12-3-2021

Exploring Non-Invasive Features for Continuous Glucose Monitoring

Brian Allen Bogue Jimenez

EXPLORING NON-INVASIVE FEATURES FOR CONTINUOUS GLUCOSE MONITORING

by

Brian Bogue-Jimenez

A Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Major: Electrical and Computer Engineering

The University of Memphis

December 2021

Acknowledgements

This document is dedicated to my family for their support in my pursuit of an education and career. To Dr. Doblas for her support in this project and for helping me push myself further in my endeavors. To my roommates for helping with the rent. To my dog, Koa, for being a good boy. To Drs. Huang and Powell for their guidance and assistance. Finally, this project would not be possible without the assistance of several faculty mentors at the University of Memphis, as well as the institution itself.

Abstract

Bogue-Jimenez, Brian. M.S. The University of Memphis. October 2021. Exploring Continuous Non-invasive Solutions for Blood Glucose Monitoring. Major Professor: Dr. Ana Doblas.

Glucose monitoring technologies allow users to monitor glycemic fluctuations (e.g., current glucose levels in their blood, also known as glycemia). This is particularly important for individuals who suffer from diabetes mellitus (DM), commonly referred to as diabetes. Traditional self-monitoring blood glucose (SMBG) devices require the user to prick their finger and extract a blood drop to measure the blood glucose based on chemical reactions with the blood. Unlike traditional glucometer devices, non-invasive continuous glucose monitoring (NICGM) devices aim to solve these issues by consistently monitoring users' blood glucose levels (BGL) and without invasively acquiring a sample. This Master Thesis aims to investigate the feasibility of a novel approach to NICGM via the use of off-the-shelf wearable sensors and the integration of learning-based models (i.e., machine learning). Several sensors were purchased to generate our own dataset with an increased feature set for studying possible relationships between glucose and non-invasive biometric measurements. Two datasets were collected for this study: (1) the OhioT1DM dataset, which is a publicly available dataset that can be obtained by contacting Ohio University; and (2) the UofM dataset, which was created by this research team. Both the Ohio dataset and our UofM dataset are passed through a machine learning pipeline that tests several models to determine whether the features are sufficient for predicting blood glucose concentrations. While preliminary results seem optimistic, a larger dataset is required to make conclusions about the feasibility of this approach.

Table of Contents

Abbreviations

Abbreviation	Meaning
BGL	Blood Glucose Level
BP	Blood Pressure
BVP	Blood Volume Pulse
CGM	Continuous Glucose Monitoring
DM	Diabetes Mellitus
EDA	Electrodermal Activity
EIS	Electrical Impedance Spectroscopy
GON	Glucose Oxidase Needle
GSR	Galvanic Skin Response
HBGM	Home Blood Glucose Monitoring
HR	Heart Rate
HRV	Heart Rate Variability
IBI	Inter-beat Interval
ISF	Interstitial Fluid
ML	Machine Learning
NDDG	National Diabetes Data Group
NICGM	Non-Invasive Continuous Glucose Monitoring
NIR	Near-infrared
SMBG	Self-Monitoring Blood Glucose
SpO2	Oxygen Saturation
TDC	Tissue Dielectric Constant

List of Tables

List of Figures

Chapter 1: Introduction

Diabetes Mellitus, colloquially known as diabetes, has been estimated to affect 450 million people of the global population [1]. This condition is characterized by abnormal levels of blood sugar. Furthermore, there are three main types of diabetes, classified by the National Diabetes Data Group (NDDG) as type 1, type 2, and Gestational Diabetes [2]. Type 1 diabetes, formerly known as juvenile diabetes, is an autoimmune disorder that a person is born with. This form of diabetes arises from the pancreas's inability to produce enough, or any, insulin. An individual who suffers from this condition must undergo daily insulin therapy via insulin injections or an insulin pump. If insulin levels are too low, the result will be that the blood glucose levels (BGL) will be too high, which is known as hyperglycemia. If too much insulin is administered, this will cause the BGL to be too low, which is known as hypoglycemia. Type 2 diabetes, also known as adult-onset diabetes and the most prevalent of the three, frequently causes hyperglycemia due to insulin resistance. In other words, the body builds a tolerance to insulin and can no longer adequately process it [2]. The third type of diabetes, gestational diabetes, also results in hyperglycemia. This condition is usually not chronic but can be dangerous to both the mother and child. Aside from these three types of diabetes mentioned, the NDDG also states several "Impaired glucose intolerance" disorders can result in symptomatic and asymptomatic individuals.

Complications often arise in individuals that suffer from diabetes. Many of these can be life-threatening, such as cardiovascular disease and renal failure. Other, less severe complications include nerve damage, ketosis, and various skin conditions. All of which dramatically affect the quality of life of a patient. Furthermore, diabetes incurs a significant cost on the economy [3]. In the United States alone, it is estimated that 24.7 million adults (i.e., 9.7%

of the population) have diabetes. This, in turn, costs the economy $327 billion in 2017. Direct medical costs comprise $237 billion, and the remaining is attributed to indirect costs such as the loss of productivity. For those with diabetes, medical costs are 2.3 times greater than their non-diabetic peers, meaning that, on average, a diabetic person will spend $9,601 per year managing their diabetes. This cost is only expected to rise as it has in the past, by 26% from 2012 to 2017.

As previously mentioned, both hyper- and hypoglycemia, if not corrected within a timely manner, can result in long-term health complications. Even so, many diabetic people do not regularly check their BGL due to how much of a chore it is. Traditional glucose monitoring devices require obtaining a blood sample invasively that is then measured via an electrochemical sensor. Along with being an uncomfortable ordeal for diabetic people of all ages, this process only results in a *snapshot* of the user's overall condition, as eloquently put by Gonzales *et al.* 2019 [1]. In this paper, the author further defines these types of devices as self-monitoring blood glucose (SMBG) devices and differentiates them from continuous glucose monitoring (CGM) devices, which allow its user to have a greater overall idea of how their body's BGL fluctuate throughout the day with continuous and automatic measurements.

1.1: Overview

Accurately being able to measure blood glucose is an essential step in the healthcare of diabetes patients. The proposed system takes a synergistic approach in which non-invasive biometrics measurements are combined with machine learning algorithms to predict blood glucose levels (BGLs) non-invasively and accurately estimate BGLs using a non-invasive system. The investigated non-invasive features include heart rate (HR), skin temperature, heat flux, electrodermal activity (EDA, also known as galvanic skin response, i.e., GSR), pulse oximetry, ambient temperature, and ambient humidity.

The goal of this research is to test different non-invasive features using machine learning (ML) approaches, and their performance in estimating the blood glucose of an individual based on the features obtained. Two datasets have been collected which are relevant to this project. The first dataset is from the "OhioT1DM Dataset for Blood Glucose Level Prediction" (a.k.a. Ohio Dataset) [4]. The second is a dataset created by the research team from the University of Memphis (i.e., the UofM Dataset) for this Master thesis.

1.2: Objectives/Aims

This thesis is divided into two main parts. First, we investigate the application of different ML algorithms to the existing Ohio dataset. In the Ohio dataset, the BGLs range from 40 mg/dl to 400 mg/dl. After this, we investigate a scaled-down version of the Ohio dataset, whose BGL ranges from 60 mg/dl to 200 mg/dl to assess if the combination of the features from the Ohio dataset and ML algorithms is sufficient to predict BGLs within the healthy/normal range. We aim to answer whether features currently available in smartwatch-like wearable devices are enough to monitor and predict glucose values or, oppositely, there is a need for adding new features.

The second objective addresses the need for additional features to generate a more complete dataset (the UofM dataset). This dataset was collected by a research team at the University of Memphis, utilizing several non-invasive sensors. The UofM dataset has been generated using three individual healthy subjects.

1.3: Contributions and Impact

This project provides the following contributions. Firstly, it explores the combination of non-invasive features integrated onto NICGM device and machine learning to predict and

monitor blood glucose values of a healthy and diabetic individual. Such a device would significantly enhance the lifestyle of diabetic patients and have broader applications in the medical industry, such as sports science and nutrition. This investigation also provides a strong foundation for future works in this field of study.

In addition, this research study results in a more complete non-invasive dataset that could be useful for a broad range of applications, enabling the study of personal versus universal health monitoring models. In other words, having information from multiple participants would allow us to compare the viability of a global model for all participants versus personalized models for each individual. Finally, this work describes and implements popular methods of analyzing datasets and the performance of machine learning models

Chapter 2: Background and Literature Review

This chapter provides a literature review of current monitoring methods for blood glucose levels. The two most accurate methods for measuring BGL in a laboratory setting are the enzymatic-amperometric and hexokinase methods [5]. Both these methods rely on an enzymatic reaction produced by exposing a sample of blood to these enzymes. Once this happens, a chemical reaction takes place, which creates some energy that can be measured as a current. This current is then measured, and since the amount of current produced is proportional to the amount of glucose present in the blood, a numeric value for the glucose can be obtained. Both these methods are highly accurate and sensitive, making them ideal clinical "ground truth" for other systems to be compared to.

A nearly identical approach is employed in SMBG devices, commonly referred to as glucometers. A person uses a lancelet to prick their finger to obtain blood. Once this blood is obtained, it is absorbed by a sensor strip that contains an enzyme, as stated above [5]. The current is then measured, and a numeric value for the user's BGL is shown on the screen. Although these devices are very beneficial to diabetic patients, they do present some drawbacks. The first and most apparent is the invasiveness of the procedure. As one can imagine, creating an open wound multiple times a day can be uncomfortable and presents the opportunity for the open wound to become infected. Additionally, these home monitoring techniques have a low frequency of measurements that entirely depend on the user. As a hypothetical scenario, if a diabetic user takes 3-5 measurements a day, this is all the information they must go on. This level of discretization leads to a great deal of uncertainty in between measurements.

To accommodate for the weaknesses of traditional home monitoring techniques, several methods have arisen. Some of these methodologies have even made it to the market as

commercial devices. What follows is a discussion of several devices, including a small summary of their working principle, and benefits and/or drawbacks.

2.1: Glucose Oxidase Needle (GON) (e.g., Dexcom G6 and Freestyle Libre)

The first method is referred to as the glucose-oxidase needle (GON) approach. The review written by Cappon *et al.* 2019[6] refers to them simply as CGM devices. This approach has seen a great deal of commercial success, and several devices have used this technology. For these devices to work, the user must wear a glucose-oxidase-doped platinum electrode deposited on a needle inserted in the subcutaneous tissue to ignite and catalyze glucose oxidation [6]. Therefore, this method has been classified a minimally invasive since the needle must pierce the subcutaneous layer of skin. These sensors also contain a transmitter that sends the measurements to a complementary device or, more recently, smartphones. Since their inception in 1999, these devices have improved their accuracy, and nowadays, they perform similarly to traditional SMBG devices. These devices also provide additional features to the user such as trend arrows, alerts and alarms, remote monitoring for physicians or parents, and compatibility with insulin pumps. Their success has proven how useful CGM can be to diabetic patients. However, these devices still present some drawbacks. For example, these devices rely on sensors with a limited lifespan, needing to be replaced every two weeks at most. The continuous purchase of sensors becomes costly and leads to unnecessary waste. In addition, this approach relies on measuring glucose level in the subject's interstitial fluid (ISF), rather than their BGL. This results in a delay from the true BGL measurement of about 10-20 minutes, not providing real-time measurements [7].

