        $\hat{\beta} \longleftarrow \beta_2$
```

Algorithm 2: GetSDMTestError

```
Data:
$D_{full}$ The full labeled dataset
$\beta$ The $\beta$ to be evaluated
$N_1/N_2$ Relative size training- and test dataset
$Folds$ Number of repetitions
$K$ Number of dimensions in reduced dataset
$C$ Classifier used
Result: $Error$ : The mean test error
begin
    for $i \longleftarrow 1 : Folds$ do
        [$D_{train}$,$D_{test}$] $\longleftarrow$ splitDataset($D_{full}, N_1/N_2$)
        $\Omega \longleftarrow$ applySDM($D_{train}$, $\beta$, $K$)
        $D_{Red.train} \longleftarrow D_{train}\Omega$
        $D_{Red.test} \longleftarrow D_{test}\Omega$
        $DecisionRule$ = trainClassifier($D_{Red.train}$,C)
        $TestErrorArray(i) \longleftarrow$ applyDecisionRule($D_{Red.test}, DecisionRule$)
    $Error \longleftarrow$ mean($TestErrorArray$)
```

Chapter 4

Experiments

This chapter will describe all performed experiments along with the datasets used. The next chapter, Chapter 5, will present and interpret the results. First, all used datasets are described in section 4-1 and in section 4-2 all experiments are defined and motivated.

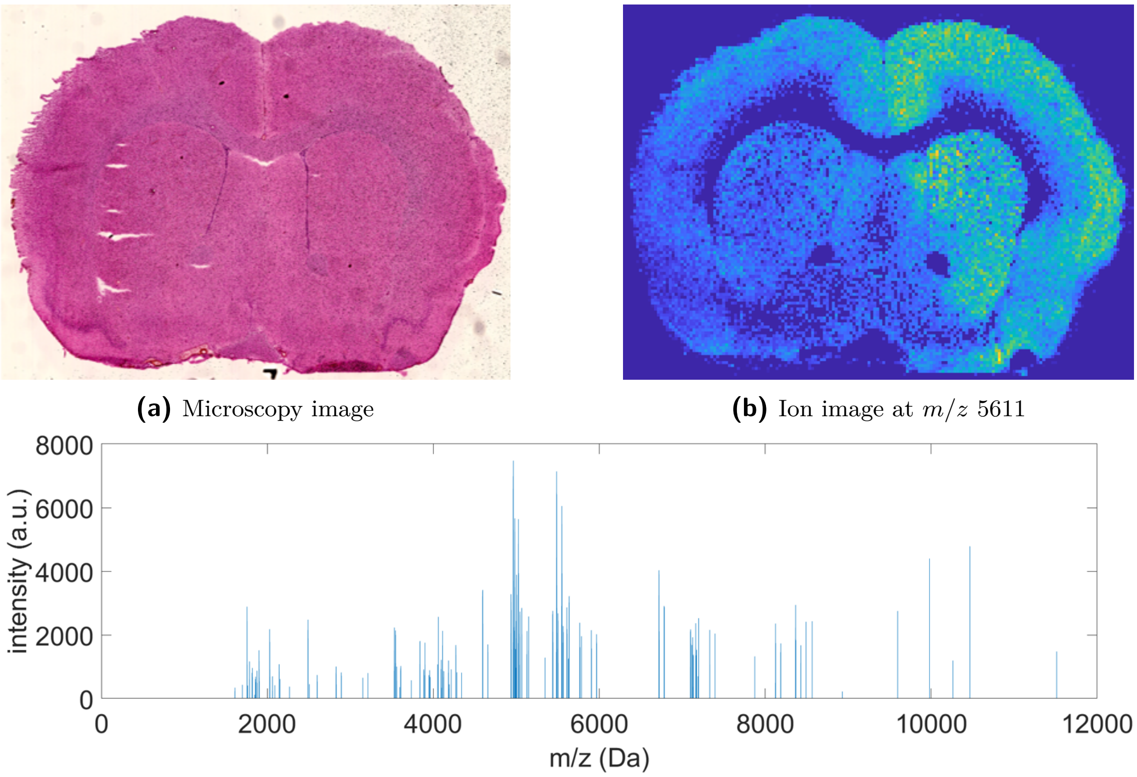
4-1 Datasets

Three different datasets are used throughout the thesis: two real-world datasets and one artificial dataset. This section describes the inherent characteristics of the datasets and the way the datasets were generated.

4-1-1 IMS-RBDS

One of the real-world datasets considered in this thesis will be the Imaging Mass Spectrometry Rat brain Dataset (IMS-RBDS). The dataset originates from an Imaging Mass Spectrometry (IMS) experiment measuring the molecular contents from a rat brain tissue section. A microscopy image of this slice is shown in Figure 4-1a.

Scientists have aimed to simulate Parkinson's disease by depriving the rat's dopamine receptors in one hemisphere of the brain. The other hemisphere is untouched and acts as a control. After the brain was harvested from the rat, the tissue was frozen and sectioned into 10 μm sections. Matrix-Assisted Laser Desorption Ionization (MALDI) is used as the ionization method, and a 15T Fourier transform ion cyclotron resonance (FT-ICR) mass analyzer performed the mass spectral measurements. Verbeeck et al. [45] give a comprehensive description of the procedure. The acquired m/z-range is set from 1300 to 24000 with approximately 21000 pixels. The full-spectrum dataset is peak-picked after the measurements are taken, resulting in a dataset with 809 m/z-peaks. The mass spectrum of pixel 4500 is shown in Figure 4-1c and the ion image corresponding to $m/z = 5611$ is shown in Figure 4-1b.



(a) Microscopy image

(b) Ion image at m/z 5611

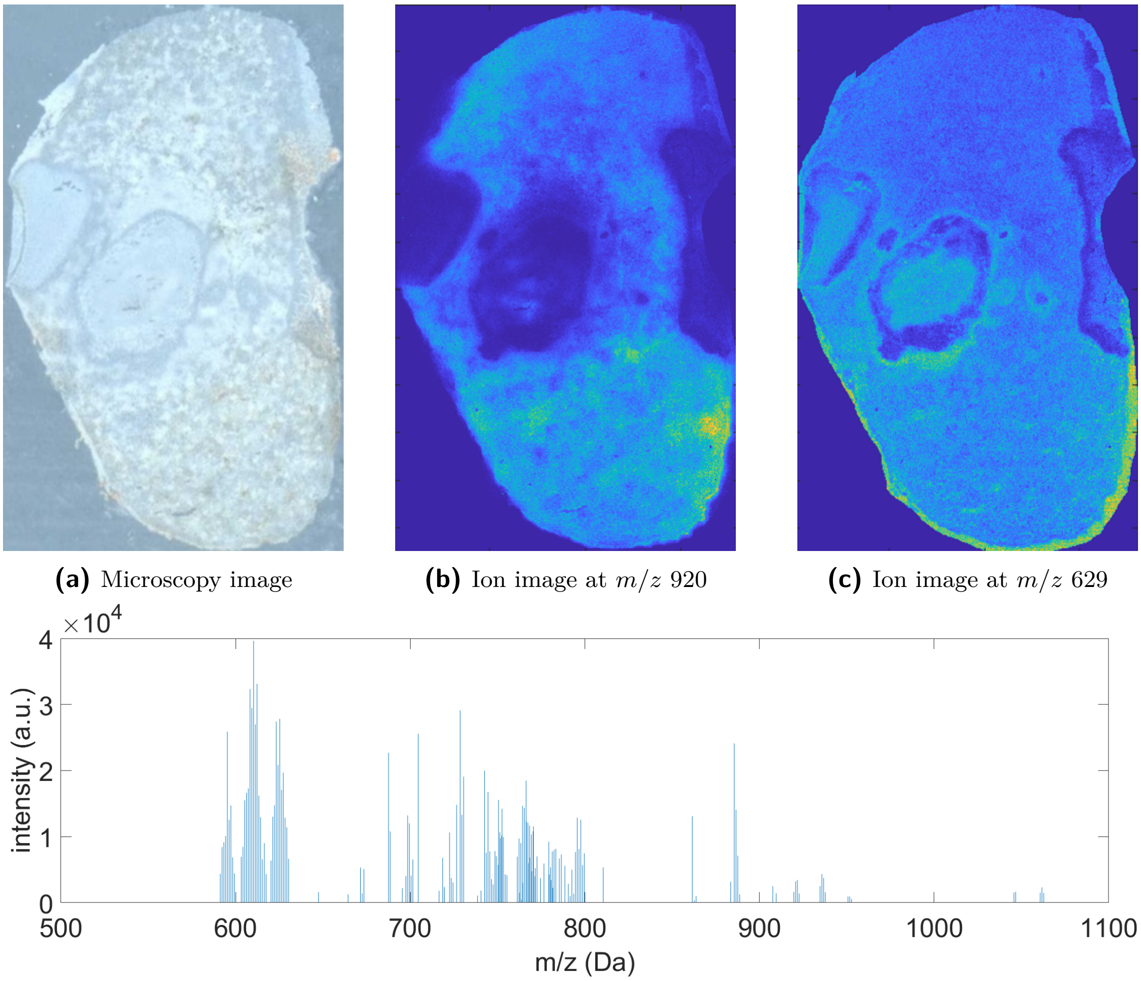
(c) Mass spectrum of pixel 4500

Figure 4-1: Visualizations of the rat brain dataset

4-1-2 IMS-MKDS

The second real-world dataset describes the kidney of a mouse. The mouse is infected with *Staphylococcus aureus*, a common bacteria that causes a diverse array of diseases such as skin infection, pneumonia, and meningitis. The bacteria can form tissue abscesses, which visually represent the immune response of the mouse. By applying IMS on this tissue, the goal is to characterize molecularly the host-pathogen interactions present in the abscesses. The dataset was acquired by William Perry at the Vanderbilt University located in Nashville, Tennessee under the supervision of Eric Skaar, Jeffrey Spraggins and Richard Caprioli

The mouse was 6-8 weeks old and euthanized 96 hours post-infection. The organs were snap-frozen in liquid nitrogen, cryosectioned in 12μm thick sections, and thaw mounted on a glass slide. A microscopy image of the tissue is shown in Figure 4-2a. The matrix 1,5-diaminonaphthalene was applied using a TM-Sprayer. The ionization process is MALDI and the m/z-intensities are measured using a prototype timsToF fleX mass spectrometer (Bruker Daltonik, Brement, Germany)[68]. The pixel size is 25 μm and the range of measured m/z-values is set to 400-1400 Da. The dataset consists of a total of approximately 150000 pixels. The dataset considered in this thesis is pre-processed with a peak-picking procedure resulting in 152 distinct m/z-peaks. Example ion images corresponding to m/z-values 920 and 629 are shown in Figures 4-2b and 4-2c respectively. A mass spectrum of a pixel 80000 of this dataset is shown in 4-2d.



(a) Microscopy image

(b) Ion image at m/z 920

(c) Ion image at m/z 629

(d) Mass spectrum of pixel 80000

Figure 4-2: Visualizations of the mouse kidney dataset

4-1-3 Artificial dataset

A drawback of using real-world datasets is that it is unknown *a priori* which m/z-values contain biological patterns and which values carry technical variation, noise, or are otherwise irrelevant to the task at hand.

In this thesis, an artificial dataset is constructed to understand the ability of a linear feature extraction method to filter out the m/z-values that are distinctive for class differences and discard noise. By constructing the 'biological' patterns ourselves in this dataset, it is known beforehand which m/z-values should definitely be kept and which should be discarded in the newly constructed feature-space. To quantify this ability, we propose a new performance metric called Peak Conservation Score (PCS), defined in section 4-2-4.

Spatial class distribution The artificial dataset is created to have similar characteristics to that of a real-world IMS dataset. However, accurately modelling IMS-datasets from first principles is beyond the scope of this thesis, and therefore no quantitative conclusions are drawn from the experiments performed on this dataset. However, by comparing the performance of Soft Discriminant Map (SDM) to Principal Component Analysis (PCA) in terms of PCS, conclusions can be made about the difference general behaviour of the methods. The parameters of the artificial dataset can be varied to study the effect on the PCS, such as the number of training data points, number of classes and dimensionality.

The human-specified spatial layout of our artificial dataset is shown in Figure 4-3. The dataset consists of data points divided into two classes: class 1 and class 2. Each square in the figure represents a pixel of a real-world dataset, each containing a feature vector with m/z measurements.

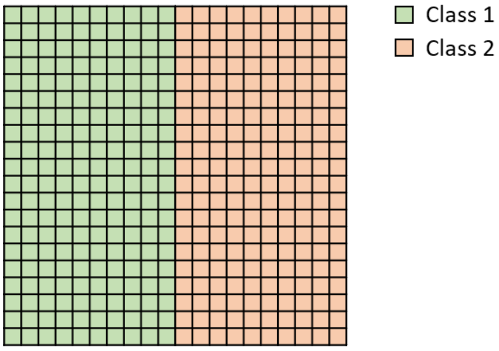


Figure 4-3: Spatial class distribution of the artificial dataset The artificial dataset consists of two classes: class 1 and class 2. Class 1 is located at the left of the dataset, class 2 is located at the right.

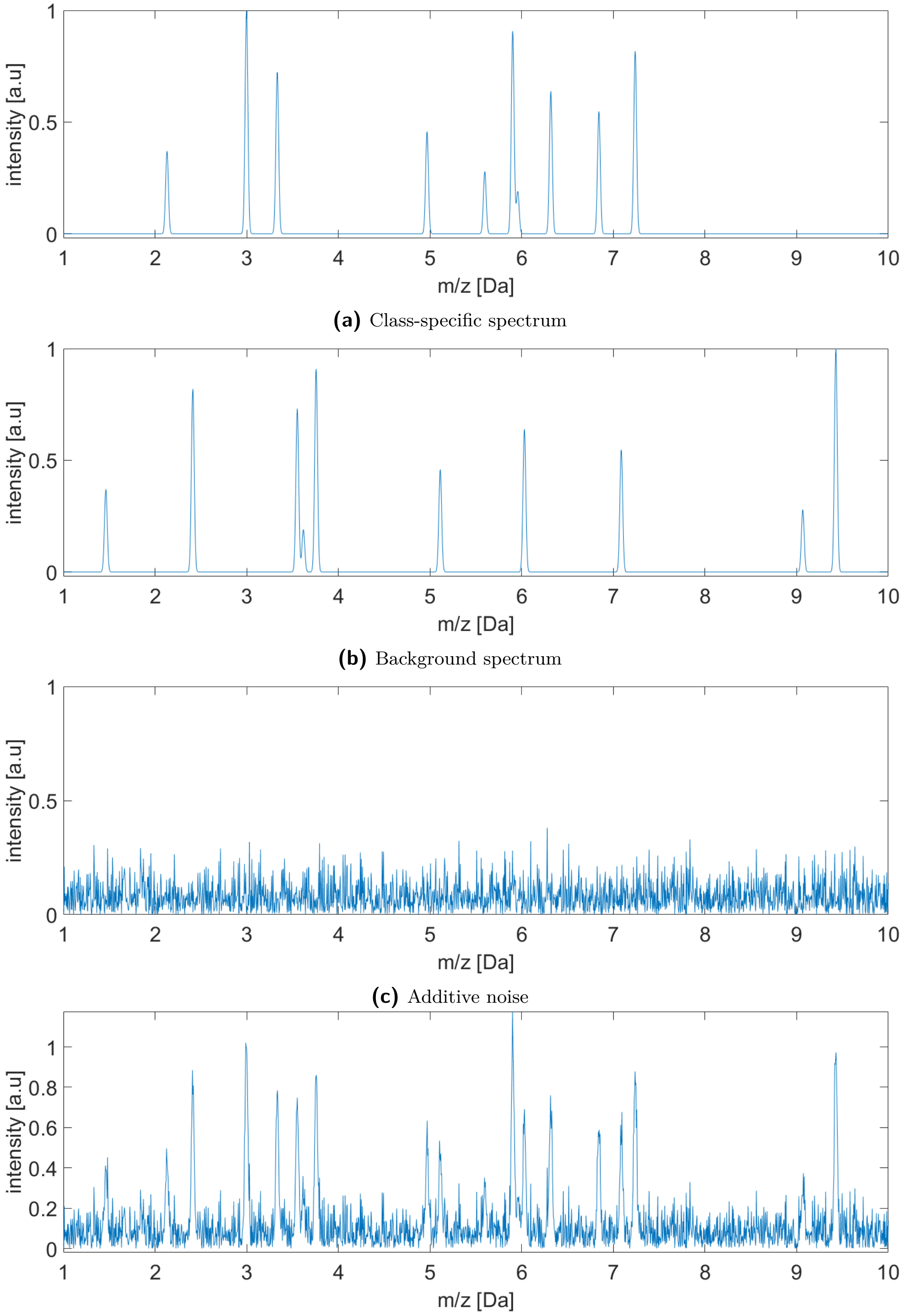
Full spectrum generation The features corresponding to each data point are constructed in a way that resembles the intensity measurements of distinct m/z-values corresponding to each pixel in an IMS dataset. This section will describe the process used to construct the full artificial mass spectra of each data point.Each full spectrum is build up as a summation of several mass spectra as seen in Figure 4-4. This approach is based on the work of Verbeeck [69].

The first spectrum is a *class-specific spectrum* that models the biological information that represents a class, for example the molecular characteristics of an abscess in the Imaging Mass Spectrometry Mouse Kidney Dataset (IMS-MKDS), in a real-world dataset. Ideally, a dimensionality reduction technique should conserve the features resulting from this spectrum. Figure 4-4a shows an example of a class-specific spectrum. These spectra will be denoted as $P_{\omega 1}$ and $P_{\omega 2}$ for class 1 and class 2 respectively.

First, a spectral axis is defined between Mass-over-charge ratio (m/z) 1 and 10. The profile consists of a series of synthetic ion peaks, modelled as Gaussian probability density curves, the means of which are uniformly randomly distributed across the m/z-range. In experiment 4, described in section 4-2-4, a standard deviation of 0.015 is used. The profiles are normalized to be within the intensity interval [0,1]. The amplitude of the resulting profiles are distributed between [0.5-1] to simulate a spread of intensities.

The second spectrum used in the summation is called a *background spectrum.* This spectrum will be the same for both classes. Dimensionality reduction methods should ideally discard the features resulting from this spectrum since they are not indicative for the class differences. Figure 4-4b shows a background spectrum and will be denoted as P_b. The background spectrum is generated in the same way as the class specific spectrum: normalized, Gaussian probability density functions with a standard deviation of 0.015 and uniformly distributed means between 1 and 10.

To simulate uncertainties in the datasets such as measurement noise, a noise profile P_n is added. The noise is generated from a Gaussian probability distribution with mean 0 and standard deviation 0.1. The absolute value is taken to prevent negative intensities, which is physically impossible in mass spectral measurements. Since dimensionality reduction methods that are robust against noise are naturally preferred, the features resulting from this profile should ideally be discarded.



(a) Class-specific spectrum

(b) Background spectrum

(c) Additive noise

(d) Total spectrum

Figure 4-4: Composition of the mass spectrum of a data point in the artificial dataset A mass spectrum from a data point in the artificial dataset consists of the addition of a class-specific, background, and noise spectrum.

As a final step, spatial variance in intensity between data points is modeled by multiplying the class-specific and background spectra with a factor denoted as $a_{\omega}^{i,j}$ and $a_{b}^{i,j}$ respectively. The superscript i,j denotes the pixel's index and is not an exponent. The factor is named pixel-variation factor.

For each pixel, two pixel-variation factors are generated, $a_{\omega}^{i,j}$ and $a_{b}^{i,j}$, from two distinct uniform probability distributions. Both $a_{\omega 1}^{i,j}$ and $a_{\omega 2}^{i,j}$ are drawn between 0 and 1. The background pixel-variation factor $a_{b}^{i,j}$ is drawn between 0 and 5. The upper-bound of $a_{\omega 1}^{i,j}$ and $a_{\omega 2}^{i,j}$ will be varied between 0 and 5 in experiment 4 (section 4-2-4).

The full spectrum $P_{i,j}$ of pixel i,j as shown in Figure 4-4d for class 1 can be denoted as follows:

$$P_{i,j} = a_{\omega 1}^{i,j} P_{\omega 1} + a_{b}^{i,j} P_{b} + P_{n} \tag{4-1}$$

Peak-picking The real-world datasets described earlier are pre-processed with a peak-picking procedure. A peak picker is an algorithm that detects the m/z values with a peak in a single full-spectrum mass spectrum (such as the spectrum in Figure 4-4d). The artificial dataset was also peak-picked to mimic the real-world datasets.

Since an IMS dataset consists of a vast amount of spectra (one for every pixel) and the peak-picker works on a single spectrum, a representative spectrum has to be produced. Multiple strategies to accomplish this are used in literature, such as taking the mean, sum and maximum of all spectra. From these options, the maximum spectrum managed to identify the most peaks corresponding to class-specific and background peaks and is therefor used in the experiments.

The MATLAB function *mspeaks* was used with the default settings to perform the peak-picking. An example of the result of a peak-picking procedure is shown in Figure 4-5. The blue line represents the maximum spectrum retrieved from the spectra of all data points in the artificial dataset. At the m/z-value of the vertical orange lines, the peak-picker identified a peak. Using the list of m/z-values returned from the peak-picker, a peak-picked dataset is constructed by retrieving the intensities according to these m/z values from the spectrum of every pixel.

Constructing the ground truth In the full spectrum, it is known which Gaussian density function corresponds to which class. We developed a method to retain this information through the peak-picking procedure.

Each Gaussian density function corresponds to either class 1, class 2 or the background spectrum. They all have a mean μ and standard deviation σ . For each peak in a full-spectrum,z an interval is defined referred to as the "hitwindow" corresponding to that Gaussian:

$$[\mu - \sigma; \mu + \sigma]$$

If a picked peak lies within one standard deviation of the mean, the peak is labelled corresponding to that profile. In Figure 4-5, the hitwindows are visualized by arrows. The bottom Figure shows a zoomed-in version of the top Figure. When an orange vertical line lies between two arrows, the peak is labelled according to the corresponding profile. Note that it is possible that a picked m/z-value hits multiple hitwindows and therefore corresponds to multiple classes. To keep the number of distinct classes small, all such cases are given the label ω_z. All possible labels are summarized in Table 4-1.

Table 4-1: All possible labels of the profile-peaks with their description

Label	Description
ω_1	the m/z value corresponds to class 1.
ω_2	the m/z value corresponds to class 2.
ω_b	the m/z value corresponds to the background profile.
ω_z	the m/z value corresponds to multiple profiles.
ω_n	the m/z value does not correspond to any class.

