Functional Principal Component Analysis based Machine Learning Algorithms for Spectral Analysis

by

Yifeng Bie
B.Sc., Shanghai Normal University, 2018

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Functional Principal Component Analysis based Machine Learning Algorithms for Spectral Analysis

by

Yifeng Bie
B.Sc., Shanghai Normal University, 2018

Supervisory Committee

Dr. Tao Lu, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Lin Cai, Departmental Member
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. Tao Lu, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Lin Cai, Departmental Member
(Department of Electrical and Computer Engineering)

ABSTRACT

The ability to probe molecular electronic and vibrational structures gives rise to optical absorption spectroscopy, which is a credible tool used in molecular quantification and classification with high sensitivity, low limit of detection (LoD), and immunity to electromagnetic noises. Spectra are sensitive to slight analyte variations, so they are often used to identify a sample's components. This thesis proposes several methods for quick classification and quantification of analysts based on their absorbance spectra. *functional Principal Component Analysis* (fPCA) is employed for feature extraction and dimension reduction. For 1,000-pixel spectra data, fPCA can capture the majority variance with as few output scores as the number of expected analytes. This reduces the amount of calculation required for the following machine learning algorithms. Further, the output scores are fed into XGBoost and logistic regression for classification, and fed into XGBoost and linear regression for quantification.

Our models were tested on both synthesized datasets and experimentally acquired dataset. Our models demonstrated similar performance compared to deep learning but with much faster processing speeds. For the synthesized 30 dB dataset, our model XGBoost with fPCA could reach a micro-averaged f_1 score of 0.9551 ± 0.0008, while FNN-OT [1] could obtain 0.940 ± 0.001. fPCA helped the algorithms extract the feature of each analyte; furthermore, the output scores nearly had a linear relationship with their concentrations. It was much easier for the algorithm to find the mapping function between the inputs and the outputs with fPCA, which shortened the training and testing time.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

My families and my friends, for unconditionally supporting my decisions.

Supervisor Prof. Tao Lu, for mentoring, support, encouragement, and patience.

Grant Organization Name, for funding me with a Scholarship.

Yifeng Bie

DEDICATION

Just hoping this thesis is useful.

Chapter 1

Introduction

1.1 Context

First popularized by Arthur Lee Samuel in 1959 [2], machine learning (ML) has been subject to rapid development in the past decade with more intelligence incorporated [3]. For example, ML can cope with more complex tasks such as face recognition [4] and human image synthesis [5], natural language processing [6] and self-driving [7]. A well-trained learning machine can even outperform human beings in various scenarios, including video games [8] and two-player games [9]. Machine learning has been widely used in data processing [10, 11], especially with enormous datasets [12].

Based on its label, ML can be divided into two major categories: supervised and unsupervised learning [13]. A supervised learning algorithm learns from the training set with the label; while unsupervised learning mainly deals with unlabeled datasets.

In general, dealing with unsupervised learning, it is referring to clustering [14] and grouping [15]. For example, the height and weight of a group of people is given, and the goal is to assign each person to their suitable T-shirt size. Since the T-shirt size of each individual is unknown, the dataset comes without a label. The algorithm tries to group the instances using the input data (height and weight). However, the number of T-shirt sizes can always be increased to make it more fitted. Therefore, for this type of question, there is no optimized solution.

As for supervised learning, the algorithm can be divided into two categories: (a) classification and (b) regression [16]. If the prediction is to classify the instance into a class, then it is a classification problem; if the output is continuous, such as predicting

the house price, it is a regression problem. A classification problem is further divided into (1) two-class (binary) classification and (2) multi-class classification, depending on how many classes the label has. Supervised learning has a broader implementation, such as natural language processing [17]. Since there are clear labels of the datasets, this project will be based on supervised learning.

1.2 Unknown Detection

In general, the point of interest is to detect the presence of toxic gas in a gas mixture, or determine the concentration of a specific solute in a solution. A technique called spectroscopic analysis is used to find these parameters [18]. When radiation interacts with an analyte, such as a mixture of gas or a solution sample, a part of the energy is absorbed. The intensity of the absorption varies as a function of wavelength and is usually called the absorbance spectrum [19]. The absorbance spectrum is sensitive to slight analyte variations, so the spectrum is often employed as an analytical tool to determine and quantify the amount of the substance present. The application of spectroscopic analysis has drawn more attention to areas such as environmental monitoring [20], food industrial regulatory system [21], and gas emission detection [22].

Spectroscopic analysis is one application for machine learning. A more specific implementation of spectroscopic analysis is in mineral exploration, where the spectra of hydrothermal alteration minerals are used to deter the mineral composition, crystallinity, and relative abundance of rock [23]. In field-based scenarios, the processing speed of the algorithm is a huge concern because results are needed as soon as possible. The proposed research problem is based around a field-based situation.

The goal is to obtain a fast and accurate prediction based on the absorbance spectrum of the analyte. Once the spectrum of the sample has been obtained, then one must do the following:

- Classify if a specific component exists or not. The algorithm needs to detect multiple components in the sample. If a certain component is detected, the output will be a 1, otherwise this value will be 0.

- Quantify the concentration of a particular component. The algorithm is expected to output the concentration accurately.

Figure 1.1 shows the flowchart of this project. Section 1.3 proposes several approaches for the classification and quantification problems. Section 5 compares the

performance of each approach.

1.3 Proposed Approaches

This project aims to design algorithms that can make fast and accurate predictions using ML algorithms. The first stage is to collect the datasets. For training and testing the algorithms, there are two datasets, (a) synthetic datasets and (b) experimentally acquired datasets. Two synthetic datasets with two distributed probability density functions were generated for testing the performance of the classification and quantification algorithm. The mid-infrared absorption cross-section spectra of C_2H_6, CH_4, CO, H_2O, HBr, HCl, HF, N_2O, NO from the high-resolution transmission molecular absorption (HITRAN) database [24] was used to generate the synthetic datasets. The absorption spectra of a group of Orange G (OG) [25] and Crystal Violet (CV) [26] solution was measured using a spectral system. Since the solution preparation is complicated, only one distribution is shown for the experimentally acquired dataset. (The details are discussed in Section 4.)

Four approaches for the two tasks are discussed in Section 1.2. Before feeding into the machine, the spectroscopic datasets were preprocessed by fPCA [27] for feature extraction and dimension reduction. Subsequently, labels were prepared for the next step. For classification, the label is transferred into a categorical value. If the concentration is zero moles, then the instance is assigned as 'False' or '0'; if the concentration is a non-zero mole, the instance is assigned as 'True' or '1'. For quantification, a method called *label rescale* is used, which will be discussed further in Section 2.4.

After preparing the data, the output fPCA scores and labels are fed into the machine learning algorithms for training and testing. The algorithms used in this project are discussed in Section 2.2. In short, XGBoost and logistic regression are used for classification whereas, XGBoost and linear regression are used for quantification algorithms, respectively. Furthermore, loss functions [28] are designed (Section 2.5) for different tasks to achieve better performance.

Once the predictions are obtained, the performance will be evaluated based on the metrics discussed in Section 2.7. Concretely, *Precision*, *recall*, F_1 score and minimum detectable concentration were computed for the classification algorithms. Moreover, mean-absolute-error, root-mean-square error and R_2 score [29] were computed for the quantification algorithms to evaluate how the predictions fit the labels. Furthermore,

the testing time for each algorithm is counted. At this stage, the training time was not very important because the training phase was conducted off-line. The testing time is more important during the final implementation (field-based spectroscopic analysis) because the processing speed or the response time is directly related to user experience.

1.4 Method Illustration

Two datasets were prepared for the proposed approach. At first glance, one might exclusively use the experimentally acquired dataset. However, this could lead to two problems: a) the Signal to Noise Ratio (SNR) of the dataset is uncontrollable in the experimentally acquired dataset, thus, the algorithms' performance cannot be tested under different SNRs; b) the distribution of the label is fixed. The solution used for obtaining a specific concentration was diluted, therefore, only a simple distribution for the experimentally acquired dataset was obtained; c) the limit of detection (LoD) of the spectrometer restricts the concentrations to be measured. The "outlier" samples cannot be obtained, and the LoD of the algorithm is unknown when only experimentally acquired data is available.

To alleviate this problem, the absorbance spectrum was synthesized, the synthetic datasets were used to test the algorithm, and the experimentally acquired dataset was further tested. Optical absorption spectroscopy is capable of probing molecular electronic and vibrational state structures, and serves as a proven tool for molecular quantification and classification with high sensitivity, low LoD, and immunity to electromagnetic noises. Spectra are sensitive to slight analyte variations, so they are often used to identify a sample's components.

The input data is a function of radiation frequency or wavelength, in which the intensity represents the energy absorbed by a sample. The input data provides enough information for the algorithms to make a prediction. The algorithms (XGBoost, linear regression, and logistic regression) can only predict one output for one input sample, so nine irrelevant machines for nine outputs need to be trained since the labels are not correlated.

1.5 Research Contributions

The main contributions of this project are listed as follows:

1. fPCA is introduced for data preprocessing.
   (a) fPCA treats the input data as functional data and fits the eigenfunctions. This procedure is similar to the ground truth that the spectra are generated, which is that a group of analytes superimpose the absorbance spectra.
   (b) The output fPCA scores are in a linear relationship with the actual label. Therefore, straightforward algorithms such as linear regression and logistic regression can be used to make a prediction.
   (c) fPCA can be made to output a certain number of output scores. Moreover, approximately one score explains the variance of one analyte. It is demonstrated that fPCA can capture the absorbance spectra information for each gas. The order of the eigenfunction is the descending of the cross-sectional area of each analyte.
   (d) The fitting time for fPCA is longer than PCA[30] because fPCA has to fit the eigenfunctions. However, for the online testing phase, the processing time is acceptable.
2. Introduction of a new approach for classification.
   (a) It has been shown that XGBoost is faster than the existing algorithms in training and testing, with similar accuracy.
   (b) The performance was evaluated using standard metrics: micro-averaged *Precision*, micro-averaged *Recall*, and micro-averaged F_1 score.
3. Introduction of a new approach for quantification.
   (a) Two algorithms were demonstrated for quantification which were compared with the results of partial least squares (PLS) [31]; the performance shows that the fPCA with linear regression and PLS yields a better result with less processing time.
   (b) It is demonstrated that for one analyte, only a single score from fPCA is needed to make an accurate prediction.

1.6 Thesis Outline

The structure of the thesis is as follows:

- Chapter 2: background information and techniques used throughout this project.
- Chapter 3: proposed models for classification and quantification.
- Chapter 4: demonstration of obtaining the synthetic datasets and experimentally acquired datasets.
- Chapter 5: presentation and evaluation of results.
- Chapter 6: conclusions, discussion, limitations and suggestions for future studies.

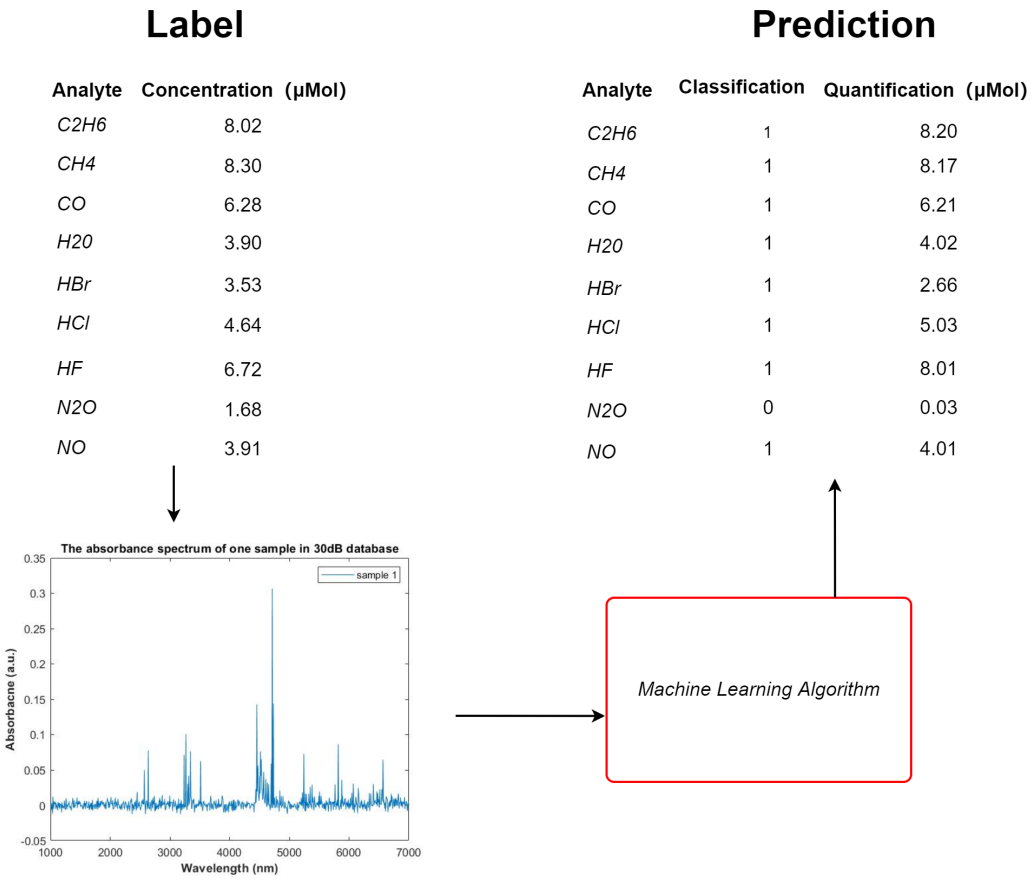

Figure 1.1: The labels of the spectrum are known and listed on the left. A machine learning algorithm is employed to predict whether a specific component exists as well as the concentration of the component. For classification outcomes, '0' indicates that the component does not exist in the sample, while '1' suggests that the component exists. For quantification outcomes, the algorithm would output its concentration. There is a misclassified instance (N_2O), and the predicted concentrations have a small gap between the labels. The performance will be evaluated using metrics introduced in Section 2.7.

Chapter 2

Background and Related Work

Two models were built for each problem of (a) classification and (b) quantification, respectively. This chapter will discuss the background and related work involved in the models.

2.1 Functional Principal Component Analysis

With a more powerful computer, machine learning algorithms are now processing data of increasing volume and complexity. The advantage is that the algorithm can handle more complex situations and make more accurate predictions with a sufficiently large training dataset. The drawback is that the input data are often in high dimensions and take extra time for the algorithms to do offline training and online prediction. Therefore, it is common to use the data pre-processing method to reduce the dimension and save computational cost. For example, the synthesized absorbance spectra have 1,000 samples ranged from $1\mu m$ to $7\mu m$. Therefore, the input data will be in 1,000 dimensions. If the dimension of the input data can somehow be reduced while keeping the majority variance (use fewer variables to represent the features), the algorithm can be trained faster. As a result, it becomes easier to find the pattern behind the surface.

High-dimensional space is mostly empty, which implies that multivariate data is usually in a lower-dimensional structure [32]. For example, the absorbance spectra are a continuous function of wavelength, which means if the function value at λ_1 is large, the function value near λ_1 will also be large. The signal strength is related in a small region. Therefore, such high-dimensional data may not be required to provide

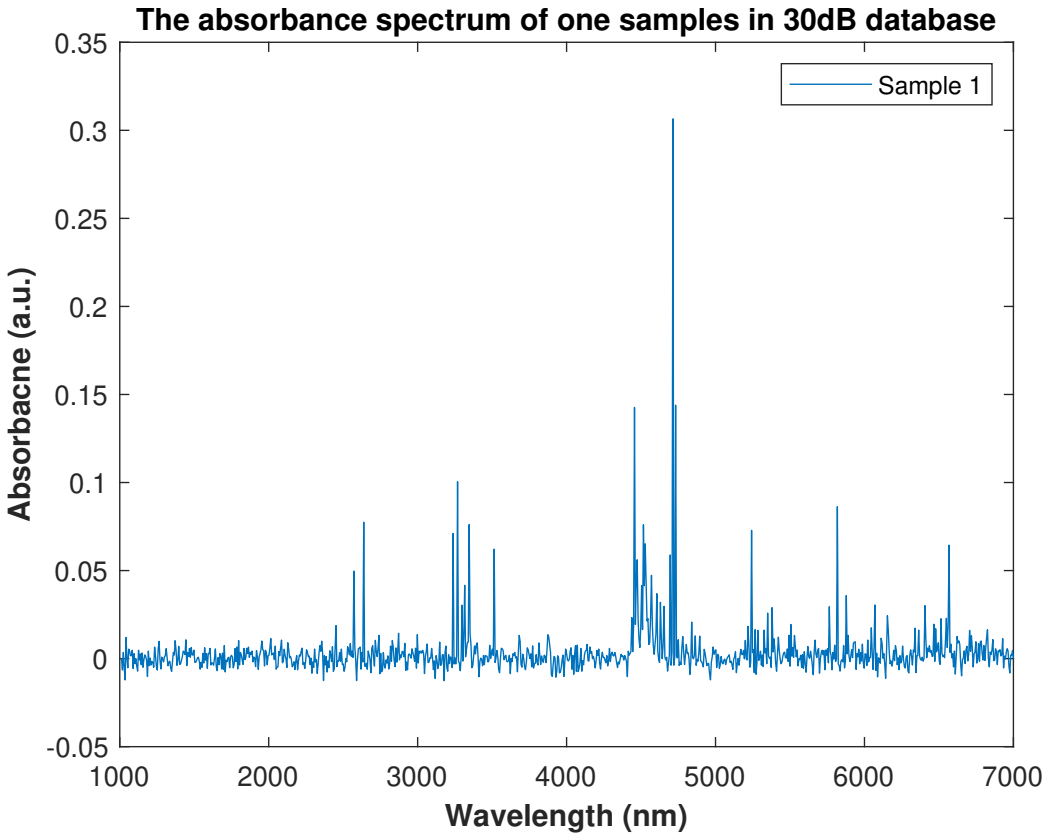

Figure 2.1: This figure shows the spectra signal of one sample in the 30 dB synthesized dataset. The concentrations of the nine gases C_2H_6, CH_4, CO, H_2O, HBr, HCl, HF, N_2O, NO are (8.3091, 6.2872, 3.8999, 3.5304, 4.6371, 6.721, 1.6761, 3.9138) (ppm). It shows that the peaks of the spectrum are all ranged between $2200 - 6700$ nm.