2.2: Electrical Impedance Spectroscopy

The second method is electrical impedance spectroscopy (EIS), which was utilized in a device designed by Caduff *et al* called the PENDRA. This device utilized capacitive sensors to measure the dielectric properties and impedance of human skin. A good correlation was shown between these parameters and blood glucose [8]. This device showed great promise as the first commercial NICGM device; however, it ultimately failed in the market due to its poor performance in real-world setting (i.e., it was only capable of successfully estimating the BGL of two-third of users). Moreover, it required a difficult calibration procedure that had to be performed by a team of health care professionals [9].

Most of these EIS devices rely on the implementation of capacitance sensing via interdigital sensors [10]. These interdigital sensors (a.k.a. fringing field sensors) can measure the dielectric properties of the skin. When these electromagnetic fields cross the human skin, the equivalent impedance of the subcutaneous tissue can be measured at frequencies higher than 200 kHz. This impedance has been shown to be well correlated with the amount of glucose concentration in the ISF of the skin. However, a significant challenge in the application of these devices has been the sensitivity of these devices under different levels of applied force.

Additionally, research by Caduff's team has indicated that global models perform significantly worse in human trials [11]. Their study found that the individual model performed better than the global model for all their test subjects, presenting an interesting question of how accurate these types of devices are for different types of skin characteristics. They proposed the use of more sophisticated non-linear statistical modeling techniques in future works.

2.3: Metabolic Heat Confirmation

Blood glucose can also be estimated through a technique known as metabolic heat conformation (MHC). This technique takes advantage of the fact that most of the heat generated by the human body is a result of the cellular process that converts glucose into energy [12]. This heat is then dispersed into the surrounding environment in the form of convection, evaporation, and radiation. Within the MHC technique, other features are also considered, including ambient information (i.e., temperature and humidity), hemoglobin concentration, oxyhemoglobin concentration, and blood flow rate. These features are measured using multi-wavelength spectroscopy at the fingertip of a patient's hand [12]. The main drawback of this approach is that it is sensitive to environmental factors, and current implementations are not continuous.

2.4: GlucoTrack

The last device to be discussed is the GlucoTrack, which employs a combination methodology approach [13]. This device combines three different techniques for non-invasively estimating the BGL from the user's earlobe. These three techniques are ultrasonic, electromagnetic, and thermal. The ultrasonic sensor measures the speed that an acoustic wave travels through the user's earlobe. The electromagnetic sensor does something equivalent but with electromagnetic impedance. The thermal approach applies a known amount of energy for a predetermined period and obtains the heat transfer characteristics of the tissue. Although none of these techniques directly measure the BGL, a strong correlation has been shown to exist between the individual measurement characteristics and the BGL in the earlobes' tissue [13]. The procedure for a user to use this device is quite simple, they simply attach the sensor clip onto their earlobe and press a button. Although this novel approach is the most successful non-

invasive device, it does not provide continuous measurements. Nonetheless, more frequent measurements are likely to be performed by a user considering there is no need for bloodletting.

2.5: Reasoning for this Thesis

Since a truly NICGM has not yet managed to find any foothold in the commercial market, the quest continues. Several researchers are still investigating different approaches and trying to develop a device that can compete with current GON devices. A large portion of the community seems to be investigating the approaches classified as impedance spectroscopy or optical detection. Some approaches have even combined the two methods, a concept similar to the GlucoTrack, and are pursuing a combination methodology approach [13].

Caduff's research team, mentioned previously as developers of the PENDRA, have combined different impedance spectroscopy sensors into one device. This new device, dubbed the "Multisensor", was published in 2015 and used multiple fringe field sensors combined with interdigital and optical sensors. This allows the Multisensor to measure other external perturbations to reduce noise [14].

Optical approaches have also garnered some attention as they remove the concerns associated with a delayed estimation that comes from measuring ISF's parameters. Two methods of note are near-infrared (NIR) absorption and Raman spectroscopy. NIR spectroscopy suffers from shallow penetration and can only be used on parts of the body such as the finger and earlobes [15]. Raman scattering has been shown promising results but minimizing such a device into a wearable CGM device comes with its own challenges.

As stated previously, a true NICGM has not achieved commercial success. Previous attempts have had fatal flaws (PENDRA) or have not been truly non-invasive (GON). Therefore,

the need for an utterly NICGM device cannot be understated. This technology would be life-changing for many people worldwide by giving them more insight into how their own body works. If symptoms of their condition were to present themselves, a user could instantly know their BGL without the need for bloodletting. Furthermore, having continuous data throughout the day could lead to better administration of insulin therapy. Such a system could also decrease costs associated with devices that have one-use sensors

Chapter 3: Methodology

This section discusses the methodology employed during this study. This study limited its scope to two datasets, the Ohio and UofM datasets collected by research teams from the Ohio University and the University of Memphis, respectively. Several features have been identified and are specificized in the coming sections. All features used are non-invasive, and those collected in the UofM dataset used off-the-shelf sensors for ease of data collection and to ensure the quality of measurements. Finally, the procedure by which the datasets were processed, compared, and used in combination with machine learning modeling techniques are discussed in detail.

3.1: Aim 1.1 – Obtain and analyze OhioT1DM

To gain a more significant intuition of how to achieve the goals outlined previously, a dataset is needed. A suitable dataset was found named OhioT1DM, which was collected by researchers at Ohio University in 2018 and 2020. The dataset contains eight weeks of data for each patient, with six patients per year (e.g., a total of twelve participants within the two years). In this dataset, all patients have type 1 diabetes. Several features were collected using either 530G or 630G model insulin pumps or the ancillary device, the Medtronic Enlite CGM. Also, the dataset includes features from the Empatica Embrace (2018) or Basis Peak (2020). By changing the wristband worn device, galvanic skin response (GSR) was no longer included in the datasets from the year 2020. Therefore, only the first 6 participant datasets from 2018 are reviewed from in this study. A comprehensive list of all features can be found in [4].

Due to the scope and purpose of this thesis, only features that could be collected via a wristband-like device (i.e., non-invasive features) are considered. This includes the following features: GSR, skin temperature, air temperature, and heart rate (HR).

These datasets are stored as XML files, which have been turned into '.csv' file. Some preliminary preprocessing was performed on the data to remove apparent outliers (those that span beyond the reasonable domain for each feature), and to match sample values by time signatures. Since the values were continuous and with different sampling rates, a down-sampling was performed on those features with an excess of samples. This was done by rounding their time signatures to the nearest minute and averaging all values with matching time signatures. Then, the values of individual features were matched based on the intersection of their times. This results in approximately twelve thousand samples per subject.

For each subject, his/her dataset was fed through a custom-made machine learning pipeline written in the Python programming language, which can be found in the Appendix A. This pipeline enables the testing of several different ML models and scaling methods automatically. To verify the results of this pipeline, we also show the results from the MATLAB regression learner app.

3.2: Aim 1.2 – Reduced range of target values in OhioT1DM

To answer the question if currently available non-invasive features are sufficient to predict blood glucose levels within the healthy range, we created a new dataset truncating the target variable (i.e., glucose) range from the Ohio dataset. This reduced dataset is then passed through the Python code and MATLABs regression app to analyze the performance of the current non-invasive features to predict normal blood glucose levels.

3.3: Aim 2 – UofM Dataset

As we will show in the results' section, there is a need for a dataset containing more features to predict blood glucose within abnormal and normal ranges. For this reason, this a custom dataset was created using an array of sensors. These sensors were chosen based on the literature review. The sensors chosen are as follows: g-Skin heat flux sensor, Empatica E4 wristband, Delfin MoistureMeterD, Viatom Checkme O2, Omron 3 Series upper arm blood pressure monitor, and finally an Adafruit DHT11. The features that these sensors provide are shown in the following diagram.

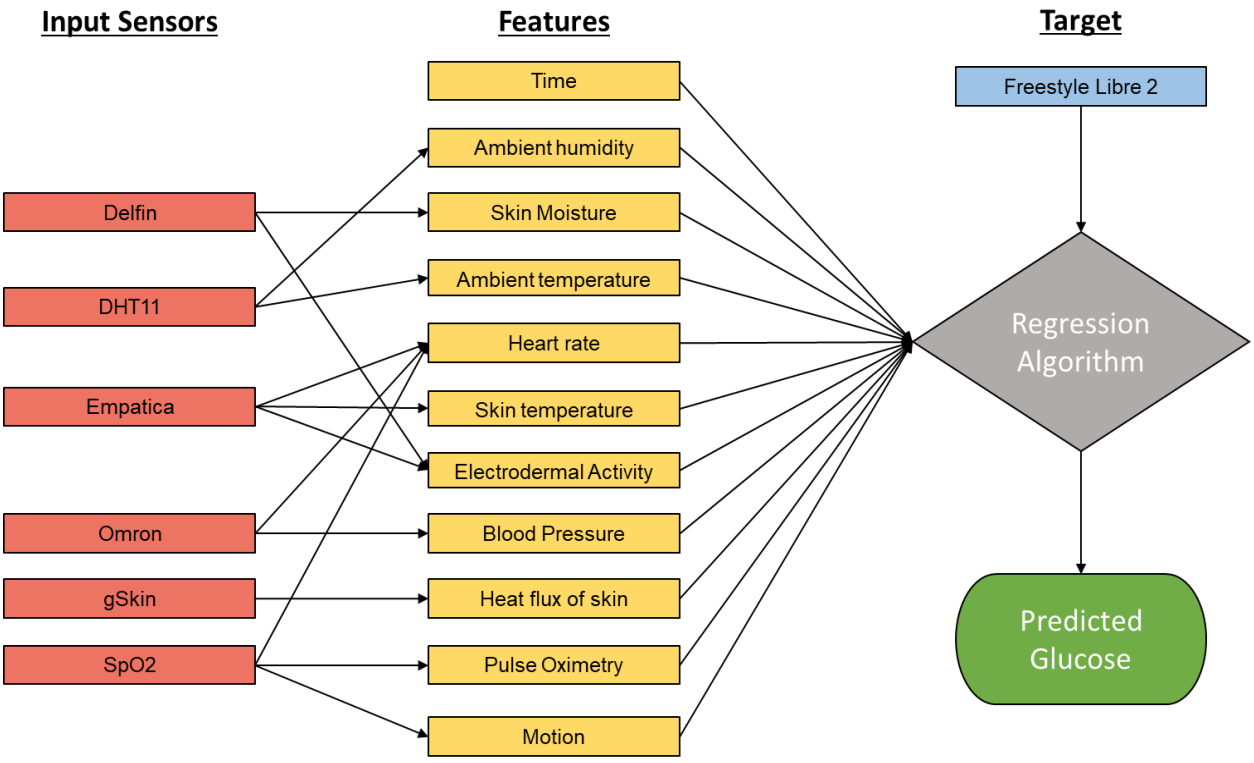


Figure 1: Sensors chosen for UofM data and the features they provide.