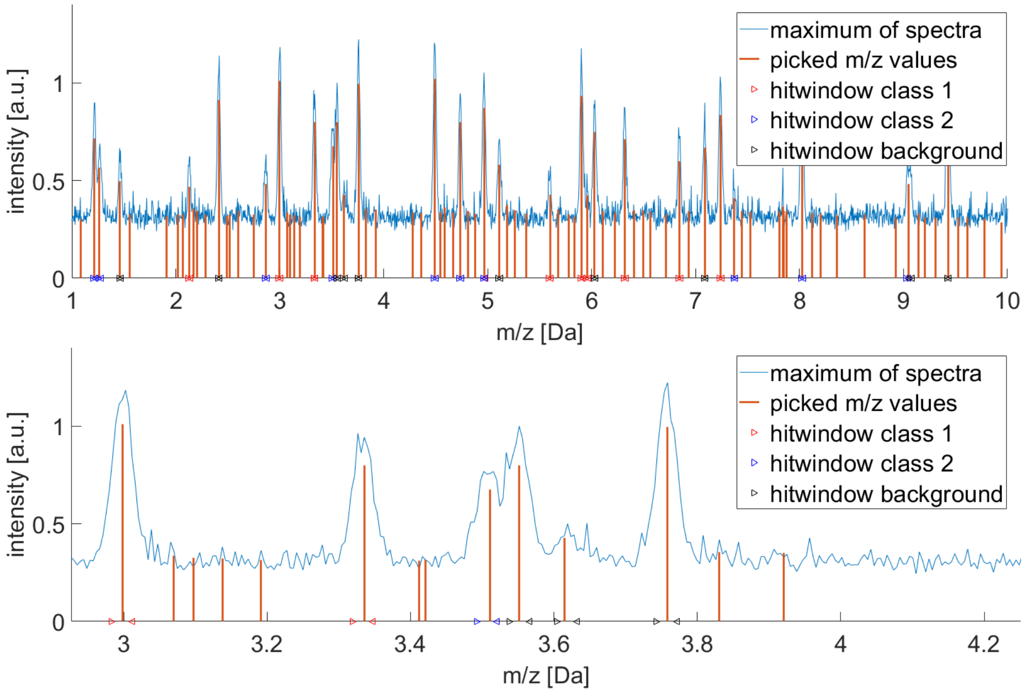


Figure 4-5: Peak-picking procedure and labelling for the artificial dataset A peak picking algorithm is applied on the maximum of the full-spectrum artificial dataset.Afterwards, a labelling procedure assign each peak to a class when the picked m/z value lies within one standard deviation of the means of the original Gaussian peak. This interval is named a *hitwindow*. The second row is a zoomed-in version of the first row to make the hitwindows better visible.

The peak picker succeeds in finding the peaks represented by the Gaussian distributions. The picker also picks m/z-values generated by the noise. The final dataset is an artificial dataset which is peak-picked and similar to the real-world datasets. All m/z-values that remain after the peak-picking procedure have a label indicating whether the peak corresponds to a 'biological pattern' that is class-specific, part of the common background profile, or noise. Using this information, SDM can be evaluated in its ability to conserve the class-specific m/z-values.

4-2 Experiment descriptions

The main research goals of this thesis can be stated as four main questions:

1. When applied to a real-world IMS dataset, how does the classification performance after applying Soft Discriminant Map (SDM) compare to after applying Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA)?

2. What is the sensitivity of the classification performance to the value of the parameter β in SDM?
3. How can β be set in a automated, data-driven way that maximizes classification performance for a given dataset? How efficient is golden section search compared to grid search?
4. Is SDM able to conserve features that are class-specific and discard features that have similar values between classes?

These questions are examined using the experiments defined in this section. In the following subsections, the research question to be answered, the motivation for performing this experiment, and the experiment design are stated.

4-2-1 Experiment 1: Classification performance

As described in chapter 2-3, applying Dimensionality Reduction (DR) is not only a way to reduce the computational costs in building a classifier. The classification performance can also be improved by reducing the "curse of dimensionality" as discussed in section 2-3-1-1. Furthermore, overfitting can also be diminished by reducing the dataset's dimensionality, resulting in higher accuracy. This experiment aims to compare SDM to benchmark DR methods PCA and LDA.

The research question that this experiment aims to answer is " *When applied to a real-world IMS dataset, how does the classification performance after applying SDM compare to after applying PCA or LDA?*"

The classification performance will be quantified using the training error and test error, which are obtained as described in section 3-2. Primarily the test error is a useful metric for ensuring the classifier learned the underlying biological patterns. The test error is compared to the training error since a big difference between the two indicates that the model was prone to overfitting (as discussed in section 2-3-1-3).

The dimensionality reduction methods will reduce the number of features in the full dataset $D_{Full} \in \mathbb{R}^{N \times M}$ into a reduced set of features in the reduced dataset $D_{Red} \in \mathbb{R}^{N \times K}$. By varying the number of variables K in the reduced dataset and calculating corresponding error rates, the DR methods are compared in the number of features needed for the corresponding error rates.

It is expected that SDM is less prone to overfitting than LDA when the dimensionality is high. By minimizing within-class variance by LDA, the generalization capability decreases[9] caused by overfitting. Since SDM uses label information and PCA does not, we hypothesize that SDM needs fewer features (a lower value for K) to reach its optimal classification performance.

Three versions of SDM with different values for β are considered. First, Data-Driven Soft Discriminant Map (DD-SDM) is used to find the optimal value for β. Then the other β values are taken lower and higher than this optimal value.

The experiment parameters are shown in Table 4-2. Note that only one classifier type is considered. It would make sense to evaluate the effect of different classifiers in follow-up research.

Table 4-2: Parameters of experiment 1

Parameters	Experiment 1
dataset	IMS-RBDS
number of classes	2
dimensionality M	809
classifier	K-Nearest Neighbours
observations in trainingset N_1	800

4-2-2 Experiment 2: Sensitivity of β with regard to classification performance

The research question that this experiment aims to answer is as follows: *What is the sensitivity of the classification performance to the value of the parameter β in SDM?*

By tuning β in SDM, the relative importance of within-class variance against between-class distance can be controlled to reduce overfitting and improve classification performance. A low β (close to zero) maximizes the distance between class-means. A high β minimizes the variance of data points within a class in the reduced dataset D_{red}.

Since this parameter β needs to be set beforehand, the effect of a change in this parameter should be investigated. In the previous experiment, only three values are considered for β. In the current experiment, β will be varied for a wide range of β from 0 to 100. This experiment aims to interpret the consequence of varying β to the classification performance, the two-dimensional class distributions of the data points in $D_{red} \in \mathbb{R}^{N \times 2}$ and ion images when used on an IMS-dataset.

The results are visualized in multiple ways. The relation between β and the training and test errors is plotted, and a latent variable false-colour image is generated by scoring all pixels in the new feature space. These images show areas within the tissue that have similar values in the reduced feature space. Hopefully, different classes, and therefor the different abscesses in the mouse kidney tissue, have distinct values. Lastly, the data points of the reduced dataset are plotted in a scatter plot. Together these visualizations aim to show the consequence of setting β on the training- and test error rate as described in section 3-2.

We consider two cases, a high dimensional dataset is used in experiment 2a and a low dimensional dataset is considered in experiment 2b. Experiment 2a represents the case where the number of data dimensions M of the training dataset $D_{train} \in \mathbb{R}^{N_1 \times M}$ is higher than the number of data points N_1, so the case where $M > N$. As explained in section 2-3-3-3, the risk of overfitting is high when β is set at a high value to minimize within-class variance. In this setting, the within class-variance can be reduced to an artificially low value value which causes the projected training dataset to be unrepresentative for the projected data points in the test dataset[9]. Therefore, the hypothesis for this experiment is that this feature space will not generalize.

In experiment 2b, the number of training data points N_1 is higher than the number of features M. This case will be referred to as the $M < N$ case. In this case, the risk of overfitting is lower since the feature space is more densely filled with data points as explained in 2-3-1-1. This experiment aims to research the behaviour of SDM in this data setting. It is expected that setting a high value for β no longer necessarily results in overfitting. This case was not considered in the original paper from Liu and Gillies [9].

The IMS-MKDS is used, containing 152 picked peaks. Section 5-2 explains the process of constructing the labels for this dataset. Two variants of the datasets are used, the first one having 145 training data points, the second one having 1000 data points, as shown in Table 4-3. The data points for each training dataset D_{Train} are are randomly drawn without replacement from the full dataset D_full in such a way that each class has an equal number of data points. The linear Bayes normal classifier is used to compute the error rates.

Table 4-3: Parameters of experiment 2

Parameters	Experiment 2a	Experiment 2b
dataset	IMS-MKDS	IMS-MKDS
number of classes	3	3
dimensionality	152	152
classifier	Linear Bayes Normal	Linear Bayes Normal
observations in trainingset	**145**	**1000**

4-2-3 Experiment 3: Data-Driven Soft Discriminant Maps Search Strategies

The research question that this experiment aims to answer is as follows: *How efficient is golden section search compared to grid search in the search for the optimal β?*

Section 3-3 describes DD-SDM, a new framework to set β in a way that aims to maximize classification performance. The search strategy used is golden-section search[67]. To evaluate the effectiveness of golden section search, the number of iterations needed to find the optimal β of DD-SDM is compared to a grid search approach. First, the number of iterations by golden section search is determined. The search interval will then be divided into the same number of sections of equal length. The best classification performance of both methods is finally compared to conclude on the improvement golden-section search can accomplish.

Table 4-4 summarizes the parameters used in this experiment. The training dataset $D_{train} \in \mathbb{R}^{N_1 \times M}$ originates from the IMS-RBDS. The number of data points in the training dataset N_1 is 800 and the dimensionality M is 809. The stopping criterion ϵ is set at 0.1. The initial search interval is set from 0 to 100. The number of repetitions of the calculation of the test error per considered β *Folds* is set at 5 as described in section 3-3-4.

Table 4-4: Parameters of experiment 3

Parameters	Experiment 3
dataset	IMS-RBDS
data points in training dataset N_1	800
dimensionality M	809
initial search interval	0-100
classifier	Linear Bayes Normal
folds	5
stop accuracy	0.1
grid search distribution	linearly distributed

4-2-4 Experiment 4: Class-specific feature conservation

The advantage of using a supervised dimensionality reduction method as opposed to an unsupervised approach is the ability to use class information in defining new features that actively incorporate class-specific features of the original dataset.

To validate this statement, the research question that this experiment aims to answer is as follows: *Is SDM able to conserve features that are class-specific and discard features that have similar values between classes in a better way than PCA?*

The hypothesis is that by using the class-labels, SDM would be better capable than methods that do not use these labels such as PCA in creating features that give a high weight to class-specific features. However, with real-world datasets, the ground truth stating which features are 'class-specific' is not available.

By creating an artificial dataset from first principles as explained in section 4-1-3, it becomes possible to construct the ground truth of stating which features are class-specific. The construction of the ground truth was described in section 4-1-3.

Feature-weight plot The features defined by both SDM and PCA are a linear combination of the original features. This linear combination is stored in a transformation matrix Ω as described in section 3-1. The weights of the linear combination of one latent variable can be visualized as a feature-weight plot, an example of which is shown in Figure 4-6. The y-axis indicates for every original data feature the weight in the new feature space for this latent feature. These weights are directly obtained from the columns of the transformation matrix Ω. In addition to the weight, the label of the peaks is also visualized by the color of the marker. Features corresponding to class 1, class 2 and the background are colored red, blue and green respectively. The features not belonging to any class and were peak-picked due to the noise in the artificial dataset are colored white with a blue outline.

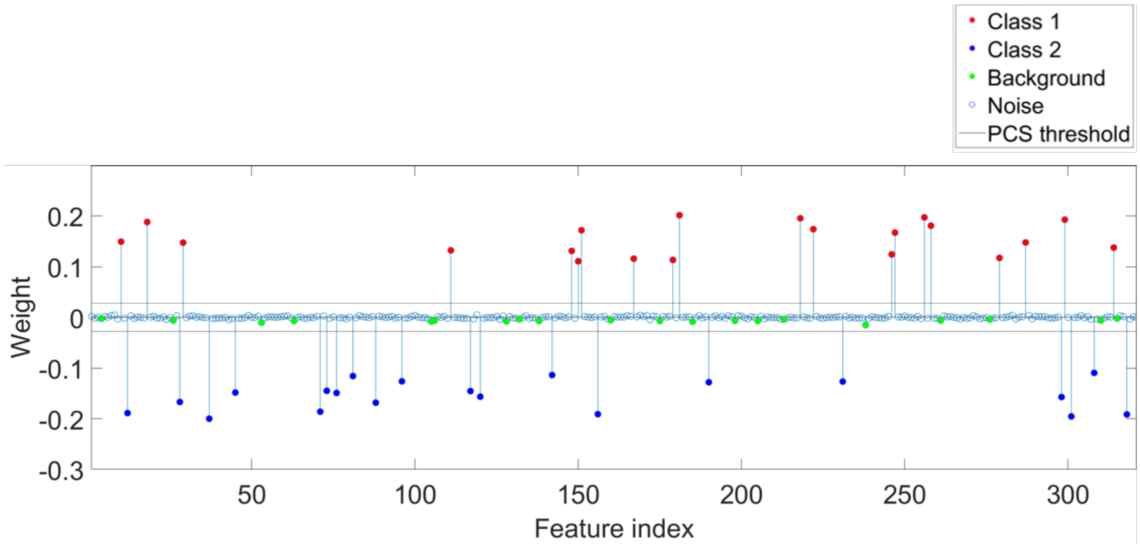


Figure 4-6: Example feature-weight plot constructed with SDM The weights for each feature as stored in transformation matrix Ω. The peaks are labelled corresponding to their origin. Peaks corresponding to class 1 and 2 are shown in red and blue, respectively. Common peaks present in both classes 1 and 2 are shown in green and referred to as background peaks. Peaks picked due to noise are shown in white with a blue outline. Two horizontal lines called the Peak Conservation Threshold are drawn at 0.5σ and -0.5σ.

Peak Conservation Score Ideally, in a classification setting where class labels are known beforehand, a pre-classification DR method should select features that discriminate between the different classes, the class-specific peaks, by giving them a high absolute weight in these graphs. Features that are common between classes, background peaks, should be given a low weight since those are not as useful for the classification process that will need to follow. Besides being able to visualize this as in Figure 4-6, we also want to quantify this goal. To this end, a threshold is defined in the feature weight plot, which is visualized as two horizontal black lines in Figure 4-6. When a feature is given a weight higher than this threshold (either lower than the lower threshold or higher the upper threshold), we consider that the feature has been 'conserved' by the DR method. When the weight is lower than the threshold, we consider the feature to be 'discarded'. We propose the metric Peak Conservation Score (PCS), which is the ratio of 'conserved' peaks with the same peak label:

$$\text{PCS} = \frac{\text{\#peaks above PCS-threshold}}{\text{total \# of peaks}}, \tag{4-2}$$

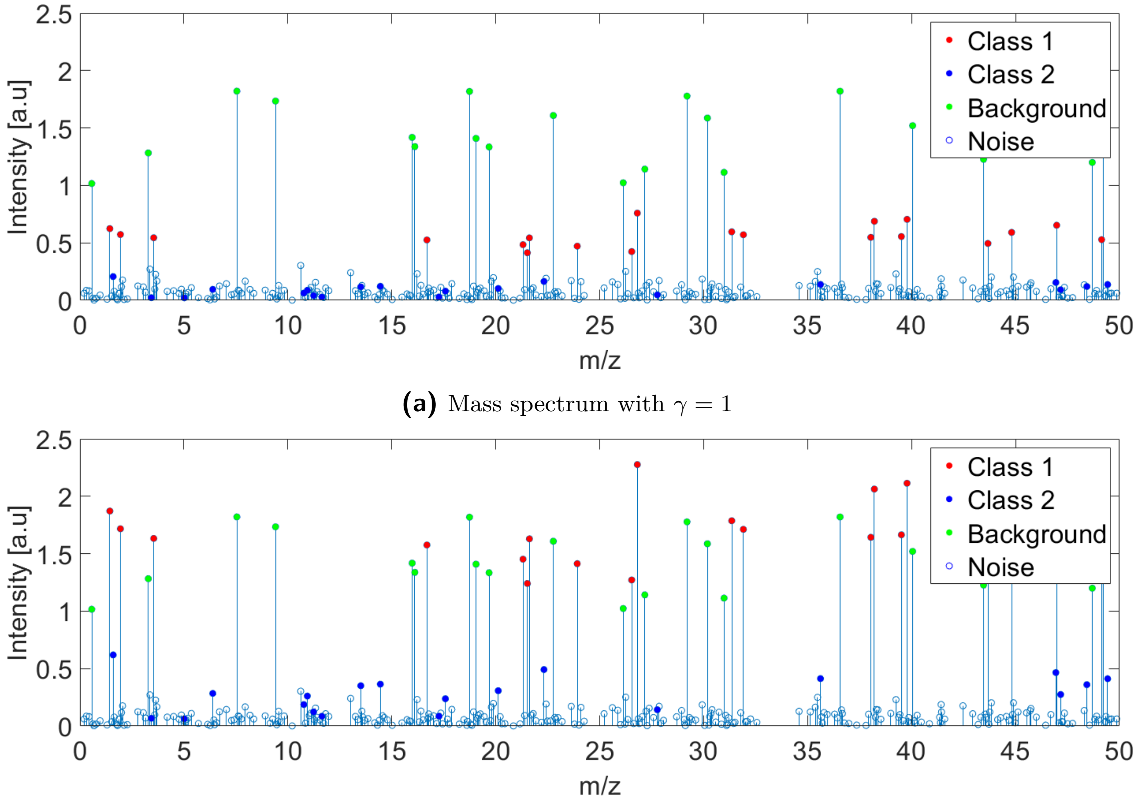
where the numerator is the number of 'conserved' peaks within a peak category and the denominator states the total number of peaks within that category. The standard deviation of σ over all weights is calculated to construct the threshold. The threshold is defined to be minus and positive a half standard deviation $[-0.5\sigma, 0.5\sigma]$ of all intensity values. Note that other values for this threshold are also possible and will effect the PCS. Using this definition, a PCS can be calculated for every peak category. The possible peak categories for this experiment are defined in Table 4-1. Note that the artificial dataset was constructed in such a way that all the Gaussian profiles were picked in the peak-picking procedure and that no picked peak belongs to multiple categories. As an example, the PCS for class 1 can be calculated using Figure 4-6. There are a total of 20 red peaks belonging to class 1. All the peaks lie above the black PCS-threshold. The PCS score for class 1 of the feature computed by SDM is therefore $20/20 = 1$.

Altering the artificial dataset PCA aims to find a linear combination of features that maximize the variance in the constructed latent variables as explained in section 2-3-3-1. When the class-specific peaks are of low intensity and therefor have low absolute variance, we expect PCA to discard these peaks. Since SDM has access to label information, we expect that this method will succeed in conserving class-specific peaks, even when these peaks have a low intensity. To validate this statement, this experiment will show the effect of height of the class-specific peaks on the PCS for both SDM and PCA.

Since an artificial dataset is used, the intensity of the class-specific features can be scaled directly. The characteristics of the artificial dataset are provided in Table 4-5. The procedure of scaling the class-specific peaks is visualized in Figure 4-7. Note that this figure is different to the feature-weight plot. Figure 4-7 shows the mass spectrum of a single data point (representing a pixel), as described in section 2-1-2. The same colors are used as in the feature-weight plot to indicate to which category a peak belongs. As described in section 4-1-3, each pixel has a pixel-variation factor a which the class-specific and background peaks get multiplied with. This value is drawn from a random uniform distribution with range [0-γ]. For the background peaks, γ is always set at 5. The value of γ for the class-specific peaks will be varied from 0 to 5. Consequently, when γ is set to 5 for the class-specific peaks, it has the same uniform distribution as the background peaks. Note that the seed used in the pseudo-random generator is set to be the same to ensure that a pixel retains the original drawn pixel-variation for each considered γ.

As an example, Figure 4-7-a shows the mass spectrum of pixel 1 with a γ of 1. This means that the pixel-variation factor for the class-specific peaks where drawn from a uniform random distribution with bounds of [0-1]. Figure 4-7-b show the same pixel with $\gamma = 3$. The only difference between the figures is the scaling of the class-specific peaks. This is the preferred behaviour since this isolates variation in intensity of the class-specific peaks, and therefor the variance contained in these features, in the different versions of the artificial dataset.

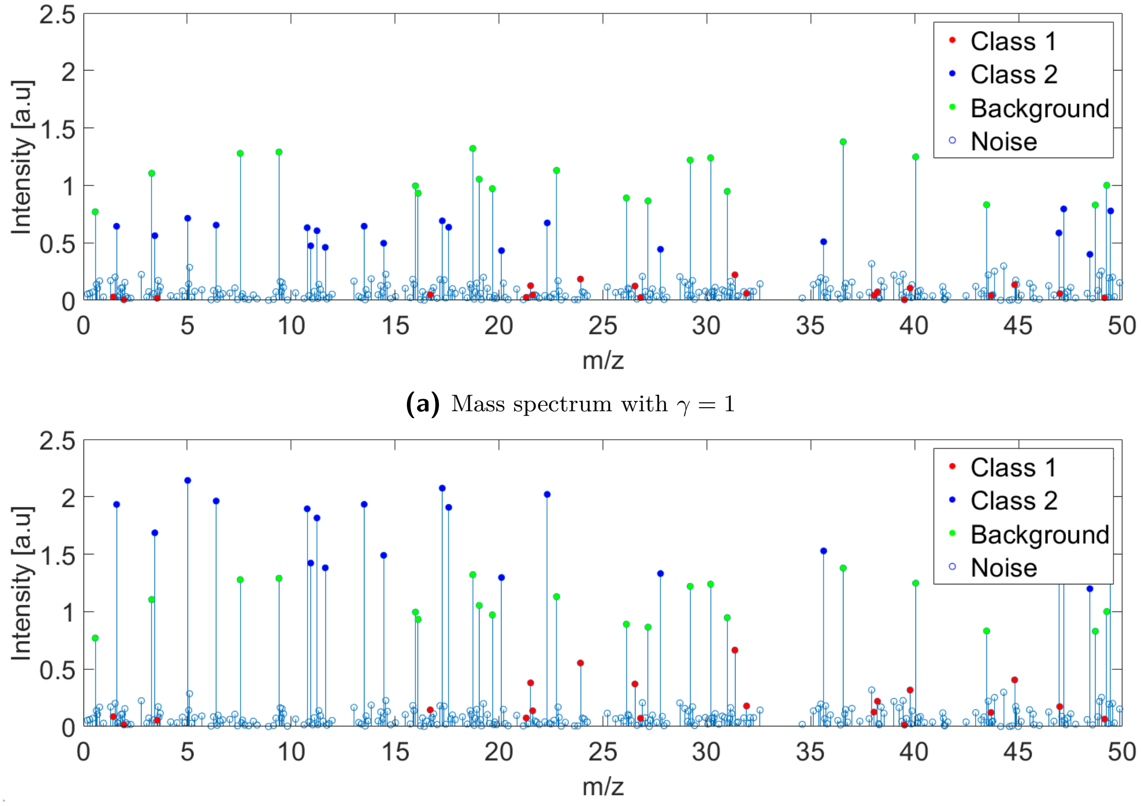
By calculating the PCS for versions of the artificial dataset with different heights of class-specific peaks for both SDM and PCA, this experiment aims to show the ability of SDM to use class information to conserve the class-specific peaks and discard the background peaks better than the unsupervised technique PCA.



(a) Mass spectrum with $\gamma = 1$

(b) Mass spectrum with $\gamma = 3$

Figure 4-7: Mass spectrum of pixel with index 1 belonging to class 1 for different values of γ



(a) Mass spectrum with $\gamma = 1$

(b) Mass spectrum with $\gamma = 3$

Figure 4-8: Mass spectrum of pixel with index 201 belonging to class 2 for different values of γ

Table 4-5: Parameters of experiment 4

Parameters	Experiment 4
dataset	Artificial dataset
m/z range	[0-50]
number of full spectrum m/z bins	5000
number of picked peaks M	321
number of class 1 peaks	20
number of class 2 peaks	20
number of background peaks	20
number of data points in full dataset N	400
number of data points in training dataset N_1	100
number of data points in test dataset N_2	300
noise type	additive Gaussian
noise standard deviation	0.1
pixel variation factor $a_b^{i,j}$ range background peaks	0-5
pixel variation factor $a_\omega^{i,j}$ range class-specific peaks	0-γ
considered γ range	[0-5]

Chapter 5

Discussion and Results

This chapter will state the results of the experiments and interpret them. The order is the same as in the experiments chapter.