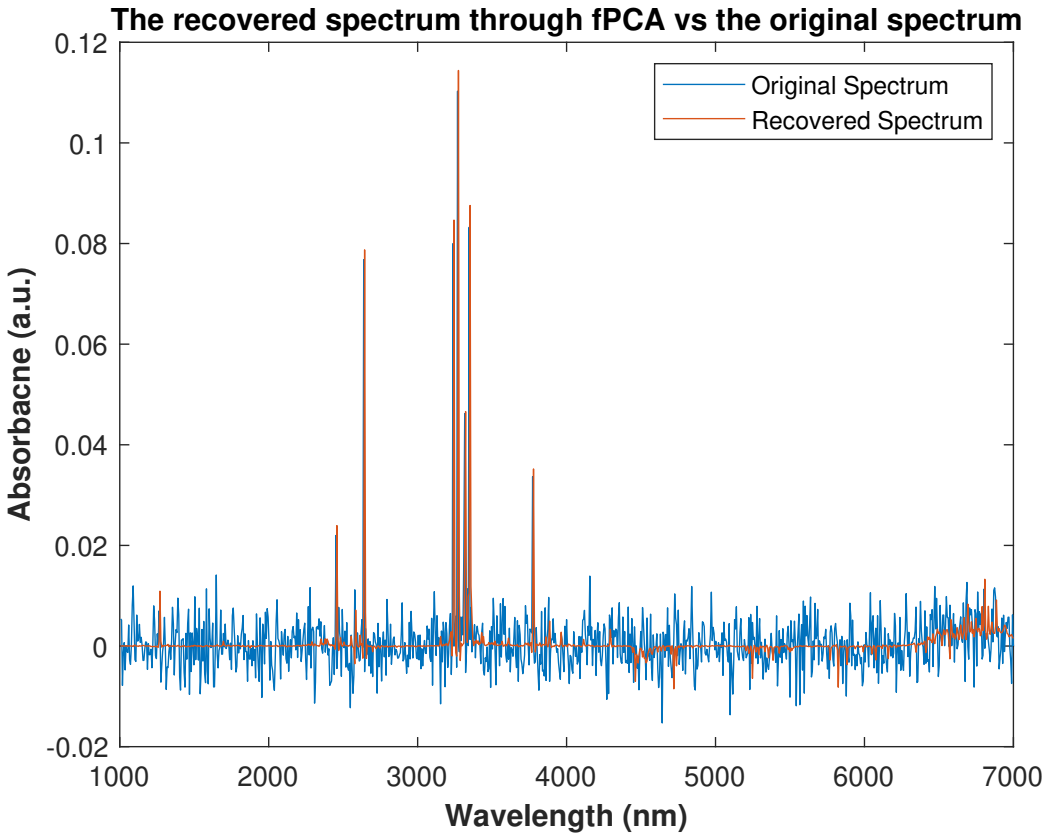

Figure 2.2: Recovered absorbance spectrum based on Equation 2.2. The spectrum regenerated through fPC scores (red) is smoother than the original spectrum (blue), because the small peaks around 0 were removed. It also shows that fPCA can capture the majority variance while removing the noise in the signal.

a certain amount of information. Furthermore, the dimension of the sample space is, in fact, a sub-space of the original high-dimensional space. Traditionally, people use Principal Component Analysis (PCA) as a data pre-processing procedure for dimension reduction and feature extraction [33]. However, in this project's approach, a method called functional principal component analysis (fPCA) is used. fPCA is a method for functional data analysis (FDA). Functional data [34], by definition, is multivariate data with an ordering on the dimensions. Since absorbance spectra have a trajectory of wavelength, they can be seen as functional data. fPCA can be applied for displaying the modes of functional variation [35]. By fitting eigenfunctions, fPCA is trying to explain the majority variance. Figure 2.1 plots the spectra signal of one samples in the 30dB synthesized dataset. Referring to Figure 2.1, the peaks of the spectrum are all ranged in certain areas, thus some functions can be used to represent the patterns or the peaks to reduce the dimension. While PCA ignores the information carried by the trajectory, this allows fPCA to outperform PCA.

Consider each absorbance spectra as an observation of the continuous family $\mathcal{A}$:

$$\mathcal{A} = \{\mathbf{A}(\lambda) : \lambda \in (\lambda_{min}, \lambda_{max})\} \tag{2.1}$$

where $\mathbf{A}(\lambda)$ is the absorbance spectra, each spectral signal is functional data and can be used for fPCA. fPCA helps extract hidden features, reduces computing time, and "de-noises" by smoothing the eigenfunctions.

The input 1,000-dimensional spectral signal was pre-processed by fPCA to extract fPC scores (fPCs). fPCA discovered the significant source of variability in observed curves and can reduce dimensions [36]. By finding an orthonormal basis containing k eigenfunctions, each spectral data $\boldsymbol{X}_m(\lambda)$ could be composed by

$$\boldsymbol{X}_m(\lambda) = \boldsymbol{\mu}(\lambda) + \sum_{k=1}^{K} \epsilon_k \boldsymbol{\psi}_k(\lambda) \tag{2.2}$$

where $\boldsymbol{\mu}(\lambda)$ is the mean observation over the trajectory, $\boldsymbol{\psi}_k(\lambda)$ is the k-th eigenfunction and ϵ_k is the corresponding fPC score. The mixture has nine gases, thus the first nine eigenfunctions are kept. The nine eigenfunctions should preserve enough information for the prediction.

Another way to interpret fPCA is to assume that the nine analytes are $\{a_1, a_2, \cdots, a_9\}$ and their absorbance spectra are $\{\boldsymbol{s_1}, \boldsymbol{s_2}, \cdots, \boldsymbol{s_9}\}$. In this case, the sample space is a

nine dimensional space because the sample space is $\mathcal{S}$:

$$\mathcal{S} = span\{\boldsymbol{s_1}, \boldsymbol{s_2}, \cdots, \boldsymbol{s_9}\} \tag{2.3}$$

the sample space $\mathcal{S}$ with a dimension higher than nine can be explained by the noise. By smoothing the input sample and explaining the majority variance, the noise can be minimized. Therefore, the nine fPCs $\{p_1, p_2, \cdots, p_9\}$ can be interpreted as a value that can be used to predict the concentration of the nine analytes. Note, these fPCs are uncorrelated because the original concentrations of the analytes are uncorrelated.

Figure 2.2 shows an absorbance spectrum regenerated by the eigenfunctions. The blue curve is the original spectrum, and the red curve is recovered from the fPC scores. The spectrum regenerated through fPC scores is smoother than the original spectrum. The blue curve has more noise around the x-axis. The random permutation does not provide much information and is mostly caused by noise. Figure 2.2 shows that fPCA can capture the majority variance while removing the noise in the signal.

A gas with a small cross-sectional area will contribute a slight variance in the absorbance spectra; a gas with a large cross-sectional area will significantly contribute to the absorbance spectra. Therefore, the first few eigenfunctions denote significant cross-sectional area gases, while the last few eigenfunctions are for minor cross-sectional area gases.

Noise is a serious problem that will block the prediction. For example, if the noise rate is high, or if the power of the noise is at the same level as a small cross-sectional area gas, the noise is likely to blur the signal from this gas. In this case, the algorithm cannot detect the small cross-sectional area gas.

2.2 Machine Learning Algorithms

In this section, algorithms for making predictions based on the output scores of the fPCA are presented. The goals of machine learning algorithms are to map the input scores $\{x_1, x_2, \ldots, x_n\}$ to the expected output Y. The algorithms were chosen based on the required task. As described in Section 1.2, there are mainly two tasks of interest:

- Classify whether a specific gas exists or not.
- Quantify the concentration of a particular gas.

For the classification problem, the goal is to determine whether or not a specific gas exists. Since there are nine gases for each sample in the database, there are nine binary classifications that need to be done. A binary (two-class) classification means that the output is either 0 or 1. As such, this is a multi-label classification problem. Additionally, Extreme Gradient Boosting (XGBoost) and Logistic Regression were used for the predictions of this question.

The algorithm has to predict the concentrations of the nine gases from its fPCA scores for the quantification problem. This would be a regression problem, which means that the outputs are continuous numbers. Therefore, each sample in the dataset will have nine concentrations

2.2.1 Extreme Gradient Boosting

Chen *et al.* introduced Extreme Gradient Boosting (XGBoost) in 2014 [37]. XGBoost is a supervised machine learning algorithm and that is used in a variety of applications in the medical industry [38, 39], image recognition [40], fault detection [41], etc. XGBoost is an open source package that is easily accessed by many users. More importantly, XGBoost has been empirically proven to be fast while yielding accurate results. This algorithm is widely used in Kaggle, and the outcomes can even beat deep learning [42]. The structure of XGBoost is omitted to avoid complicating this paper. Instead, this paper will focus on the advantages and disadvantages of XGBoost.

The advantages of the XGBoost algorithm are as follows.

- XGBoost can regularize learning objects and provide column blocks for parallel learning [37].
- XGBoost is a highly effective and versatile approach to predictive modeling because additive tree models can be seen to determine the local neighborhoods' size adaptively. [42].
- XGBoost implements parallel processing and is fast compared with deep neural nets [37].

The drawback is that XGBoost can only produce one score at one time, so for multi-label classification, the task is treated as several irrelevant classifications, therefore, scores are computed independently.

XGBoost can be suitable for both tasks as long as the loss function is changed, therefore, the loss function and hyperparameters can be changed so that XGBoost

can be used for classification and quantification. Loss functions are further discussed in section2.5

2.2.2 Linear Regression

Linear regression [43] is a statistical analysis method that uses regression analysis in mathematical statistics to determine the quantitative relationship between two or more variables. This model is widely used in various areas, such as price prediction. In regression analysis, a unary linear regression model contains one independent variable and one dependent variable. Therefore, a straight line can approximate the relationship between the two. However, if the regression analysis includes two or more independent variables and the relationship between the dependent and independent variables is linear, then it is called a multiple linear regression. The expression for multiple linear regression is formulated as:

$$y = \boldsymbol{w'} \cdot \boldsymbol{x} + e \tag{2.4}$$

where $\boldsymbol{w}$ is the coefficient vector, $\boldsymbol{x}$, is the input scores that are in the same dimension of $\boldsymbol{w'}$ and e is a normal distribution with an error that obeys a mean value of 0.

Linear regression is the very first type of regression analysis that has undergone rigorous research and is widely used in practical applications. This model is popular because a model that linearly depends on its unknown parameters is more accessible to fit than a model that non-linearly depends on its unknown parameters. In addition, the statistical properties of the resulting estimates are easier to determine.

The least-squares approximation often fits linear regression models to minimize the linear regression by the least-squares method. On the contrary, the least-squares method can sometimes be used in the approximation of fitting non-linear models. Despite least-squares and linear regression models being closely connected, they are not equal.

Least-square Method

The least-squares method is a mathematical optimization technique[44]. It finds the best function match of the data by minimizing the sum of squares of the error. Thus, the least-squares method can quickly obtain the label of anonymous data and minimize the sum of the squared error between the original label and the mapping

label.

The least-squares method is the most commonly used method to solve curve-fitting problems. Let the input sample in the n-dimension be $\boldsymbol{x} = [x_1\ x_2\ \dots\ x_m]^T$, the label can therefore be expressed as:

$$f(\boldsymbol{x}) = \alpha_1\phi(\boldsymbol{x_1}) + \alpha_2\phi(\boldsymbol{x_2}) + \cdots + \alpha_m\phi(\boldsymbol{x_m}) + b \tag{2.5}$$

where $\phi(\boldsymbol{x_i})$ is a set of linearly independent functions selected in advance, α_i is the undetermined coefficient. The fitting criterion is used to minimize the sum of the squared-error between the distance from y_j and $f(\boldsymbol{x_j})$. Specifically, the fitting criterion $L(\boldsymbol{x})$ is:

$$L = \sum_{j=1}^{m}(y_j - f(\boldsymbol{x_j}))^2 \tag{2.6}$$

the fitting criterion L-2 distance is used to describe how much the function fits the input samples. If the prediction is close to the label, then the overall $L(\boldsymbol{x})$ is small; otherwise, the squared-distance is big and the $L(\boldsymbol{x})$ is big as well.

Formula 2.5 and 2.6 can be rewritten in matrix form:

$$f(\boldsymbol{\alpha}, \boldsymbol{x}) = \boldsymbol{x}\boldsymbol{\alpha} \tag{2.7}$$

$$L(\boldsymbol{\alpha}, \boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}\boldsymbol{\alpha} - \boldsymbol{y})^T(\boldsymbol{x}\boldsymbol{\alpha} - \boldsymbol{y}) \tag{2.8}$$

where $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2\ \dots\ \alpha_m]^T$, $\boldsymbol{x}$ is the n samples input with m dimensions $\boldsymbol{x} \in R^{n\times m}$, $\boldsymbol{y}$ is the label of the samples $\boldsymbol{y} \in R^{n\times 1}$.

To minimize $L(\boldsymbol{\alpha}, \boldsymbol{x})$, it's gradient needs to be zero:

$$\frac{\partial}{\partial\boldsymbol{\alpha}}L(\boldsymbol{\alpha}, \boldsymbol{x}) = \boldsymbol{x}^T(\boldsymbol{x}\boldsymbol{\alpha} - \boldsymbol{y}) = 0 \tag{2.9}$$

This results in:

$$\boldsymbol{\alpha} = (\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}^T\boldsymbol{y} \tag{2.10}$$

The least-squares can yield to the global minimum of L. However, computing the inverse of Formula 2.10 is time-consuming when $\boldsymbol{x}$ is large, therefore, the gradient descent method is often used for finding $\boldsymbol{\alpha}$. The technology used for this method will be discussed in section 2.6.

2.2.3 Logistic Regression

Logistic regression [45], also known as logistic regression analysis, is a generalized linear regression analysis model, often used in data mining, automatic disease diagnosis, economic forecasting, etc. The dependent variables for logistic regression are categorical labels. For example, logistic regression can be used to explore the risk factors that cause disease and predict the probability of the disease based on the risk factors. Two groups of people can be selected; one group is the gastric cancer group, and the other is the non-gastric cancer group. The two groups of people may have different physiologies and lifestyles. Therefore, the dependent variable is gastric cancer, and the value is "yes" or "no," which is a binary categorical label. The independent variables can include many items, such as age, gender, eating habits, and Helicobacter pylori infection. Independent variables can be continuous or categorical. Through logistic regression analysis, the weights of the independent variables can be obtained, therefore making it possible to approximate which factors are risk factors. Concurrently, the weight obtained from logistic regression can be used to predict the likelihood of a person suffering from cancer based on risk factors.

Logistic regression is a generalized linear regression, so it has many similarities with multiple linear regression in the previous section. The output of logistic regression P is:

$$p(\boldsymbol{x}) = \alpha_1\phi(\boldsymbol{x_1}) + \alpha_2\phi(\boldsymbol{x_2}) + \cdots + \alpha_m\phi(\boldsymbol{x_m}) + b \tag{2.11}$$

where α_i and b are the parameters to be sought. The difference between the linear regression lies in their different dependent variables. Multiple linear regression directly from $y = \boldsymbol{\alpha}^T \cdot \boldsymbol{x}$, and logistic regression uses the function L to map $y = \boldsymbol{\alpha}^T \cdot \boldsymbol{x}$ to a hidden state p,

$$p = L(y) = \begin{cases} 1 & p \geq 0.5 \\ 0 & p < 0.5 \end{cases} \tag{2.12}$$

formula 2.12 shows how to map the output y to the probability p and $1 - p$. Then the formula can determine the coefficient of dependent variables according to the training set label. Similar to the linear regression, a loss function will be used to describe the likely-hood of the model and the training set. Loss functions will be discussed in detail in the next section.

The dependent variable of logistic regression can be binary (two-category) or multi-category, however binary variables are more common and easier to interpret.

Multi-category can be processed using the "softmax" method. Details regarding the "softmax" method are omitted because it was not used in this project. In practice, the most commonly used model is the binary logistic regression. Predicting the existence of a specific gas is a binary logistic regression problem.

2.3 Optimal Threshold

For the classification task, the output scores from machine learning models are compared with a threshold t_i for classification. The output scores can be treated as the probability of existence, so the threshold is usually $T = 0.5$. That is to say, if the output score is higher than 0.5, then the instance is classified as positive for this gas. However, some gases may be hard to detect due to nature, or in low SNR, high-level noise may blur the valid signal; a lower or higher than 0.5 threshold may help minimize misclassified instances. One way to increase the performance is to use accuracy as the metric to find the nine optimal thresholds.

The F_1 score (Section 2.7.4) is the harmonic mean of the precision and recall. The F_1 score is commonly used for evaluating the performance of a model, therefore, people may use F_1 score as the metric to find the optimal thresholds [46]. However, if the optimal threshold is found via F_1 scores, the algorithm may fail to work when the F_1 scores of one dataset is less than 0.66 when $T = 0.5$. Consider an unsatisfactory SNR situation, where the noise is more prominent than valid signals, and the algorithm is randomly guessing. The threshold can be set to 0 in order to predict every positive sample. The confusion matrix (discussed this concept in Section 2.7) for a balanced dataset contains $2a$ samples which would look like Table 2.2. The precision (Section 2.7.2) and recall (Section 2.7.3) are 0.5 and 1.0, respectively, which yields the F_1 score 0.67. If the optimal threshold is chosen via F_1 scores, then the performance for the low SNR dataset is "false high". Thus, F_1 score should not be used as the metric.

	Actual class 1	Actual class 0
Predicted class 1	a	a
Predicted class 0	0	0

Table 2.1: The confusion matrix for balanced dataset when each instance is predicted as positive.

The algorithm for this step is quite simple. Once the output scores from XGBoost

are obtained, then the output scores will be compared with 1,000 thresholds that are equally spaced from 0 to 1 where the values that yield the highest accuracy are chosen. Optimal thresholding was employed to enhance the performance of the classification algorithms.

2.4 Label Rescale

For quantification, the instance's label is rescaled in order to make it more suitable for ML. Since the prediction is based on the scaled label, the prediction is scaled to the original concentration level before the output. The initial label of each instance in the synthetic datasets are in $O(10^{-6})$. The labels need to be rescaled to $O(1)$ before output the final concentrations. The reasons are as follows:

- The original labels are too small for the double-precision floating-point format to handle. Since the IEEE standard double-precision [47] has a 52-bit fraction, which yields the precision of 10^{-16}. $O(10^{-6})$ is too small for double-precision and is inconvenient for calculation.
- The original are labels piled together which makes it hard to find a pattern and make a precise prediction.

Therefore, the label $\mathbf{L}$ is re-scaled as follows:

$$\mathbf{L}_i = \frac{\mathbf{L}_i}{std(\mathbf{L})} \tag{2.13}$$

where $\mathbf{L}_i$ is the label of the i-th instance, and $std(\mathbf{L})$ is the standard deviation of each gas. Then the concentrations are in $O(1)$, and in unit variance.

Rescaling the output scores $\mathbf{s}$ is done as follows:

$$\mathbf{s}_i = \mathbf{s}_i \times std(\mathbf{L}) \tag{2.14}$$

where $\mathbf{s}_i$ is the prediction for the i-th sample.

2.5 Loss Function

The loss function quantifies the amount by which the prediction deviates from the actual values. That is to say, the loss functions should be defined in a way that can

reflect how much the prediction fits the label. Two loss functions were used for the two tasks.