These features were chosen due to an expectation they will have a moderately high correlation with blood sugar based on reported studies [12,13]. Most of the features in this list were heavily inspired by the MHC technique discussed in Section 2.3 (i.e. heat flux, pulse

oximetry, skin temperature). In contrast to MHC, all the sensors that provide these features using the proposed approach could be implemented in a smartwatch-like wearable device. This was not the case in MHC because of its reliance on the multiwavelength spectroscopy technique, which for the necessary wavelengths would be far to bulky. The remaining sensors were chosen as they are equivalents are already prevalent in smartwatch devices.

As shown in Figure 1, there is some redundancy in selecting some of the sensors to ensure accurate measurements. The sensors listed have also been chosen specifically due to their compatibility with this application. As previously stated, the intended use of this device would include a form factor that is compatible with wrist-worn bands. These have become increasingly popular in recent years due to comfort and convenience. Moreover, some of these sensors have already been implemented in other devices. For example, the Empatica E4 band implements sensors for EDA, HR, and skin temperature while maintaining the form factor of a smartwatch wristband device.

One of the most significant sources of inspiration for this project was the metabolic heat confirmation technique previously discussed in Chapter 2. However, this approach differs in that it does not rely on optical spectroscopy. As a matter of fact, the only optical sensors in this proposed method would only be photoplethysmography sensors, already commonly found in nearly all smartwatch-like wearable devices. This sensor provides heart rate and SpO2 information. The heat flux sensor and skin moisture sensors were also chosen as they would capture information about the energy released out from the body into the environment in the form of radiation or evaporation. This also required ambient information, which is why the ambient temperature and humidity sensors are needed.

Furthermore, the Freestyle Libre 2 has been selected as the target variable sensor for this dataset. It will provide glucose values at a minimum sampling frequency of once every 15 minutes. Glucose measurements are then stored in the ancillary data logger or a compatible phone application. Manual measurements are also allowed for up to a maximum sampling frequency of once per minute. We expect that some of the measurements provided by the selected sensors and the blood glucose measured by the Freestyle Libre 2 would be highly correlated to give the necessary predictive power in the ML models.

Figure 2: Sensors worn for the first 2 trials of the UofM dataset collection procedure

Two procedures for the data collection process were defined. The first of the two procedures included all the sensors previously mentioned. An image of all the sensors used in this first iteration of the procedure is shown in Figure 2. Data was collected for a period of 7 hours, sampling the target variable every 5 minutes. This led to approximately 84 samples for

two test subjects. Subjects were asked to fast overnight. Breakfast was administered in the first 90 minutes of the procedure. Then the subjects were asked to exercise for 60 minutes at low intensity. Low intensity was defined as maintaining a heart rate at 160% their resting heart rate for the hour's duration. Then, the subject increased the intensity level to 175% of their resting heart rate (a high-intensity exercise) for another 30 minutes. The aim of this exercise period was to cause the subject's blood sugar to drop dramatically after the breakfast. After the exercise, lunch was administered, and subjects were asked to sit idly for 90 minutes. To test the sensor's accuracy under different conditions, two thermal challenges were simulated. In the first 30 minutes of the thermal challenge, the subjects were placed in hot temperatures and under direct sunlight for 30 minutes. The temperature for this thermal challenge was kept above 26.6 degrees Celsius (79 F). During the second 30-minute portion of the thermal challenge, the subject sat in a room where the temperature did not exceed 21 degrees Celsius (70 F). This thermal challenge aimed to see if the blood glucose levels are affected by sudden changes in ambient temperature, or if the ambient temperature had any role. Finally, the test subjects rested once again for 90 minutes in idle sitting positions.

Figure 3: Sensors worn for the third trial of the UofM dataset collection procedure

Due to time constraints and the small sample size resulting from the datasets collected in procedure one, a second procedure was developed. An image showing how the sensors worn during this procedure is shown in Figure 3. This procedure aimed to be more lenient so that data could be collected automatically and passively for five days. However, this necessitated that certain sensors were not used at all during procedure 2. In particular, we excluded the Delfin MoistureMeterD, and the Omron Blood pressure cuff since these sensors require manual measurements. Furthermore, since the data was no longer taken manually every 5 minutes, the target data sampling frequency was reduced to once every 15 minutes. Data for procedure 2 was collected for 10 hours per session over the period of 5-day sessions and 2-night sessions. This resulted in 70 hours of data with four samples per hour, summing 280 samples.

All these datasets were preprocessed by the methods as follows. Firstly, since each sensor presented a different sampling rates, the data points for each feature needed to be down-sampled to match the sampling rate of the target variable. This was done in a process similar to the one described for Aim 1. In essence, each of the feature had their sample's time signatures rounded to the nearest minute. Then, all the samples with equal time signatures were averaged. Finally, only those values with the same time signature down to the minute with respect to the target variable were kept.

After the dataset had been matched by time, outliers were removed. The most obvious outliers are those that fall outside the domain that is expected for the feature in question. For example, some of the SpO2 samples had values exceeding 100%, which is impossible. These wrong measurements of the sensors could be the result of getting a loose connection during data collection.

3.4: Model Selection and Evaluation

To adequately assess the predictive power of the features, several models were considered. The field of Data Science utilizes a broad array of techniques and different theories to derive knowledge from data. Although this is an expansive field with much to learn from and apply different practices, in this thesis, we have only explored the standard practices. This section describes the practices employed to examine model performance on the formerly mentioned datasets.

3.4.1: Metrics

Quantifying one's results is an important step in determining the quality of the predictions. Therefore, some reasonable metrics must be established. In this work, we have used common metrics used in the literature and associated with the application of wearable sensors.

The first to be discussed is Root Mean Squared Error (RMSE), whose equation is given below, where $\hat{y}_i$ is the predicted value and y_i is the true value, and *n* is the number of samples:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \quad (1)$$

As one realizes, Equation 1 is similar to the equation for standard deviation, which is why it is also known as the standard deviation of the residuals. The benefit of using RMSE is that it maintains the units of the original target variable. This allows the results to be easily understood in context. It is also a favorite among metrics used by research into other BGL monitors. Another benefit is that this metric is easily used in tools developed specifically to evaluate the accuracy of blood glucose meters (e.g., Clarke Error Grid Analysis). Additionally, this metric is commonly used for quantifying the performance of regression algorithms and is usually implemented in the programming toolboxes.

The second metric used is coefficient of determination (R^2) which is defined in Equation (2) where SS_{res} is the sum of squares of the residuals (a.k.a. errors), and SS_{tot} is the total sum of the squares (from the prediction to the means):

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (2)$$

In this equation, $\hat{y}_i$ is the predicted value at the observation, $\bar{y}$ is the mean of the dependent variable and y_i is the observed value (true value). Although slightly less intuitive than the previous metric, generally, this number falls between 0 and 1, however not exclusively. An R^2 of zero would indicate that the predictions are no better than if the regression algorithm simply guessed the mean of the target for every prediction. R^2 values ranging from 0 to one indicate that there is some predictive power in this model. If the coefficient of determination is less than zero, this means the model is arbitrarily performing worse than if it were to guess the mean of the target variable. Therefore, the R^2 value is a reasonably good way to understand how well a model is performing. It should be noted that there are several definitions of R^2 and how it is specifically implemented. However, Equation 2 is the most general definition. This definition is the one that is used for the remainder of this thesis.

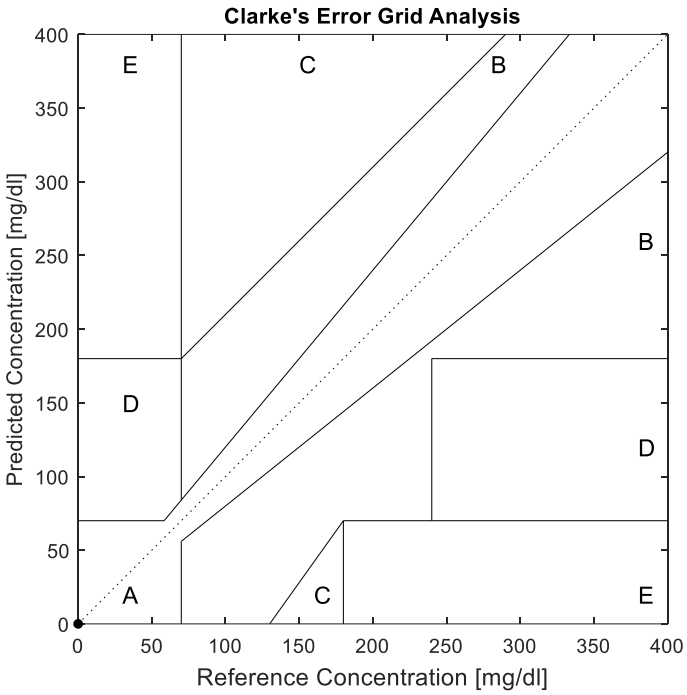


Figure 4: Blank Example of the Clarke Error Grid Analysis Plot

One final metric is the Clarke Error Grid Analysis (Figure 4) which was used to evaluate how acceptable the model's performance to predict blood glucose. This metric is a scatterplot of the reference/estimated values plotted against the true values to determine the accuracy of the prediction. In the Clarke Error Grid, the scatterplot is divided into five regions: A, B, C, D, E. The quality of the predictions depends on which region the predictions fall in [16]. Region A are values that fall within 20% of the reference values. In region B, the predictions fall outside 20% of the true value, but they would not lead to inappropriate actions to correct the condition. Region C could result in predicted values that could cause the patient to make wrong actions to manage their condition. Region D could lead to a harmful action due to the lack of detection of hyper- and hypo- glycemia. Region E confuses hyper- and hypoglycemia events (i.e., an individual with a blood sugar of 250 mg/dL can get a predicted value of 50 mg/dL). This plot is considered the gold-standard metric for measuring the quality of glucosemeter predictions.

3.4.2: Model Selection

Various machine learning algorithms were tested on this dataset. To achieve this efficiently, a pipeline script was created in the Python programming language. The core toolboxes used were the popular machine learning toolbox SciKit-Learn (a.k.a. sklearn) [17], as well as some other auxiliary toolboxes such as: Pandas, NumPy, and Matplotlib. This pipeline script will be included as Appendix A. The pipeline also tests a few different scaling methods to see if it affects the performance significantly. All models used fall under the broader category of supervised learning algorithms in the family of Artificial Intelligence. The machine learning models are listed below with descriptions informed by the SciKit-Learn website [17]:

1. Linear Regression (LR): The implementation of linear regression used is the Ordinary Least Squares (OLS) method. This model would be considered univariate multiple regression as it has a single target variable and multiple predictor variables. This is the simplest of all the models and would only be expected to perform well if there is a high correlation between some (or all) of the predictors and the target variable. OLS model attempts to fit a linear model by iteratively adjusting the coefficient and intercept of the equation by minimizing the residual sum of squares (i.e., the cost/objective function).
2. Support Vector Regression (SVR): This model is a generalization of Support Vector Machines used for classification purposes. The model produced only depends on a subset of the training data as defined by the cost function. This cost function ignores values whose prediction is close to the target. Model hyperparameters that are adjusted manually are the C and epsilon values. This model also supports several different kernel types used in the algorithm, which can fit both linear and non-linear relationships. The

implementation from the toolbox used does not perform well on dataset larger than a couple tens of thousands of samples larger. However, this is not an issue in this case.