5-1 Experiment 1: Classification performance

In the first experiment, the classification performance on Imaging Mass Spectrometry Rat brain Dataset (IMS-RBDS) dataset is considered to answer the following research question: "*When applied to a real-world Imaging Mass Spectrometry (IMS) dataset, how does the classification performance after applying Soft Discriminant Map (SDM) compare to after applying Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA)?*"

Three different Dimensionality Reduction (DR) methods were applied on the dataset to create a reduced feature space:SDM,PCA, and LDA.

The full dataset $D_{Full} \in \mathbb{R}^{N \times M}$ is transformed into a reduced set of features in the reduced dataset $D_{Red} \in \mathbb{R}^{N \times K}$. The dimensionality of the constructed subspace K is varied in this experiment to compare the DR methods in the number of features needed for their corresponding error rates.

In other words, a classifier is trained on reduced datasets D_{red} with varying dimensionality K. The classifier used in this experiment is the same as used by Liu and Gillies [9];K Nearest Neighbours (K-NN). This classifier assigns a test data point to the class of the K nearest training data points in D_{Red}. This parameter K should not be confused with the dimensionality of the reduced dataset. The implementation used is the routine knnc from toolbox PRTools [70], a pattern recognition library created by Ela Pekalska and Robert P.W. Duin, which sets the number of neighbours automatically in a way that maximizes classification performance.

In Figure 5-1 the resulting error curves are shown. The x-axis denotes the dimensionality of the subspace (K) generated by the DR method. The y-axis denotes the achieved error rates, both the training and test errors. Naturally, a low error rate is desirable overall. The methods considered are PCA, LDA and SDM with three different values for β: the β determined with DD-SDM ($\beta = 1.8$), a lower value ($\beta = 0.2$), and a higher value ($\beta = 10$). The training and test errors are obtained as described in section 3-2.

The test errors are denoted with the dotted lines and are considered a good measure for the classification model's generalization capability and they are therefor the errors we want

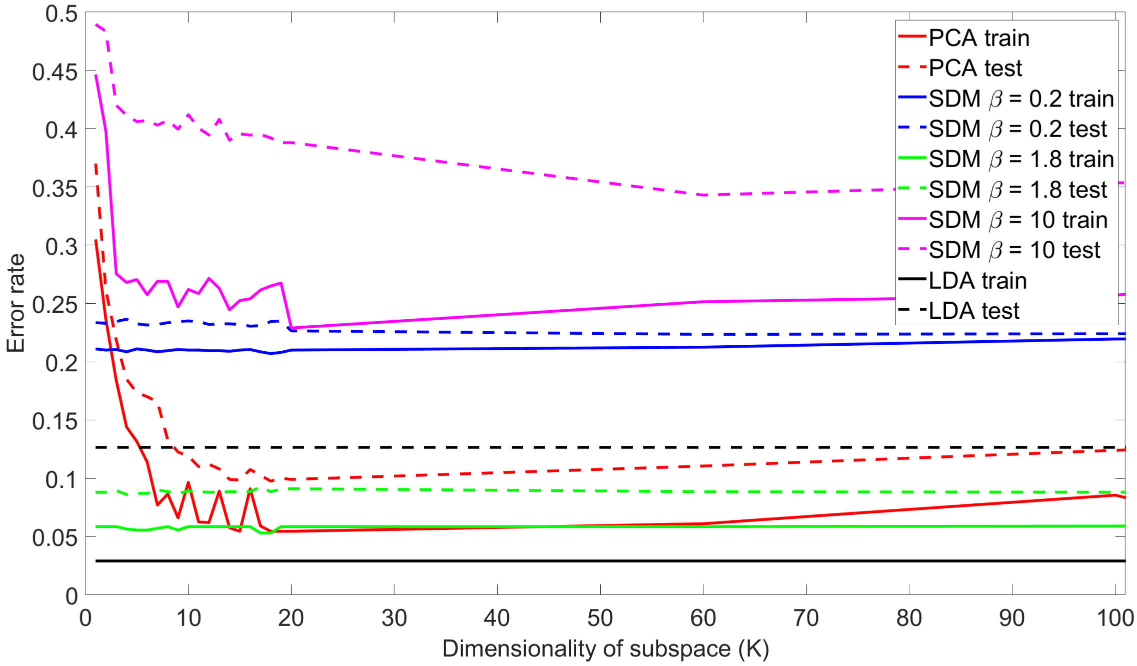


Figure 5-1: Comparison of classification errors The training- and test errors with varied dimensionality of the subspace generated by three methods: PCA,LDA, and SDM. Three different values of β are considered, one of which ($\beta = 1.8$) is set by Data-Driven Soft Discriminant Map (DD-SDM).

to minimize. As seen in Figure 5-1, DD-SDM achieves the lowest test-error overall. For $K = 1$, SDM with $\beta = 1.8$ scores best, followed by LDA, SDM with $\beta = 0.2$, PCA, and lastly SDM with $\beta = 10$.

Overfitting The key issue that SDM aims to solve is illustrated by comparing LDA to DD-SDM. The training error of LDA is lower, but at the cost of a higher test error. This bigger difference between the two errors indicates that the LDA-defined subspace and its subsequent classification is victim of overfitting on the training data. As explained in section 3-1, SDM affects the amount of overfitting by tuning β. The difference between train and test error is smaller for SDM with $\beta = 1.8$ indicating that the method successfully reduces the amount overfitting resulting in better generalization capability for the classification model trained downstream.

Number of classes In the used rat brain dataset, there are two classes as described in section 4-1-1. As mentioned earlier, LDA constructs a subspace with dimension $K <= C - 1$, with C the number of classes, so for this dataset LDA delivers a subspace with $K = 1$. As seen in Figure 5-1, the error of SDM similarly does not improve after the first feature has been found. PCA, however, reaches its lowest test error only at a subspace with $K = 15$. This hints already that the class-differentiating information in the PCA subspace is spread over several principal components, rather than being grasped by the first subspace dimension. This is to be expected, since PCA is an unsupervised method, and does not take class information into account. Moreover, with values of K higher than 15, the test error rises for PCA again, suggesting that the components added beyond that point are making classification more difficult and could lead to overfitting.

This result extends the case examined by Liu and Gillies [9], who only looked at datasets where the number of classes C is higher than the highest considered K.

Considerations The error-curves of Figure 5-1 are generated with the K-NN classifier. The natural question arises for which classifier types DD-SDM will out-perform other methods, and so in future work more classifiers could be considered. In this thesis, SDM is only benchmarked against PCA and LDA. A natural followup would be to compare SDM to other families of DR methods such as non-linear methods. The optimal β obtained is still dependent on some parameters set in DD-SDM, which are the number of training samples and the classifier type used to obtain the errors (which will usually match the classifier type that will be used subsequent to the DR-part of the processing pipeline). The current β found by the method described in section 3-3 outperforms PCA and LDA in this experiment. However, it is possible that using other approaches to determine β within DD-SDM could result in an even better performing β.

5-2 Experiment 2: Sensitivity of β with regard to classification performance

In their paper introducing SDM, Liu and Gillies [9] explain that in high dimensional datasets, the within-class variance should be reduced rather than increased to avoid overfitting. In this experiment, the Imaging Mass Spectrometry Mouse Kidney Dataset (IMS-MKDS) is used to visualize the effects of varying the parameter β in SDM to control the bias towards maximizing between-class variance and minimizing within-class variance. Next to studying the effect of β, this experiment also aims to visualize the effect of the number of data points in the training dataset N_1 relative to the dimensionality M of the full dataset $D_{Full} \in \mathbb{R}^{N \times M}$. Remember that N_2 denotes the number of data points in the test dataset and $N_1 + N_2 = N$.

Dataset creation To obtain a labelled dataset, masks are defined as shown in Figure 5-2. The blue, orange and purple masks indicate the regions where abscesses are present. The green mask represents a patch of non-abscess 'healthy' tissue. The pixels that lie within the masks are assigned to separate classes with labels. The pixels (*e.g.* data points) within the masks are stored in the full dataset D_{full} together with a class-label for each pixel denoting the tissue patch it belongs to. Note that the pixels outside the masks are discarded and are therefore not used as input for SDM.

To generate Figure 5-3, D_{Full} was split into two parts, D_{Train} and D_{Test}, and SDM was applied to D_{Train} only to determine the dimensionally reduced subspace. In constructing D_{Train}, the number of data points per class are kept equal to avoid introducing a bias towards one of the classes. The transformed data points are shown in a scatter plot in Figure 5-3. Note that the points in the left and middle abscess have substantial overlap. This overlap suggests that these tissue structures can not successfully be separated using these two latent variables and possibly need an extra latent variable. To make the effect of β more clearly visible, in the remainder of this experiment the left and middle abscess will be combined into one class. By having three classes, the number of latent variables will be $C - 1 = 2$, making it possible to fully visualize all variables in a 2-D scatter plot. The left and middle abscesses will be denoted with abscesses 1 and 2 respectively. The right abscess will be denoted with abscess 3.

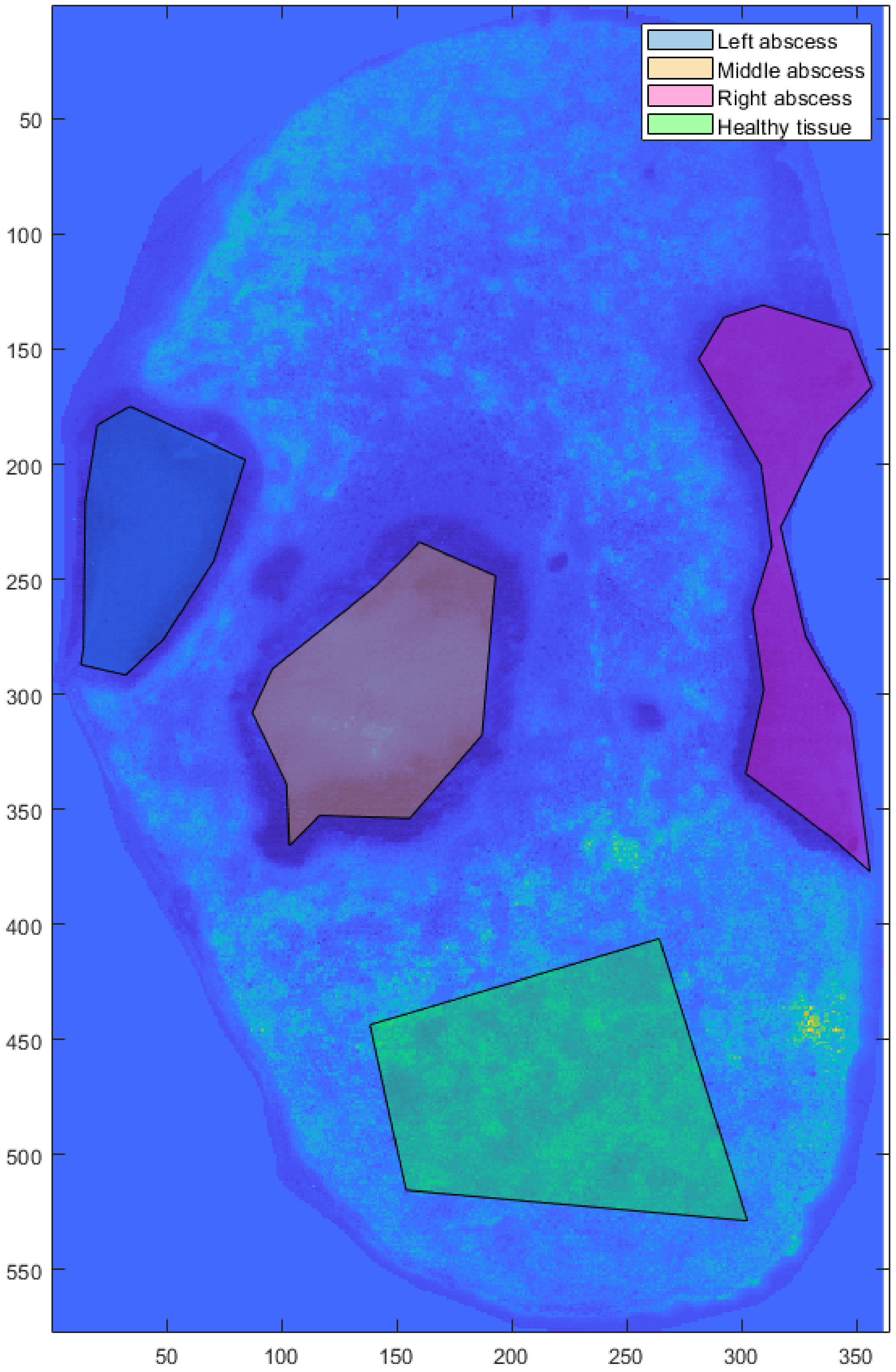


Figure 5-2: Masks indicating the classes in the IMS-MKDS Labels are assigned to four regions within the kidney tissue. Three abscesses are visually detected and marked with a mask. A sample of the remaining tissue has been masked with green. All regions are assigned to a different class in the classification pipeline.

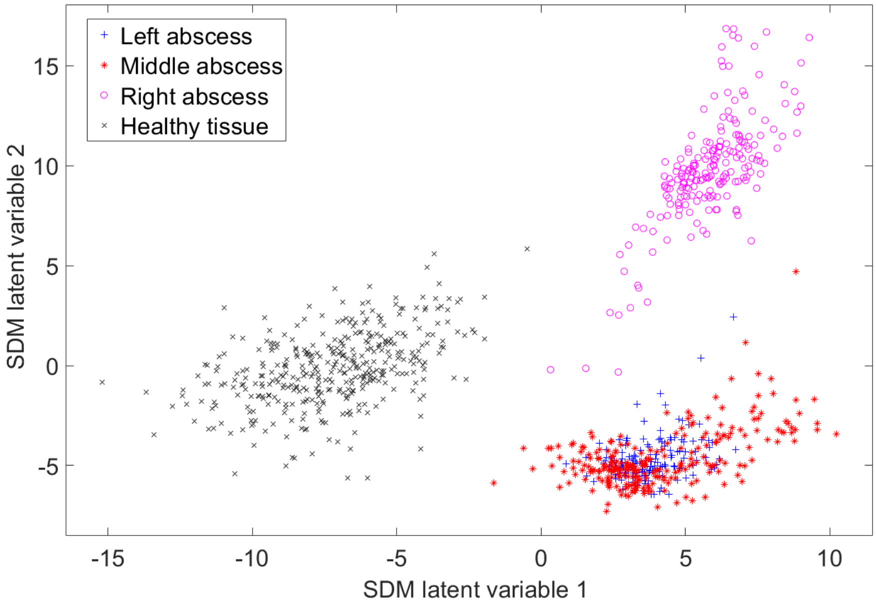


Figure 5-3: Scatter plot of the two first SDM-loadings of the IMS-MKDS The four classes are plotted in a scatter plot with the two first SDM-loadings. The left and middle abscesses have high overlap.

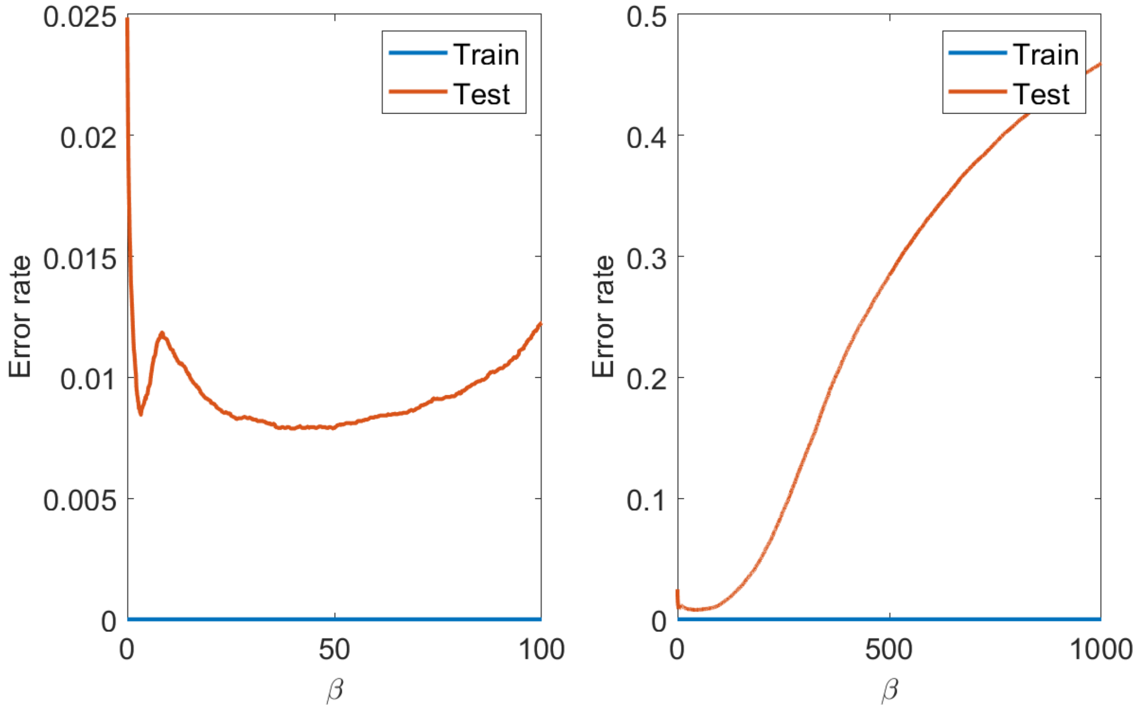
5-2-1 Experiment 2a: High dimensionality

In the first case we consider, the number of data points N_1 in the training set is chosen to be smaller than the dimensionality M, the number of features in each original measurement. This case represents high-dimensional datasets with relatively few observations compared to the number of features. This is also the case Liu and Gillies [9] considered for all their datasets, claiming that SDM reduces the risk of overfitting in this scenario.

As seen in Figure 5-4, the error rates are repeatedly computed while varying β between zero and 1000. An upper bound of 1000 was chosen since the results beyond this value did not significantly change. The train and test errors are computed as explained in Section 3-2, where number of reduced features K is set to be $C - 1 = 2$, where C is the number of classes (in this $C = 3$). For a high β value, the test error rises, while the training error goes to zero. This is an indication of overfitting. Interestingly, there are two local minimums present. In the following section, the minimum with the lowest β is considered at $\beta = 1.87$.

Optimal β The first local minimum is found at $\beta = 1.87$. Figure 5-5 shows the corresponding scatter plots. Figure 5-5a shows the data points of the training set projected into the coordinate system found by SDM. The three classes are separated from each other, and there is still variance present within the classes. The separation that determines the test error however, is in the same plot for the data points in the test set shown in Figure 5-5-b. The plot shows that the classes are still separated quite well, making it possible for the classifier to achieve a low test error.

The value that each pixel has for the latent variables provided by SDM can be used to create false-colour images similar to the ion images described in Section 2-1-2. Note that all pixels, not only the ones included in the masks, are now transformed into the new



(a) Zoomed-in

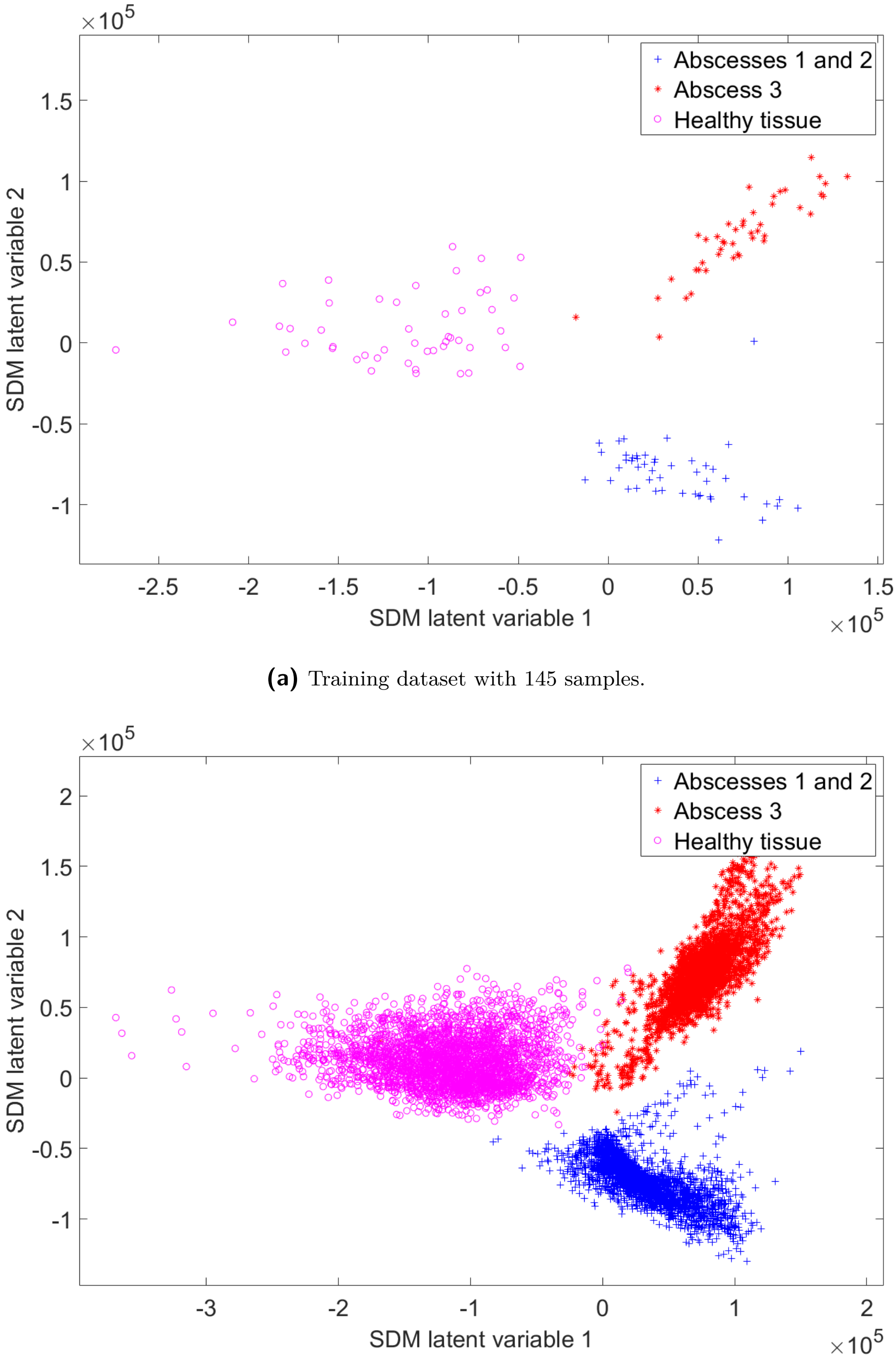
(b) Zoomed-out

Figure 5-4: The effect of β on the classification errors using the Linear Bayes Normal Classifier The training dataset consists of 145 samples (N_1).