For the classification task, each label is in binary form, therefore the loss function is chosen as a binary logistic regression. The loss function f_{L_1} is defined as follows:

$$f_{L_1}(\boldsymbol{s}, \boldsymbol{y}) = -\sum_{i=1}^{N} \Big(y_i log(s_i) + (1 - y_i) log(1 - s_i) \Big) \tag{2.15}$$

where N is the total number of training samples, $\boldsymbol{y}$ is the training label and $\boldsymbol{s}$ is the output scores from the machine learning model.

Equation 2.15 describes how much the model fits the training set. The analysis is as follows. In the case that the instance $\boldsymbol{x_i}$ in the training set is positive for one analyte, then y_i is 1. The desired output score s_i from the logistic regression model should be as high as possible to make the prediction robust. The term $-(1 - y_i)(log(1 - s_i)) = 0$ because $y_i = 1$. For the first term $(-y_i log(s_i))$, a bigger s_i would result in a smaller $-log(s_i)$; if the instance $\boldsymbol{x_i}$ in the training set is negative for one analyte, then y_i is 0. The desired output score s s_i from the logistic regression model needs to be as small as possible for same reason. The term $-y_i log(s_i) = 0$ because $y_i = 0$. For the second term $-(1 - y_i) log(1 - s_i)$, the closer s_i is to 1, the smaller $-(1 - y_i) log(1 - s_i)$ becomes. Therefore, if this loss function is minimized, the logistic regression model should output a score si close to 0 when the instance is negative and 1 when the instance is positive.

As for the quantification task, several loss functions were tested, and it turns out that the mean squared error (MSE) has the best performance. The loss function f_{L_2} is defined as follows:

$$f_{L_2}(\boldsymbol{s}, \boldsymbol{y}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - s_i)^2 \tag{2.16}$$

where N is the total number of training samples, $\boldsymbol{y}$ is the training label and $\boldsymbol{s}$ is the output scores from the linear regression model.

Equation 2.16 describes how much the model fits the dataset when the labels are continuous numbers. Clearly, for any instance in the dataset, if s_i is close to the label y_i, then the squared error $(y_i - s_i)^2$ would be small; if the s_i is far away from the label y_i, then the squared error $(y_i - s_i)^2$ would be large. Therefore, minimizing the loss function $f_{L_2}(\boldsymbol{s}, \boldsymbol{y})$ makes the prediction $\boldsymbol{s}$ as close as possible to the label $\boldsymbol{y}$).

The issue of over-fitting for linear regression can be disregarded. The logistic regression model is straightforward and is nearly impossible to overfit the training

set. For XGBoost, both $L1$ regularization (Lasso regression) and $L2$ regularization (Ridge Regression) are used to avoid the over-fitting problem.

The full name of Lasso regression is *least absolute shrinkage and selection operator*. It adds an extra term in equation 2.16:

$$f_{L_2}(\boldsymbol{s}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - s_i)^2 + \lambda||\boldsymbol{\alpha}||_1 \tag{2.17}$$

where $\boldsymbol{\alpha}$ is the coefficient of the features; Ridge Regression adds $L2$ regularization terms in the loss function and,

$$f_{L_2}(\boldsymbol{s}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - s_i)^2 + \lambda||\boldsymbol{\alpha}||_2^2 \tag{2.18}$$

where $\boldsymbol{\alpha}$ is the coefficient of the features. Both regularization terms can be used to solve the overfitting problem of standard linear regression.

Suppose the $L1$ or $L2$ regularization are added to penalize the weight. Consequently, if the loss function is being minimized, the coefficient $\boldsymbol{\alpha}$ will also be minimized, and the hyperparameter λ controls the penalty on the coefficient $\boldsymbol{\alpha}$. If λ is relatively large, then a model with all parameters being relatively small would to be constructed, because the term $\lambda||\boldsymbol{\alpha}||_1$ or $\lambda||\boldsymbol{\alpha}||_2^2$ would contribute a significant portion in the loss function. As a result, the regularization terms are the first to be minimized. Models with small parameters are relatively simple, adapt to different data sets, and avoid overfitting to a certain extent. By penalizing the model's complexity, the two additional regularization terms help smooth the final learned weights to avoid over-fitting [37].

2.6 Gradient Descent

When searching the optimized parameters of machine learning algorithms, which is an unconstrained optimization problem, *Gradient Descent* (GD) is one of the most commonly used methods [48]. Another commonly used method is the least-squares method, which was discussed in Section 2.2.2. GD is a first-order iterative optimization method used to find the local minimum of the loss functions. GD requires the optimized function to have suitably smooth properties (e.g., differentiable or subdifferentiable). Conversely, the maximum value of the loss function can be found

using the gradient ascent method. The algorithm and the pseudo code of GD will be introduced first.

The idea of the GD algorithm is as follows: given a loss function $f(\boldsymbol{p})$, and assuming that the function has suitable smoothness, then the gradient is a vector. The directional derivative of a function at a point $\boldsymbol{p} = [w_1^{'}\ w_2^{'}\ \cdots\ w_n^{'}]^T$, the function changes fastest along the direction of the gradient with the maximum rate being the modulo of the gradient. Then by definition, the components of the gradient $\nabla f(\boldsymbol{p})$ is the partial derivative of the function:

$$\nabla f(\boldsymbol{p}) = \begin{bmatrix} \frac{\partial f_{\boldsymbol{p}}}{\partial p_1} \\ \frac{\partial f_{\boldsymbol{p}}}{\partial p_2} \\ \vdots \\ \frac{\partial f_{\boldsymbol{p}}}{\partial p_n} \end{bmatrix} \tag{2.19}$$

where $\boldsymbol{p}$ can be seen as the coefficient of the machine learning algorithm. Therefore, the function would decrease the most along the opposite direction of the gradient. To account for this, a step $\gamma \nabla f(\boldsymbol{p})$ is taken away from the current point $\boldsymbol{p}$:

$$\boldsymbol{p}^{'} = \boldsymbol{p} + \gamma \nabla f(\boldsymbol{p}) \tag{2.20}$$

where $\boldsymbol{p}^{'}$ is the next point which gives $f(\boldsymbol{p}^{'}) < f(\boldsymbol{p})$. Therefore, the minimum of the function can be found using a limited loop. The hyperparameter called "learning rate" γ needs to be chosen carefully, if γ is too big, the GD cannot converge precisely around the true minimum $\boldsymbol{p}^*$; if γ is too small, then the term $\gamma \nabla f(\boldsymbol{p})$ would also be too small, which means it would take extra iterations to converge. Generally, there are two stop criteria. The first one is if the function value at $\boldsymbol{p}^{'}$ is smaller than a threshold T, the loop needs to be stopped; another stop criterion is if the step can be negligible, $||\boldsymbol{p}^{'} - \boldsymbol{p}||_2 < T$. The pseudo code of the GD procedure is:

2.7 Evaluation Metric

It is crucial to evaluate the performance of the algorithms because the results between them must be compared. Therefore, for both tasks, there are three different metrics, respectively. For classification purposes, evaluation is done using *Accuracy*, *Precision*, *Recall*, and F_1 score; for quantification purposes, evaluation is done using

Algorithm 1 Gradient Descent

Require: differentiable loss function $f(\boldsymbol{p})$, learning rate γ, stopping threshold T
Ensure: $\min f(\boldsymbol{p})$ with $\boldsymbol{p}^*$
  $\boldsymbol{d} \leftarrow \nabla f(\boldsymbol{p})$
  $\boldsymbol{p}^{'} \leftarrow \boldsymbol{p} - \gamma \boldsymbol{d}$
  while $||\gamma \boldsymbol{d}||_2 > T$ **do**
    $\boldsymbol{d} \leftarrow \nabla f(\boldsymbol{p})$
    $\boldsymbol{p}^{'} \leftarrow \boldsymbol{p}^{'} - \gamma \boldsymbol{d}$
  end while
  $\boldsymbol{p}^* = \boldsymbol{p}^{'}$
  return $\boldsymbol{p}^*$

mean absolute error (MAE), *root mean squared error* (RMSE), and R_2 score.

Now to introduce the confusion matrix [49], refer to table 2.2. The confusion matrix in the statistics tool is often used to evaluate the performance of a classifier. It is a two-dimensional matrix where the columns represent the predicted categories, and the rows represent the actual categories. The value on the diagonal indicates the number or the proportion of correct predictions; the off-diagonal element is part of the wrong prediction. By definition, higher diagonal values of the confusion matrix are better and indicates more accurate predictions. When the number of classification data is unbalanced, the confusion matrix can intuitively display the performance of the classification model corresponding to each category.

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Table 2.2: The confusion matrix for a binary classification.

In Table 2.2, *True Positive* is a positive example that was correctly predicted. It means that the actual value is positive, and it is also correctly predicted to be positive; *True Negative* is a negative example that was correctly predicted. *True Negative* indicates that the actual value is a negative example, and it was assigned as a negative; *False Positive* indicates that the actual value is a negative case, and is mispredicted as a positive case; *False Negative* indicates that the actual value is a positive example, and is mispredicted as a negative example. Next, the evaluation metrics will be introduced.

2.7.1 Accuracy

When discussing the performance of a model, *Accuracy* is often used. The formula for the *Accuracy* of a given confusion matrix is:

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.21}$$

which is the number of correctly classified samples divided by the number of all samples. Generally speaking, the higher the correct rate, the better the classifier. Is the model with the highest *Accuracy* necessarily the best model?

Suppose there are 99 females and one male on an island. A classifier is trained to classify the gender of the instance, this becomes a binary classification problem. If the classifier assigns each instance as female, the *Accuracy* would be 99% because only the male is misclassified. In this case, the classifier fails to predict the only negative sample (the male) but still gets a high *Accuracy*. The *Accuracy* is highly dependent on the balances of the dataset. In the example above, the dataset is highly unbalanced, and the *Accuracy* cannot reflect the algorithm's performance.

As such, when the dataset is balanced, that is, when the values of false positives and false negatives are almost the same, *Accuracy* can be used as a good measurement. When the dataset is unbalanced or the balances is uncertain, *Accuracy* is not an objective metric. Consequently, for the unbalanced datasets, other metrics must be examined to fully evaluate the model. Typically, metrics such as *Precision*, *Recall*, and f1 score can be used for evaluation.

2.7.2 Precision

Precision shows the probability of correct positive predictions among the samples whose predictions are positive samples. The formula of *Precision* is:

$$precision = \frac{TP}{TP + FP} \tag{2.22}$$

Precision is used as the metric of evaluation when the false positives seriously damage the system. For example, in spam detection, false positives mean that non-spam (*True Negative*) is incorrectly predicted as spam (predicted to be positive). Therefore, if the *Precision* of a spam monitoring system is not high and many non-spam emails would be classified into the spam mailbox, then the mailbox users may lose or miss some important emails. Thus, *Precision* is a important evaluation metric for ML.

2.7.3 Recall

Recall represents the proportion of samples whose predictions are positive among all samples whose actual label is positive. The formula of *Recall* is:

$$recall = \frac{TP}{TP + FN} \tag{2.23}$$

Recall is used as the primary metric when the goal is to have the rate of *False Negatives* among the actual positive samples to be minimized. For example, in bank fraud detection, if fraudulent transactions (*True Positive*) are predicted as non-fraudulent transactions (predicted negative), it may cause severe losses to the bank. Taking the recent COVID-19 as an example, if a sick person (actually positive) is predicted as healthy (predicted to be negative) through a reagent test, the risk of such a false negative or false negative is very high. Therefore, *Recall* is a critical metric for certain scenarios.

2.7.4 F_1 Score

F_1 score is an index used in statistics to measure the *Accuracy* of a binary classification model. It takes into account both *Accuracy* and *Recall*. F1 score can be regarded as a harmonic average of model *Accuracy* and *Recall*. It has a maximum value of 1 and a minimum value of 0. By definition, F_1 score is:

$$f_1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.24}$$

and generally, the F_β score is:

$$f_\beta = (1 + \beta^2) \times \frac{precision \times recall}{(\beta^2)precision + recall} \tag{2.25}$$

where β can be interpreted as the weight of *Recall* verse *Precision*. F_1 score is used in our project because *Precision* can be evaluated at the same weight as *Recall*.

2.7.5 Minimum Detectable Concentration

The *minimum detectable concentration* (MDC) is a significant metric to evaluate the performance[50]. A lower MDC can be obtained by increasing the integration time of the spectrometer properly [51] since increasing the integration time can help the spectrometer catch smaller signals.

The MDC can be defined as the lowest concentration which gives the $F_1 = 0.8$. Concretely, the testing sets must be sorted, and the instances were divided into 30 small bins. The F_1 score of each bin can be calculate, where bins with higher concentrations should have a higher F_1 score. The highest concentration of the first non-zero bin is taken with $F_1 \geq 0.8$ as the MDC.

2.7.6 Mean Absolute Error

Mean absolute error (MAE) is known as the L1 norm loss; it represents the average distance between the predicted value and the actual value in a geometric sense. The formula is:

$$error = \frac{1}{n}\sum_{1}^{n}|\hat{y}_i - y_i| \tag{2.26}$$

where n is the total number of samples, $\hat{y}_i$ is the prediction, and y_i is the actual label. By definition, the MAE would be small if the prediction is close to the actual label. However, due to the absolute value in the formula, the function is not smooth and cannot be derived at certain points.

2.7.7 Root Mean Squared Error

The *mean squared error* (MSE) is defined as:

$$error = \frac{1}{n}\sum_{1}^{n}(\hat{y}_i - y_i)^2 \tag{2.27}$$

where n is the total number of samples, $\hat{y}_i$ is the prediction, and y_i is the actual label. The MSE solves the problem when MAE is not differentiable, but its size and target variable are not on the same scale because of the square.

In order to solve this problem, the root of the MSE can be extracted, from this, the root mean square (RMSE) can be obtained via the following:

$$error = \sqrt{\frac{1}{n}\sum_{1}^{n}(\hat{y}_i - y_i)^2} \tag{2.28}$$

where n is the total number of samples, $\hat{y}_i$ is the prediction, and y_i is the actual label.

2.7.8 R_2 Score

R_2 score[29] is defined as:

$$R_2 = 1 - \frac{\sum_i^n(\hat{y}_i - y_i)^2)}{\sum_i^n(\bar{y}_i - y_i)^2)} \tag{2.29}$$

where n is the total number of samples, $\hat{y}_i$ is the prediction, y_i is the actual label, and $\bar{y}_i$ is the mean value of the label. The numerator is the model's squared error, which

reflects how close the predictions are to the label, and the denominator is the variance of the dataset. Therefore, if the prediction is perfect, then $R_2 = 1$, and $R_2 = 0$ which indicates that the model is underfitting. The R_2 score reflects how much variance this model could predict.

Chapter 3

fPCA Based Machine Learning Algorithms

In this chapter, new approaches are presented to help solve the two questions based on the technologies introduced in Chapter 2.

3.1 Classification Algorithm

For classification purposes, a typical approach can be used by computing the probability (score) of each class, and for each class, assigning the instance as positive if the output scores are higher than the threshold.

Figure 3.1(a) shows the whole process of the classification algorithm. Firstly, fPCA was used to process the spectrum signals and normalize the output functional principal component (fPC) scores. Next, the normalized fPC scores were fed into the machine learning model to predict the existence probability. At this stage, XGBoost and logistic regression is used. This task is a multi-label classification, but both models can produce one score at a time. Therefore, this task is treated as nine irrelevant binary classifications. This results in nine models for nine gases.

Generally, the threshold T is set to 0.5, however, optimal thresholding was employed (as discussed in Section 2.3) for better performance. An instance will be classified as positive if the output score is higher than the thresholds T, otherwise it will be classified as negative. Finally, the performance is evaluated using *Precision*, *Recall*, F_1 score and MDCs (section 2.7).

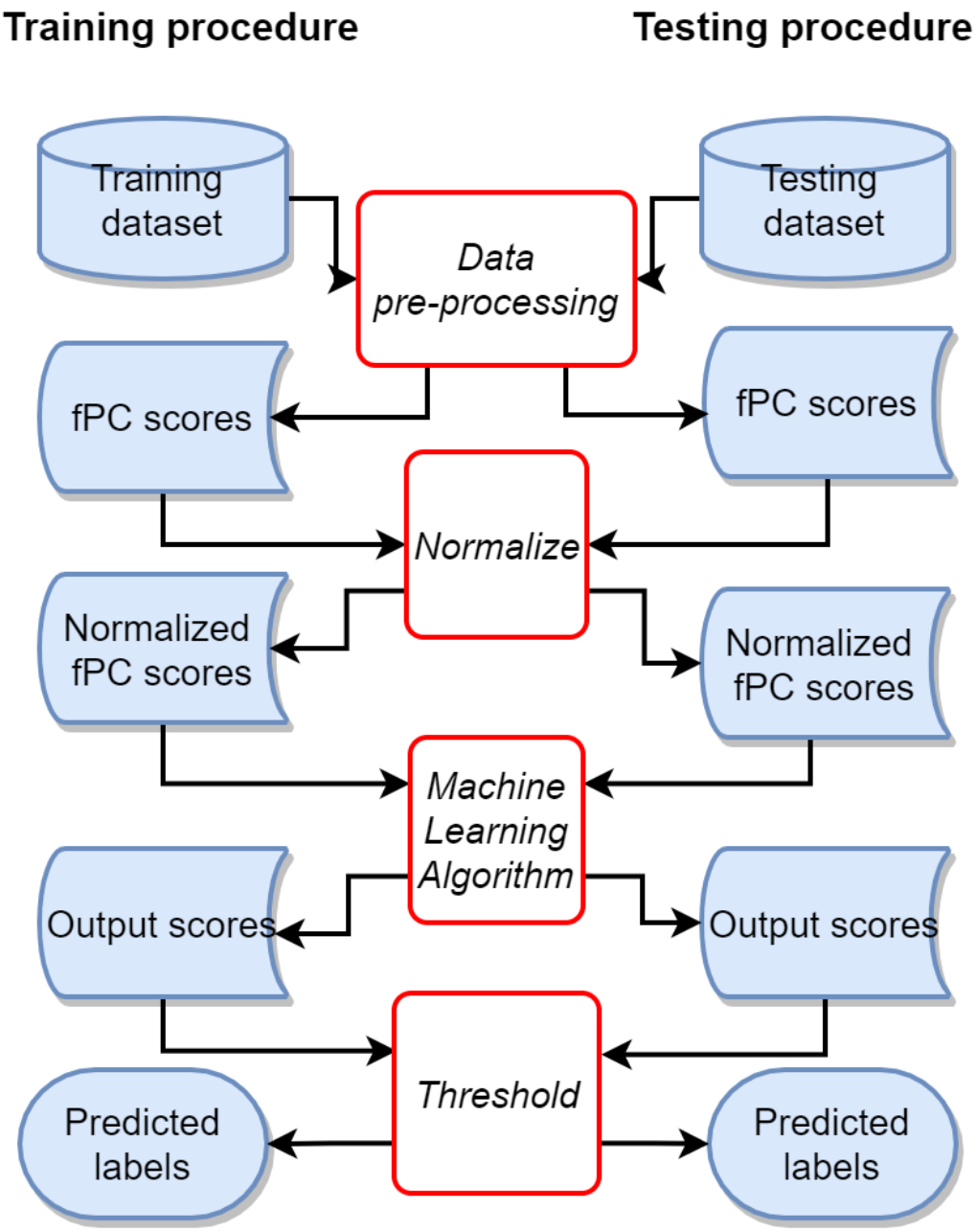

Figure 3.1: This figure shows the flow charts of the training and testing procedure for the classification task. The machine learning algorithms are XGBoost and logistic regression.