3. K Nearest Neighbors Regression (KNN): The K Neighbors regression model is another case where a classification algorithm is generalized for regression applications. Rather than just taking discrete variables as class labels for input, this method can fit continuous variables. There are several hyperparameters that can also be adjusted by the user, including but not limited to: K number of neighbors, the weight function used, and the algorithm used for calculating the nearest neighbors. The predictions are based on interpolation performed by the model after fitting a training set.
4. Decision Trees Regression (DTR, CART): Decision trees, otherwise known as Classification and Regression Trees, can be visualized as a tree of if-else statements whose branches are decisions formed by previous experience. This is a powerful regression algorithm that can fit an arbitrary dataset nearly perfectly. Nonetheless, they are heavily prone to overfitting; thereby its results should be viewed as optimistic unless thoroughly validated. Several hyperparameters can be manually adjusted in an attempt to improve performance, including the criterion by which it measures the quality of the split, how the branches are split at each node, the maximum depth of the tree, and the minimum number of samples needed to justify the creation of a new node. Decision trees have a tendency to perform poorly on new data should the original training set not be representative of the unseen new data.
5. Bagging Trees Regressor (BTR): Bagging trees is an ensemble method created from a collection of decision trees. This method takes a random subset of samples from each feature to train the statistical black box estimator, thereby reducing the variance inherent

to simple decision trees. This technique is what makes this model a sparse one. By injecting some randomness into the process, BTR becomes more robust to high variance but can still fall victim to it, nonetheless. BTR results in much larger and more complex models that cannot be easily understood and are quite indeterministic because of the randomness. The hyperparameters adjusted in BTR are the same as the ones in DTR except the number of trees in the forest.

6. Random Forest Regression (RFR): Random Forests are a sparse ensemble model created from a collection of decision trees, much like bagging trees. Much like with BTR, this method employs randomness to combat overfitting. In contrast to bagging trees, RFR takes a random number of samples from a random subset of features instead of using all features to create the random subset of samples. This makes it even more robust to high variance and overfitting since not all features are used to train any tree.
7. Gaussian Process Regression (GPR): Much like KNN, this approach interpolates observations to generalize into a regression algorithm. The benefit of GPR models is that they also return empirical confidence intervals for each prediction, providing information about the predictions. Using this information can lead to models that can be further refit in some regions of the dataset. GPR is also very versatile since it can implement several kernels. However, they are not considered sparse models like the ensemble trees previously mentioned (BTR, FRF). They also do not perform very well with a high number of predictor dimensions, suffering from the *curse of dimensionality*.
8. Multi-layer Perceptron Regression (MLP or NNR): This is a specific case of the Neural Network family of models that implements regression analysis via the multi-layer perceptron method. Neural Networks are known to fit any arbitrary decision boundary,

not being limited to linear relationship. The term MLP is loosely used to classify any Artificial Neural Network that is feedforward and uses a non-linear activation function at each neuron but the input node. The hyperparameters for this model allow a user to adjust the lidden layer sizes, the activation function for each hidden layer, and the optimization method used to adjust the weights of the nodes.

3.4.3: Validation/Testing Methods:

As mentioned, many of these algorithms will tend to overfit and therefore should be tested to determine how well they would perform on unseen data. That word *unseen* is critical, making sure the model has not been trained on the testing data is essential to avoid overly optimistic results. Before discussing the validation methods, some intricacies in regards to the data splitting should be noted. One should ensure that no scaling is performed after the train/test split to avoid data leakage, scaling the test data using the mean and standard deviation of the training data. Once again, this is done to safeguard against overly optimistic results caused by data leakage.

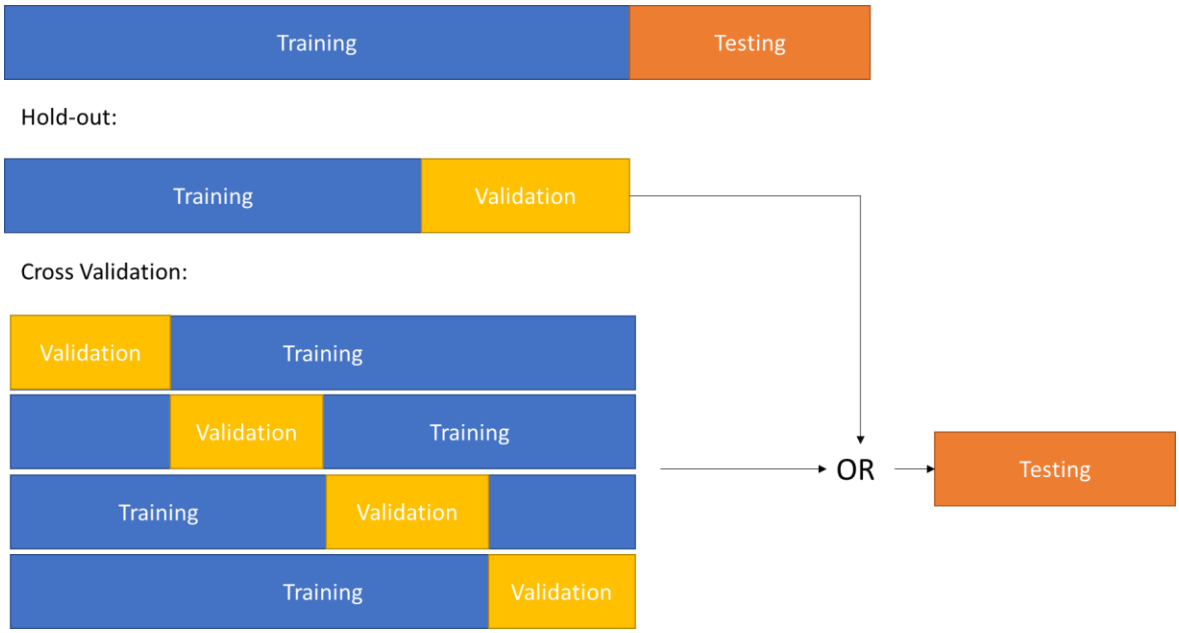


Figure 5: Hold-out versus cross-validation

Two methods for validation are presented in this Master thesis: hold-out and cross-validation. Both are represented in Figure 5. The holdout method of testing a dataset is the simplest since it only requires that the training and testing sets be separated before training. This is the preferred method when datasets are very large or computational power is limited. However, the measured performance is limited to the training set that it is trained on. In cases where the dataset is very large, and both the training and testing sets are a good representation of the total population, the hold-out method does not present any issue. Common issues arise when the dataset is relatively small, in which case how the data was split matters a lot and can affect performance on unseen data.

K-fold cross-validation solves this problem by further iteratively splitting the dataset, K number of times, into training and validation sets. Doing this essentially generalizes the hold-out procedure and repeats it K times on different combinations of training and validation sets. Once all combinations are tested, the mean and standard deviation of the scores can be computed. This

can give a user a good idea of how the model would perform on unseen data, especially when the dataset in question is relatively small. After this step, the best model can be selected based on the scores and fitted to the entire training set. Lastly, this trained model can be quantified against the unseen test data.

It is easy to see which of the two methods is superior. Aside from being more computationally expensive, cross-validation offers a lot more benefits than drawbacks. The extra information about how the models being tested perform on the dataset in question is useful in selecting a model. Those new to Machine Learning naively (and understandably) fall into the trap of assuming their models are performing great after using the hold-out method. However, cross-validation is standard practice. The authors of the textbook "Data-Driven Science and Engineering" elegantly claim "*if you don't cross-validate, you is dumb*."

3.4.4: Machine Learning Pipeline versus MATALB

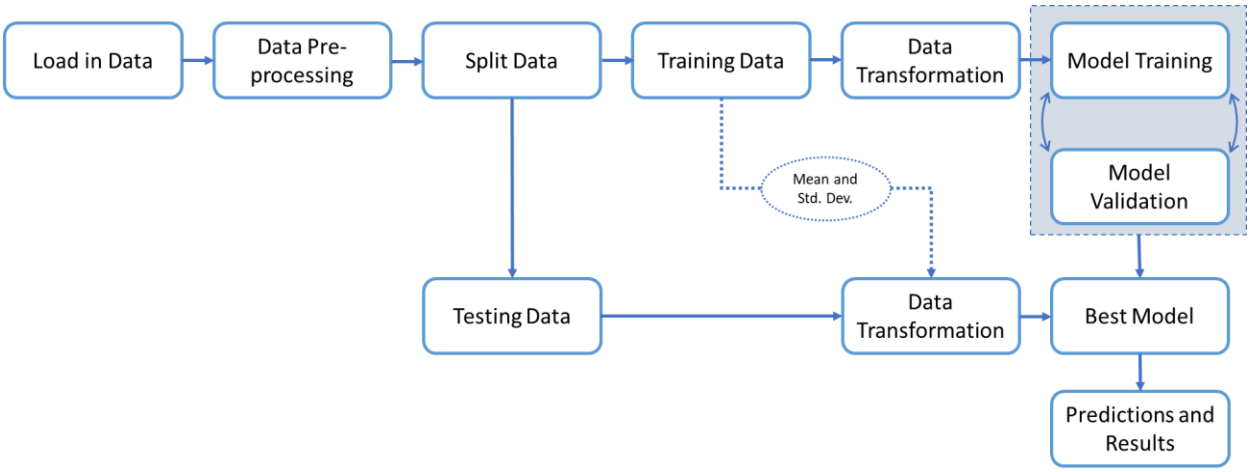


Figure 6: Machine Learning Pipeline diagram

To test all these different models efficiently and without relying on manually running each model separately, a pipeline was created. This pipeline, outlined in Figure 6, is flexible enough to allow for more models and scaling methods to be included easily. The script allows

the user to load in any dataset easily and then run all the regression algorithms and appropriate scaling methods at once. This script will compare the scores of all the models and output the best model. It also tests a couple of different scaling methods (e.g., standardization, normalization) on the dataset to determine if this improves the scores. The full script can be found in appendix A.

To verify the results of this pipeline, MATLAB's Regression Learner App will also be used to test and compare several models. This program can be found in the "APPS" tab, shown in Figure 7, if the appropriate toolboxes are installed.