coordinate system created by SDM. When looking at the false-colour image of SDM latent variable 1 in Figure 5-8a, note that the abscesses are mainly yellow, indicating a high score on Latent variable 1, which corresponds to what can be seen in the scatter plots. Latent variable 2 separates the left and middle abscesses from the right one as seen in Figure 5-8b, which also matches the scatter plots where the abscesses are spread over the vertical axis representing the second latent variable.

Decreasing β The scatter plots resulting from setting β to a lower than optimal value at 0.001 are shown in Figure 5-6. In this case, the test data points are separated well similarly to the optimal β, resulting in a low test error rate. Note that by setting β close to zero, the distance between class-means is maximized and no weight is given to minimizing the variance within the classes. Therefore, the data points lie in a bigger range, which can be seen by comparing the axis of Figures 5-6 and 5-5. The variance within the classes in the training set are higher than in the optimal case. This case resembles the behaviour of PCA, which aims to maximize the variance of all data points in the reduced feature space. Figures 5-8c and 5-8d show the corresponding false colour images. When the false colour image of SDM latent variable 1 in Figure 5-8c is compared with the optimal case of Figure 5-8a, we observe that the colors are flipped. Since flipping a coordinate system does not influence the variance of the projected data points, SDM can show this flipping behaviour. However, this behaviour does not influence classification performance.

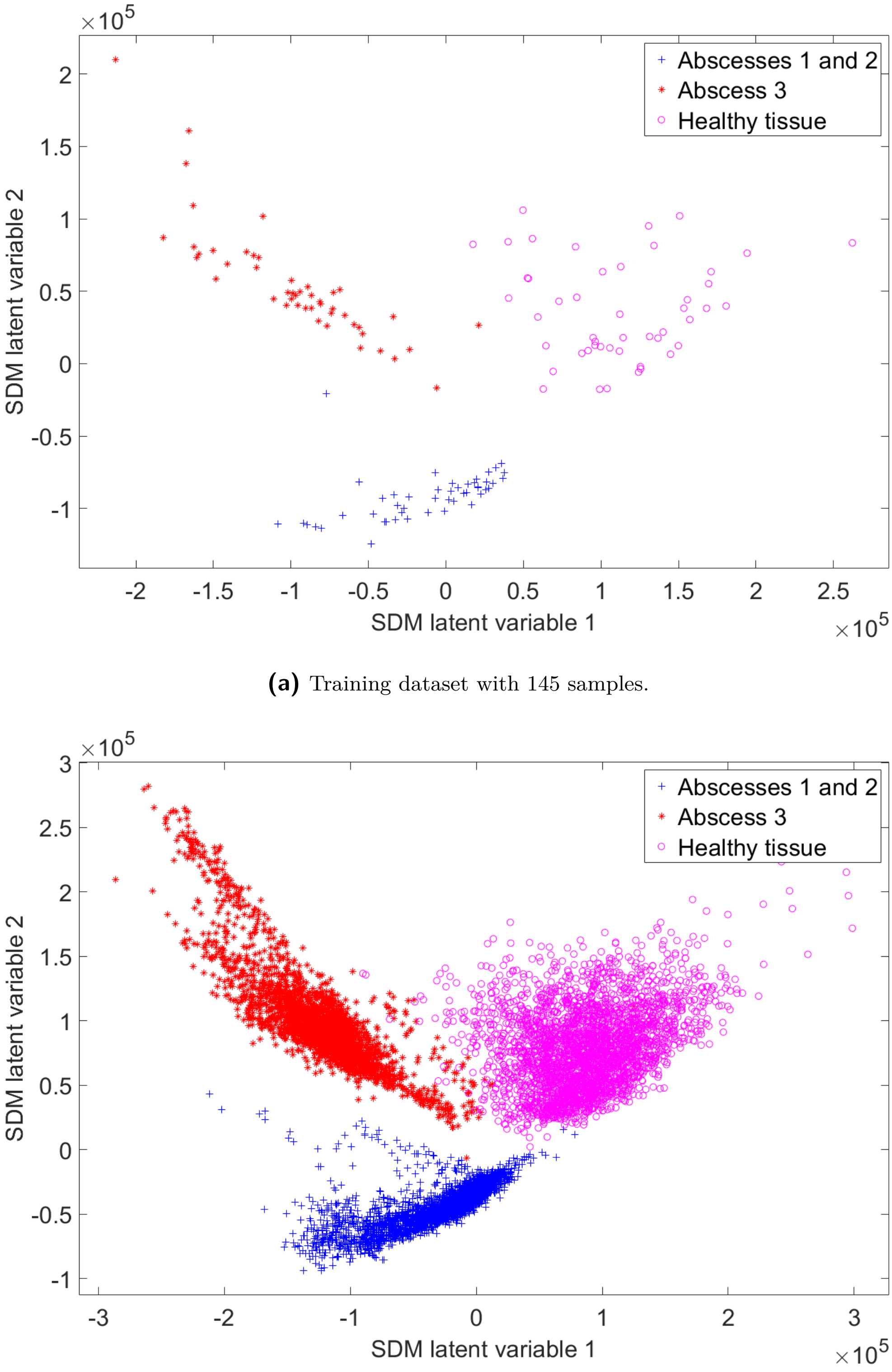
Increasing β By increasing β, the minimization of the within-class variance has been given more weight. An interesting phenomenon occurs in this situation where $M > N_1$. As seen in Figure 5-7a, SDM aims to find a coordinate system that minimizes the within class variance. It can be shown that when $M > N_1$, a coordinate system can be chosen in such a way that all data points within a class have the same feature values [9]. At first glance, achieving this amount of class separation makes it easy to create an effective classification rule. However, when looking at Figure 5-7b, when the data points in the test set are projected onto the new coordinate system, the classes have a high overlap. This indicates that by setting β too high, overfitting can occur and result in a high test error. In the resulting false colour images shown in Figures 5-8e and 5-8f, vast amounts of noise are present. This will be the effect of tuning β too high in the case that $M > N_1$.



(a) Training dataset with 145 samples.

(b) Test dataset with 9000 samples.

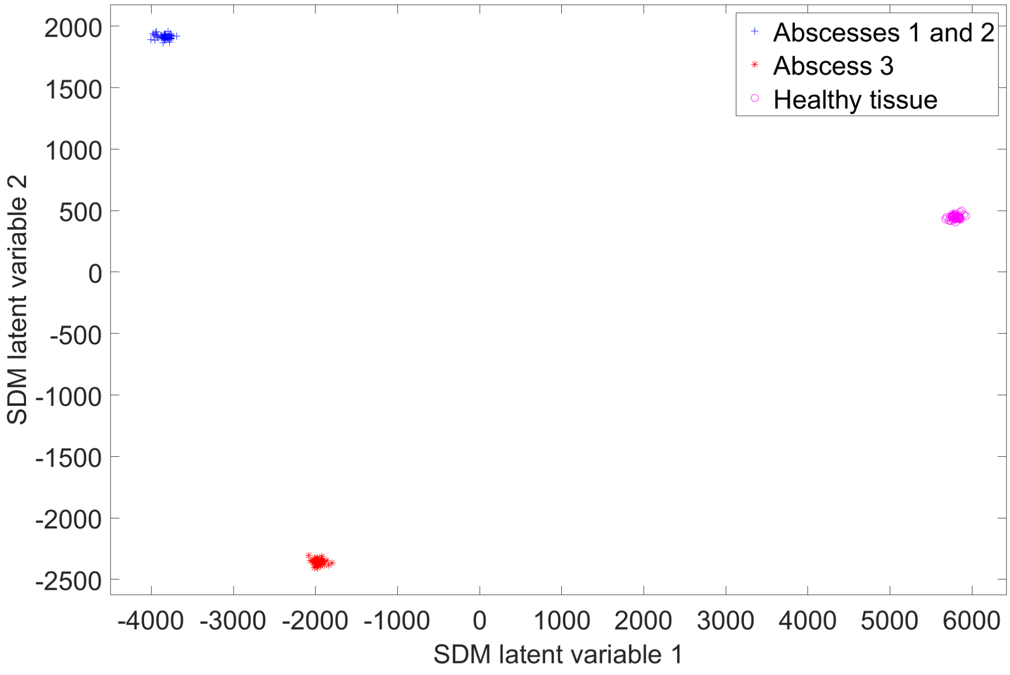
Figure 5-5: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 1.87$ The training dataset has $N_1 = 145$ data points.



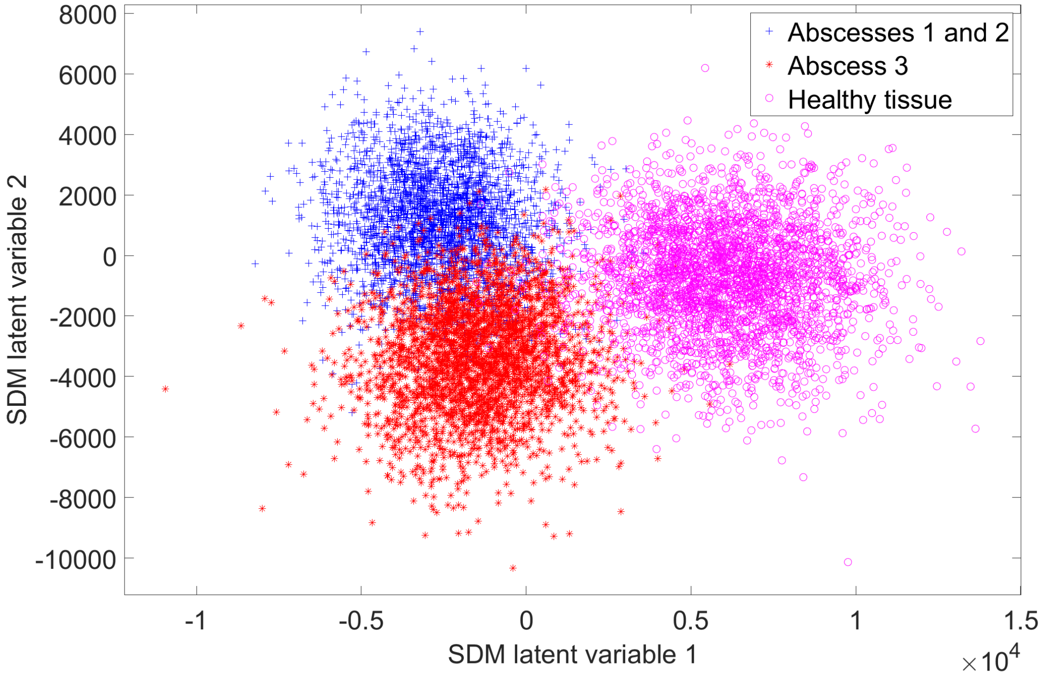
(a) Training dataset with 145 samples.

(b) Test dataset with 9000 samples.

Figure 5-6: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 0.001$ The training dataset has $N_1 = 145$ data points.

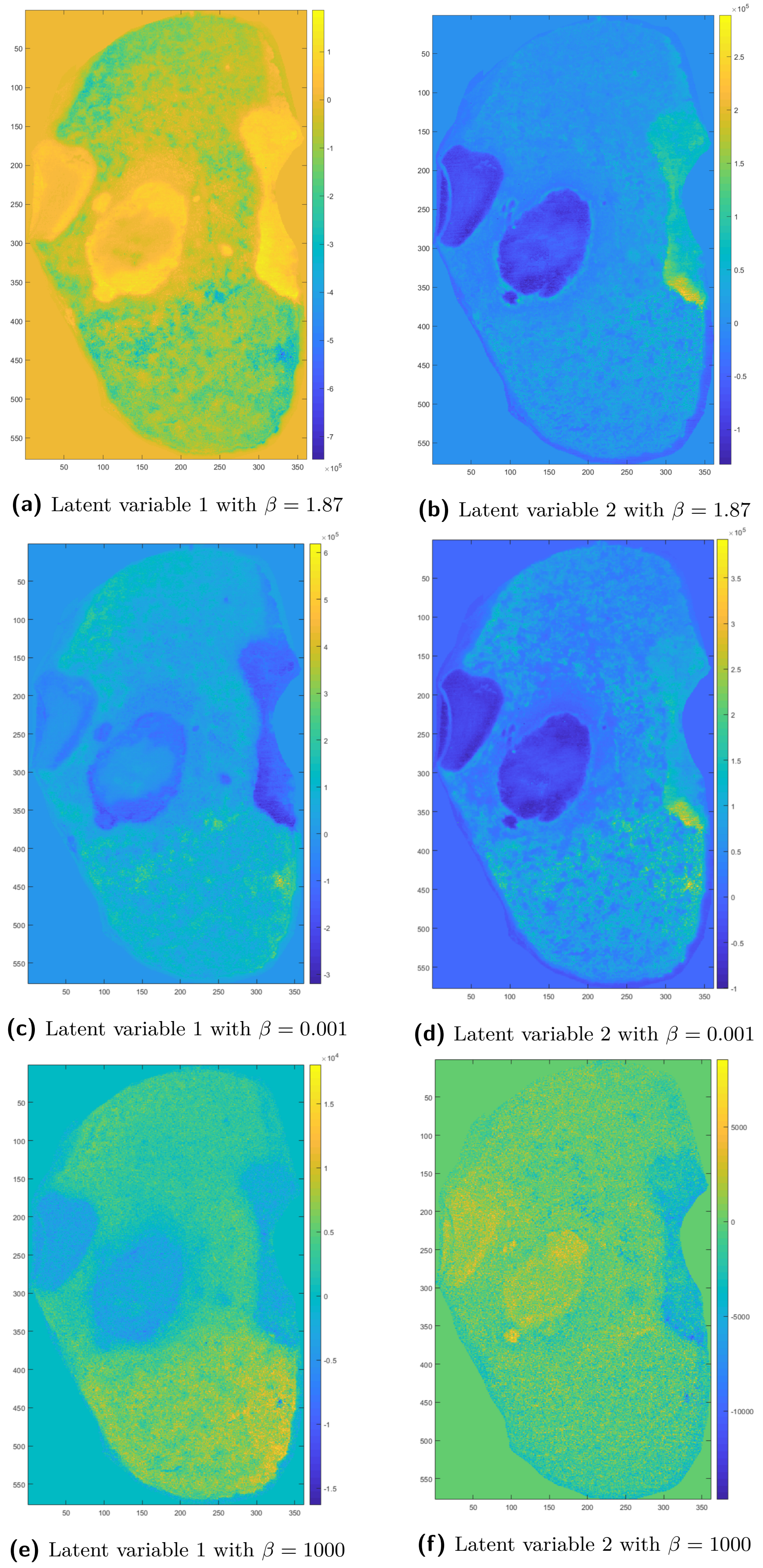


(a) Training dataset with 145 samples.



(b) Test dataset with 9000 samples.

Figure 5-7: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 1000$ The training dataset has $N_1 = 145$ data points.



(a) Latent variable 1 with $\beta = 1.87$

(b) Latent variable 2 with $\beta = 1.87$

(c) Latent variable 1 with $\beta = 0.001$

(d) Latent variable 2 with $\beta = 0.001$

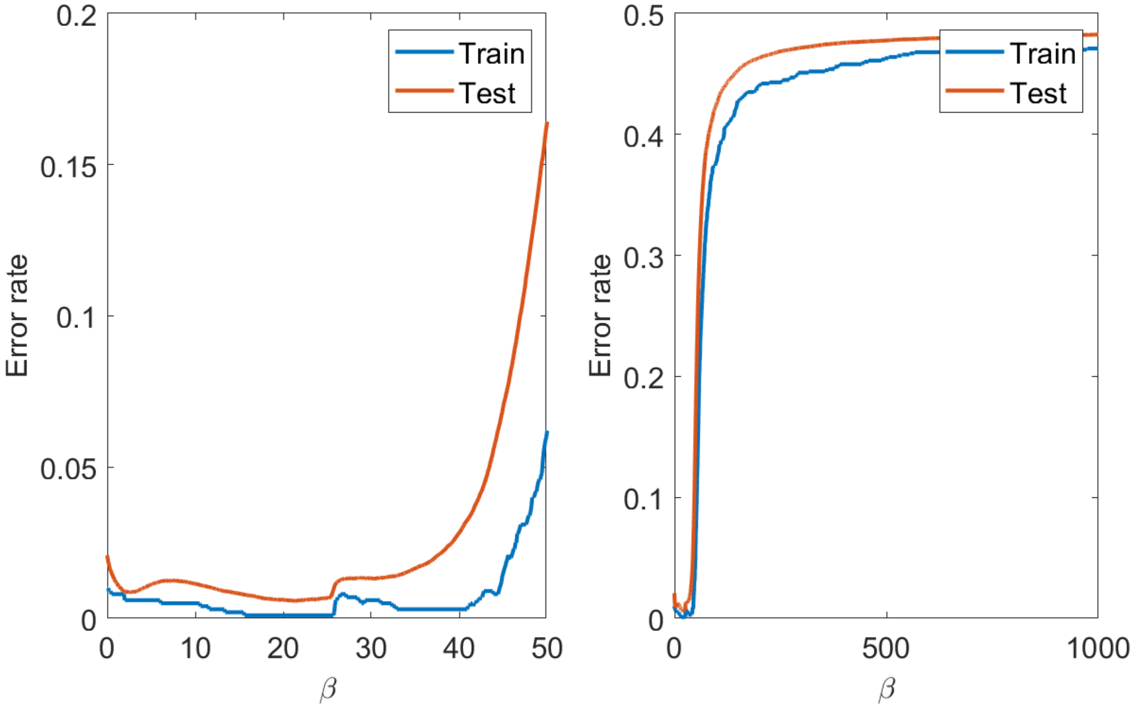
(e) Latent variable 1 with $\beta = 1000$

(f) Latent variable 2 with $\beta = 1000$

Figure 5-8: SDM latent variable images The input dataset has 145 samples and 152 m/z bins.

5-2-2 Experiment 2b: Low dimensionality

Next, the case were $M < N_1$ is considered. This case is not considered by Liu and Gillies [9] and will therefor be considered here. In Figure 5-9, the effect of β on the error rates is shown. Looking at the training error, when β rises to 26, the error keeps decreasing until the training error is zero, meaning all training data points are classified correctly using the SDM-provided coordinate system. Note that there is a local minimum around $\beta = 2$ and $\beta = 23$. When β goes above around 30, both the training and test error will start to increase rapidly and converge to a high error rate of about 45% as seen in Figure 5-9b. These effects will be further explained accompanied with visualizations in the following paragraphs.



(a) Zoomed-in **(b) Zoomed-out**

Figure 5-9: The effect of β on the classification error using the Linear Bayes Normal Classifier The training dataset consists of 1000 samples(N_1).

The dataset is visualized for three β's: the optimal β (23.1), a low β (0.001), and a high β (1000). Similar as in experiment 2a, two visualizations are presented for each β: scatter plots that plot the training and test dataset projected into the SDM-generated coordinate system, and latent-variable images for both SDM-loadings.

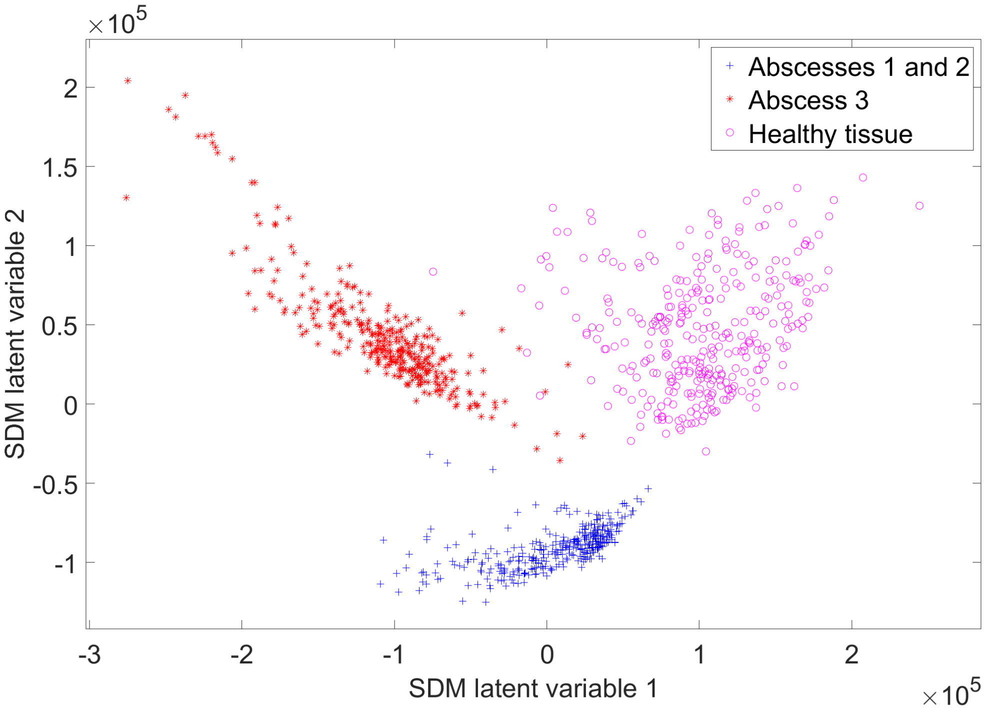
Low β Figure 5-10 shows two scatter plots for the training and test datasets where the number of data points in the training and test sets are $N_1 = 1000$ and $N_2 = 9000$ respectively. Note that since β is set at a low value of 0.001, the variance within the class is relatively high. Also note that the training data points are representative for the test dataset, resulting in only little overlap of the classes for the test dataset. The resulting latent variable images are shown in Figures 5-13a and 5-13b. Both the scatter plots and latent variable images are similar to the results in the $M > N_1$ case for low β. Interestingly, it seems that the results for low values for β are not so much impacted by the relative size of the training dataset.

Optimal β For the case with $\beta = 23.1$, Figure 5-11 shows that the variance within the classes in the training set has decreased compared to the low-β case. This results in a good separation in the test set, causing the test error to be the lowest of all β's. Note that the structure of the point-clouds have changed to an elliptical shape. The corresponding latent variable images in Figures 5-13c and 5-13d show a clear distinction between the classes. However, primarily in Figure 5-13d the tissue appears noisy in the "healthy"-class, suggesting that the second latent variable does not capture the biological characteristics of this class well. Scatter plot 5-11 confirms that most of the variance in the "healthy" class is represented by the first variable. The explanation of this noise could also lie in the earlier mentioned artificial ellipse shape of the scatter plots. Possibly suggesting that the latent variables capture inherent features of the data instead of biological patterns.