3.2 Quantification Algorithm

Figure 3.2(a) shows the flow chart for quantification. Gas concentrations typically present highly skewed distributions, thus, the labels need to be re-scaled at the start (section 2.4). The next step is data pre-processing. XGBoost is still used for predicting the concentration and re-scale the output scores back to the real concentrations. Finally, the results are evaluated based on the metrics introduced in Section 2.7 (RMSE, MAE and R_2 score).

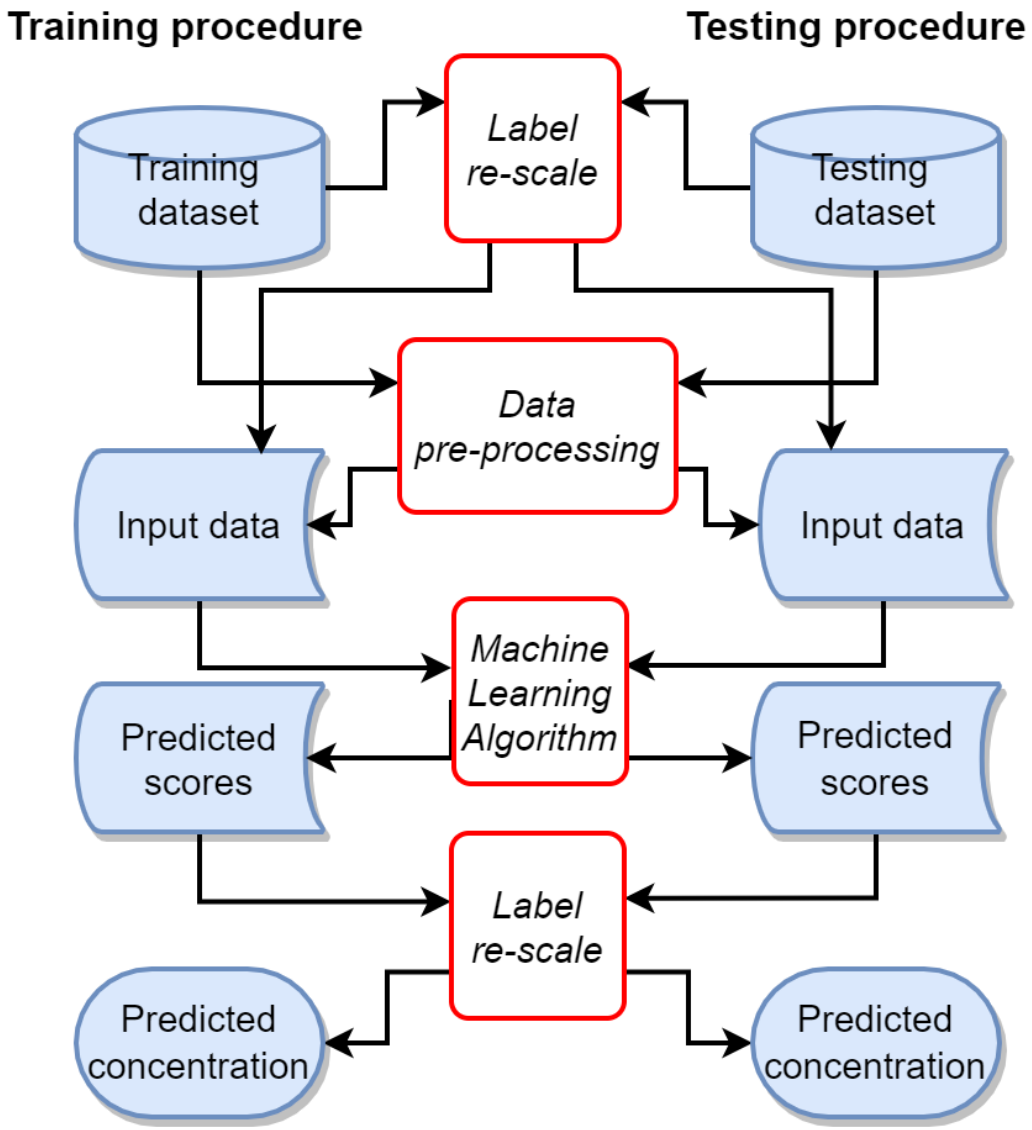

Figure 3.2: This figure shows the flow chat of training and testing procedure for the quantification task.

Chapter 4

Dataset and Code Implement

This chapter introduces the two datasets for the project. Following the line of research, two ML algorithms are applied to synthetic datasets and an experimentally acquired dataset.

4.1 Synthetic Dataset

Gan *et al.* [**?**] firstly proposed a method using Matlab to synth absorbance spectrum. Two different synthetic datasets were generated with two distributed probability density functions for testing the performance of the classification and quantification algorithm. The mid-infrared absorption cross-section spectra of C_2H_6, CH_4, CO, H_2O, HBr, HCl, HF, N_2O, NO from the high-resolution transmission molecular absorption (HITRAN) database [24] were used to simulate the synthetic datasets. Figure 4.1(a) shows the cross-sectional area of the nine gases with the wavelength ranging from $1\lambda m$ to $7\lambda m$.

Following the method described in [**?**], both datasets were generated as follows. Firstly, the two distributed probability density functions were *log-normal distribution*[52] and *uniform distribution*. Also known as *Galton distribution* or *Galton's distribution*, *log-normal distribution* is a random variable whose logarithm is normally distributed. Therefore, if the concentration C is in log-normal distribution, then $Y = ln(C)$ is in *normal distribution*. In addition, half of the concentrations were randomly set to be zero and the other half set to be in *log-normal distribution* form. In this case, both low concentrations and extremely high concentrations could be obtained. For classification purposes, the MDCs of the algorithm were tested. This

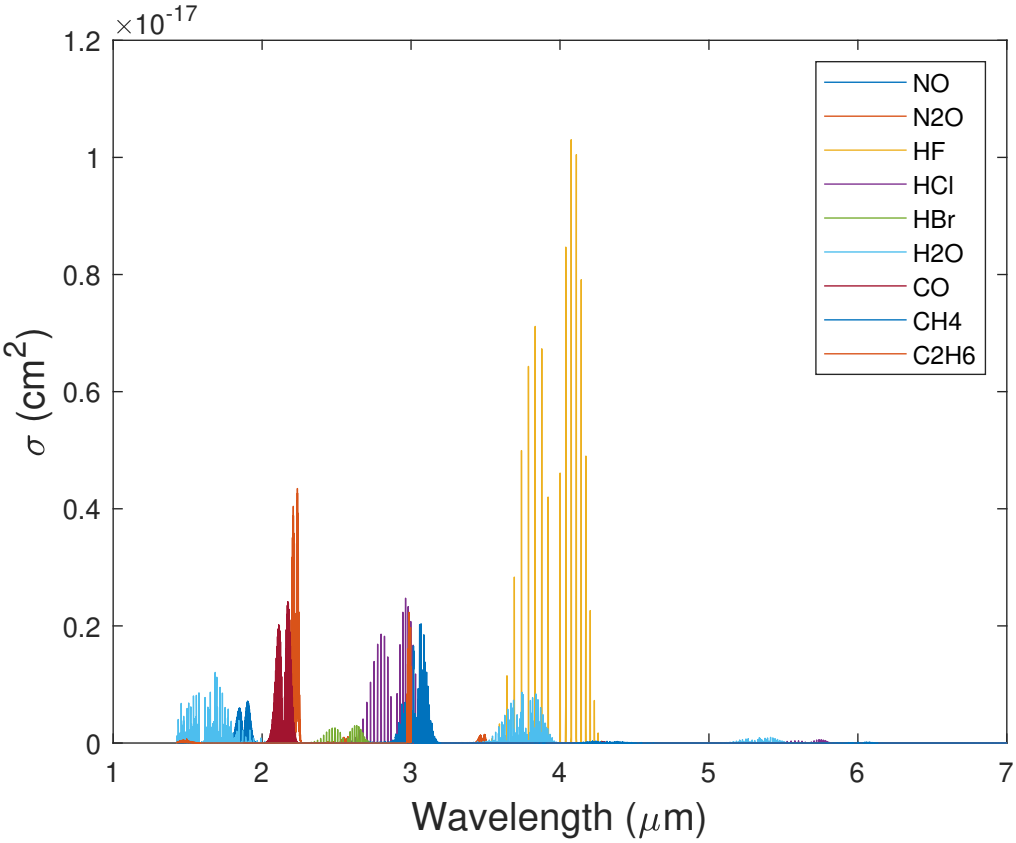

Figure 4.1: Mid-infrared absorption cross-section spectra of the nine gases ranged between $1\lambda m$ to $7\lambda m$. A higher value indicates a more substantial absorbance in the wavelength.

distribution is not suitable for testing the performance of the quantification algorithms because the majority of concentrations would narrow to a small scale, therefore, the algorithm could obtain a small RMSE by outputting the mean of the concentration. In short, *uniform distribution* was employed for training and testing the quantification algorithm. The concentrations were uniformly distributed from $0\mu M$ to $10\mu M$. The goal was to generate a larger variance dataset to test if the algorithm can make accurate predictions.

The next step is down sampling of the absorbance spectrum to $1,000$ pixels that equally spread from $1\lambda m$ to $7\lambda m$. In this case, each sample in the synthetic datasets has the size of $R^{1,000\times 1}$

In this task, a big concern is the signal-to-noise ratio (SNR). SNR is the ratio of the signal's power to the noise power. Thus, SNR can be defined as:

$$SNR = 10\log_{10}\frac{P_{Sig}}{P_{Noi}} \tag{4.1}$$

where P_{Sig} is the power of the signal and P_{Noi} is the noise power. In real life scenarios, a spectrometer is expected to have a high SNR because a high SNR spectrometer can obtain cleaner absorption which makes it easier for the algorithm to make the prediction. The simulated datasets are expected to have a wide range of SNR to detect the algorithm's capacity. Thus, different power of *Gaussian noise* was added to the synthetic datasets to obtain four datasets with SNR ranging from 10 dB to 40 dB.

4.2 Experimentally Acquired Dataset

The absorption spectra of a group of Orange G (OG) and Crystal Violet (CV) solution were also measured by a spectral system for obtaining the experiment acquired dataset. This work is done by Tianyang Zhao and Sherry Li.

The spectral system (Figure 4.2) consisted of an Ocean Optics Halogen Light source (HL-2000), two Ocean Optics optical fiber cables, an Ocean Optics (1CM) Cuvette Holder and an Ocean Optics Miniature Spectrometer (USB4000-VIS-NIR-ES). Solutions were contained in Fisherbrand 4.5 mL Disposable Polystyrene Cuvettes (No. 14955125). The cuvette has a 10 mm path length, allowing accurate measurement in visible spectra range. A lid was used to prevented the evaporation caused by the heat from the visible light beam.

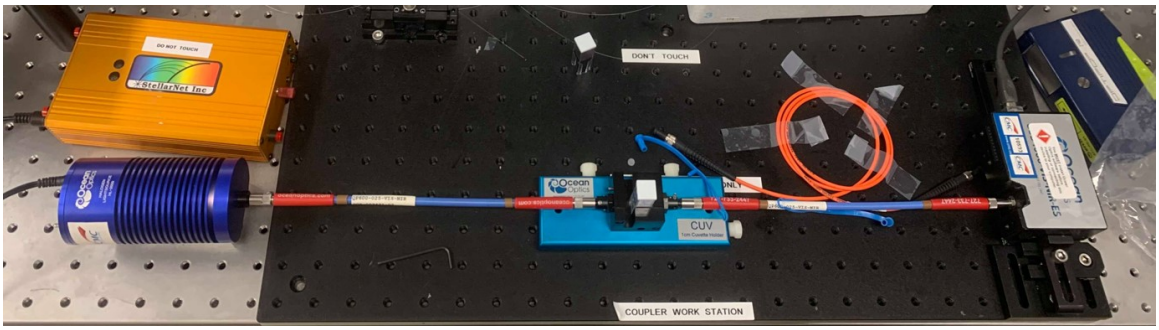

Figure 4.2: This figure is the setup of the spectral system.

Orange G	1	0.5	0.25
(mg/ml)	0.1	0.05	0.025
	0.01	0.005	0.0025
	0.001	0	
Crystal Violet	3	1.5	0.75
(mg/ml)	0.3	0.15	0.075
	0.03	0.015	0.0075
	0.003	0	

Table 4.1: The concentrations of Orange G and Crystal Violet solution for measurement. There are 11 settings for each dye, therefore, the full group is 121 settings.

4.2.1 Experimental Procedure

Two solutes, crystal violet (CV) and orange G (OG), were diluted in the deionized water with specified dilution ratios. When the solute concentration is high, too many solute molecules interact with each other and with the solvent; thus, the absorption curve is noisy. When the solute concentration is low, the detection of little solute molecules was mainly constrained by the spectral system's spectrometer. The limit of detection (LoD) of each solute was the lowest concentration distinguished from the absence of that solute. Figure 4.4(a) and 4.3(a) demonstrates the LoDs of CV and OG.

Table 4.1 summarizes the complete concentrations of each solute that were detected by the spectral system. A matrix of 11×11 is subject to this project's spectroscopy analysis. The spectrum of deionized water was measured as a full transmission label, and dark currents were denoised for each measurement. Oceanview software was used for collecting the samples' spectral data. The acquired data were analyzed and plotted using the in-house Matlab code.

4.3 Code

fPCA is a statistics tool that is not often used in spectral analysis, and it appears that the 'fda' only provides the R package whereas the 'fdapace' [53] provides both the R and Matlab package. 'fda' was chosen instead of 'fdapace' because 'fda' pro-

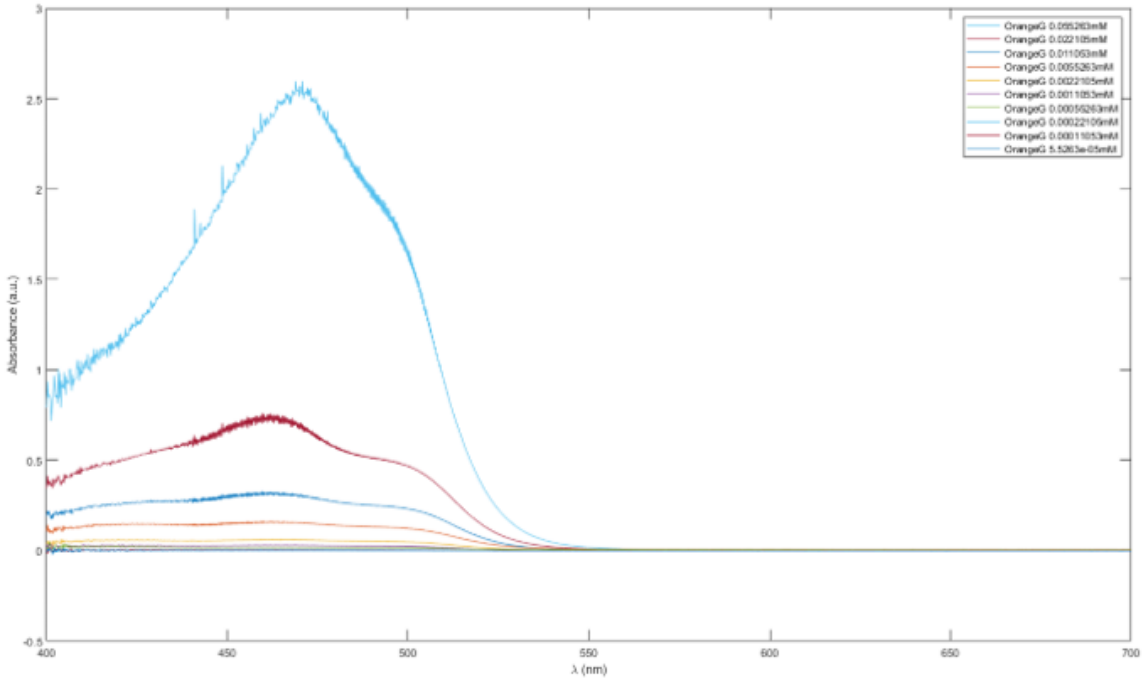

Figure 4.3: The limit of detection of orange G. The signal for concentration lower than $0.001 mg/ml$ is hard to detect.

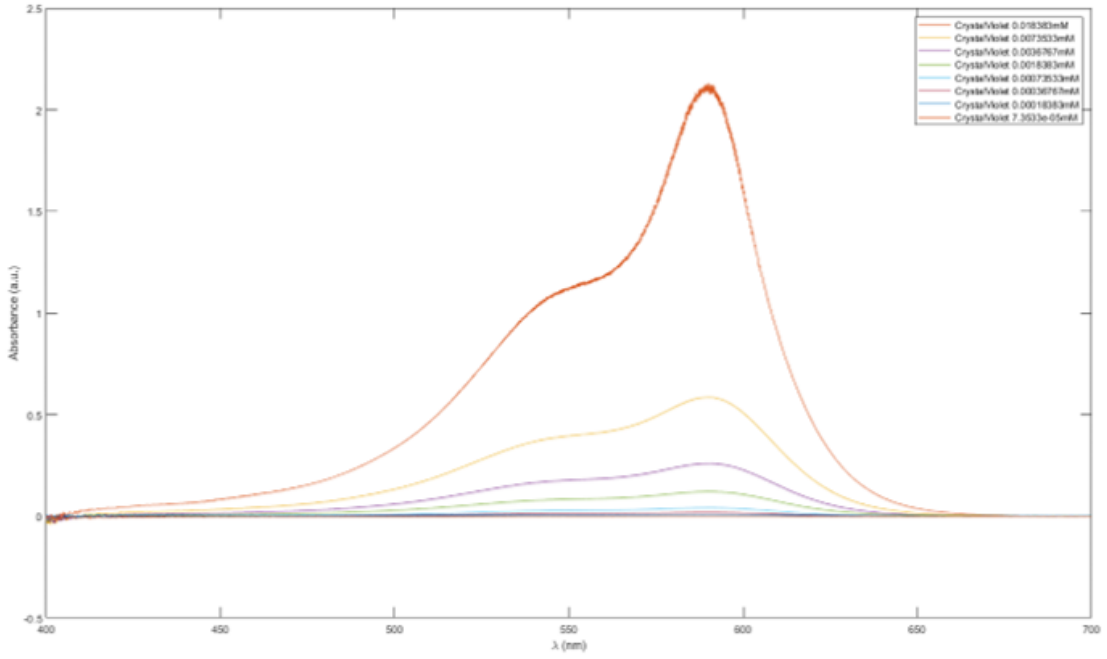

Figure 4.4: The limit of detection of crystal violet. The signal for concentration lower than $0.003 mg/ml$ is hard to detect.

vides a controllable parameter for tuning the number of output fPCs while 'fdapace' automatically generates based on its own metrics.