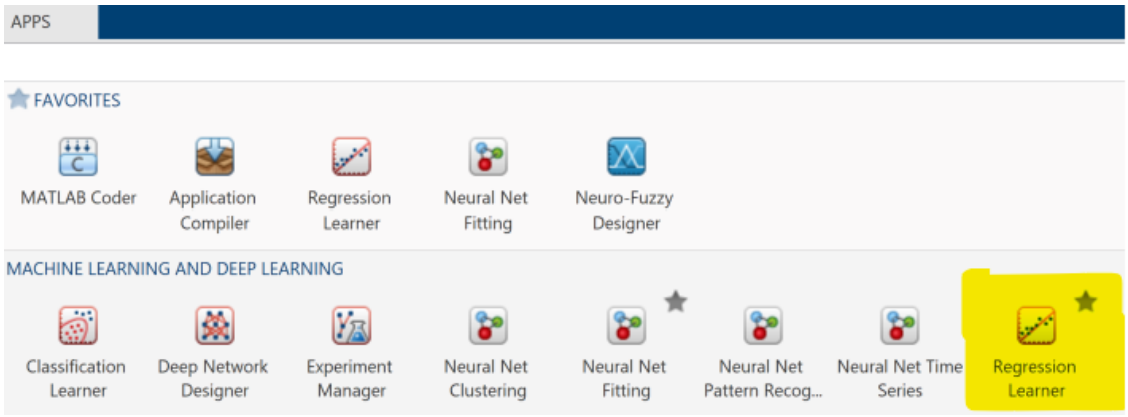


Figure 7: A visual example of where to find the Regression Learner App in MATLAB

Once opened, a new session can be started using any dataset that has been loaded into the workspace by the user, shown in figure 8.

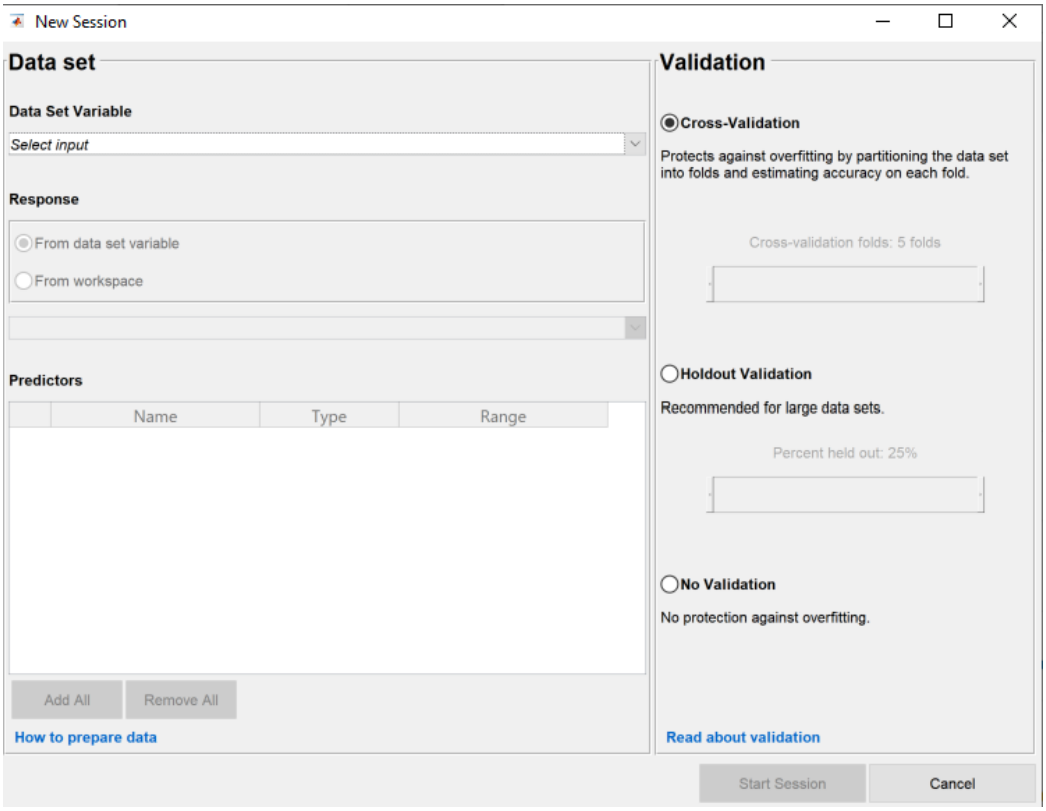


Figure 8: Starting a new session in MATLAB

To be fair in comparing the two methods, the number of folds selected for cross-validation should be equal. Therefore, for the rest of this thesis, we have asummed K = 10 folds.

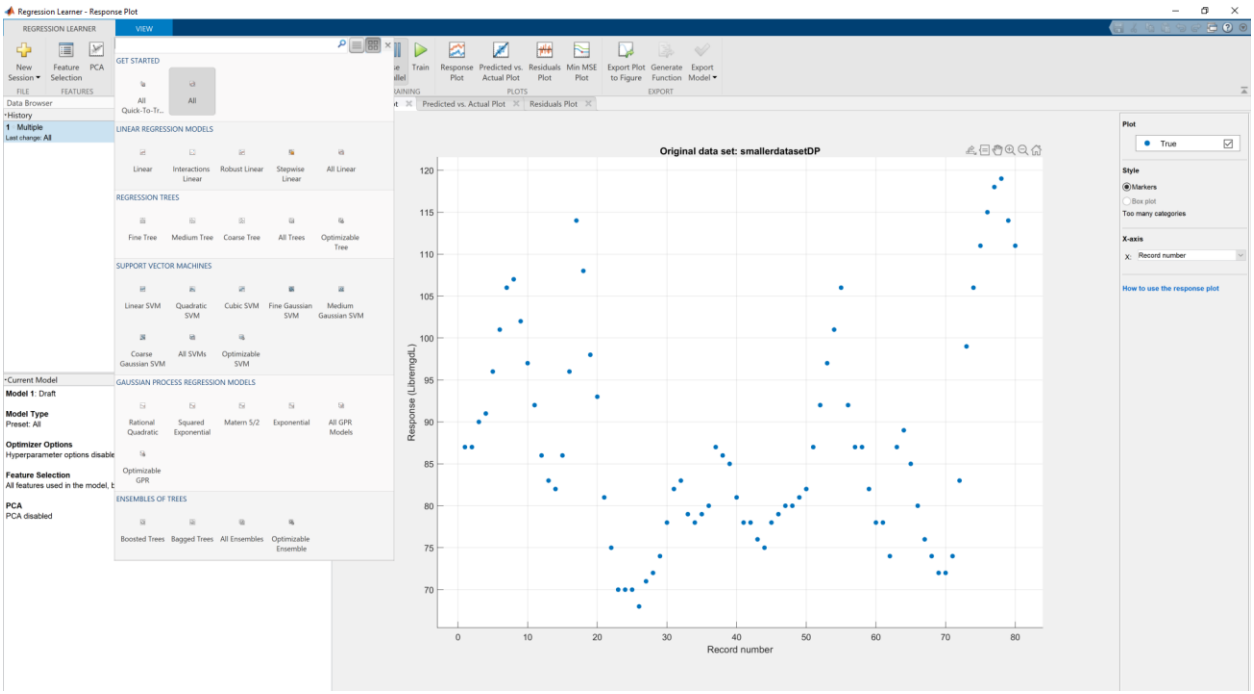


Figure 9: Selecting models to be tested

Once the dataset has been loaded in and a new session started, the user can choose which models to evaluate, shown in Figure 9. For the sake of this demo, the ‘All’ option was selected. In Figure 10, an example of the best model selected by this app is shown.

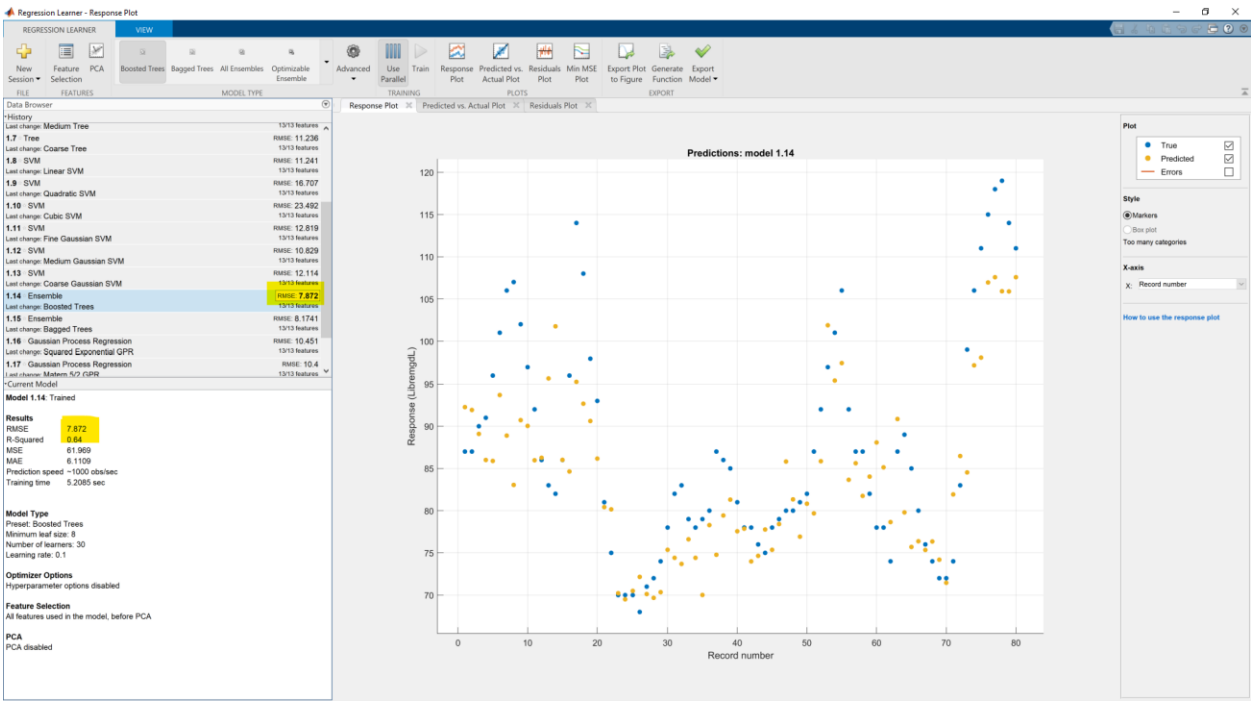


Figure 10: Best performing model for this dataset

As can be shown in Figure 10, the predictions are plotted as well as the metrics of the model which are highlighted. The plot shown can also be changed from the response plot as seen above to the Predicted vs Actual, Residuals, and Min MSE. These will be useful later.