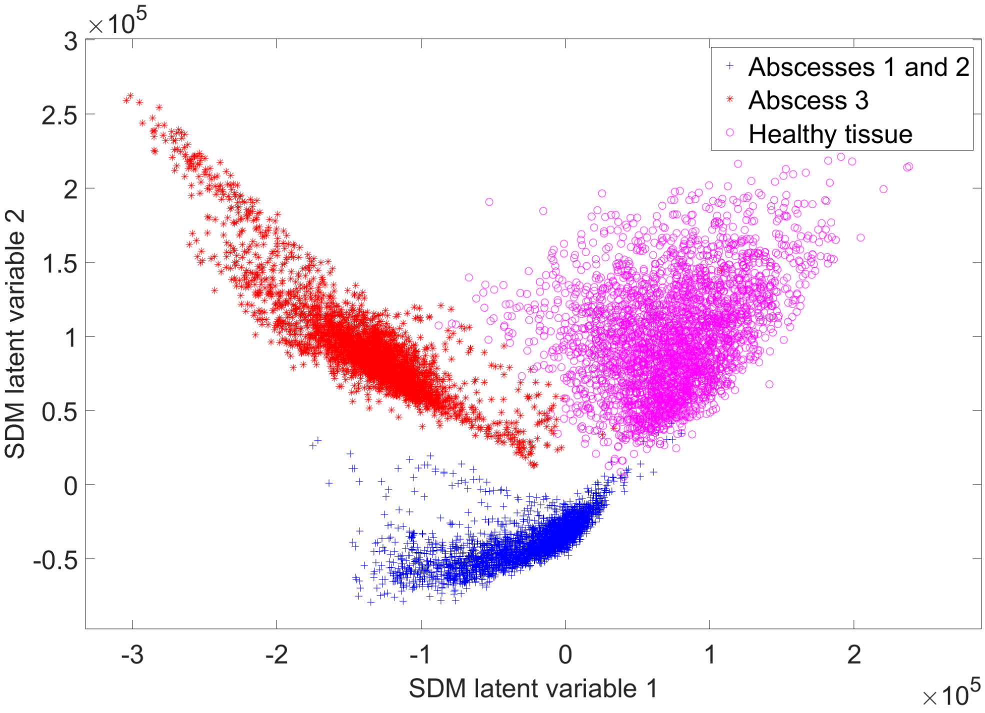
High β Figure 5-9b shows that both the training and the test error blow up for high β values. The scatter plots in Figure 5-12 show the result when $\beta = 10000$, and illustrate the reason for this poor performance. The scatter plots show that SDM fails to achieve class separation in the training dataset as well as in the test dataset. The resulting latent variable images shown in Figures 5-13e and 5-13f are therefor of poor quality.

When β is set high, SDM minimizes within-class variance, instead of maximizing the distance between the means of the classes. Since the number of data points in the training dataset is higher than its dimensionality, there no longer exists a feature in which all points coincide with each other as in the case shown in Figure 5-7a. Note the range of the axis in Figure 5-12a. All points lie within a range between -3000 and 3000, this in contrast to the low β case where the axis spans between -$2 * 10^5$ and $2 * 10^5$. The severe overlap in the high β cases clearly illustrates that minimizing within-class variance alone is a poor method for extracting features.

Conclusion To conclude, for both cases, $M > N_1$ and $M < N_1$, there is an optimal β. High values for β perform poorly in both cases but for different reasons. Low values for β perform similar in both cases, but have higher overlap of the classes, resulting in a suboptimal test error. It is clear that the classification performance is highly impacted by the chosen β. In the next section, the proposed framework DD-SDM for setting β is evaluated. DD-SDM aims to set β automatically in a way that minimizes the test error.

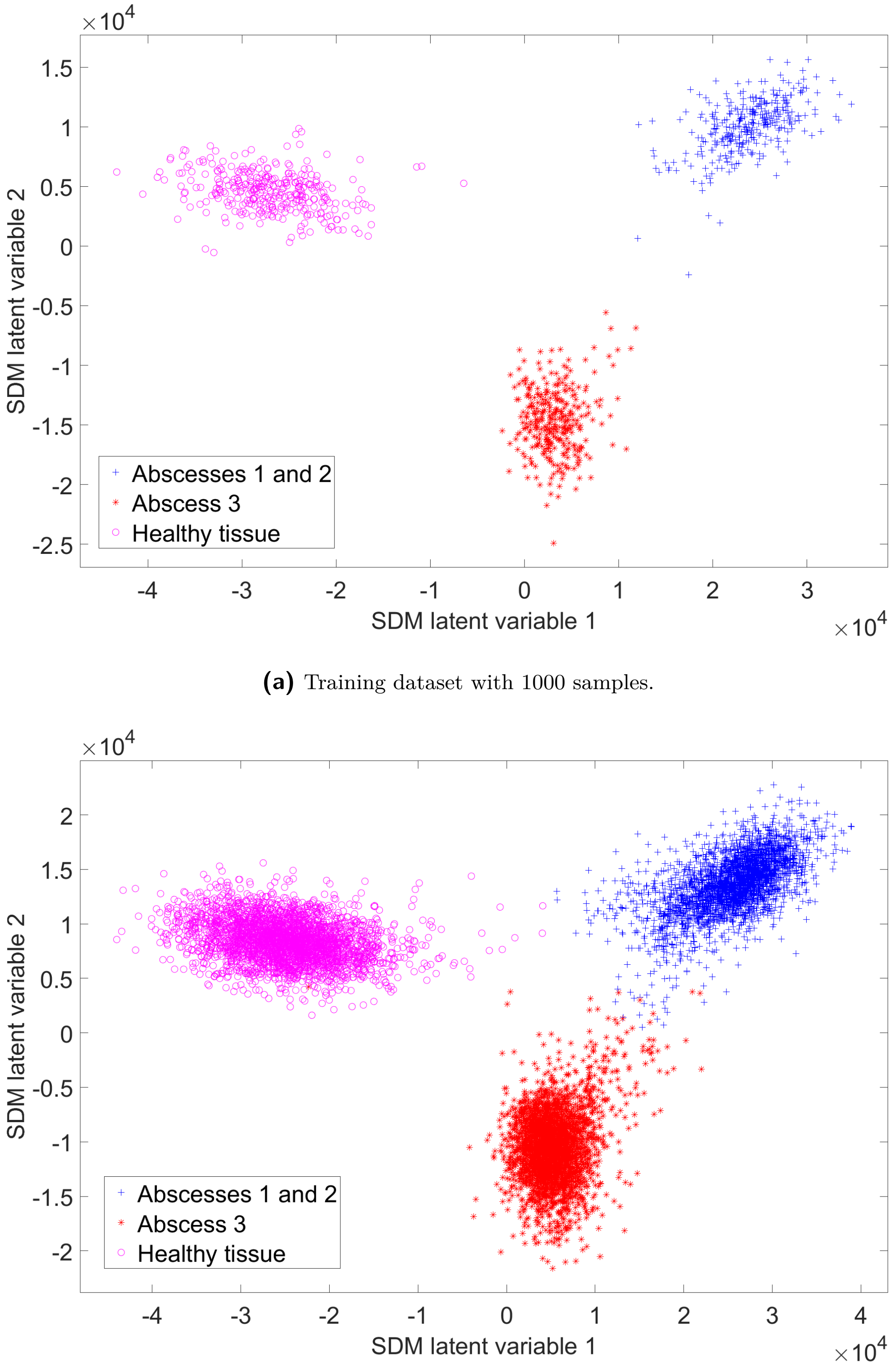


(a) Training dataset with 1000 samples.



(b) Test dataset with 9000 samples.

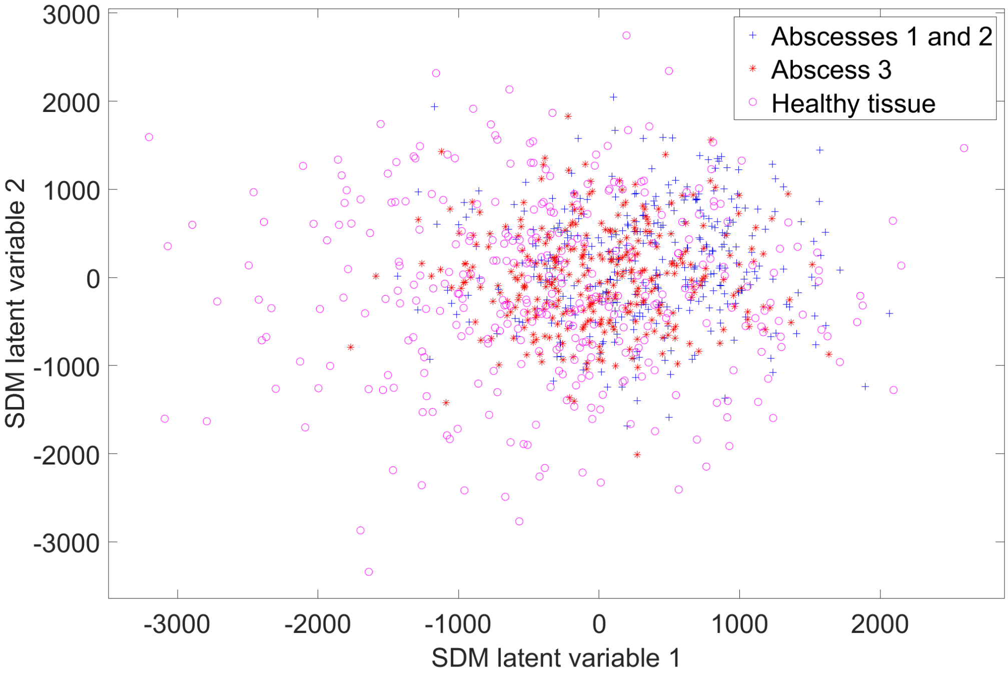
Figure 5-10: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 0.001$ The training dataset has $N_1 = 1000$ data points.



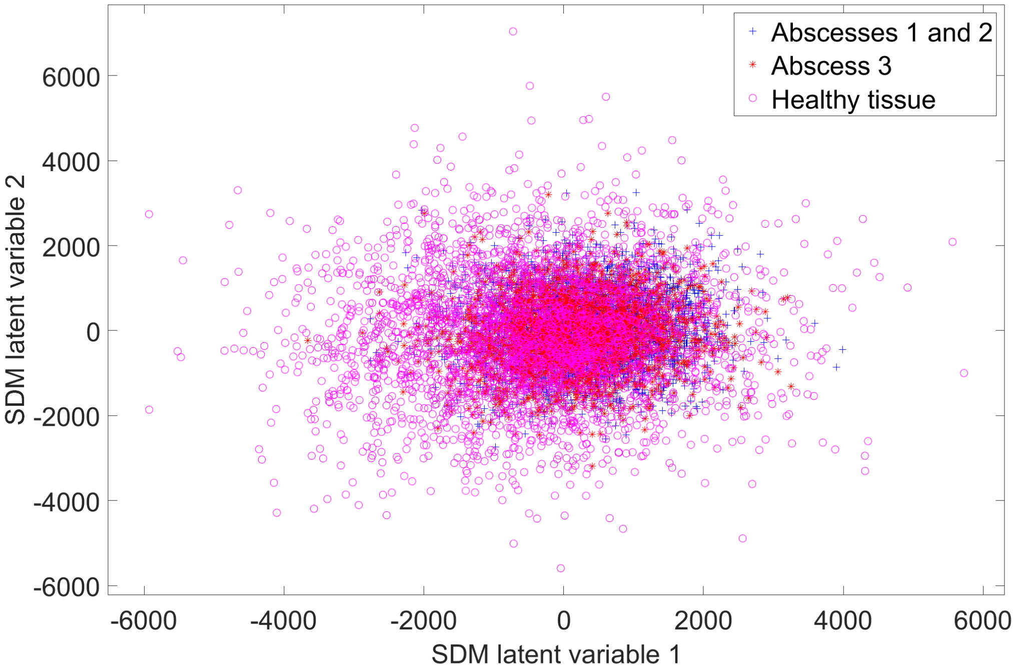
(a) Training dataset with 1000 samples.

(b) Test dataset with 9000 samples.

Figure 5-11: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 23.10$ The training dataset has $N_1 = 1000$ data points.

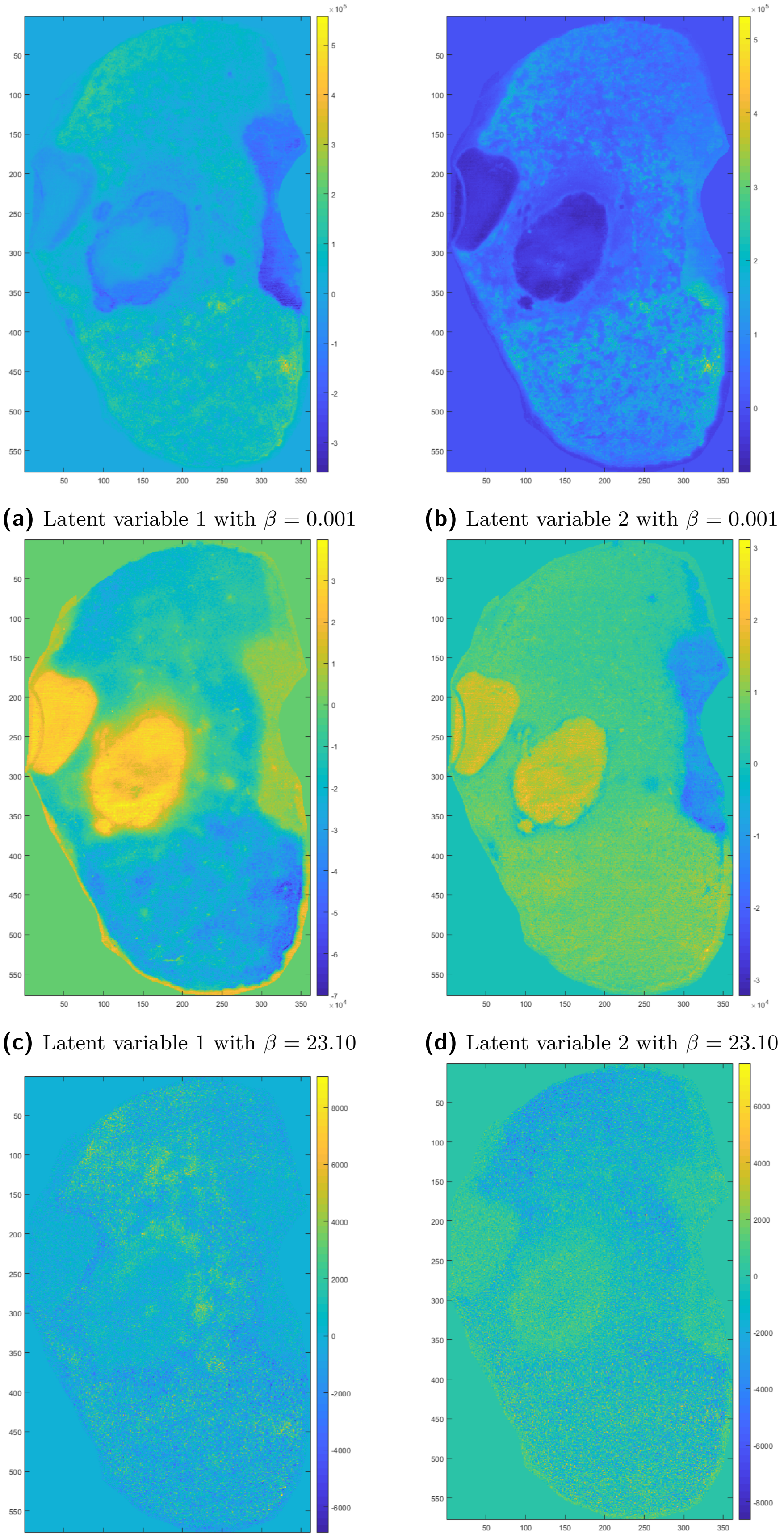


(a) Training dataset with 1000 samples.



(b) Test dataset with 9000 samples.

Figure 5-12: Scatter plot of the data points of the IMS-MKDS in the reduced feature space created by SDM with $\beta = 1000$ The training dataset has $N_1 = 1000$ data points.



(a) Latent variable 1 with $\beta = 0.001$

(b) Latent variable 2 with $\beta = 0.001$

(c) Latent variable 1 with $\beta = 23.10$

(d) Latent variable 2 with $\beta = 23.10$

(e) Latent variable 1 with $\beta = 1000$

(f) SDM Latent variable 2 with $\beta = 1000$

Figure 5-13: SDM latent variable images The input dataset has 1000 samples and 152 m/z bins.

5-3 Experiment 3: Data-Driven Soft Discriminant Maps Search Strategies

In section 3-3, the proposed framework DD-SDM for automatically setting β is described. In this experiment, the number of iterations needed to reach the optimal β using the golden section search is analyzed and benchmarked against a more naive approach, grid search. Since the upper and lower bounds of the search interval are still user-supplied parameters within DD-SDM, we would like to have a method that scales well in terms of the number of times SDM needs to be computed to reach the optimal β. As an illustrative example, Figure 5-14 shows the test error dependent on β. The data points on the blue line indicate all SDM evaluations performed using grid search, while the points on the orange line are evaluations picked by the golden section method. To quantify the certainty of the test error, every β is evaluated 5 times, with 5 different randomly selected subsets of the full dataset D_{full}. The data points in these subsets serve as training dataset D_{Train} and are randomly selected without replacement from the full dataset $D_{Full}$where each class has the same number of data points. The line goes through the averages and the error bars are constructed using one standard deviation up and down.

To find the optimal value of β with an accuracy of 0.1, with a search space between 0 and 100, grid-search would need to evaluate the test error $\frac{(100-0)}{0.1} = 1000$ times. The golden section method picks the to-be-evaluated $\beta's$ in a more efficient way, resulting in needing only 17 iterations (Table 5-1), to reach the optimum with an accuracy of 0.1.

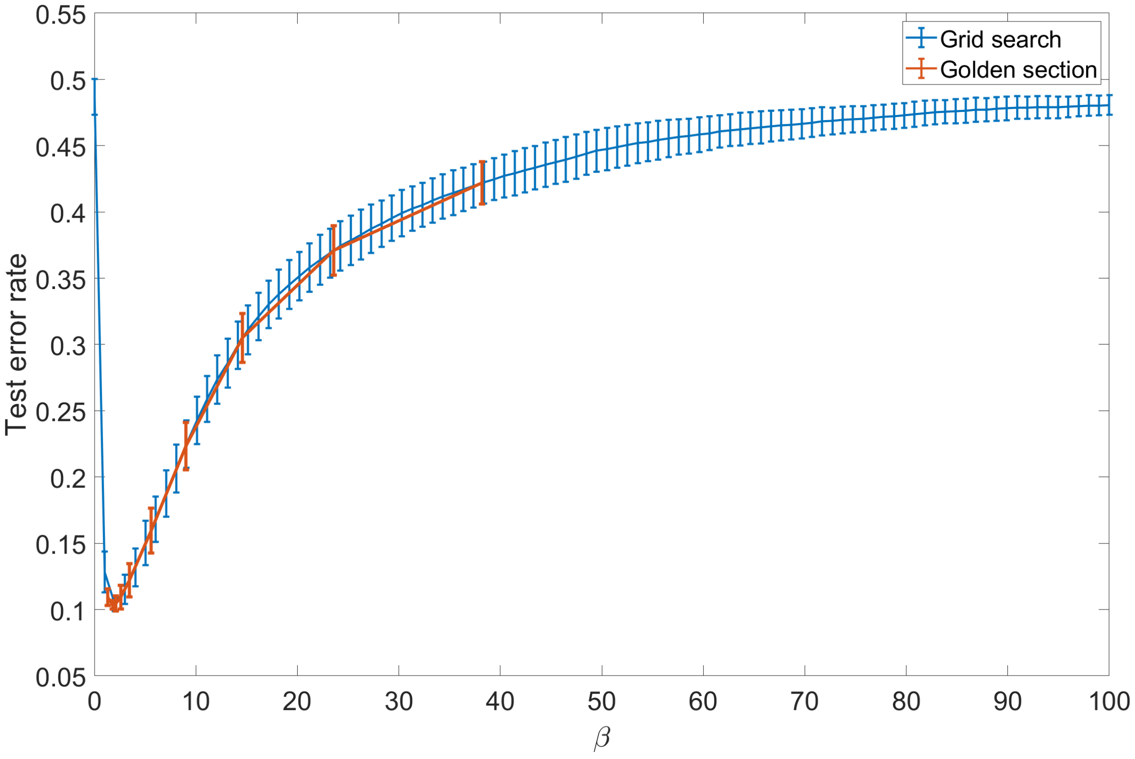


Figure 5-14: Comparison grid search versus golden section method The test error rate is computed for different values of β. The blue error bars are evaluations of linear grid search. The orange bars are $\beta's$ considered by the golden-section method. The error bars indicate one standard deviation above and below the average test error rate over five repetitions.

Table 5-1: Number of iterations needed to reach the optimal β

Iterations	Grid search	Golden section
Exp. 1	1000	17

Fair comparison Another way to visualize the performance difference between grid search and the golden section method, is shown in Figure 5-15. First, the golden section method (shown in orange) is applied, finding the optimal $\beta$of 1.83 in 17 iterations with a test error rate of 10%. Next, grid search is applied with an equal number of iterations (17), with the lowest test error rate found at $\beta = 7.14$ with a test error rate of 18%. This shows that the golden section method is more efficient in finding the optimal value of β.

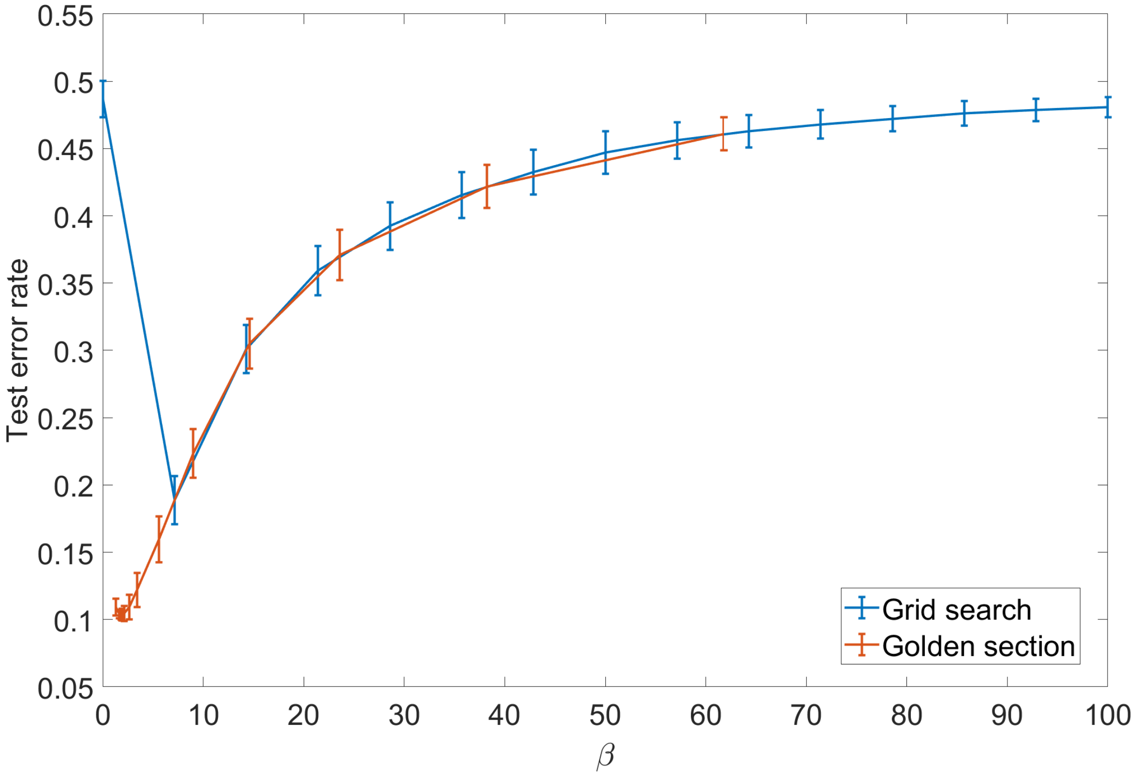


Figure 5-15: Comparison grid search versus golden section method with an equal number of iterations Grid search is not able to reach the optimal value of β when the method uses an equal number of iterations as the golden section method needs to find the optimum with an accuracy of 0.1.