The implementation of the machine learning algorithms is in the language Python. Python is a popular coding language used in ML and there are several packages available that help alleviate coding work.

In short, data preprocessing part was written in R [54] with fda package [55]. ML algorithms were written in Python. The packages are: (a) xgboost [37], (b) scikit-learn [56], (c) Numpy [57] and (d) Pandas [58]. Result evaluation and plotting were mostly done in Matlab [59].

4.4 Running Environment

The model was mainly trained and tested on Compute Canada and a personal computer (PC). Compute Canada provides essential advanced research computing (ARC) services and infrastructure for Canadian researchers and their collaborators in all academic and industrial sectors. The data processing (fPCA) jobs were done on Compute Canada because this process required a lot of computing and space for storage. The rest of the tasks were done on a PC with an Intel i7-6800K CPU. Since the dimension of the input data was nine after using fPCA, the size of the input data was small enough that the training task could be done using a PC.

Chapter 5

Result Evaluation and Analysis

This chapter will present the result of fPCA. The evaluations of the classification models and quantification models are also presented based on the metric introduced in section 2.7.

The simulation datasets contains 100,000 samples for each SNR, and 95,000 samples were used for training while the rest 5,000 samples were used for testing. The empirically acquired solution absorbance datasets contain 117,846 samples. Only 112,846 samples were used for training and 5,000 samples were used for testing. The points of interests are as follows: (a) *Precision*, (b) *Recall*, (c) F_1 score, and (d) MDC of the algorithm for the classification results. In contrast, for the quantification results, the points of interests are (a) RMSE, (b) MAE and (c) R^2 score.

5.1 fPCA

When the data is high-dimensional, there are two popular tools for dimensional reduction: (a) Principal Component Analysis (PCA) and (b) fPCA. Compared with PCA, one advantage of fPCA is that fPCA is more potent in extracting core information [60]. For approximately 97% of cumulative explained variance in the 30 dB simulation dataset, fPCA requires less than nine components, while PCA would generate about 840 principle components [**?**]. fPCA is more powerful in terms of dimensional reduction. On the other hand, fPCA requires a longer processing time compared with PCA. In a computer with an Intel i7-6800K CPU, fPCA requires around 500 seconds to fit the eigenfunctions of 95,000 samples and 15 seconds for predicting the scores for remaining 5,000 testing samples. While PCA would only require 13s and 1s for

predicting the output scores of the same amount of data, respectively. Compared with PCA, fPCA is a more effective method for dimension reduction but requires a relatively long processing time. However, fPCA alleviates the computational cost for the following algorithm, and the algorithm can train on a CPU instead of a GPU.

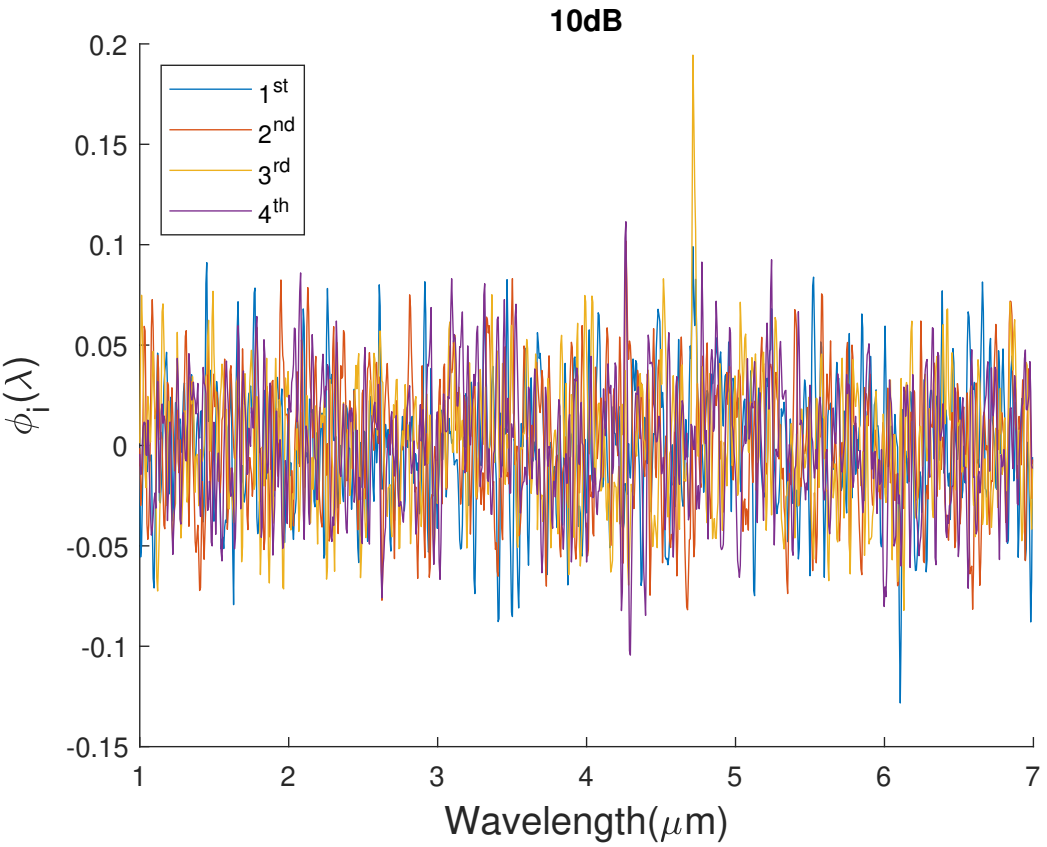

Figure 5.1: The plot of the first four eigenfunctions for the 10 dB synthetic quantification dataset. Compared with Fig. 5.2-5.4 the eigenfunctions are "jumping" and blurred by the noise.

Figure 5.1-5.4 are the plots of the first four eigenfunctions of 10 dB to 40 dB synthetic quantification dataset. The first eigenfunction in the 10 dB dataset have pulses, representing that is caught the valid signal. However, the remaining eigenfunctions in the plot look like random permutations, which indicates that the remaining fPCs mainly represent noise. The fPCs plots of the 20 dB to 40 dB datasets look similar because the data distribution of the dataset is the same. The fPCs in the three plots are smooth and clear, which means they mostly explain valid signals. However, the fourth eigenfunction of 20 dB dataset is also perturbed at $A \approx 0$, which means it is

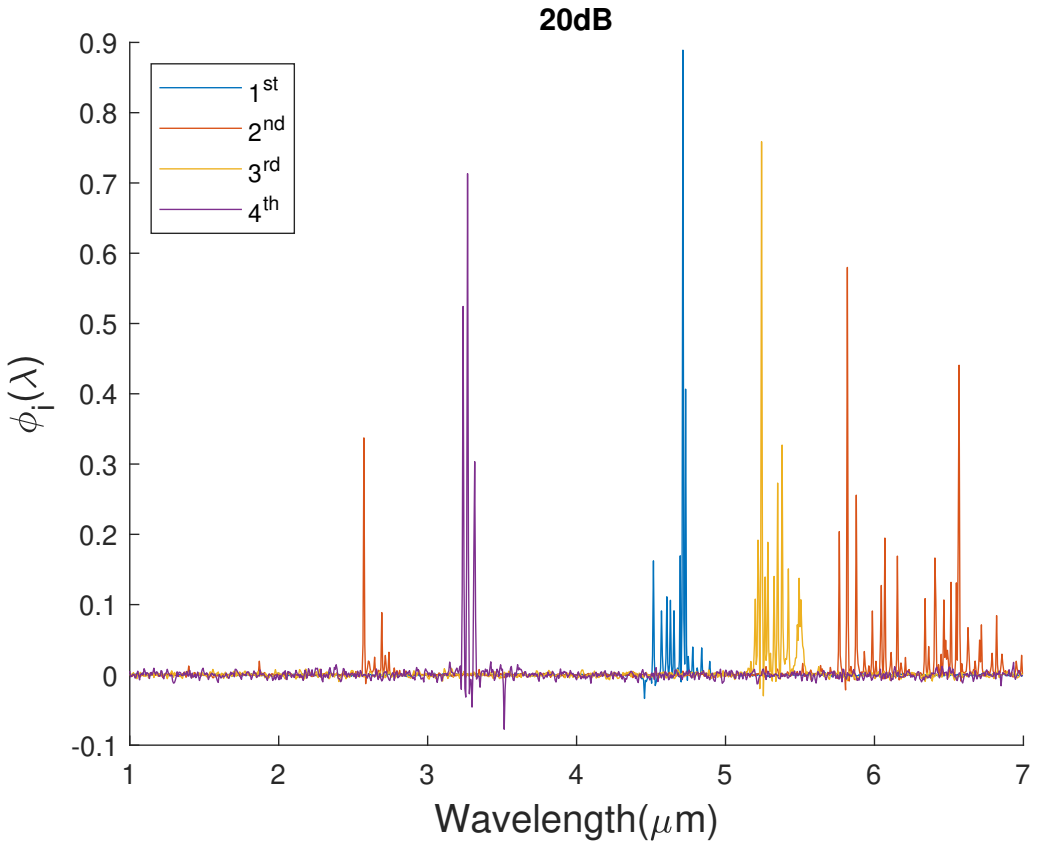

Figure 5.2: This figure shows the plot of the first four eigenfunctions for the 20 dB synthetic quantification dataset. The noise rate of the 20 dB dataset is 10 times less than 10 dB, therefore, it can obtain smooth and clean eigenfunctions. Note, the fourth eigenfunction (purple) is also perturbed around the x-axis, indicating that this eigenfunction is relatively noisy.

blurred by noise.

From these results, fPCA can be seen as a powerful tool due to it's dimensional reduction capabilities which provides a more powerful and efficient method of determining the eigenfunctions. Although the fitting time for the fPCA model is long, this process is done offline, thus, it is insensitive to processing time. In real life practice (testing phase), fPCA can process 5,000 samples per 15 seconds, which is around 300 spectra per second. The processing speed is quick enough for a field-based implementation.

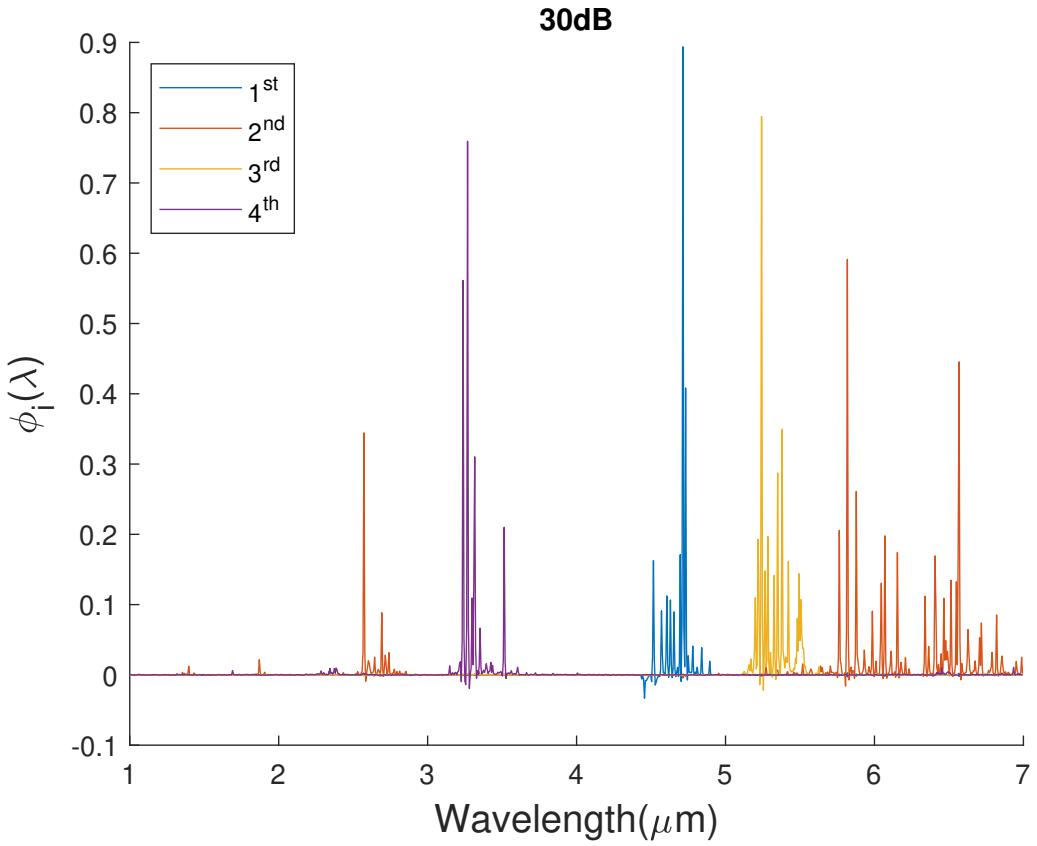

Figure 5.3: The plot shows the first four eigenfunctions for the 30 dB synthetic quantification dataset. For this dataset, the noise is minimal, therefore, smooth and clean eigenfunctions could be obtained.

5.2 Optimal Threshold

Optimal thresholding is applied to the classification models. The effectiveness of optimal thresholding will be shown in this section by comparing the results of 30 dB synthetic dataset using the XGBoost model with the following conditions: with optimal thresholding and without.

Figures 5.5-5.8 show the plots of accuracy of a particular gas as a function of threshold (t_i) in 10 dB to 40 dB. The 'x's on the plots are the peaks of each curve. In the 10 dB and 20 dB plot, each curve is approximately a parabola, and the axes of symmetry are close to 0.5; the peaks of each curve are higher than the values $t_i = 0.5$. For 20 dB, the peaks of each curve are gathered at around 0.42. On the other hand, for the 30 dB and 40 dB dataset, the curves are almost flat, except for HBr. In

this case, optimal thresholding creates an insignificant influence on accuracy. The peaks for CH_4 and CO are not in the interval $[0.2, 0.8]$, so the 'x's are not shown in the plots. Moreover, for the gases with small cross-sectional areas such as HBr, the curves' downward concave is steeper whereas gases with large cross-sectional areas such as HF have a less prominent concavity.

Figure 5.9 shows the increase in accuracy from optimal thresholding under certain SNR by calculating the gap between the maximum accuracy in the interval $[0.2, 0.8]$ and accuracy of $t_i = 0.5$. A large value in this graph indicates a significant gain in accuracy. For SNRs higher than 30 dB, optimal thresholding holds a little impact on accuracy except for HBr. For low SNRs, the optimal threshold would help improve the accuracy of each gas.

5.3 Classification

The performances of three different algorithms, XGBoost, forward neural network with optimal thresholding (FNN-OT), and Logistic regression are compared in this section. FNN-OT was proposed by Gan *et al.*, who have done classification using FNN-OT and partial least squares when binary relevance (PLS-BR) on the same simulated datasets [**?**]. Both the XGBoost and Logistic Regression models employed fPCA as the data pre-processing method for feature extraction and dimension reduction (Section 3.1).

Performance on Synthetic Data

The results are compared between XGBoost, FNN-OT and logistic regression, and it turns out that FNN-OT has a better performance when the SNR is extremely low, while the XGBoost is a lot better when SNR is higher than 20 dB. This characteristic makes XGBoost a better choice the SNR for a spectrometer is around 30 dB in a real scenario.

Table 5.1 presents the micro-averaged precision, recall and F_1 score at four different SNRs. Across all algorithms, when the SNR is low, the performances are poor. For the 10 dB dataset, all three models obtained a F_1 score of approximately 0.52, which is slightly better than a random guess. Furthermore, SNRs that were higher than 30 dB had F_1 scores of approximately 1 for all algorithms, and the XGBoost model had the highest F_1 scores among the three. This means that the XGBoost

model could almost successfully predict every sample.

As discussed in Section 2.7.4, the F_1 score is the most crucial metric for evaluation. As shown in Table 5.1, for SNRs lower than 20 dB, FNN-OT provided the highest F_1 score; on the other hand,XGBoost show the highest F_1 scores for SNRs higher than 30 dB. As mentioned earlier, 30db is the most important because spectrometers used in a real life scenario have a SNR of approximately 30 dB. As a result, XGBoost is most suitable for a real-life implementation.

When the noise level is low, PCA based algorithm has better performance, while for the high SNR datasets, fPCA based algorithm is better. One reason for this inconsistency is that high SNR functional data is smoother, and the correlation coefficient is bigger; the correlation between the pixels is stronger. The dominant eigenfunctions are easier to fit, and the lost information is less. On the other hand, noise 'decorrelates' the spectra signal, which reduces the smoothness of the functional data. If the smoothness is too bad to be treated as functional data, fPCA will perform worse than PCA.

MDC (Section2.7.5) is another important metric since it reflects the sensitivity of the algorithm. A smaller MDC means that the algorithm can detect the existence of a lower concentration analyte. The MDC is defined as the lowest concentration which gives a $F_1 = 0.8$. Concretely, the testing sets are sorted and divided into 30 small bins. The F_1 scores of each bin were then calculated. Intuitively, a bin with higher concentrations should have higher F_1 scores since they are more easily detectable. Since half of the concentrations are set to be zero, the first half of the bins will contain concentration instances of 0 ppm after sorting. The highest concentration of the first non-zero bin with $F_1 \geq 0.8$ is taken as the MDC.

Figure 5.10-5.13 show the MDCs of nine gases at SNRs ranging from 10 dB to 40 dB. As seen in figure 5.10, all the three algorithms performs bad at 10 dB, note the highest concentration of the entire dataset is $10\mu M$. Most gases cannot reach $F_1 = 0.8$ with the highest concentration. High noise rates block the prediction in the low SNR dataset. N_2O has a significant cross section area, thus is easier to detect and has a lower $MDC \approx 6$ ppm compared with other gases. Furthermore, as expected, the MDCs decline as the SNRs increase. The MDCs for XGBoost and FNN-OT are around $1.5 - 2$ ppm. These MDCs mean that both algorithms can confidently predict the existence of the gases unless the concentrations of gas are deficient. Across all SNRs except 40 dB, FNN-OT yields to smaller MDCs than XGBoost and logistic regression. This result is in consistent with the result in Table 5.1, since in low SNRs,

FNN-OT had better performance.