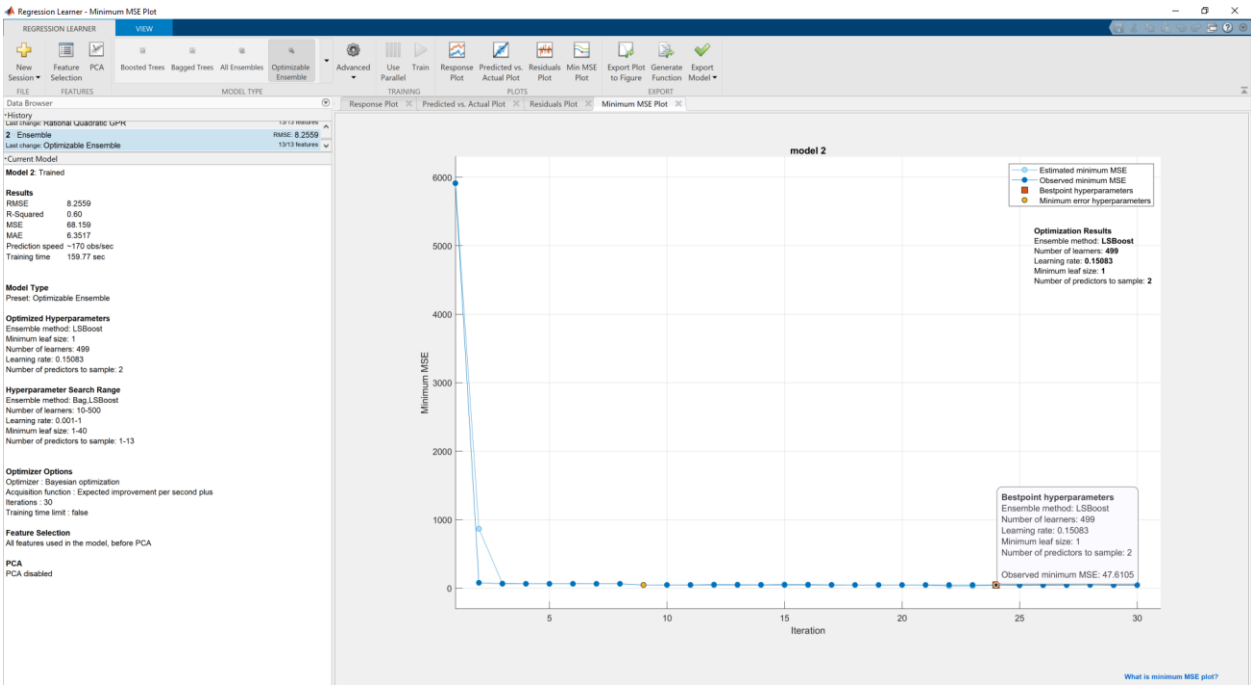


Figure 11: Min MSE plot hyperparameter tuning

Figure 11 shows that the best hyperparameters selected by minimizing the MSE after 24 iterations lead to an increase of the RMSE.

Chapter 4: Results

In the following chapter, the different datasets are used to test out several models. Individual subjects were trained and tested by using personally trained models that were then tested on unseen data from the same subject. Global models have not been explored. The best model is presented for each subject along with discussion of the results. The meaning and relevance of these results is then be examined in detail and compared to one another.

4.1: Aim 1.1 – Ohio Data Results

As previously mentioned, the following results were obtained via the use of either Python or MATLAB. In both cases, the methodology detailed in Chapter 3 was strictly followed for developing the code or using the associated applications. For the sake of brevity, only 2 subjects from OhioT1DM dataset are discussed in this section. The two patients are Subject 559 and 563. These datasets were chosen because they present the highest and lowest variance, respectively

The only features that are kept from the original datasets are: ambient temperature, glucose, heart rate, skin temperature. Tables 1 and 2 display the sample count, mean, standard deviation (std), minimum and maximum values, 25% quartile, 50% quartile, and 75% quartile for each feature in both subjects. These tables demonstrate the characteristics of the dataset, including the distribution of the target variable (glucose) and its range.

Table 1: Description of data, Subject 559.

Value	Glucose	Amb Temp	GSR	Heart Rate	Skin temp	log(GSR)
Mean	167.23	84.28	0.40	73.89	87.66	-3.31
Std. Dev.	70.36	4.38	2.04	15.94	3.44	1.37
Min.	40.00	63.86	0.00	46.00	72.32	-4.17
25%	110.00	81.32	0.00	62.00	85.10	-4.11
50%	158.00	83.66	0.00	69.00	87.44	-3.99
75%	210.00	87.62	0.00	83.00	90.50	-3.25
Max	400.00	96.98	23.02	189.00	95.90	1.36
Count	12,432	12,432	12,432	12,432	12,432	12,432

Table 2: Description of data, Subject 563

Value	Glucose	Amb Temp	GSR	Heart Rate	Skin temp	log(GSR)
Mean	150.53	84.02	0.41	96.46	87.94	-2.75
Std. Dev.	50.50	3.60	1.68	13.88	2.71	1.51
Min.	40.00	57.02	0.00	66.00	66.20	-4.36
25%	112.00	82.40	0.00	88.00	86.90	-4.03
50%	145.00	84.20	0.00	97.00	88.16	-3.32
75%	184.00	86.00	0.02	105.00	89.60	-1.76
Max	400.00	98.78	21.98	167.00	97.52	1.34
Count	13,008	13,008	13,008	13,008	13,008	13,008

The methods used to clean the data are outlined in Chapter 3. The only significant alterations to the data are the transformation of the GSR feature with a logarithmic operation to remove some of the skewness of the data. This is shown in the tables above by the feature column ‘log(GSR)’.

Tables 3 and 4 show the correlation matrices for subject 559 and 563. As a quick visual aid for identifying the magnitude of the coefficients, we have colored each cell in the tables based on the estimated correlation coefficient. From these matrices, we can deduce a lot of information about the dataset that has not been described previously. Most notably, the relationship of the features to the target data. It seems that the best correlation to the glucose is

that given by the heart rate in subject 559's data, which has a coefficient of 0.2. Nonetheless, due to the reduced correlation between the target glucose values with respect to the input features, it is expected that these features are not very good predictors of BGLs if strictly linear models are used. For this reason, we have also investigated several models within the pipeline that can hopefully account for and identify non-linear relationship between the BGLs and the features that may not be obvious.

Table 3: Correlation Matrix for Subject's 559 dataset

Features	*Glucose*	*Amb. Temp.*	*GSR*	*Heart Rate*	*Skin Temp.*	*log(GSR)*
Glucose	1.00					
Amb. Temp.	-0.16	1.00				
GSR	-0.02	-0.15	1.00			
Heart Rate	0.20	-0.49	0.26	1.00		
Skin Temp.	-0.15	0.97	-0.14	-0.49	1.00	
log(GSR)	-0.04	0.05	0.61	0.17	0.07	1.00

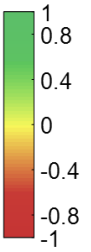
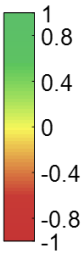


Table 4: Correlation Matrix for Subject's 563 dataset

Feature	*Glucose*	*Amb. Temp.*	*Heart Rate*	*Skin Temp.*	*log(GSR)*
Glucose	1.00				
Amb. Temp.	0.07	1.00			
Heart Rate	0.04	-0.09	1.00		
Skin Temp.	0.09	0.95	-0.07	1.00	
log(GSR)	-0.04	-0.01	0.25	0.01	1.00



Once the data have been processed, it was passed through the machine learning pipeline previously described. The results of the training procedure are shown in Figure 12. These are the results of the best model after being trained by using 10-fold cross-validation. In this case, the best model chosen was an Ensemble method called Bagged Trees. These results indicated that the features from the Ohio data do not have sufficient predictive power to be used for blood glucose level (BGL) predictions. The mean and standard deviation lines are also shown on the plot to highlight how the model predictions are doing a poor job. Rather than accurately predict

the output, the model has the ‘safest’ predictions that reduce the target metric (e.g., RMSE or R2). This results in the model clustering around the mean of the target data, rarely ever making predictions outside of one standard deviation from this average. The same behavior is observed in the results for the dataset of Subject 563 (figure not shown in this Master dissertation).

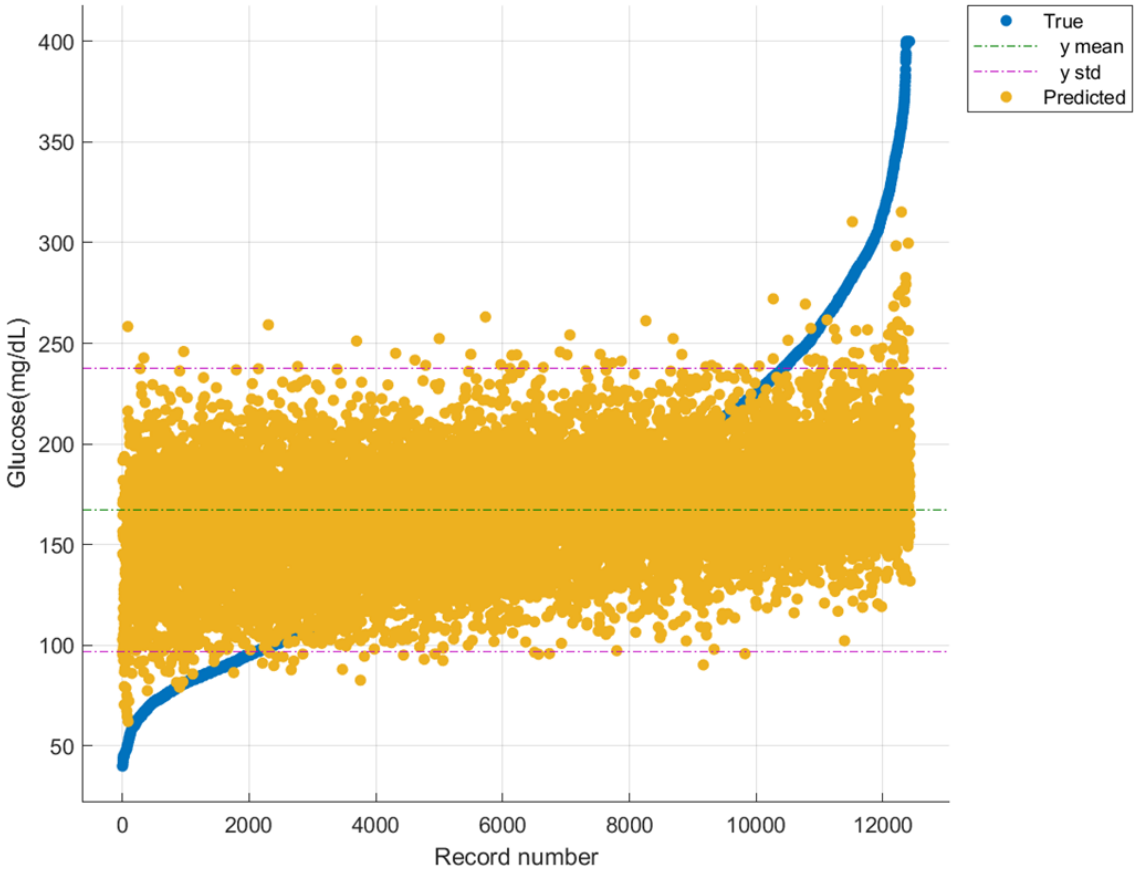


Figure 12: Best Model for Subject 559.

Another useful way of displaying the results is the Clarke Error Grid Analysis, shown in Figure 13. This plot, which is essentially a scatter plot of predicted vs actual values, informs us about the behavior of the predictions to the ideal scenario. Ideally, all the points on this plot should fall on the dotted diagonal line. When the points are close to this line, the model has achieved a very high R^2 value. The further the points are from this line, the worse the

predictions. In addition, the Clarke analysis has the added benefit of dividing the scatter plot into five regions, which rate how good the predictions are for determining treatment; see Section 3.4.1 in Chapter 3. The goal is that the predictions should stay within region A (i.e., predicted values may differ up to 20% of the reference values). Based on this Clarke grid, the predictions are clustering in all the regions, indicating that the model is poorly predicting accurate blood glucose values. We think the model is underfitting the data due to the lack of a high correlation between the predicted data and the features (i.e., the lack of useful features).