Considerations The golden section method assumes that the curve will have only one minimum. In particular with evaluating high values of β, where the errors are equally high, the golden section method could get stuck in a local minimum as seen in Figure 5-16. A first method to overcome this is by smoothing the curve by calculating the error over different subsets of the dataset, similarly as has been done in Figure 5-14. However, this comes at the cost of evaluating SDM more times, which could be infeasible for big datasets.

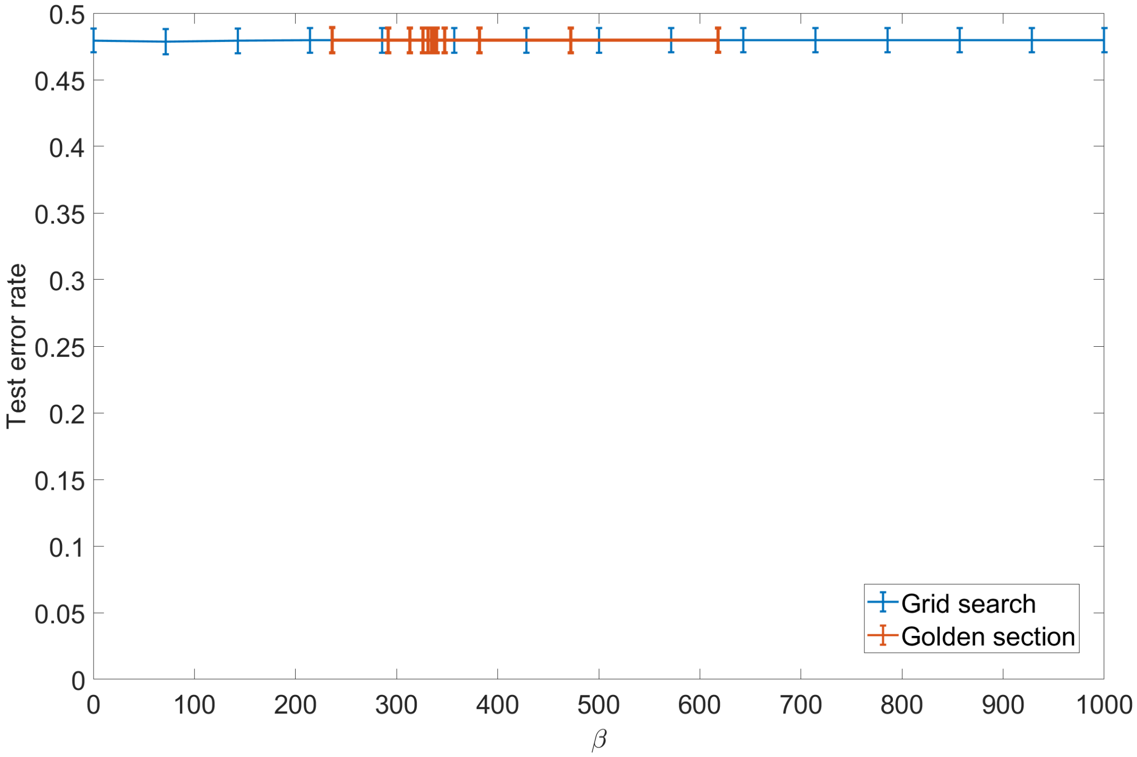


Figure 5-16: Stuck in local minimum for high values of β When the upper bound for DD-SDM is set to high, the error rates are equally high. This results in the golden section method to be stuck in a local minimum. Grid search is also not able to achieve the optimal value with the same number of iterations.

A second solution could be to run the golden section search with a multi-start setting. This means that the entire method is run multiple times with different initial upper bounds, denoting the β corresponding to the lowest test error as the global optimum. This method will also come at the cost of extra evaluations of SDM, which is undesirable.

5-4 Experiment 4: Class-specific feature conservation

In this experiment, the artificial dataset, constructed as described in section 4-1-3, is used in combination with the Peak Conservation Score (PCS) to compare SDM and PCA in their ability to extract features that conserve the class-specific peaks (i.e. class-specific features) and discard noise and peaks or features that are common between classes. For both methods, the PCS is calculated for artificial datasets with scaled class-specific ion intensity peaks as described in section 4-2-4.

SDM Figure 5-17 shows the analysis of the PCS of SDM with a β of 0.1. The top plot shows the PCS score calculated for a range of scale factors of the class-specific peaks. Ideally, the PCS for class 1 and 2 should be 100%, indicating that SDM conserves these peaks. The PCS for the background and noise peaks should ideally be zero, indicating that the method discards these peaks while constructing a lower-dimensional subspace. As seen in the top figure in Figure 5-17, this scenario is realized for intensity or amplitude scaling factors above 1.3. The corresponding feature weight plot for the subspace's first latent dimension is shown in the bottom right corner. Indeed, all class-specific peaks are above the threshold and are taking part in the lower-dimensional subspace provided, while all noise and background-induced peaks fall below the threshold and do not participate in the produced subspace. The thresholds, as explained in Section 4-2-4, are set at -0.5σ and 0.5σ. When the class-specific peaks are scaled-down in intensity, however, SDM fails to conserve the class-specific peaks and selects the non-class-specific background peaks as part of the first dimension of the provided subspace, as seen in the bottom left figure of Figure 5-17. This observation suggests that SDM is prone to picking common peaks when the variance of the class-specific peaks not sufficiently high.

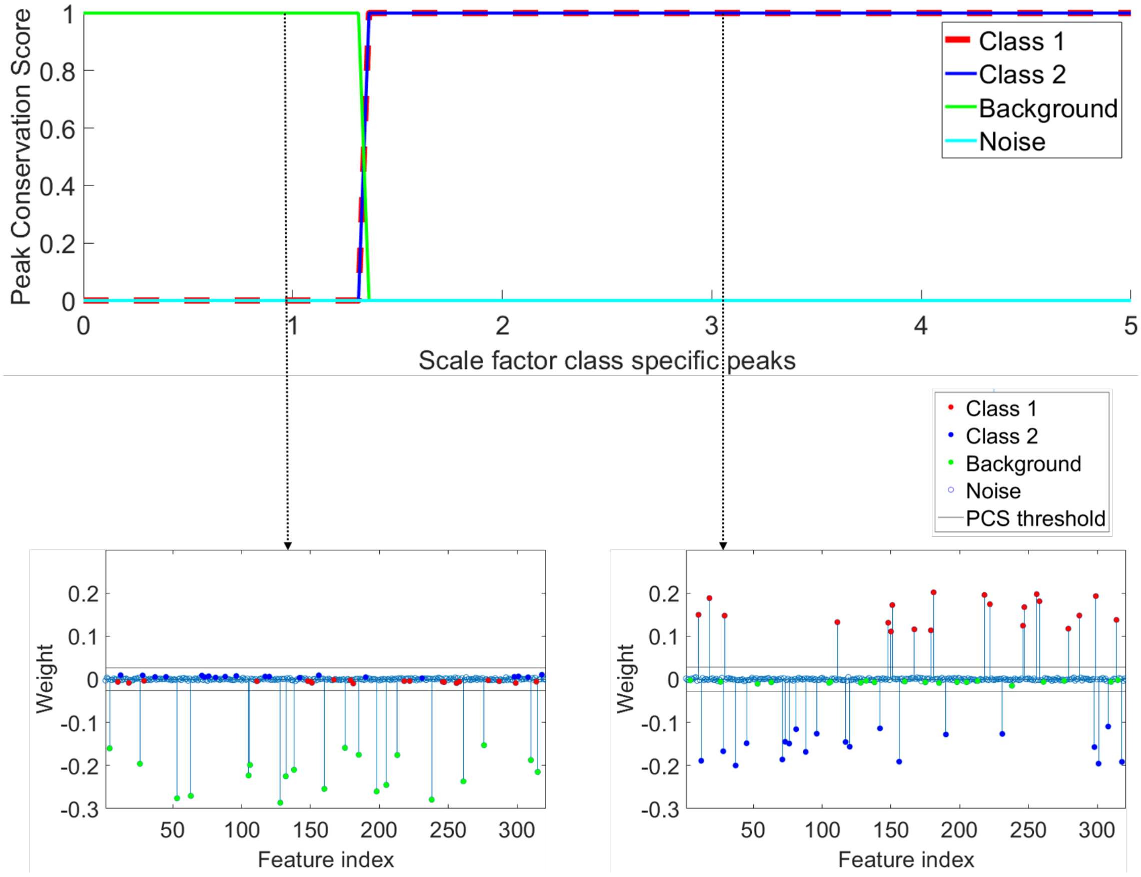


Figure 5-17: SDM: Peak conservation score with corresponding feature-weight plots The top figure shows the PCS for all peak-classes with varying scale factors of the class specific peaks. Feature-weight plots are shown for scale factors 1 and 3. The arrows indicate which feature-weight plot corresponds to which scale factor.

PCA - Principal component 1 Figure 5-18 shows the analysis for the first principal component, the latent variable of the PCA-produced subspace that corresponds to the latent variable in the SDM-produced subspace. As seen in the PCS-plot at the top, for low intensity values up to a scaling factor of around 2, the first principal component only conserves the background peaks. This is to be expected since the background peaks are responsible for the most variance in this scenario, and the class-specific peaks are insufficiently high to compete with the variance of the non-class-specific background peaks. With a scaling factor of three, however, the class-specific peaks have enough variance to be selected for and to rise above the PCS-threshold, as seen in the feature-weight plot at the right. However, the background peaks are still conserved as well. When the scaling factor is 5, the desired scenario is reached where only the class-specific peaks are conserved as shown in the feature-weight plot at the bottom, while the non-class-specific peaks are actively deselected for in the PCA-produced subspace. Although the background peaks lie below the threshold, the peaks still get some nonzero weight.

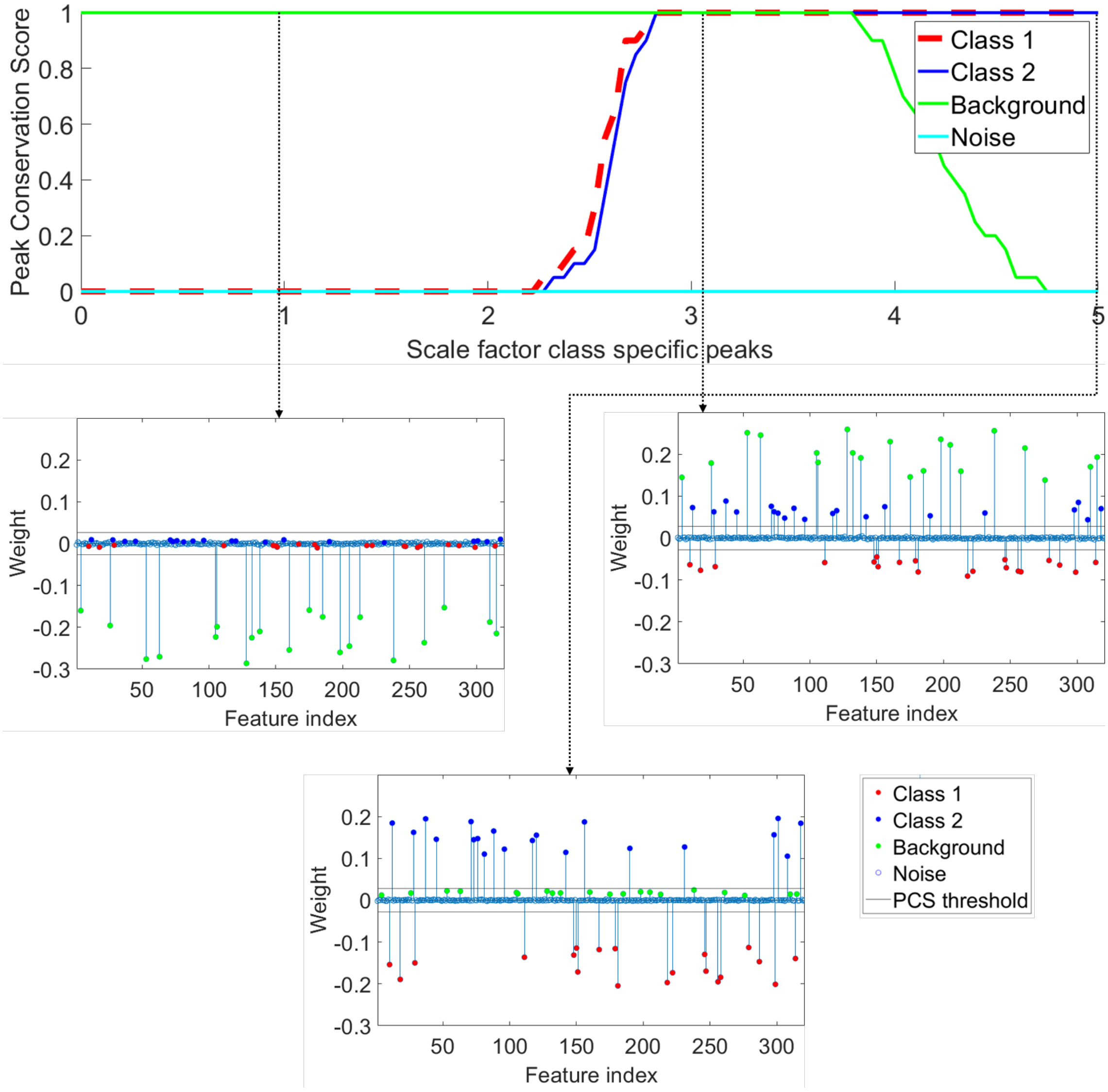


Figure 5-18: Principal component 1: Peak conservation score with corresponding feature-weight plots The top figure shows the PCS for all peak-classes with varying scale factors of the class specific peaks. Feature weight plots are shown for scale factors one, three and five. The arrows indicate which feature-weight plot corresponds to which scale factor.

PCA - Principal component 2 Figure 5-19 shows the same analysis for the second principal component that defines the PCA-produced subspace. For a scaling factor of zero, this component does not conserve the class-specific peaks as expected, since they carry no variance. Since the principal components together are generated to be orthogonal to each other, it is expected that the background peaks are not primarily conserved when the first component does so. This observation helps explain the behaviour of the weight plots. The top-left feature-weight plot corresponding to a scale factor of zero reveals that this component consists primarily of the peaks generated by noise. The top-right feature-weight plot corresponding to a scaling factor of 1 shows the desired scenario. Only the class-specific peaks are conserved, showing that PCA is also able to capture the class-specific peaks in one feature in this scenario. The bottom-left feature weight plot corresponding to a scale factor of 3 starts giving weight to the background peaks. When the scale factor is 5, the bottom-right feature-weight plot shows that only the background peaks are conserved.

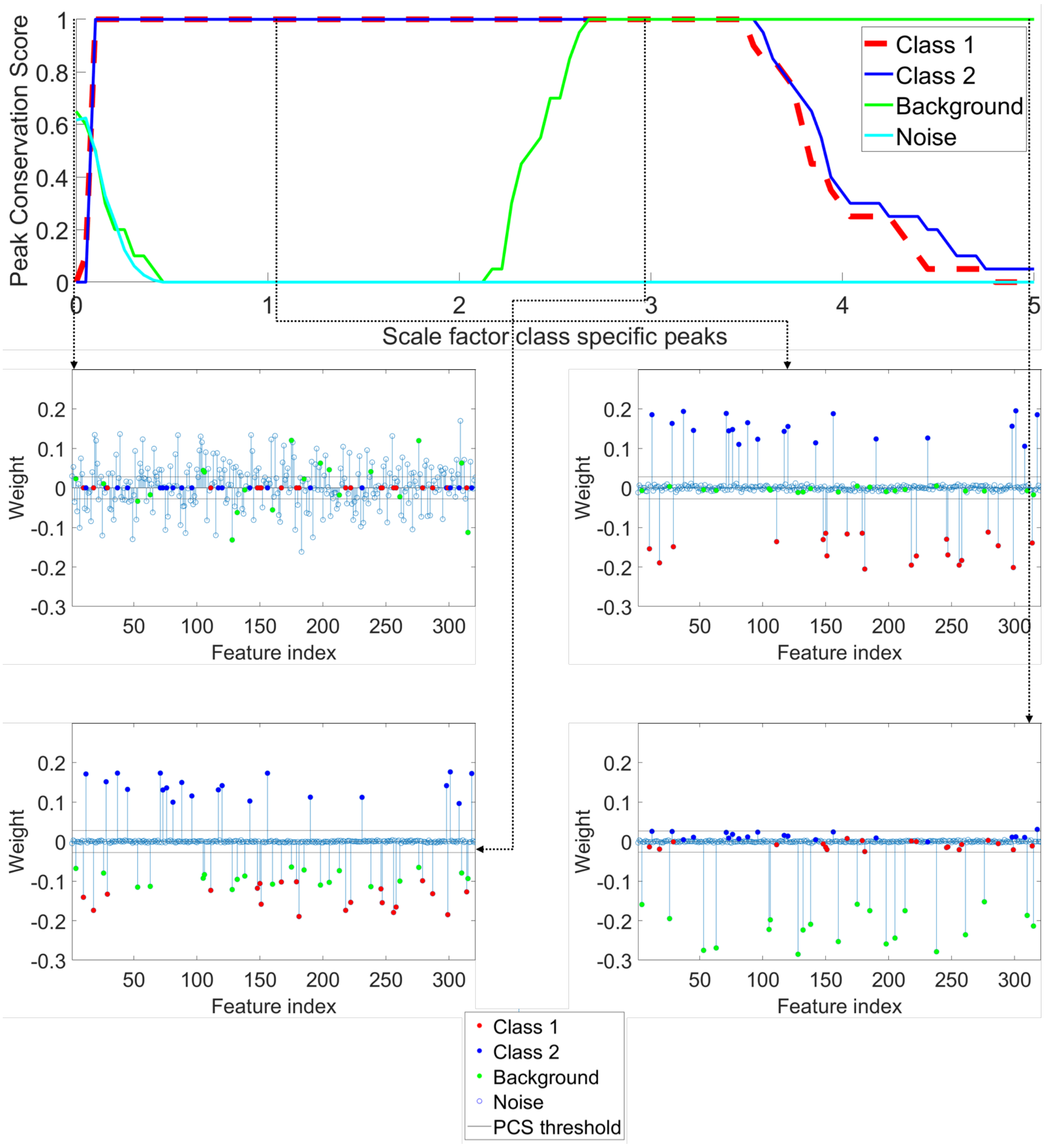


Figure 5-19: Principal component 2: Peak conservation score with corresponding feature-weight plots The top figure shows the PCS for all peak-classes with varying scale factors of the class specific peaks. Feature weight plots are shown for scale factors zero, one,three and five. The arrows indicate which feature-weight plot corresponds to which scale factor.

PCA - Principal component 3 Figure 5-20 shows the analysis for the third principal component. The top-left feature weight plot shows the case where the scaling factor is zero. Similar to the second component, primarily the peaks due to noise are conserved. Interestingly, when the scaling factor rises a little bit, the class-specific peaks start to get conserved as seen in the top right feature-weight plot. For all scaling factors above one, the third component conserves all class-specific peaks and discards all other peaks, as shown in the bottom feature-weight plot. Previously we described this behaviour as the ideal scenario. However, the class-specific peaks are all on the same (positive) side of the plot, failing to distinguish between the classes. This fact makes these feature-weights less desirable than the ones generated by SDM for high scale factors.

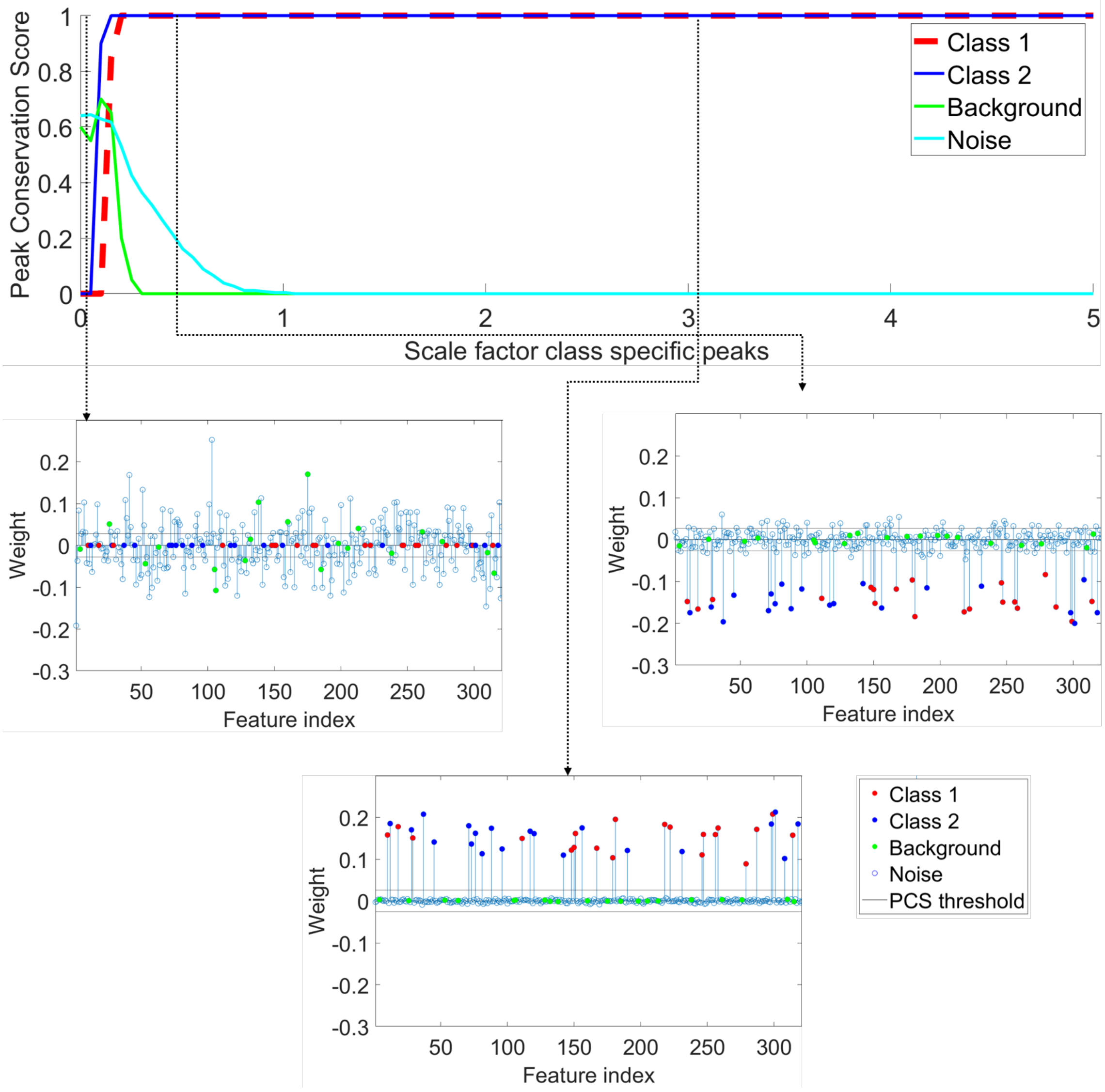


Figure 5-20: Principal component 3: Peak conservation score with corresponding feature-weight plots The top figure shows the PCS for all peak-classes with varying scale factors of the class specific peaks. Feature weight plots are shown for scale factors zero, 0.5 and three. The arrows indicate which feature-weight plot corresponds to which scale factor.