Micro-Averaged Precision			
SNR	XGBoost	FNN-OT	Logistic Regression
10dB	0.5304 ± 0.0028	0.562 ± 0.002	0.532 ± 0.002
20dB	0.8045 ± 0.0041	0.81 ± 0.03	0.742 ± 0.003
30dB	0.9786 ± 0.0010	0.983 ±0.002	0.9926 ± 0.0005
40dB	0.9971 ± 0.0004	0.9992 ±0.0002	1± 0.0
Micro-Averaged Recall			
SNR	XGBoost	FNN-OT	Logistic Regression
10dB	0.5197 ± 0.0074	0.594 ± 0.005	0.546 ± 0.010
20dB	0.7244 ± 0.0064	0.78 ± 0.01	0.742 ± 0.003
30dB	0.9328 ± 0.0016	0.908 ± 0.002	0.882 ± 0.002
40dB	0.9896 ± 0.0003	0.953 ± 0.002	0.887 ± 0.002
Micro-Averaged F_1			
SNR	XGBoost	FNN-OT	Logistic Regression
10dB	0.5249 ± 0.0032	0.578 ± 0.003	0.538 ± 0.005
20dB	0.7622 ± 0.0024	0.79 ± 0.02	0.767 ± 0.001
30dB	0.9551 ± 0.0008	0.9439 ± 0.0007	0.934 ± 0.001
40dB	0.9933 ± 0.0003	0.9755 ± 0.0009	0.941 ± 0.001

Table 5.1: The three tables are micro-averaged *Precision*, *Recall* and F_1 at different SNRs for simulated gas dataset. XGBoost and Logistic Regression (Section2.2) employ fPCA as the data pre-processing method for feature extraction and dimension reduction. The second column is the results from [**?**].

Performance on Experimentally Acquired Data

The 10-fold cross-validation results for the experimentally acquired dataset are shown in Table 5.2. It shows that XGBoost and FNN-OT outperforms logistic regression. However, the F_1 scores of each algorithm are approximately one, indicating that the

	XGBoost	FNN-OT	Logistic Regression
Precision	0.992± 0.002	0.995± 0.004	0.92±0.01
Recall	0.999±0.0003	0.996±0.001	1.0±0.0
F_1 score	0.995±0.0009	0.996±0.002	0.960±0.005

Table 5.2: The micro-averaged *Precision*, *Recall* and F_1 score of the Orange G (OG) and Crystal Violet (CV) solution dataset obtained using XGBoost, FNN-OT and logistic regression. The F_1 scores of each algorithm are approximately one, indicating that the algorithm can make an accurate prediction.

algorithm can make an accurate prediction.

In general, the SNR of a spectrometer is approximately 30 dB. Despite this, the algorithm's performance was better on the experimentally acquired dataset (30 dB) compared to it's performance on the 40 dB simulated datasets. This phenomenon may come from the following reasons:

- Data distributions for these two datasets are different. For the simulated dataset, half of the samples are assigned with 0 *ppm* for classification; whereas the CV and OG solution, approximately 9% of instances were 0 mg/ml for both solutes. The imbalance empirical dataset reduces the chance that the algorithm mislabels low concentration as negative.

- The minimum concentrations for both datasets are different. The distribution for non-zero samples in the simulation dataset is a log-normal distribution. Samples with extremely low concentrations are needed to test the MDC of the algorithm. For the solution absorbance dataset, the solution cannot be overly diluted due to the limitation of a spectrometer's measurement accuracy and background noise.

5.4 Quantification

XGBoost and Linear Regression were implemented to predict the concentration of a specific gas, and a standard method *PLS* was also implemented for comparison purposes. As shown in Table 5.3 and Figure 4.1(a), the maximum cross-sectional area of HBr is the smallest among the nine gases, while HF and N_2O have the largest ones.

Maximum Cross-section ($\times 10^{-19} cm^2$)				
C_2H_6	CH_4	CO	H_2O	HBr
22.31	20.36	24.15	12.05	2.99
HCl	HF	N_2O	NO	
24.73	103.02	43.46	7.17	

Table 5.3: Table of the maximum cross-section of the nine gases. HBr has the smallest cross-sectional area while HF has the largest cross-sectional area.

A small cross-section gas contributes a slight variance in the absorbances spectra, making the prediction more difficult. Therefore, the performance for HBr should be bad while the performance for HF and N_2O should be the best.

Performance using Synthetic Datasets

Table 5.4-5.6 display the results of the three algorithms using the synthetic dataset. Intuitively, quantification is a stricter job compared with classification and therefore requires more information to make an accurate prediction. The results of the $10dB$ dataset suggests that both models failed to predict accurate concentrations but generated the mean of the label to get a lower RMSE/MAE. For the $20dB$ dataset, either three algorithms can make precise predictions for gases with substantial cross-section areas, such as N_2O, and fails to predict other gases. The high noise rate in lower SNRs makes quantification nearly impossible unless the gas has a significant cross-section area. For high SNRs, both RMSE and MAE are approaching 0 ppm and R^2 score is almost 100%, suggesting that the model can explain nearly all variance. Across the three algorithms, XGBoost and linear regression yield a better result when the SNR is high, or the cross-section is immense, while PLS performs better when SNR is low.

Thus, it can be concluded that SNRs higher than 30 dB allow for accurate predictions. The SNR of a spectrometer can always be increased to get a more accurate prediction. Moreover, for gases with a substantial cross-sectional area, all the three algorithms could precisely predict the concentration even if the SNR is low (20 dB); for gases with small cross-section, higher SNRs are needed for more precise results.

Figure 5.14 is the plot of R^2 scores as a function of the amount of fPC score being used in the $30dB$ dataset. This plot shows that one eigenfunction contains the information of one gas, which means the information carried by each fPC is irrelevant.

Root Mean Squared Error (ppm)				
C_2H_6				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.89±0.03	2.53±0.01	0.421±0.003	0.0061±0.0009
PLS	2.50±0.02	2.41±0.02	0.422±0.004	0.0172±0.0078
LR	2.87±0.03	2.51±0.01	0.424±0.003	0.0043±0.0004
HBr				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.91±0.02	2.91±0.01	1.26±0.01	0.1150±0.0009
PLS	2.50±0.01	2.84±0.02	1.27±0.01	0.173±0.008
LR	2.88±0.01	1.30±0.01	1.28±0.01	0.1432±0.0010
HF				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.91±0.02	2.31±0.02	0.339±0.004	0.0337±0.0004
PLS	2.49±0.02	2.21±0.02	0.342±0.002	0.126±0.001
LR	2.30±0.02	2.30±0.02	0.342±0.004	0.035±0.0002
N_2O				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.36±0.02	0.368±0.003	0.0413±0.0006	0.0135±0.0003
PLS	1.91±0.02	0.372±0.003	0.098±0.002	0.0816±0.0010
LR	2.35±0.02	0.373±0.003	0.0375±0.0004	0.00375±0.00004

Table 5.4: The table of the RMSE of C_2H_4, HBr, HF and N_2O using XGBoost, PLS and linear regression (abbreviated as LR in the table).

Performance using Experimentally Acquired Dataset

Table 5.7 shows the RMSE, MAE and R^2 score of XGBoost, linear regression and PLS on the CV and OG solution dataset. Note this dataset's distribution is different from simulated dataset's distribution. Thus, it is not reasonable to directly compare RMSE and MAE with the two datasets. The results show that XGBoost also performs better than PLS and linear regression, which is in consistent with the synthetic datasets' results. The linear regression algorithm has the worst performance among the three methods. However, the R^2 scores of the three algorithms are approaching 100%, suggesting that the models could explain the full variance of the dataset. Note, these

Mean Absolute Error (ppm)				
C_2H_6				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.50±0.03	2.10±0.01	0.334±0.002	0.0061±0.0001
PLS	2.50±0.02	1.98±0.02	0.336±0.003	0.014±0.002
LR	2.49±0.03	2.09±0.01	0.337±0.003	0.0043±0.0004
HBr				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.51±0.02	2.51±0.02	0.999±0.010	0.1150±0.0009
PLS	2.50±0.01	2.44±0.02	1.02±0.02	0.138±0.006
LR	2.50±0.01	2.50±0.02	1.04±0.01	0.143±0.001
HF				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	2.51±0.02	1.89±0.02	0.269±0.003	0.0337±0.0004
PLS	2.49±0.02	1.80±0.02	0.272±0.002	0.101±0.001
LR	2.50±0.02	1.89±0.02	0.273±0.003	0.035±0.0002
N_2O				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	1.93±0.02	0.293±0.003	0.0329±0.0005	0.0135±0.0003
PLS	1.90±0.02	0.297±0.003	0.0784±0.002	0.0651±0.0009
LR	1.94±0.02	0.298±0.003	0.0299±0.0003	0.00375±0.00004

Table 5.5: The table of the MAE of C_2H_4, HBr, HF and N_2O using XGBoost, PLS and linear regression (abbreviated as LR in the table).

results were very similar to the results of gases with large cross-sections found in the 30 *db* simulated dataset. Therefore, when quantifying a analyte with large cross-section area, an accurate prediction can be make as long as the SNR is higher than 30 dB.

R^2 score				
C_2H_6				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	-0.16±0.003	0.23±0.01	0.421±0.003	0.999±0.000
PLS	-0.005±0.002	0.30±0.01	0.422±0.004	0.996±0.000
LR	0.000±0.0002	0.25±0.01	0.424±0.003	0.9998±0.0000
HBr				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	0.013±0.002	-0.016±0.02	0.9787±0.0003	0.9975±0.0001
PLS	-0.010±0.002	0.027±0.007	0.9786±0.0005	0.9964±0.0003
LR	0.0000±0.0002	0.0000±0.0003	0.9784±0.0003	0.9975±0.0001
HF				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	-0.016±0.004	0.36±0.01	0.9862±0.0003	0.9998±0.0000
PLS	-0.003±0.005	0.412±0.006	0.9860±0.0002	0.998±0.000
LR	0.000±0.004	0.35±0.01	0.9869±0.0004	0.9998±0.0000
N_2O				
SNR	$10dB$	$20dB$	$30dB$	$40dB$
XGBoost	0.33±0.01	0.9837±0.0004	0.9998±0.0001	0.99996±0.00000
PLS	0.357±0.008	0.9834±0.0002	0.9988±0.0001	0.9992±0.0000
LR	0.3409±0.0005	0.9833±0.0004	0.9998±0.0000	0.999998±0.000000

Table 5.6: The table of the R^2 score of C_2H_4, HBr, HF and N_2O using XGBoost, PLS and linear regression (abbreviated as LR in the table).

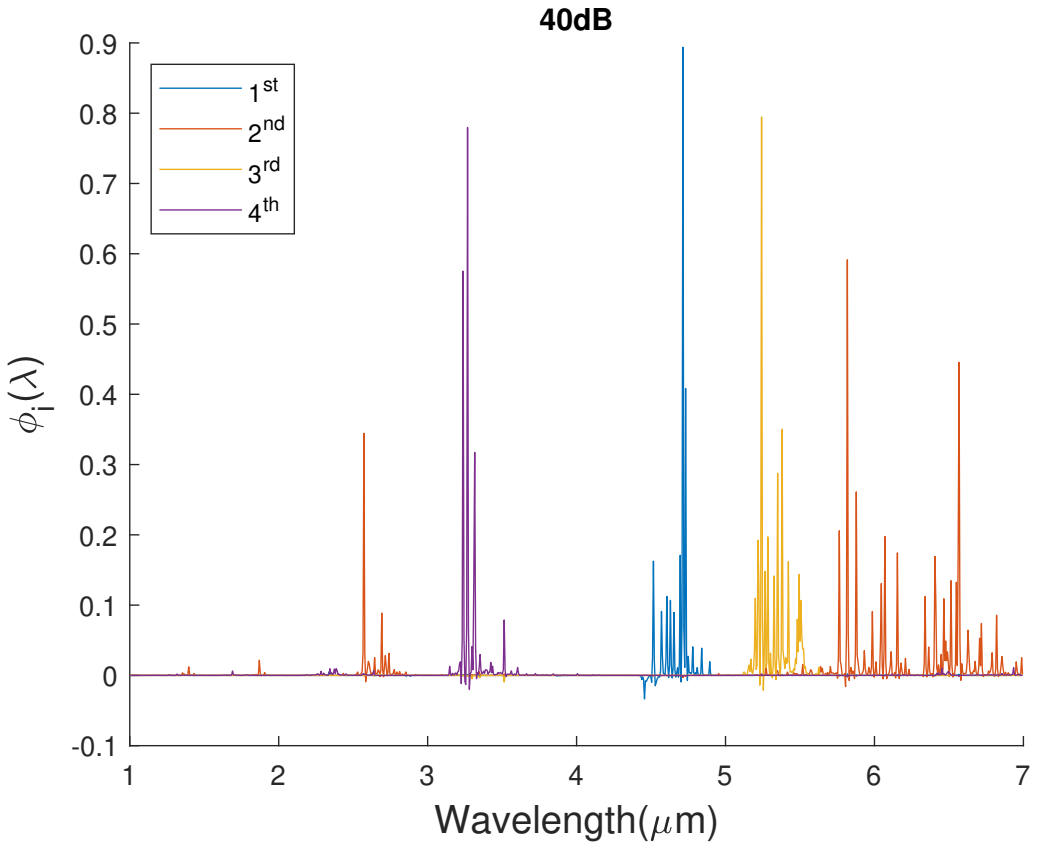

Figure 5.4: The plot of the first four eigenfunctions for the 40 dB synthetic quantification dataset. This plot is similar to Figure 5.3. Since the data distribution for the four datasets is the same, the dominant modes should be approximately the same. The biggest factor that differentiates Figures 5.1-5.4 is the noise level. In this dataset, when the SNR is high, the first nine eigenfunctions could approximately regenerate any instance in the dataset using Equation 2.2 (Figure 2.2). Moreover, from results discussed in Section 5.4, approximately one fPC represents one gas. Additionally, the first few eigenfunctions represent gases with large cross-sectional areas which contribute more to the absorbance spectra.

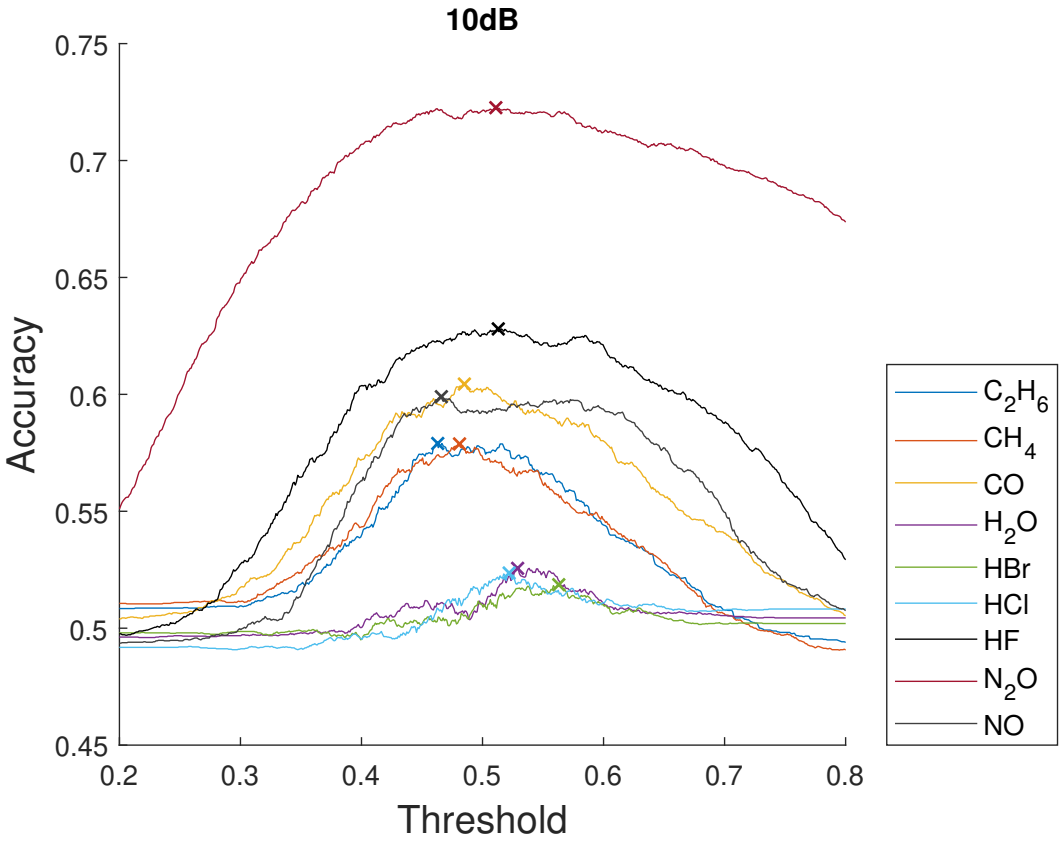

Figure 5.5: Accuracy for each gas molecule as a function of threshold (t_i) in the 10 dB synthetic dataset, the peaks of each curves are marked as crosses. The peaks often appears when $t_i \neq 0.5$, indicting that the optimal thresholding increases the performance.

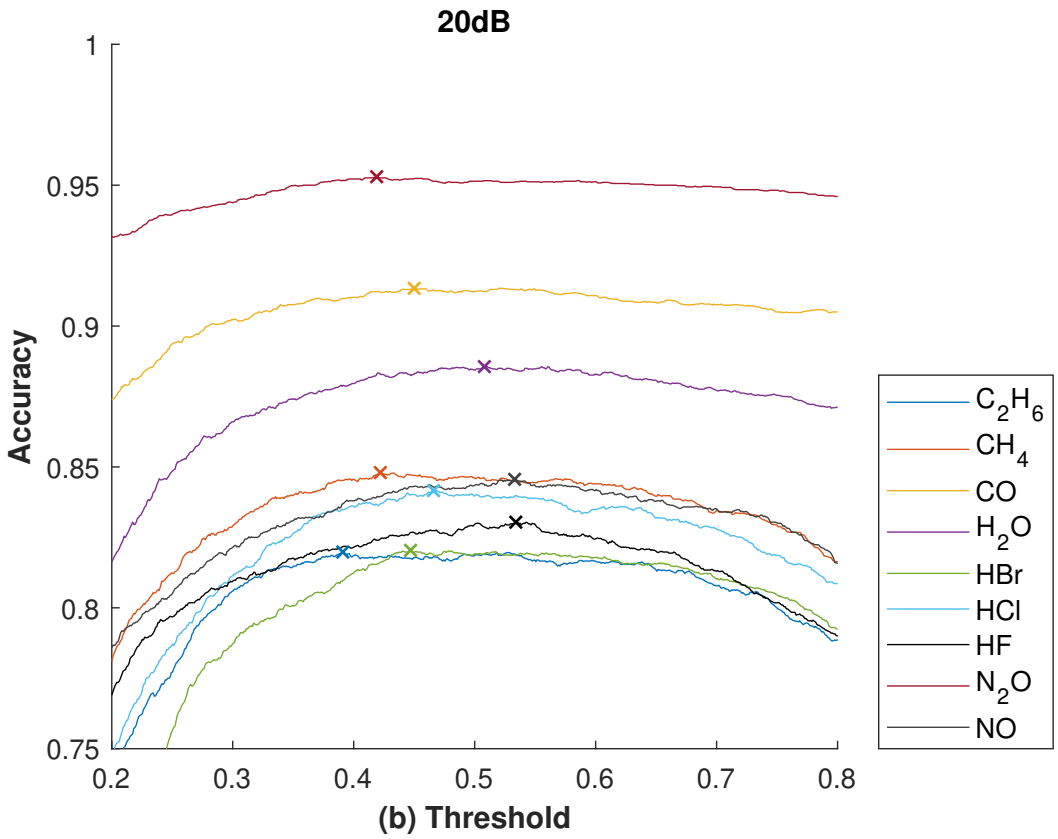

Figure 5.6: Plot of the accuracy for each gas molecule as a function of threshold (t_i) in the 20 dB synthetic dataset, the peaks of each curve are marked as crosses.