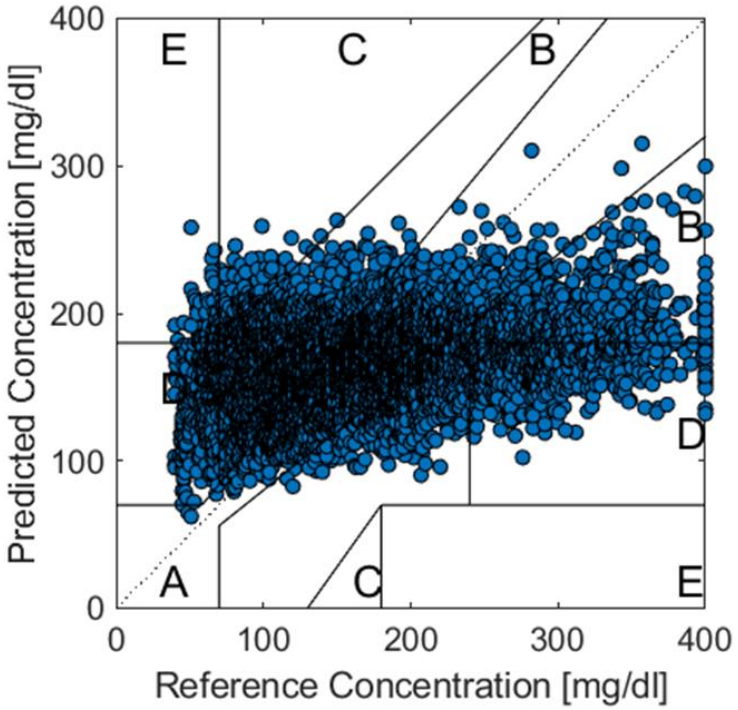


Figure 13: Clarke Error Grid for the results of Subject 559.

Finally, the results for both subjects 559 and 563 are displayed above in Table 5. Keep in mind that although a lower RMSE is shown for 563's dataset, this is only a result of the reduced variance in his target data. For both subjects, the best model has an abysmally low value of R^2, which shows a terrible fit to the target data. The RMSE is also very similar to the standard deviation, meaning the results are no better than if random input values are given as predictors.

Furthermore, a significant portion (~6-12%) of the results fall outside the clinically acceptable regions of A and B. This would result in individuals taking action at inappropriate moments, potentially leading to fatal outcomes.

Table 5: Relevant metrics for Subjects 559 and 563

Metric	Subject 559	Subject 563
A	37.89%	49.91%
B	49.02%	43.83%
C	1.67%	0.12%
D	10.80%	6.03%
E	0.62%	0.12%
Total	12432	13008
RMSE	66.32	46.38
R^2	0.11	0.16

4.2: Aim 1.2 – Reduced Range Ohio Data Results

The final hope for the features of the Ohio dataset rests on the performance improvement if the range of the target is reduced to that of a healthy individual. Even though we have just shown that these non-invasive features cannot adequately predict BGL values in the range of 40-400 mg/dL, we are interested in investigating if one can predict blood glucose levels using non-invasive features for healthy individuals. Therefore, for this task, the datasets for subjects 559 and 563 have been reduced by deleting all data points whose target blood glucose did not fall within the healthy range (e.g., 60-200 mg/dL). After this, the same procedure was performed, we repeated the same steps as in Aim 1. Table 6 and Figure 14 shows the best results. These results show the same problem as the one discussed in Aim 1. Once again, in Figure 14, all the scatter plot's points are arranged in shape like a rectangle. This behavior can be linked to the fact that the target data is underfitted based on the lack of trustable features. Even in a healthy range of

BGL, these features simply are not sufficient to predict accurate values of BGL. Therefore, it is mandatory that a dataset with better features should be created.

Table 6: RMSE and R^2 metrics for Reduced Datasets of Subjects 559 and 563

	Subject 559	Subject 563
RMSE	37.168	32.756
R^2	0.06	0.13

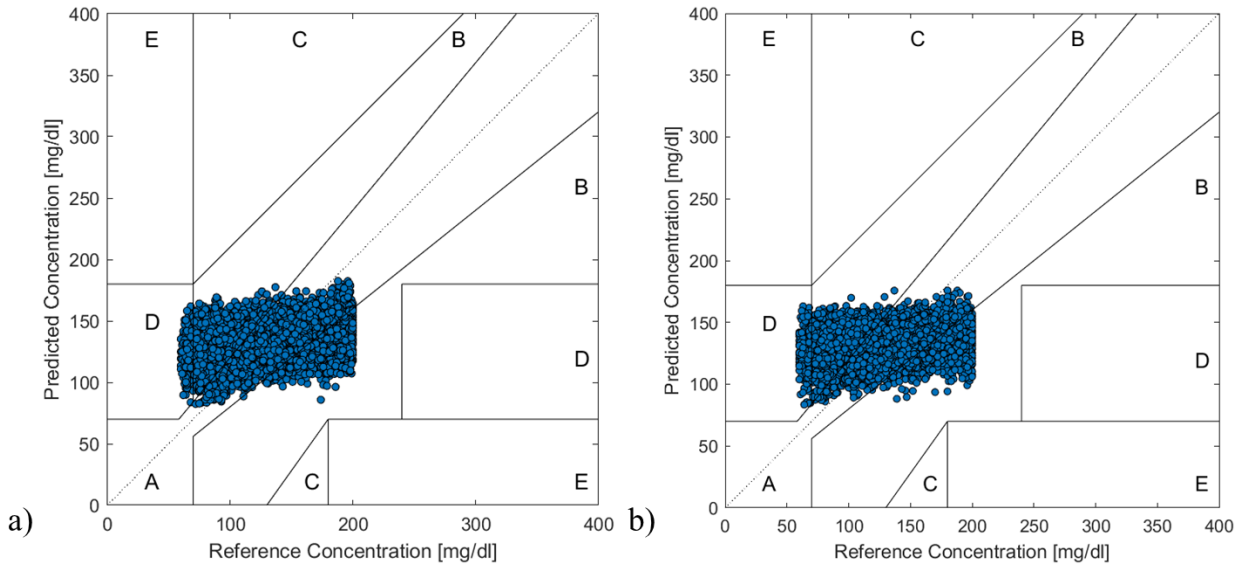


Figure 14: Clarke Error Grids for the Reduced Datasets of Subjects 559 and 563

4.3: Aim 2 – UofM Dataset Results

Now that new datasets have been created with a comprehensive feature list, these new datasets must be fully understood. As stated, for convenience, a similar naming scheme was maintained for the three new datasets. The subject's ID are: 1, 2, and 3. Subject's 1 and 2 had an identical procedure for data collection. Below is a sample table describing the characteristics of Subject 1's dataset.

Table 7: Description of Data, Subject 1

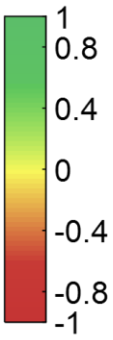
Value	Glucose	SpO2(%)	Heart rate	Motion	Amb. Humidity	Amb. Temp.	Moisture(%):Ventral	Moisture(%):Dorsal	Heat Flux (W/m)	Empatica:BVP	Empatica:IBI2	Empatica:TEMP	Empatica:EDA	Systolic	Diastolic
Mean	87.30	96.72	75.05	2.32	49.40	21.52	41.56	40.93	622.63	-1.42	0.44	28.76	9.78	125.24	81.07
Std. Dev.	12.96	1.04	14.15	1.57	9.83	5.20	9.57	9.36	375.18	10.51	0.32	4.48	18.77	11.67	5.33
Min.	68.00	92.00	56.00	0.33	0.00	0.00	0.00	19.30	-438.21	-33.30	0.00	0.00	0.00	106.00	65.00
25%	78.00	96.31	64.67	1.27	48.35	20.63	39.20	35.80	325.07	-1.38	0.21	27.95	0.18	117.00	78.00
50%	83.00	96.97	71.43	1.97	50.10	21.00	43.20	39.70	615.68	-0.20	0.45	28.49	0.44	121.00	80.00
75%	96.00	97.28	83.32	2.83	52.75	21.70	46.55	46.35	819.54	0.69	0.65	29.36	6.97	130.25	83.63
Max	119.00	98.30	112.63	8.13	65.30	34.20	55.40	56.60	1248.14	57.16	1.33	39.34	80.99	163.00	94.00
Count	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00	80.00

As can be seen in Table 7, we have collected 11 additional features. Each of the datasets were preprocessed appropriately and in accordance with the preprocessing procedure described in Chapter 3 of this thesis. Because these subjects are healthy subjects, the ranges of the target data are greatly reduced when compared to the Ohio dataset. This is an unfortunate side effect of only having non-diabetic patients as test subjects. Therefore, in this aim, we have investigated the performance of a model to predict accurate blood glucose values in healthy subjects, discussing if the addition of higher features has improved the performance of the predictors (as compared with Aim 1.2).

We have used the correlation coefficient again to identify the features that are most likely to contribute to the model's predictive power, see Table 8. The most important is the first column in this table which identifies the bivariate correlation between the target feature and all other predictors. For this column, four key features can be identified: heart rate, ventral skin moisture, heat flux and electrodermal activity (EDA). All these features have a moderate degree of negative correlation to glucose. This makes intuitive sense, and is supported by the literature, as it is expected that whatever stressors caused these features to rise will also cause glucose to drop. In the case of our study, the stressor was exercise.

Table 8: Correlation Matrix for Subject 1

Features	Glucose	SpO2(%)	Heart rate	Motion	Amb. Humidity	Amb. Temp.	Moisture(%): Ventral	Moisture(%): Dorsal	Heat Flux (W/m)	BVP	IBI	Skin Temp	EDA	Systolic	Diastolic
Glucose	1.00														
SpO2(%)	-0.02	1.00													
Heart rate	-0.32	-0.07	1.00												
Motion	-0.24	0.06	0.48	1.00											
Ambient Humidity	-0.08	0.18	0.02	-0.17	1.00										
Ambient Temp.	-0.10	0.13	-0.05	-0.18	0.86	1.00									
Moisture(%): Ventral	-0.24	0.05	0.22	-0.04	0.79	0.62	1.00								
Moisture(%): Dorsal	-0.19	-0.19	0.42	0.14	-0.10	-0.21	0.16	1.00							
Heat Flux (W/m)	-0.45	0.08	0.35	0.44	-0.10	-0.30	0.03	0.09	1.00						
Empatica: BVP	-0.04	-0.12	0.15	-0.02	-0.07	-0.05	-0.03	0.09	-0.02	1.00					
Empatica: IBI	0.05	-0.04	0.06	-0.07	-0.08	-0.23	0.06	0.27	0.06	0.09	1.00				
Empatica: TEMP	-0.05	0.23	-0.10	-0.30	0.16	0.25	0.10	0.04	-0.43	-0.01	-0.04	1.00			
Empatica: EDA	-0.45	-0.10	0.71	0.47	-0.04	-0.17	0.30	0.57	0.49	0.10	0.21	-0.05	1.00		
Systolic	-0.09	-0.02	0.71	0.46	-0.17	-0.38	0.12	0.49	0.32	0.10	0.24	-0.15	0.61	1.00	
Diastolic	-0.08	-0.09	0.39	0.14	-0.07	-0.27	0.10	0.42	0.30	-0.01	0.29	-0.12	0.48	0.47	1.00



Aside from identifying good features that contribute to the model's predictive power, this matrix can also aid in identifying potentially redundant features that could be removed. For example, EDA and skin moisture (dorsal) have a fairly high degree of correlation (>0.50), and therefore both may not be needed. Another example is EDA and the systolic and diastolic measurements. These also appear highly correlated, so a similar conclusion could be reached.