Discussion This experiment showcases the use of the proposed analysis framework in making conclusions about the peak height of the class-specific peaks. This framework provides a more direct analysis of the performance of a linear feature extraction method instead of an analysis on the classification performance alone, which is inherently also dependent on the used classifier. When a method succeeds in conserving the class-specific peaks and discarding the common patterns and noise while building its lower-dimensional subspace, the resulting feature-weights have the added benefit of identifying important features, making supervised dimensionality reduction a potentially powerful tool for biomarker discovery.

It is clear from the PCS analysis that SDM can use class information to conserve only the class-specific peaks with sufficient variance in the class-specific peaks. PCA is, in some situations, also able to conserve the class-specific peaks. However, the PCA approach does introduce a new problem in having to identify which of its principal component dimensions captures the class-specific behaviour of the dataset.

Note that SDM needs only one feature to conserve the class-specific peak conservation where PCA needs multiple features. When comparing principal component 1 with principal component 2, the change in variance of the class-specific features causes a shift in class-specific peak conservation from the second to the first component. Hence it is hard to determine, without available ground truth in real-world datasets, which component, if any, captures the class-specific features.

Future work This analysis approach provides a means for research into linear feature extraction methods. In this experiment, we have looked at the influence of height (and therefore also variance) of the class-specific peaks on the PCS. The experiment is easy to extend to look at a wide array of other dataset characteristics such as the number of classes, dimensionality of the original dataset and signal to noise ratio. In this experiment, we only looked at the comparison between SDM and PCA. However, this same approach can be used to analyze any other linear feature extraction method as well.

Chapter 6

Conclusions and future work

6-1 Main conclusions

In this thesis, we have explored the use of a relatively new linear feature extraction method Soft Discriminant Map (SDM) introduced by Liu and Gillies [9]. The method was extended to set the previously user-specified parameter β in a data-driven way, referring to the extension as Data-Driven Soft Discriminant Map (DD-SDM).

The (extended) method was evaluated in different ways. The experiments cover the effect on classification performance, the influence of β on the extracted features, and the computational efficiency of DD-SDM in finding the optimal β for the given data. In the last experiment, an artificial dataset is used in combination with a new performance metric, the Peak Conservation Score (PCS), to evaluate the method on its ability to conserve class-specific features. The following sections will state the conclusions specific to each of the experiments.

Classification By applying SDM on a real-world Imaging Mass Spectrometry (IMS) dataset, a comparison can be made of the classification performance between SDM and other conventional linear feature extraction methods such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). We showed that using the features extracted with DD-SDM, a better classification performance could be achieved when the dataset is reduced to a lower-dimensional subspace than with PCA and LDA being used for that same purpose.

Using the features extracted by SDM with a K Nearest Neighbours (K-NN) classifier, resulted (in our specific setup) in an improved test error above that achieved by using PCA or LDA as a dimensionality reduction method. The experiment showed that LDA is more prone to overfitting than DD-SDM. This overfitting was observed by the more significant difference between the training and test error. This observation suggests that SDM, with a β set by our extension DD-SDM, succeeds in controlling the amount of overfitting, successfully decreasing the test error.

The minimal test error using DD-SDM is reached with a reduced subspace of $K = 1$. Using the class labels, DD-SDM succeeds in capturing the class separation data in 1 feature. This behaviour suggests that DD-SDM behaves like LDA, which also extracts a maximum of $C - 1$ number of features.

Influence of β SDM has been applied to several real-world IMS datasets. Two scenarios have been studied: a high dimensional setting where $M > N_1$ and a low dimensional setting where $M < N_1$. The original paper by Liu and Gillies [9] did not consider the latter scenario. In both cases, an optimal value for β has been observed in terms of the test error. When β is set lower than the optimum determined by DD-SDM, it only maximizes the distance between class means, allowing for overlap between classes when the variance within classes is high.

For values above the optimum, SDM minimizes the variance within the classes, allowing the means of the classes to become very close to each other. In the case where $M > N$, SDM can extract a feature that minimizes the within-class variance to close to zero. This results in the data points within a class in the training set to be artificially close to each other. Therefore, when the data points in the test dataset are projected on this new feature, high overlap occurs, indicating the SDM-produced subspace is prone to overfitting.

If $M < N$ and β is set to a high value, SDM no longer succeeds in separating the classes in the training dataset. By minimizing the variance within the classes, the distance between class-means is also low, creating a high overlap of the classes in both the training- and test dataset. We can conclude that minimizing within-class variance alone is a sub-optimal way to extract features that separate classes well.

Data-driven Soft Discriminant maps An extension to SDM has been proposed to avoid setting the β parameter by hand. The method calculates SDM repeatedly with different values for β, choosing the β with the lowest test error. The values of β to be evaluated are picked using the golden-section search method. It was shown that, using this scheme, the number of iterations necessary to reach the optimal β was significantly reduced when compared to a more naive grid search method. However, in this experiment, SDM still had to be calculated 17 times to obtain the optimal β. In a real-world scenario, this could still prove to be too computationally expensive, depending on the particularities of the dataset at hand.

Class-specific feature conservation To evaluate SDM in terms of its ability to actively select class-specific features while disregarding non-class-specific variation in the dataset while constructing its subspace, a novel evaluation pipeline was constructed using a synthetic dataset. A method for generating artificial peak-picked IMS datasets was described. The dataset was constructed from first principles, mimicking the characteristics of a simple Imaging Mass Spectrometry dataset. A new performance metric was introduced to quantify the ability of SDM to conserve class-specific peaks in this artificial dataset; the Peak Conservation Score (PCS). SDM was compared to PCA in its ability to extract features that conserve the class-specific peaks when these peaks are varied in intensity. Since no guarantees can be given for the artificial dataset's representativeness of real-world datasets, only qualitative conclusions are made here. We concluded that SDM can actively select for class-specific peaks and discard the background non-class-specific and noise peaks when the variance within the class-specific peaks is sufficiently high for them to be discerned.

6-2 Recommendations for future work

This thesis proposed and analyzed an extension to SDM to set β in a data driven manner. The provided analysis can furthermore be extended in the following ways to gain a deeper understanding of the performance of SDM in general:

1. Investigate which classes of classifiers benefit the most of the extracted features of SDM.
2. Validate the findings in this thesis by applying DD-SDM to more real-world datasets with other characteristics, *e.g.* datasets with more classes.
3. Benchmark DD-SDM to other Dimensionality Reduction (DR) methods other than PCA and LDA, *e.g.* with non-linear feature extraction methods.
4. Research the ability of DD-SDM to be used as a tool for biomarker discovery.

DD-SDM can possibly get stuck in a local minimum. Attempts to improve the method to avoid that could include:

1. Use Brent's method[67] instead of golden section search. By aiming to fit parabolas on the minimization function, an optimum could be reached in fewer evaluations of the test error.
2. Implement a multi-start approach to avoid getting stuck in a local minimum. DD-SDM could be initialized with different upper bounds of the search interval and return the β corresponding to the lowest test error.
3. Instead of a multi-start approach, the bracketing algorithm described by Press et al. [67] could also be investigated to effectively set a reasonable initial search interval.

The implemented artificial dataset and the Peak Conservation Score (PCS) can be used to analyze linear feature extraction techniques in general. This thesis only varied the peak height of the class-specific peaks. However, the artificial dataset could be manipulated in other ways to analyze other dataset characteristics' effect on the PCS. Parameters that could be varied are, for example:

1. Number of classes;
2. Dimensionality of the training dataset;
3. Signal-to-noise ratio;
4. Number of class-specific or background peaks;
5. The linear feature extraction method to be analyzed.

This thesis has extended the linear feature extraction method SDM to set the user-supplied parameter β automatically. We have shown that DD-SDM successfully captures class-specific peaks, resulting in a reduced feature space that can improve classification performance when applied to a real-world IMS dataset. By improving DR methods, large datasets, not limited to those generated by IMS, can be more successfully reduced in size while retaining the vital information within. The resulting improved classification performance aids in retrieving actionable information from datasets that will continue to grow in size in the coming years.

Appendix A

Appendix

A-1 Feature selection

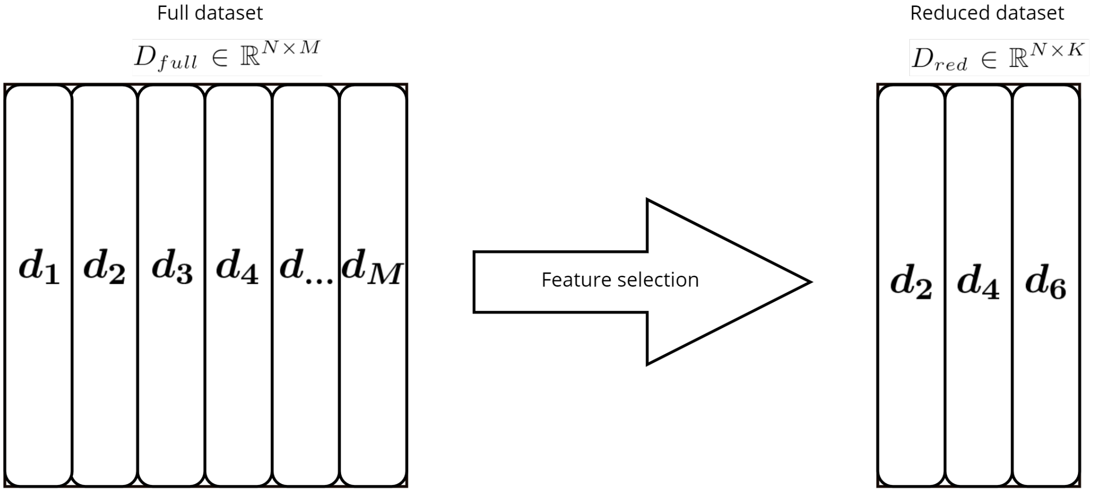


Figure A-1: Schematic representation of feature selection Feature selection reduces the dimensionality of the full dataset D_{full} by selecting a subset of its columns and putting them in the reduced dataset $D_{red} \subset D_{full}$. In this example the algorithm selected three columns ($K = 3$): $\boldsymbol{d_2}, \boldsymbol{d_4}$ and $\boldsymbol{d_6}$.

One family of Dimensionality Reduction (DR) methods is called feature selection. Figure A-1 shows the full dataset D_{full} with columns $[\boldsymbol{d_1}, \boldsymbol{d_2}, \boldsymbol{d_3}, \dots, \boldsymbol{d_M}] \in \mathbb{R}^{N \times 1}$. A feature selection algorithm constructs the reduced dataset D_{red} with a subset of these columns: $D_{red} \subset D_{full}$. In this example three columns are selected by the algorithm: $\boldsymbol{d_2}, \boldsymbol{d_4}$ and $\boldsymbol{d_6}$. Feature selection algorithms aim to find an optimal subset of the original features by eliminating irrelevant, redundant and noisy features [58]. The concepts of relevance and redundancy are further explored in section 2-3-1-5.

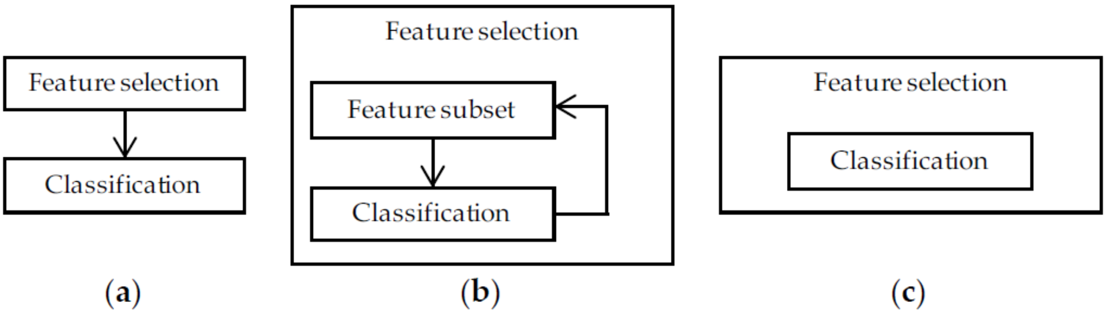


Figure A-2: Overview of feature selection methods(a) Filter, (b) wrapper, and (c) embedded feature selection methods. Filter methods perform the feature selection independently of construction of the classification model. Wrapper methods iteratively select or eliminate a set of features using the prediction accuracy of the classification model. In embedded methods the feature selection is an integral part of the classification. Source of image: Supper *et al.* [71]

There are many algorithms present in literature for finding the optimal subset of features. In the following section these strategies are categorized and examples are given. In literature there is a consensus that feature selection methods can be divided into three [72].

1. **Filters** extract features from the data without any learning involved.
2. **Wrappers** use learning techniques to evaluate which features are useful.
3. **Embedded** techniques combine the feature selection step and the classifier construction.

It is also possible to combine a filter and wrapper method, this is referred to as a **hybrid** method.

A-1-1 Filtering methods

Characterization The filtering feature selection techniques asses the relevance of the features by only looking at the intrinsic properties of the data [73]. In most cases, the features are ranked on a statistical measure and the lowest ranking features are removed. Afterwards, the top-ranking features are used as input for a classification algorithm.

Pros and cons Advantages of filter methods are that they easily scale to large datasets, since they are computationally simple and fast. Since there is no classifier involved, the generated feature subset is classifier independent. After the subset has been generated, multiple classifiers can be evaluated to choose the one which performs the best. However, the resulting prediction error will be generally lower than the following categories, since filter methods ignore the interaction with the classifier. The fast filter methods are often univariate, which means that feature interdependencies are ignored. The complications of using univariate methods is further discussed in section A-1-3-2.

Categories To overcome the pitfalls of univariate methods, a number of multi-variate filters were introduced to incorporate feature dependencies to some degree. In this thesis the focus will lie on dimensionality reduction techniques for supervised classification. However, there are also unsupervised filter methods that do not use class labels to select features to use in a clustering task.

Examples Some commonly used univariate methods are information gain [74], reliefF [75], and Fisher score [76]. Examples of multi-variate methods are minimum redundancy maximum relevance (mRmR) [77], correlation-based feature selection(CFS) [78] and multi-cluster feature selection(MCFS) [79]. The fisher score will be explored as an example in the following section.

A-1-1-1 Example: Fisher Score

A popular example of a filter method is fisher scoring. The features are given a score based on a statistical measure one at a time. The features are ranked according to these scores. The top-scoring features are selected and put in a representative dataset to obtain dimensionality reduction.

The fact that a score is calculated for every feature independently makes the fisher criterion a univariate method. As a consequence, fisher scoring does not take feature interdependencies into account. Multivariate variants of the fisher score which jointly selects features are constructed for this reason. An example of such a variant is the generalized fisher score [76]. A further discussion on the specific pitfalls of univariate methods is provided in Section A-1-3-2.

Since fisher scoring is a relatively fast method, it's performance is often used as a benchmark to compare against more sophisticated methods [80]. The computational complexity of the method is $\mathcal{O}(N \times M)$, suggesting that the computation time scales only linearly with the amount of dimensions and data points, making it suitable for very large datasets. [81]

The main idea of the fisher score is to give a high score to features having large separation between classes and small variance within classes. Since the method uses class information it classifies as a supervised feature selection method.

If we have a full dataset $D_{full} \in \mathbb{R}^{N \times M}$ with N data points with label $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$, $y \in \{1, \dots, c\}$.. Let n_i be the number of data points in class i. Let μ_i and σ_i^2 be the mean and variance of class i, $i = 1, ..., c$ corresponding to the rth feature. Let μ and σ^2 be the mean and variance of all classes of feature r combined. The fisher score for feature r is given by:

$$F_r = \frac{\sum_{i=1}^{c} n_i(\mu_i - \mu)^2}{\sum_{i=1}^{c} n_i \sigma_i^2} \tag{A-1}$$

The numerator becomes large when the means of the classes lie far apart. The denominator becomes small when the variance within a class is low. The class size is present in both terms to account for unequal class sizes.

Dimensionality reduction is achieved by choosing the top K scoring features and stacking them to produce a reduced dataset $D_{reduced} \in \mathbb{R}^{N \times K}$.

A-1-2 Wrapper methods

The second category in the feature selection domain contains the wrapper methods. This category is characterized by the involvement of a classifier to directly evaluate feature subsets. The search through possible feature subsets and the evaluation of them are decoupled, making it possible to use any classifier for the evaluation.

Search methods The most straightforward way to find the optimal subset is by evaluating all possible subsets, in the literature referred to as an exhaustive search. There are 2^M possible feature subsets possible with M being the number of features in a dataset. The evaluation of all these possible subsets will quickly become infeasible since the number of possible feature subsets grows exponentially with the number of features in the dataset [56].

To prevent going through all possible subsets, different search methods are developed in the literature. These methods can be divided into two categories: sequential and heuristic procedures. Sequential algorithms are deterministic in nature and search the feature space by adding or removing features sequentially. An example of a heuristic method is Sequential Backward Selection and is discussed in section A-1-2-1.

The heuristic approaches are probabilistic in nature and are mainly evolutionary algorithms such as Particle Swarm Optimization(PSO), Ant Colony Optimization, Genetic Algorithms and others [82]. An overview of the most used search methods can be found in [72].

Heuristic approaches generally consider a higher number of possible feature subsets. Hence, they are more computationally expensive than sequential methods. Furthermore, they are more sensitive to overfitting the data. However, they are less prone to obtain stuck in a local minimum. A comparison between the deterministic sequential and probabilistic heuristic methods is shown in Table A-1. [83]

Table A-1: Comparison between sequential and heuristic search methods Recreated from Hira and Gillies [83]

Sequential	Heuristic
Small overfitting risk	High overfitting risk
Prone to local optima	Less prone to local optima
Classifier dependent	Classifier dependent
-	Computationally intensive

.

Evaluation of a subset A classifier evaluates every subset generated by the search strategy. Cross-validation ensures to obtain an almost unbiased estimate for the prediction accuracy as described in Section 2-2-3-2. After each subset has received a score, the search strategy will use this information to generate the subset for the next iteration.

Depending on the search strategy used, the algorithm will terminate after a predefined number of features is reached, a pre-defined number of iterations is completed or when the classification error does not increase anymore with a certain threshold amount.

Pros and Cons Since the evaluation of a feature set involves learning a classifier on the data, wrapper methods are generally more computationally expensive than filter methods. The chosen classifier should not be too complex to train since this has an immediate negative effect on the computation time.

Leading to the next point: the subsets returned by the wrapper feature selection routine depend on the classifier used. They will therefore not ensure that the subset of features will perform equally well when used thereafter in combination with another classifier in the learning stage. This can be seen as a positive and negative point. This also means that the subset returned is optimized for prediction accuracy and will tend to perform better than feature selection methods without any learning involved *e.g.* the Fisher scoring [83]. This is to be expected because wrapper methods do take feature interdependencies into account as opposed to univariate filter methods and directly use the prediction accuracy to select features.

A-1-2-1 Sequential Backward Selection

A well-known wrapper method is called Sequential Backward Selection(SBS). This method sequentially eliminates the worst-performing feature from the feature set. The routine will continue until the feature set has the number of features defined *a priori*.

To determine the currently worst performing feature, each feature is removed from the full set one at a time, creating M subsets with $M - 1$ features. Next, each subset is used to train a classifier and evaluated using a cross-validation routine. The score obtained is used as a performance measure to evaluate the subsets. The feature that was left out of the best scoring subset is eliminated permanently and the subset is used in the next iteration. When the subset reaches a pre-defined size the routine terminates and outputs the current subset.

Once a feature is removed, it has no possibility of obtaining in the final feature set. Therefore, the subsets generated by each step are nested:

$$X_k \subset X_{k-1} \subset X_{k-2} \subset X_{k-\ldots} \subset X_1 \tag{A-2}$$

The nested structure increases the risk of being trapped in a sub-optimal solution [84]. In an attempt to reduce this risk, a variant of SBS was introduced by Novovicova and Kittler in 1994 called Sequential Backward Floating Selection (SBFS) [85], which allows the algorithm to reconsider eliminated features. This technique increases the number of subsets considered and thereby increases the computational load.

A-1-3 Embedded methods

The final category feature selection methods are embedded methods. These techniques try to overcome the computational load of wrapper methods by incorporating the feature selection step with the model building step. A classifier is trained on the training data and, dependent on the method used, the model is 'opened up' to view which features were most important in creating the decision boundary. The obvious advantage of this technique is that the model has to be built once, instead of the vast amount of models that are trained with wrapper methods. Similar to wrapper methods, the selected feature set depends on the classifier used, making the feature set sub optimal if used with a different classifier in the prediction stage. However, since the model hypothesis is involved in the feature selection process, embedded methods generally perform better in terms of classification error than filter methods.

An example of an embedded technique is feature selection via concave minimization(FSV) [86], where the selection process is injected into the training of an SVM by a linear programming technique. Other examples are Relevant Sample-Feature Machine (RSFM) [87] and Random Forest [88]. The latter is explained in the following section.

A-1-3-1 Random Forest

Random forest is a popular choice of classifier since it generally provides a good predictive performance,low overfitting and easy interpretability. The easy interpretability comes from the fact that it is straightforward to derive the importance of each feature on the prediction made by the classifier.

Feature selection Random forest classifiers combine a high number of decision trees to construct a decision boundary. Each tree is built from a random subset of features and a random subset of data points. A decision tree is built by considering each feature sequentially and defining the optimal threshold for that feature by optimizing its so called impurity. In classification, the impurity measure is the Gini impurity, which is measure for the amount of misclassified data points in a decision tree. Using this, it is possible to calculate how much each feature decreases the impurity. The more a feature decreases the impurity, the more important a feature is. With random forests, the impurity decrease from each feature can be averaged across the trees to determine the final importance of a feature. Sandri *et al.* [88] give a mathematical description of this process.

A-1-3-2 Comparison univariate- vs. multivariate methods

As discussed in Section A-1-1, univariate filter methods have the advantage to be computationally cheap. They are successful in ranking the features on relevance when only the considered feature would be used in training a classifier.

However, univariate methods do not take feature interdependencies into account. The following example retrieved from Jones et al. [5] shows that features that score under par on univariate tests are able to separate the classes perfectly when they are combined.

Example An artificial dataset is considered with six features. In Figure A-3 the features are shown in separate plots. In each plot the red dots indicate class A and the blue dots class B. Univariate methods will examine these features independently and score them based on statistical measures for separability. Under each plot three of these measures are given. $p(t)$ gives the p-value of a t-test, $p(w)$ that of the Wilcoxon test and AUROC indicates the area under the receiver operator curve. In this example, the ranking based on the AUROC would be in decreasing relevancy: 1,3,5,6,2,4. It could be stated that only feature one separates the classes moderately well and that the others overlap too much to generate a reasonable decision threshold.