XGBoost	RMSE (mg/ml)	MAE (mg/ml)	R^2 (%)
Crystal Violet	0.039±0.0017	0.016±0.0005	99.97±0.03
Orange G	0.122±0.011	0.065±0.0030	99.97±0.005
Linear Regression	RMSE (mg/ml)	MAE (mg/ml)	R^2 (%)
Crystal Violet	0.155±0.012	0.116±0.002	99.65±0.001
Orange G	0.482±0.088	0.277±0.003	99.55±0.002
PLS	RMSE (mg/ml)	MAE (mg/ml)	R^2 (%)
Crystal Violet	0.104±0.008	0.077±0.0006	99.84±0.03
Orange G	0.244±0.090	0.128±0.0034	99.87±0.107

Table 5.7: The table shows the RMSE, MAE and R^2 for Orange G (OG) and Crystal Violet (CV) solution dataset using XGBoost, linear regression and PLS.

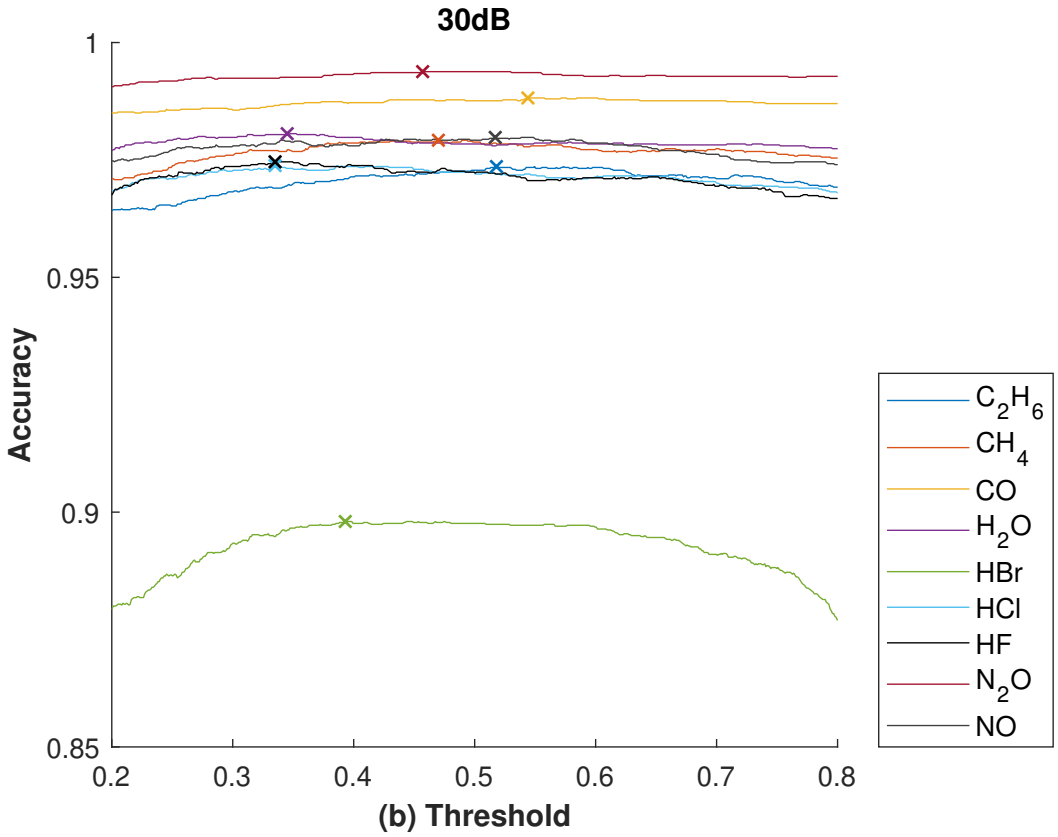

Figure 5.7: Plot of the accuracy for each gas molecule as a function of threshold (t_i) in the 30 dB synthetic dataset, the peaks of each curve are marked as crosses.

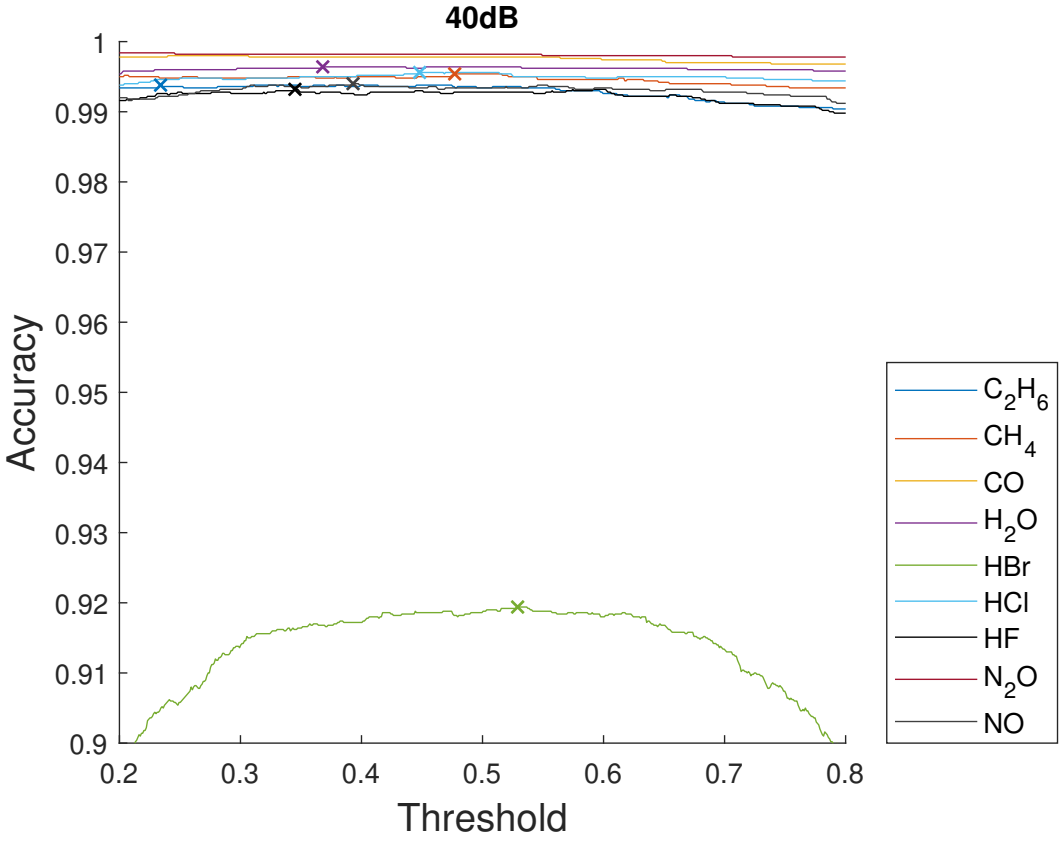

Figure 5.8: Plot of the accuracy for each gas molecule as a function of threshold (t_i) in the 40 dB synthetic dataset for classification, the peaks of each curves are marked as crosses. The curves are nearly flat, indicting that optimal thresholding does not not significantly increase the performance when $SNR = 40\ dB$.

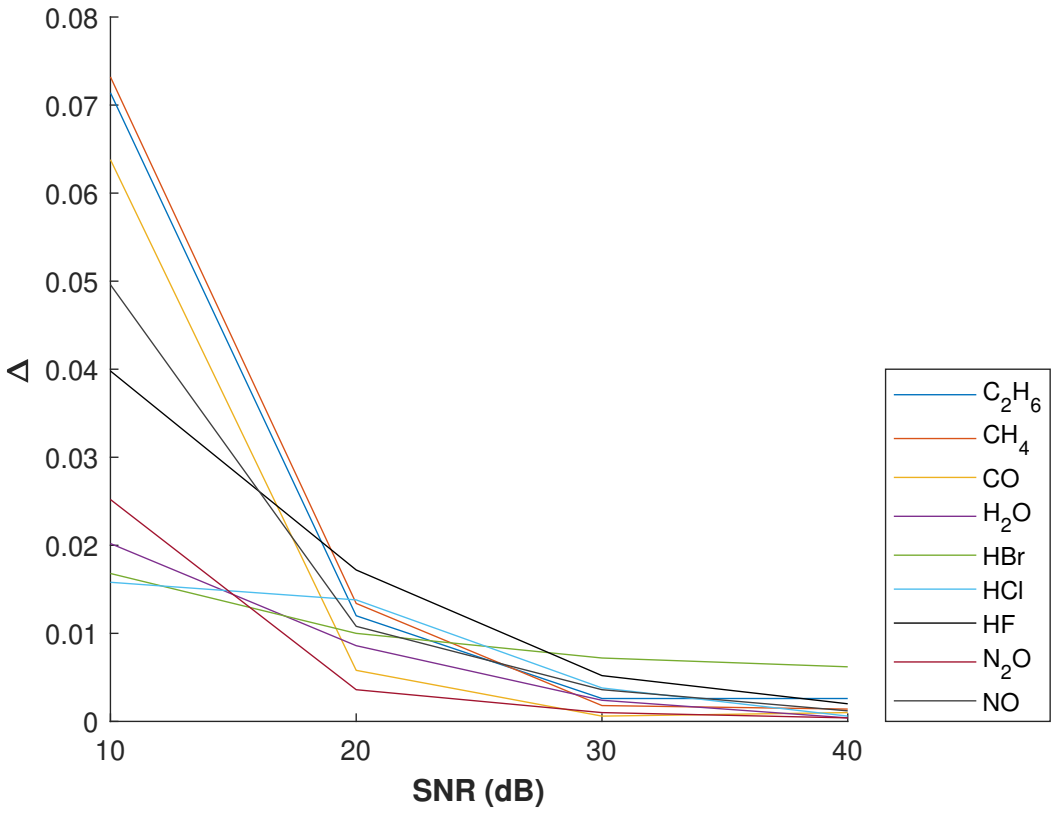

Figure 5.9: Plot of the gap between the maximum accuracy obtained with threshold in the interval $t_i \in [0.2, 0.8]$ and the accuracy of the threshold equals to 0.5 (A_{max} - $A_{t=0.5}$) for each gas molecule under certain SNRs. From the plot, it can be concluded that the gap decrease as the SNRs increase.

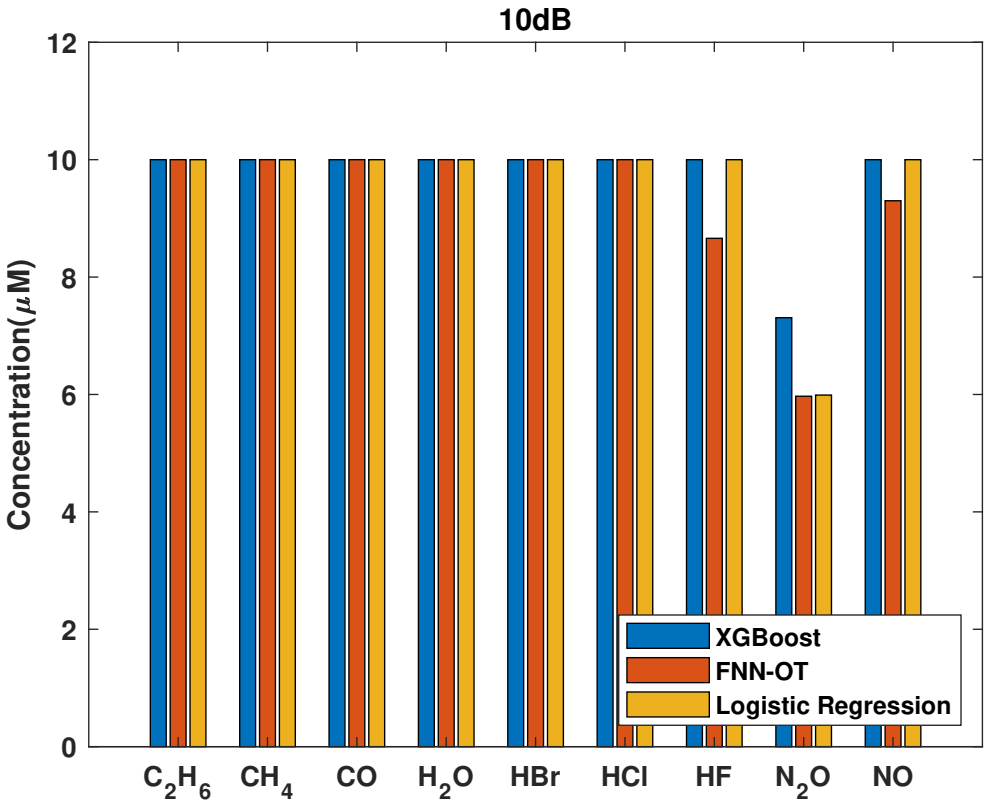

Figure 5.10: Bar graph plotting the MDCs of nine gases at the $10dB$ synthetic dataset for classification. The highest concentration of the dataset is $10\mu mol$ for all the synthetic dataset from $10dB$ to $40dB$. The four bar graphs show a general trend that a lower MDC could be obtained when the SNR is higher.

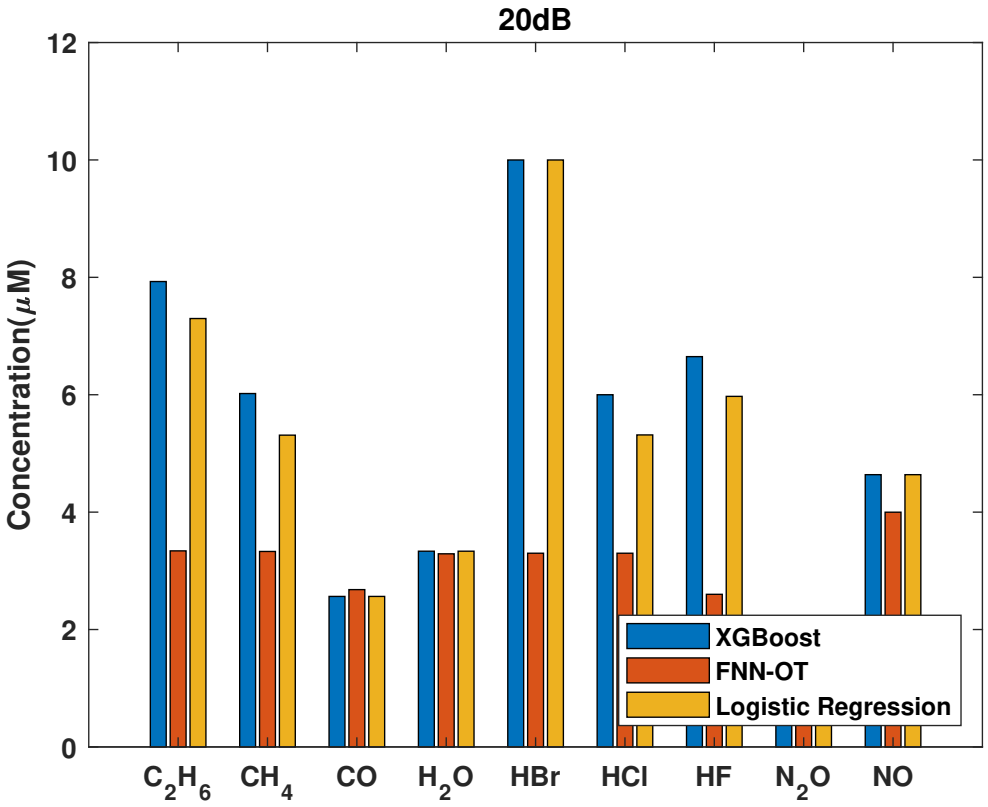

Figure 5.11: Bar graph plotting the MDCs of nine gases at the $20dB$ synthetic dataset for classification.

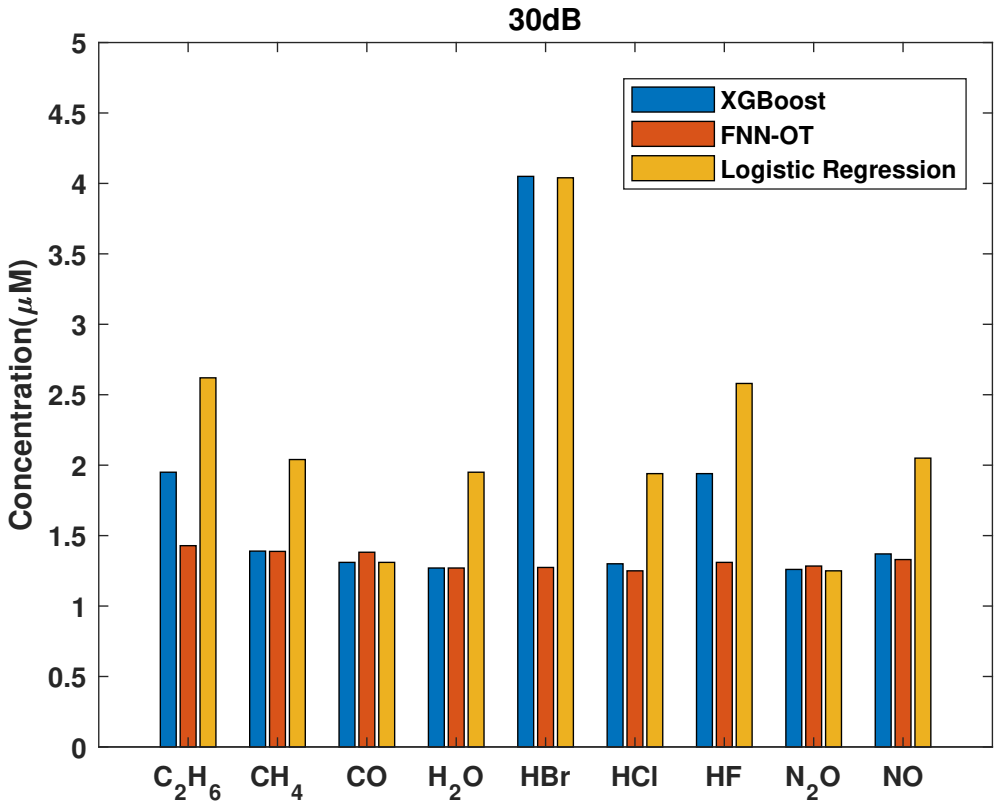

Figure 5.12: Bar graph plotting the MDCs of nine gases at the $30dB$ synthetic dataset for classification.