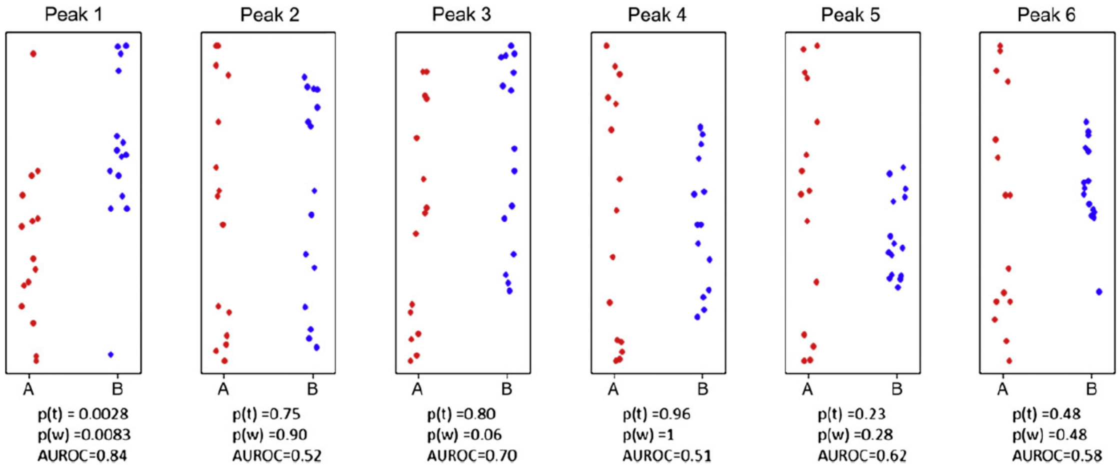


Figure A-3: Scatter plots for the simulated dataset of six MS peaks The six features shown are evaluated independently based on three statistical measures for class separability: The p-value of a t-test, p(t), the p-value of the Wilcoxon test, p(w) and the area under the receiver operator curve (AUROC). The red dots belong to class A and the blue dots belong to class B. Source of image: Jones et al. [5]

However, it is shown in Figure A-4 that by combining feature 2 and 3, the classes can be separated perfectly. This shows that univariate methods could undervalue powerful features that score well when combined with each other.

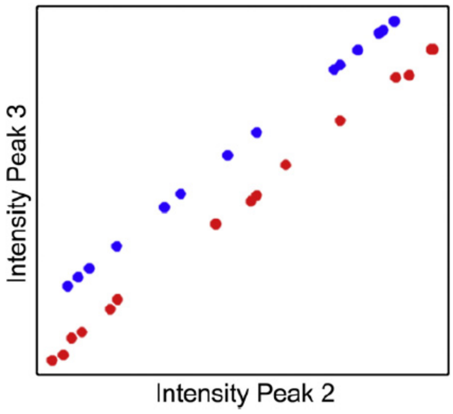


Figure A-4: Scatter plot of features two and three When features two and three are plotted together, the classes become fully separable. Source of image: Jones et al. [5]

Bibliography

[1] S. Meding et al., “Tumor Classification of Six Common Cancer Types Based on Proteomic Profiling by MALDI Imaging”, *Journal of Proteome Research* **2012**, *11*, 1996–2003, DOI 10.1021/pr200784p.

[2] B. Balluff et al., “Classification of HER2/neu Status in Gastric Cancer Using a Breast-Cancer Derived Proteome Classifier”, *Journal of Proteome Research* **2010**, *9*, 6317–6322, DOI 10.1021/pr100573s.

[3] J. A. Bauer et al., “Identification of markers of taxane sensitivity using proteomic and genomic analyses of breast tumors from patients receiving neoadjuvant paclitaxel and radiation”, *Clinical Cancer Research: An Official Journal of the American Association for Cancer Research* **2010**, *16*, 681–690, DOI 10.1158/1078-0432.CCR-09-1091.

[4] B. Balluff et al., “De novo discovery of phenotypic intratumour heterogeneity using imaging mass spectrometry”, *The Journal of Pathology* **2015**, *235*, 3–13, DOI 10.1002/path.4436.

[5] E. A. Jones et al., “Imaging mass spectrometry statistical analysis”, *Journal of Proteomics*, Special Issue: Imaging Mass Spectrometry: A Users Guide to a New Technique for Biological and Biomedical Research **2012**, *75*, 4962–4989, DOI 10.1016/j.jprot.2012.06.014.

[6] G. B. Eijkel et al., “Correlating MALDI and SIMS imaging mass spectrometric datasets of biological tissue surfaces”, *Surface and Interface Analysis* **2009**, *41*, 675–685, DOI 10.1002/sia.3088.

[7] N. Verbeeck, R. M. Caprioli, R. V. d. Plas, “Unsupervised machine learning for exploratory data analysis in imaging mass spectrometry”, *Mass Spectrometry Reviews* **2019**, *0*.

[8] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed., Springer-Verlag, New York, **2002**, DOI 10.1007/b98835.

[9] R. Liu, D. F. Gillies, “Overfitting in linear feature extraction for classification of high-dimensional image data”, *Pattern Recognition* **2016**, *53*, 73–86, DOI 10.1016/j.patcog.2015.11.015.

[10] D. Cai, X. He, J. Han in 2008 IEEE 24th International Conference on Data Engineering, 2008 IEEE 24th International Conference on Data Engineering, ISSN: 2375-026X, **2008**, pp. 209–217, DOI 10.1109/ICDE.2008.4497429.

[11] M. Aichler, A. Walch, “MALDI Imaging mass spectrometry: current frontiers and perspectives in pathology research and practice”, *Laboratory Investigation* **2015**, *95*, 422–431, DOI 10.1038/labinvest.2014.156.

[12] K. J. Boggio et al., “Recent advances in single-cell MALDI mass spectrometry imaging and potential clinical impact”, *Expert Review of Proteomics* **2011**, *8*, 591–604, DOI 10.1586/epr.11.53.

[13] R. J. Goodwin, “Sample preparation for mass spectrometry imaging: Small mistakes can lead to big consequences”, *Journal of Proteomics* **2012**, *75*, 4893–4911, DOI https://doi.org/10.1016/j.jprot.2012.04.012.

[14] V. I. Slaveykova et al., “Dynamic NanoSIMS ion imaging of unicellular freshwater algae exposed to copper”, *Analytical and Bioanalytical Chemistry* **2009**, *393*, 583–589, DOI 10.1007/s00216-008-2486-x.

[15] M. R. Groseclose et al., “High-throughput proteomic analysis of formalin-fixed paraffin-embedded tissue microarrays using MALDI imaging mass spectrometry”, *Proteomics* **2008**, *8*, DOI https://doi.org/10.1002/pmic.200800495.

[16] S. Khatib-Shahidi et al., “Direct Molecular Analysis of Whole-Body Animal Tissue Sections by Imaging MALDI Mass Spectrometry”, *Analytical Chemistry* **2006**, *78*, 6448–6456, DOI 10.1021/ac060788p.

[17] P. Chaurand et al., “Monitoring Mouse Prostate Development by Profiling and Imaging Mass Spectrometry”, *Molecular & Cellular Proteomics* **2008**, *7*, 411–423, DOI 10.1074/mcp.M700190-MCP200.

[18] R. J. A. Goodwin et al., “Time-dependent evolution of tissue markers by MALDI-MS imaging”, *PROTEOMICS* **2008**, *8*, 3801–3808, DOI 10.1002/pmic.200800201.

[19] R. J. A. Goodwin et al., “Use of a Solvent-Free Dry Matrix Coating for Quantitative Matrix-Assisted Laser Desorption Ionization Imaging of 4-Bromophenyl-1,4-diazabicyclo(3.2.2)nonane-4-carboxylate in Rat Brain and Quantitative Analysis of the Drug from Laser Microdissected Tissue Regions”, *Analytical Chemistry* **2010**, *82*, 3868–3873, DOI 10.1021/ac100398y.

[20] L. Cole et al., “Investigation of protein induction in tumour vascular targeted strategies by MALDI MSI”, *Methods* **2011**, *54*, 442–453, DOI 10.1016/j.ymeth.2011.03.007.

[21] M. R. Groseclose et al., “Identification of proteins directly from tissue:in situ tryptic digestions coupled with imaging mass spectrometry”, *Journal of Mass Spectrometry* **2007**, *42*, 254–262, DOI 10.1002/jms.1177.

[22] P. Chaurand et al., “Integrating Histology and Imaging Mass Spectrometry”, *Analytical Chemistry* **2004**, *76*, 1145–1155, DOI 10.1021/ac0351264.

[23] J. Franck et al., “On-Tissue N-Terminal Peptide Derivatizations for Enhancing Protein Identification in MALDI Mass Spectrometric Imaging Strategies”, *Analytical Chemistry* **2009**, *81*, 8305–8317, DOI 10.1021/ac901043n.

[24] M. Karas et al., “Matrix-assisted ultraviolet laser desorption of non-volatile compounds”, *International Journal of Mass Spectrometry and Ion Processes* **1987**, *78*, 53–68, DOI 10.1016/0168-1176(87)87041-6.

[25] R. M. Caprioli, T. B. Farmer, J. Gile, “Molecular Imaging of Biological Samples: Localization of Peptides and Proteins Using MALDI-TOF MS”, *Analytical Chemistry* **1997**, *69*, 4751–4760, DOI 10.1021/ac970888i.

[26] L. MacAleese, J. Stauber, R. M. A. Heeren, “Perspectives for imaging mass spectrometry in the proteomics landscape”, *PROTEOMICS* **2009**, *9*, 819–834, DOI 10.1002/pmic.200800363.

[27] K. Dreisewerd, “Recent methodological advances in MALDI mass spectrometry”, *Analytical and Bioanalytical Chemistry* **2014**, *406*, 2261–2278, DOI 10.1007/s00216-014-7646-6.

[28] L. McDonnell, R. Heeren, “Imaging mass spectrometry”, *Mass Spectrometry Reviews* **2007**, *26*, 606–643, DOI 10.1002/mas.20124.

[29] A. R. Buchberger et al., “Mass Spectrometry Imaging: A Review of Emerging Advancements and Future Insights”, *Analytical Chemistry* **2018**, *90*, 240–265, DOI 10.1021/acs.analchem.7b04733.

[30] L. M. Cole, M. R. Clench, “Mass spectrometry imaging for the proteomic study of clinical tissue”, *PROTEOMICS Clinical Applications* **2015**, *9*, 335–341, DOI 10.1002/prca.201400103.

[31] R. Aebersold, M. Mann, “Mass spectrometry-based proteomics”, *Nature* **2003**, *422*, 198–207, DOI 10.1038/nature01511.

[32] W. C. Wiley, I. H. McLaren, “TimeofFlight Mass Spectrometer with Improved Resolution”, *Review of Scientific Instruments* **1955**, *26*, 1150–1157, DOI 10.1063/1.1715212.

[33] A. G. Marshall, C. L. Hendrickson, G. S. Jackson, “Fourier transform ion cyclotron resonance mass spectrometry: A primer”, *Mass Spectrometry Reviews* **1998**, *17*, 1–35, DOI 10.1002/(SICI)1098-2787(1998)17:1<1::AID-MAS1>3.0.CO;2-K.

[34] J. Kriegsmann, M. Kriegsmann, R. Casadonte, “MALDI TOF imaging mass spectrometry in clinical pathology: A valuable tool for cancer diagnostics (Review)”, *International Journal of Oncology* **2015**, *46*, 893–906, DOI 10.3892/ijo.2014.2788.

[35] A. Römpp, B. Spengler, “Mass spectrometry imaging with high resolution in mass and space”, *Histochemistry and Cell Biology* **2013**, *139*, 759–783, DOI 10.1007/s00418-013-1097-6.

[36] P. Chaurand, “Imaging mass spectrometry of thin tissue sections: A decade of collective efforts”, *Journal of Proteomics*, Special Issue: Imaging Mass Spectrometry: A Users Guide to a New Technique for Biological and Biomedical Research **2012**, *75*, 4883–4892, DOI 10.1016/j.jprot.2012.04.005.

[37] K. Schwamborn, R. M. Caprioli, “Molecular imaging by mass spectrometry looking beyond classical histology”, *Nature Reviews Cancer* **2010**, *10*, 639–646, DOI 10.1038/nrc2917.

[38] K. Gorzolka, A. Walch, “MALDI mass spectrometry imaging of formalin-fixed paraffin-embedded tissues in clinical research”, *Histology and Histopathology* **2014**, *29*, 1365–1376, DOI 10.14670/HH-29.1365.

[39] Y. Fujimura, D. Miura, “MALDI Mass Spectrometry Imaging for Visualizing In Situ Metabolism of Endogenous Metabolites and Dietary Phytochemicals”, *Metabolites* **2014**, *4*, 319–346, DOI 10.3390/metabo4020319.

[40] C. Eriksson et al., “MALDI Imaging Mass Spectrometry-A Mini Review of Methods and Recent Developments”, *Mass Spectrometry (Tokyo Japan)* **2013**, *2*, S0022, DOI 10.5702/massspectrometry.S0022.

[41] P. M. Angel, R. M. Caprioli, “Matrix-Assisted Laser Desorption Ionization Imaging Mass Spectrometry: In Situ Molecular Mapping”, *Biochemistry* **2013**, *52*, 3818–3828, DOI 10.1021/bi301519p.

[42] A. Thomas et al., “Mass spectrometry for the evaluation of cardiovascular diseases based on proteomics and lipidomics”, *Thrombosis and Haemostasis* **2011**, *106*, 20–33, DOI 10.1160/TH10-12-0812.

[43] M. M. Gessel, J. L. Norris, R. M. Caprioli, “MALDI imaging mass spectrometry: spatial molecular analysis to enable a new age of discovery”, *Journal of Proteomics* **2014**, *107*, 71–82, DOI 10.1016/j.jprot.2014.03.021.

[44] J. L. Norris, R. M. Caprioli, “Analysis of Tissue Specimens by Matrix-Assisted Laser Desorption/Ionization Imaging Mass Spectrometry in Biological and Clinical Research”, *Chemical Reviews* **2013**, *113*, 2309–2342, DOI 10.1021/cr3004295.

[45] N. Verbeeck et al., “Connecting imaging mass spectrometry and magnetic resonance imaging-based anatomical atlases for automated anatomical interpretation and differential analysis”, *Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics*, MALDI Imaging **2017**, *1865*, 967–977, DOI 10.1016/j.bbapap.2017.02.016.

[46] N. E. Mascini et al., “Tumor classification with MALDI-MSI data of tissue microarrays: A case study”, *Methods*, Health Informatics and Translational Data Analytics **2018**, *151*, 21–27, DOI 10.1016/j.ymeth.2018.04.004.

[47] N. Kurabe et al., “Accumulated phosphatidylcholine (16:0/16:1) in human colorectal cancer; possible involvement of LPCAT4”, *Cancer Science* **2013**, *104*, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cas.12221, 1295–1302, DOI 10.1111/cas.12221.

[48] Z. Zhou, R. N. Zare, “Personal Information from Latent Fingerprints Using Desorption Electrospray Ionization Mass Spectrometry and Machine Learning”, *Analytical Chemistry* **2017**, *89*, 1369–1372, DOI 10.1021/acs.analchem.6b04498.

[49] V. Vapnik, *Statistical Learning Theory*, WileyInterscience, **1998**.

[50] L. Breiman, “Random Forests”, *Machine Learning* **2001**, *45*, 5–32, DOI 10.1023/A:1010933404324.

[51] S. Theodoridis, K. Koutroumbas in *Pattern recognition (fourth edition)*, (Eds.: S. Theodoridis, K. Koutroumbas), Academic Press, Boston, **2009**, pp. 261–322, DOI https://doi.org/10.1016/B978-1-59749-272-0.50007-4.

[52] A. K. Jain, R. P. W. Duin, J. Mao, “Statistical Pattern Recognition: A Review”, *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 4–37, DOI 10.1109/34.824819.

[53] V. Spruyt, The Curse of Dimensionality in classification, **2014**, https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/ (visited on 10/07/2019).

[54] K. Beyer et al. in Database Theory ICDT99, (Eds.: C. Beeri, P. Buneman), Springer, Berlin, Heidelberg, **1999**, pp. 217–235, DOI 10.1007/3-540-49257-7_15.

[55] A. Zimek, E. Schubert, H.-P. Kriegel, “A survey on unsupervised outlier detection in high-dimensional numerical data”, *Statistical Analysis and Data Mining: The ASA Data Science Journal* **2012**, *5*, 363–387, DOI 10.1002/sam.11161.

[56] L. Yu, H. Liu, “Efficient Feature Selection via Analysis of Relevance and Redundancy”, *J. Mach. Learn. Res.* **2004**, *5*, 1205–1224.

[57] J. Li et al., “Feature Selection: A Data Perspective”, *ACM Comput. Surv.* **2017**, *50*, 94:1–94:45, DOI 10.1145/3136625.

[58] J. Miao, L. Niu, “A Survey on Feature Selection”, *Procedia Computer Science*, Promoting Business Analytics and Quantitative Management of Technology: 4th International Conference on Information Technology and Quantitative Management (ITQM 2016) **2016**, *91*, 919–926, DOI 10.1016/j.procs.2016.07.111.

[59] X. Xu et al., “Review of classical dimensionality reduction and sample selection methods for large-scale data processing”, *Neurocomputing*, Chinese Conference on Computer Vision 2017 **2019**, *328*, 5–15, DOI 10.1016/j.neucom.2018.02.100.

[60] B. Schölkopf, A. Smola, K.-R. Müller in Artificial Neural Networks ICANN'97, (Eds.: W. Gerstner et al.), Springer, Berlin, Heidelberg, **1997**, pp. 583–588, DOI 10.1007/BFb0020217.

[61] W. S. Torgerson, "Multidimensional scaling: I. Theory and method", *Psychometrika* **1952**, *17*, 401–419, DOI 10.1007/BF02288916.

[62] J. B. Tenenbaum, V. d. Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction", *Science* **2000**, *290*, 2319–2323, DOI 10.1126/science.290.5500.2319.

[63] L. A. Klerk et al., "Extended data analysis strategies for high resolution imaging MS: New methods to deal with extremely large image hyperspectral datasets", *International Journal of Mass Spectrometry*, Imaging Mass Spectrometry Special Issue **2007**, *260*, 222–236, DOI 10.1016/j.ijms.2006.11.014.

[64] A. M. Race et al., "Memory Efficient Principal Component Analysis for the Dimensionality Reduction of Large Mass Spectrometry Imaging Data Sets", *Analytical Chemistry* **2013**, *85*, 3071–3078, DOI 10.1021/ac302528v.

[65] AT&T Database of Faces, https://kaggle.com/kasikrit/att-database-of-faces (visited on 01/14/2020).

[66] R. Liu, "Feature extraction in classification", PhD thesis, Imperial College London, **2013**.

[67] W. H. Press et al., *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3rd ed., Cambridge University Press, USA, **2007**.

[68] J. M. Spraggins et al., "High-Performance Molecular Imaging with MALDI Trapped Ion-Mobility Time-of-Flight (timsTOF) Mass Spectrometry", *Analytical Chemistry* **2019**, *91*, Publisher: American Chemical Society, 14552–14560, DOI 10.1021/acs.analchem.9b03612.

[69] N. Verbeeck, "Datamining of Imaging Mass Spectrometry Data for Biomedical Tissue Exploration", PhD thesis, Leuven, **2014**.

[70] E. Pekalska, R. P. W. Duin, *PRTools*.

[71] A. Suppers, A. J. v. Gool, H. J. C. T. Wessels, "Integrated Chemometrics and Statistics to Drive Successful Proteomics Biomarker Discovery", *Proteomes* **2018**, *6*, 20, DOI 10.3390/proteomes6020020.

[72] A. Jovi, K. Brki, N. Bogunovi in 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), **2015**, pp. 1200–1205, DOI 10.1109/MIPRO.2015.7160458.

[73] Y. Saeys, I. Inza, P. Larrañaga, "A review of feature selection techniques in bioinformatics", *Bioinformatics* **2007**, *23*, 2507–2517, DOI 10.1093/bioinformatics/btm344.

[74] N. Hoque, D. Bhattacharyya, J. Kalita, "MIFS-ND: A mutual information-based feature selection method", *Expert Systems with Applications* **2014**, *41*, 6371–6385, DOI https://doi.org/10.1016/j.eswa.2014.04.019.

[75] M. Robnik-ikonja, I. Kononenko, "Theoretical and Empirical Analysis of ReliefF and RReliefF", *Machine Learning* **2003**, *53*, 23–69, DOI 10.1023/A:1025667309714.

[76] Q. Gu, Z. Li, J. Han, "Generalized Fisher Score for Feature Selection", *Computing Research Repository* **2012**, DOI arXiv:1202.3725.

[77] M. Radovic et al., "Minimum redundancy maximum relevance feature selection approach for temporal gene expression data", *BMC Bioinformatics* **2017**, *18*, 9, DOI 10.1186/s12859-016-1423-9.

[78] F. Azuaje, "Witten IH, Frank E: Data Mining: Practical Machine Learning Tools and Techniques 2nd edition", *BioMedical Engineering OnLine* **2006**, *5*, 51, DOI 10.1186/1475-925X-5-51.

[79] D. Cai, C. Zhang, X. He in Unsupervised feature selection for Multi-Cluster data, **2010**, pp. 333–342, DOI 10.1145/1835804.1835848.

[80] H. Frohlich, O. Chapelle, B. Scholkopf in Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence, Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence, ISSN: 1082-3409, **2003**, pp. 142–148, DOI 10.1109/TAI.2003.1250182.

[81] G. Roffo et al., "Infinite Latent Feature Selection: A Probabilistic Latent Graph-Based Ranking Approach", *Computing Research Repository* **2017**, DOI arXiv:1707.07538[cs].

[82] N. Abd-Alsabour in 2014 European Modelling Symposium, 2014 European Modelling Symposium, **2014**, pp. 20–26, DOI 10.1109/EMS.2014.28.

[83] Z. M. Hira, D. F. Gillies, "A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data", *Advances in Bioinformatics* **2015**, *2015*, DOI 10.1155/2015/198363.

[84] H. Mi et al., "Robust feature selection to predict tumor treatment outcome", *Artificial Intelligence in Medicine* **2015**, *64*, 195–204, DOI 10.1016/j.artmed.2015.07.002.

[85] P. Pudil, J. Novoviová, J. Kittler, "Floating search methods in feature selection", *Pattern Recognition Letters* **1994**, *15*, 1119–1125, DOI 10.1016/0167-8655(94)90127-9.

[86] P. Bradley, O. L. Mangasarian in Feature selection via concave minimization and support vector machines, Morgan Kaufmann, **1998**, pp. 82–90.

[87] Y. Mohsenzadeh et al., "The Relevance Sample-Feature Machine: A Sparse Bayesian Learning Approach to Joint Feature-Sample Selection", *IEEE Transactions on Cybernetics* **2013**, *43*, 2241–2254, DOI 10.1109/TCYB.2013.2260736.

[88] M. Sandri, P. Zuccolotto in Data Analysis, Classification and the Forward Search, (Eds.: S. Zani et al.), Springer, Berlin, Heidelberg, **2006**, pp. 263–270, DOI 10.1007/3-540-35978-8_30.

Glossary

List of Acronyms

MALDI	Matrix-Assisted Laser Desorption Ionization
IMS	Imaging Mass Spectrometry
MS	Mass Spectrometry
PCA	Principal Component Analysis
TMA	Tissue Microarrays
FT-ICR	Fourier transform ion cyclotron resonance
TOF	Time-of-Flight
SIMS	Secondary Ion Mass Spectrometry
SDM	Soft Discriminant Map
LDA	Linear Discriminant Analysis
DR	Dimensionality Reduction
PCS	Peak Conservation Score
IMS-MKDS	Imaging Mass Spectrometry Mouse Kidney Dataset
IMS-RBDS	Imaging Mass Spectrometry Rat brain Dataset
DD-SDM	Data-Driven Soft Discriminant Map
m/z	Mass-over-charge ratio
K-NN	K Nearest Neighbours