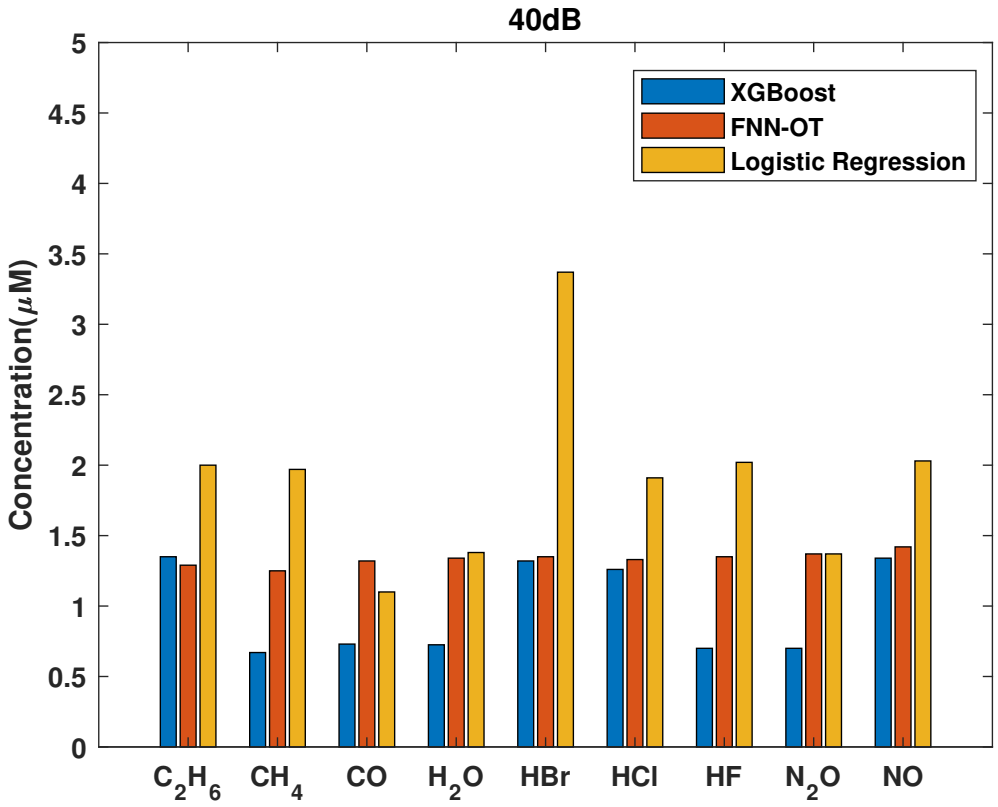

Figure 5.13: Bar graph plotting the MDCs of nine gases at the 40 dB synthetic dataset for classification.

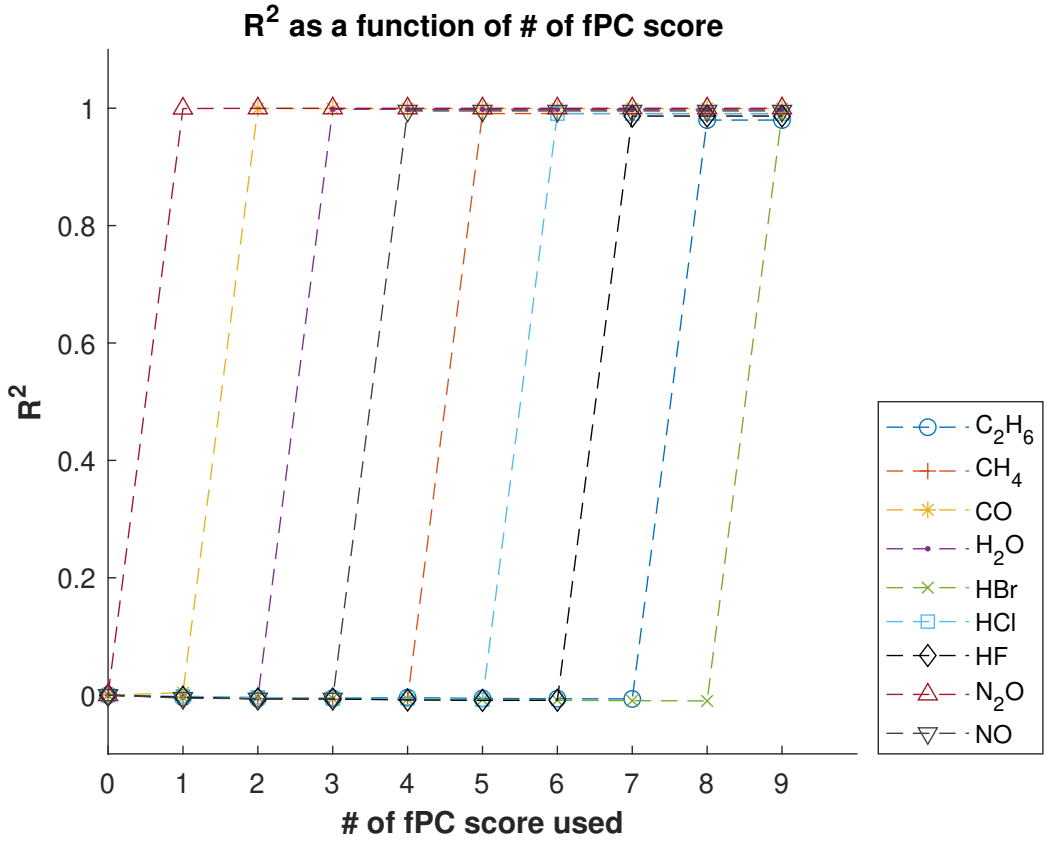

Figure 5.14: Plot of R^2 scores as a function of the amount of fPC score used in the $30dB$ dataset. This plot shows that one eigenfunction contains the information of one gas.

Chapter 6

Conclusions

This chapter summarizes the methods and results of this project. This chapter also demonstrates the advantages of the proposed algorithms. Furthermore, this chapter shows our contributions, and provides a discussion section which examines the limitations of the methods

6.1 Summary and Contribution

Several approaches for classifying and quantifying the absorbance spectrum data were proposed in this project. Based on the results of this project it has been shown that:

- fPCA is more potent than PCA in terms of feature extraction; however, the small cross-section gas signal may be blotted out by fPCA. This is because fPCA has a hard time distinguishing between the signal of the small cross-section area gas and noise.
- Compared with PCA, fPCA requires more time to fit the eigenfunctions (in the training phase); however, the time for computing output is tolerable.
- Nearly one output fPC represents the concentration for one gas. The order is the descending order of all gases' cross-sectional area. Thus, a linear transformation can be used to convert the fPCs to the output (concentrations).
- The cross-sectional area of an analyte is the critical feature for the performance of both classification and quantification. A gas with a more significant cross-sectional area is easier to detect.

- The optimal threshold method is helpful when the SNR is low; however, in the high SNR dataset, the increase in performance coming from optimal thresholding is negligible because the input signal is clear enough for making accurate predictions.
- The performance on the experimentally acquired dataset is better than the synthesized dataset. The main reason is that the label distribution for the experimentally acquired dataset is more straightforward than the one for the synthesized dataset.
- For the 30 dB synthesized dataset, the best classification algorithm can obtain a F_1 score of 0.9551 ± 0.0008, and the best quantification algorithm can obtain a RMSE value of 0.0794 ± 0.0006 ppm for CO and 1.26 ± 0.01 ppm for HBr, given the average concentration as around 5 ppm.

6.2 Limitations and Future Work

During this project, the following limitations were found:

- Compared with PCA, fPCA takes a longer time to predict the output scores.
- The model may not be complex enough for predicting the correlation dataset. In this project, the labels were uncorrelated, but in the real world, the concentration for two materials may be correlated. The proposed model cannot take advantage of this.
- The experimentally acquired dataset is not big enough to fully test the algorithm's performance.

Based on the current limitations and the potential solutions, several suggestions are proposed to improve performance in future studies:

- Test a new method for fitting the eigenfunctions and generating the output fPCs. fPCA is proven to "denoise" the input samples. However, the algorithm can be optimized for a shorter processing time.
- Combine the proposed approach with existing models such as FNN-OT [**?**] for better performance. FNN-OT has better performance in certain SNRs, so the ensemble model should outperformance the individual model across all SNRs.

- Experiment on other models to replace fPCA and machine learning algorithms. A More complicated model, such as Convolutional Neural Network (CNN), may be more reliable in extracting information from spectroscopic input data.
- Use dynamic thresholding instead of a uniform threshold for better performance.
- Acquire a larger experimental dataset for training and testing. The solution contains two analytes in this project, and the concentration has 121 sets. By increasing the number of analytes and the combination of concentrations, a more complex dataset can be obtained. Additionally, the background noise can be deliberately increased to test the performance under different SNRs.

Bibliography

[1] L. Gan, B. Yuen, and T. Lu, "Multi-label classification with optimal thresholding for multi-composition spectroscopic analysis," *Machine Learning and Knowledge Extraction*, vol. 1, no. 4, pp. 1084–1099, 2019.

[2] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[3] S. Patterson, "Letting the machines decide," *The Wall Street Journal*, vol. 13, 2010.

[4] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708, 2014.

[5] B. Chesney and D. Citron, "Deep fakes: a looming challenge for privacy, democracy, and national security," *Calif. L. Rev.*, vol. 107, p. 1753, 2019.

[6] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, "Natural language processing: an introduction," *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 544–551, 2011.

[7] Z. Xiong, W. Li, Q. Han, and Z. Cai, "Privacy-preserving auto-driving: a gan-based approach to protect vehicular camera data," in *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 668–677, IEEE, 2019.

[8] R. High, "The era of cognitive systems: An inside look at ibm watson and how it works," *IBM Corporation, Redbooks*, pp. 1–16, 2012.

[9] F.-Y. Wang, J. J. Zhang, X. Zheng, X. Wang, Y. Yuan, X. Dai, J. Zhang, and L. Yang, "Where does alphago go: From church-turing thesis to alphago thesis

and beyond," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 2, pp. 113–120, 2016.

[10] G. B. Goh, N. O. Hodas, and A. Vishnu, "Deep learning for computational chemistry," *Journal of computational chemistry*, vol. 38, no. 16, pp. 1291–1307, 2017.

[11] A. L. Tarca, V. J. Carey, X.-w. Chen, R. Romero, and S. Drăghici, "Machine learning and its applications to biology," *PLoS Comput Biol*, vol. 3, no. 6, p. e116, 2007.

[12] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[13] R. Sathya and A. Abraham, "Comparison of supervised and unsupervised learning algorithms for pattern classification," *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, no. 2, pp. 34–38, 2013.

[14] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[15] P. Perona and W. Freeman, "A factorization approach to grouping," in *European Conference on Computer Vision*, pp. 655–670, Springer, 1998.

[16] P. Strecht, L. Cruz, C. Soares, J. Mendes-Moreira, *et al.*, "A comparative study of classification and regression algorithms for modelling students' academic performance.," *International Educational Data Mining Society*, 2015.

[17] E. D. Liddy, "Natural language processing," 2001.

[18] J. Kong and S. Yu, "Fourier transform infrared spectroscopic analysis of protein secondary structures," *Acta biochimica et biophysica Sinica*, vol. 39, no. 8, pp. 549–559, 2007.

[19] I. N. Evdokimov, N. Y. Eliseev, and B. Akhmetov, "Assembly of asphaltene molecular aggregates as studied by near-uv/visible spectroscopy: I. structure of the absorbance spectrum," *Journal of Petroleum Science and Engineering*, vol. 37, no. 3-4, pp. 135–143, 2003.

[20] H.-P. Horz, A. Barbrook, C. B. Field, and B. J. Bohannan, "Ammonia-oxidizing bacteria respond to multifactorial global change," *Proceedings of the National Academy of Sciences*, vol. 101, no. 42, pp. 15136–15141, 2004.

[21] J. Claßen, F. Aupert, K. F. Reardon, D. Solle, and T. Scheper, "Spectroscopic sensors for in-line bioprocess monitoring in research and pharmaceutical industrial application," *Analytical and bioanalytical chemistry*, vol. 409, no. 3, pp. 651–666, 2017.

[22] C. Oppenheimer and P. R. Kyle, "Probing the magma plumbing of erebus volcano, antarctica, by open-path ftir spectroscopy of gas emissions," *Journal of Volcanology and Geothermal Research*, vol. 177, no. 3, pp. 743–754, 2008.

[23] W. Herrmann, M. Blake, M. Doyle, D. Huston, J. Kamprad, N. Merry, and S. Pontual, "Short wavelength infrared (swir) spectral analysis of hydrothermal alteration zones associated with base metal sulfide deposits at rosebery and western tharsis, tasmania, and highway-reward, queensland," *Economic Geology*, vol. 96, no. 5, pp. 939–955, 2001.

[24] L. S. Rothman, I. E. Gordon, Y. Babikov, A. Barbe, D. C. Benner, P. F. Bernath, M. Birk, L. Bizzocchi, V. Boudon, L. R. Brown, *et al.*, "The hitran2012 molecular spectroscopic database," *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 130, pp. 4–50, 2013.

[25] H. Lachheb, E. Puzenat, A. Houas, M. Ksibi, E. Elaloui, C. Guillard, and J.-M. Herrmann, "Photocatalytic degradation of various types of dyes (alizarin s, crocein orange g, methyl red, congo red, methylene blue) in water by uv-irradiated titania," *Applied Catalysis B: Environmental*, vol. 39, no. 1, pp. 75–90, 2002.

[26] A. Mittal, J. Mittal, A. Malviya, D. Kaur, and V. Gupta, "Adsorption of hazardous dye crystal violet from wastewater by waste materials," *Journal of colloid and interface science*, vol. 343, no. 2, pp. 463–473, 2010.

[27] J. O. Ramsay, "Functional data analysis," *Encyclopedia of Statistical Sciences*, vol. 4, 2004.

[28] T. Sypherd, M. Diaz, L. Sankar, and P. Kairouz, "A tunable loss function for binary classification," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 2479–2483, IEEE, 2019.

[29] A. Ash and M. Shwartz, "R2: a useful measure of model performance when predicting a dichotomous outcome," *Statistics in medicine*, vol. 18, no. 4, pp. 375–384, 1999.

[30] S. Roweis, "Em algorithms for pca and spca," *Advances in neural information processing systems*, pp. 626–632, 1998.

[31] J. Hulland, "Use of partial least squares (pls) in strategic management research: A review of four recent studies," *Strategic management journal*, vol. 20, no. 2, pp. 195–204, 1999.

[32] L. O. Jimenez and D. A. Landgrebe, "Supervised classification in high-dimensional space: geometrical, statistical, and asymptotical properties of multivariate data," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 28, no. 1, pp. 39–54, 1998.

[33] I. T. Jolliffe, "Principal components in regression analysis," in *Principal component analysis*, pp. 129–155, Springer, 1986.

[34] F. Ferraty and P. Vieu, *Nonparametric functional data analysis: theory and practice*. Springer Science & Business Media, 2006.

[35] M. Jones and J. A. Rice, "Displaying the important features of large collections of similar curves," *The American Statistician*, vol. 46, no. 2, pp. 140–145, 1992.

[36] Z. Lin, L. Wang, and J. Cao, "Interpretable functional principal component analysis," *Biometrics*, vol. 72, no. 3, pp. 846–854, 2016.

[37] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[38] L. Torlay, M. Perrone-Bertolotti, E. Thomas, and M. Baciu, "Machine learning–xgboost analysis of language networks to classify patients with epilepsy," *Brain informatics*, vol. 4, no. 3, pp. 159–169, 2017.

[39] J. Wang and M. Gribskov, "Irespy: an xgboost model for prediction of internal ribosome entry sites," *BMC bioinformatics*, vol. 20, no. 1, p. 409, 2019.

[40] H. Dong, X. Xu, L. Wang, and F. Pu, "Gaofen-3 polsar image classification via xgboost and polarimetric spatial information," *Sensors*, vol. 18, no. 2, p. 611, 2018.

[41] D. Chakraborty and H. Elzarka, "Early detection of faults in hvac systems using an xgboost model with a dynamic threshold," *Energy and Buildings*, vol. 185, pp. 326–344, 2019.

[42] D. Nielsen, "Tree boosting with xgboost-why does xgboost win" every" machine learning competition?," Master's thesis, NTNU, 2016.

[43] G. A. Seber and A. J. Lee, *Linear regression analysis*, vol. 329. John Wiley & Sons, 2012.

[44] H. Abdi *et al.*, "The method of least squares," *Encyclopedia of measurement and statistics*, vol. 1, pp. 530–532, 2007.

[45] R. E. Wright, "Logistic regression.," 1995.

[46] Z. C. Lipton, C. Elkan, and B. Narayanaswamy, "Thresholding classifiers to maximize f1 score," *stat*, vol. 1050, p. 14, 2014.

[47] W. Kahan, "Ieee standard 754 for binary floating-point arithmetic," *Lecture Notes on the Status of IEEE*, vol. 754, no. 94720-1776, p. 11, 1996.

[48] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[49] J. T. Townsend, "Theoretical analysis of an alphabetic confusion matrix," *Perception & Psychophysics*, vol. 9, no. 1, pp. 40–50, 1971.

[50] K. Foltz Biegalski and S. Biegalski, "Determining detection limits and minimum detectable concentrations for noble gas detectors utilizing beta-gamma coincidence systems," *Journal of Radioanalytical and Nuclear Chemistry*, vol. 248, no. 3, pp. 673–682, 2001.

[51] J. Sun, G. Zhu, X. Guo, L. Zhang, X. Zou, and Z. Gan, "Using wavelength modulation spectroscopy technique to detect trace ammonia gas in near-infrared spectral region," in *2016 17th International Conference on Electronic Packaging Technology (ICEPT)*, pp. 1085–1089, IEEE, 2016.

[52] R. L. Mitchell, "Permanence of the log-normal distribution," *JOSA*, vol. 58, no. 9, pp. 1267–1272, 1968.

[53] Y. Chen, C. Carroll, X. Dai, J. Fan, P. Hadjipantelis, K. Han, H. Ji, H. Müller, and J. Wang, "fdapace: Functional data analysis and empirical dynamics," 2020.

[54] R Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[55] J. Ramsay, H. Wickham, M. J. Ramsay, and S. deSolve, "Package 'fda'," 2020.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[57] T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.

[58] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, Austin, TX, 2010.

[59] MATLAB, *version 7.10.0 (R2020a)*. Natick, Massachusetts: The MathWorks Inc., 2020.

[60] Y. Nie and J. Cao, "Sparse functional principal component analysis in a new regression framework," *Computational Statistics & Data Analysis*, p. 107016, 2020.