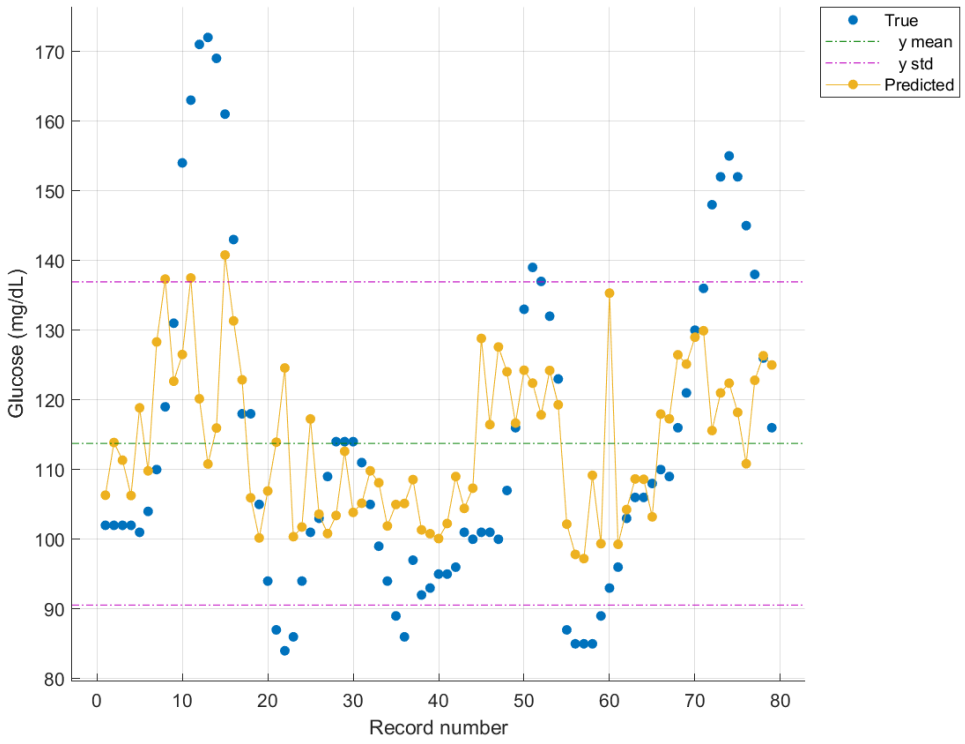


Figure 16: Cross-Validation Results, Subject 1

Next, the same cross-validation is performed to compare all different model's fit to this dataset. Figure 16 shows the best model from this experiment. These results are attributed to the predictions of the Ensemble model: Bagged Trees, as in Aim 1. It becomes immediately apparent from looking at the plot that this dataset has a much higher performance. The predicted data points do a much better job following the trend of the true values, as indicated from a higher R^2 value. However, it would not be accurate to make final conclusions based on these results (Figure 16) since they are the result of cross-validation on the entire dataset. Therefore, a step back must be taken to split the dataset into training and testing to avoid any data leakage and adequately measure the performance of this model.

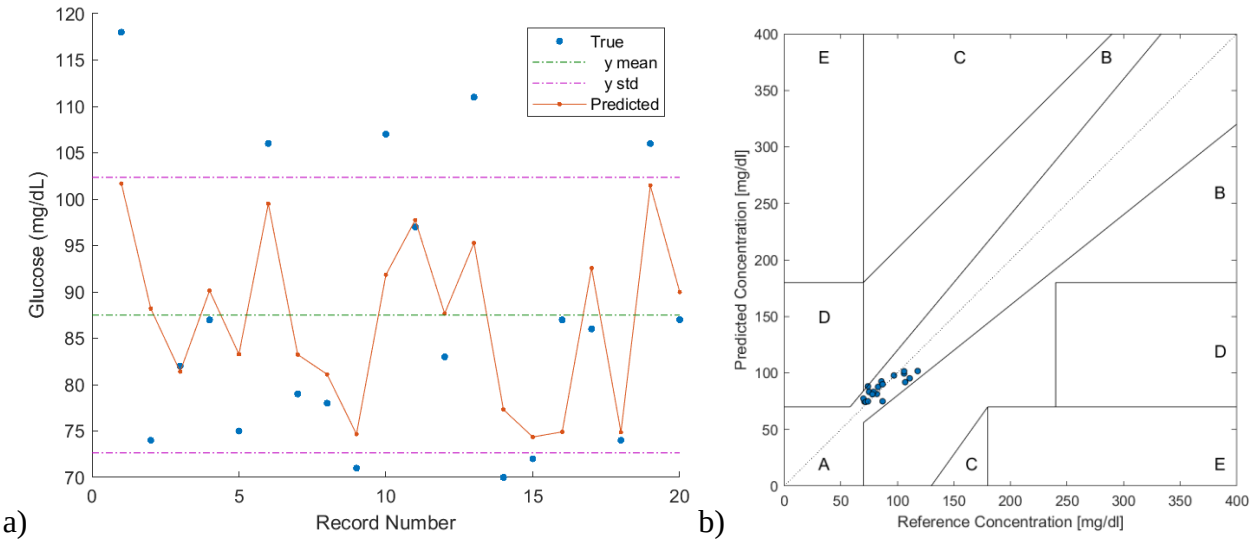


Figure 17: Results of testing set for Subject 1: (a) glucose values at different time events and (b) Clarke Error Grid

Now that the data is split into training and testing sets, the model can be retrained on the training set and tested on an utterly unseen test set. Figures 17 and 18 show the results of this experiment. While not quite as apparent as before (on account of far fewer samples), the model is doing a decent job at following the overall trends of the target variable. A quantitative evaluation of the accuracy of the predictions provided by the model is shown in Table 9. This table also shows the quantitative metrics for the other two subjects. The best-selected model for subjects 2 and 3 was a Gaussian Process Regression with a rational quadratic kernel function for subject 2 and a Matern 5/2 kernel function for subject 3. Based on these values, one can realize that the RMSE values have been reduced, mostly related to the reduction in the variance of the target variable. Also, it is expected the residuals would have been decreased in this range. From the analysis of the 'goodness of fit' (R^2) , we can conclude that the performance of the model is dramatically higher. Whereas all the predicted values (Figure 17b) for subject 1fall in the clinically acceptable range of being within 20% of the reference values (Region A), the

predictions for subjects 2 and 3 fall within Region B. In particular, 10% and 21.13% of the predicted values lie on Region B for subjects 2 and 3, respectively. Nonetheless, it is important to mention that although predicted values in Region B differ a factor higher than 20% from the reference value, they would not necessarily lead to inappropriate treatment. Finally, it is also important to highlight some predicted values (1.41%) for subject 3 are in Region D. As described in Chapter 3, subject 3's data collection was different from the rest since it was collected for several days and fewer sensors were used. This means that fewer features were used to predict the target glucose values. Furthermore, this dataset was likely prone to more noise as Subject 3 went about their day-to-day activities.

Table 9: Relevant metrics for Subjects 1, 2, and 3

Metric	Subject 1	Subject 2	Subject 3
A	100.00%	90.00%	77.46%
B	0.00%	10.00%	21.13%
C	0.00%	0.00%	0.00%
D	0.00%	0.00%	1.41%
E	0.00%	0.00%	0.00%
Total	20	20	71
RMSE	8.39	13.72	15.84
R^2	0.71	0.72	0.40

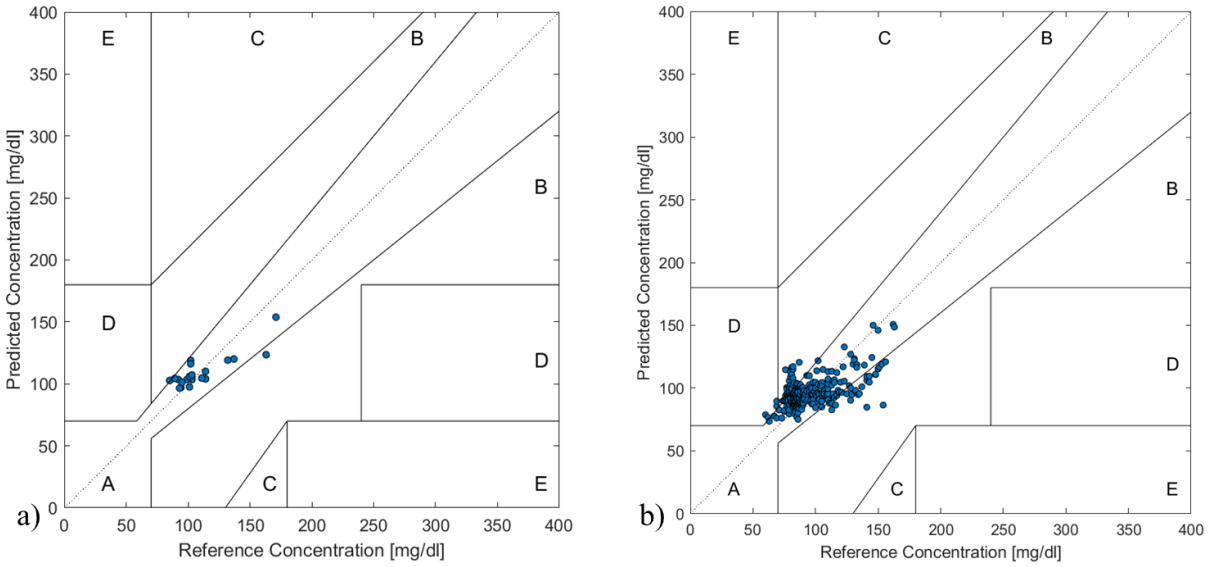


Figure 18: Clarke Error Grids, (a) Subject 2 and (b) Subject 3

Finally, for comparison purposes, Table 10 and Figure 20 (next page) shows the results obtained using the Python pipeline. The overall best performing model for all 3 subjects is the random forest regressor. This model is similar to the ensemble bagged forests regressor. For subjects 1 and 2, the random forest regressor provides a better performance predicting the values. Note that the R^2 value has been increased from 0.71 to 0.822 (+11.2%) for subject 1, and from 0.72 to 0.76 (+4%) for subject 3. Nonetheless, this trend has not followed for subject 3 whose performance has been reduced by a factor of 3%. However, this small difference could be related to the random state used to initialize the models and randomly split the dataset.

Table 10: Relevant metrics for Python pipeline results

	Subject 1	Subject 2	Subject 3
RMSE	6.359	10.15	15.246
Std. Dev. of train set	15.585	21.454	19.405
R^2	0.822	0.76	0.3718

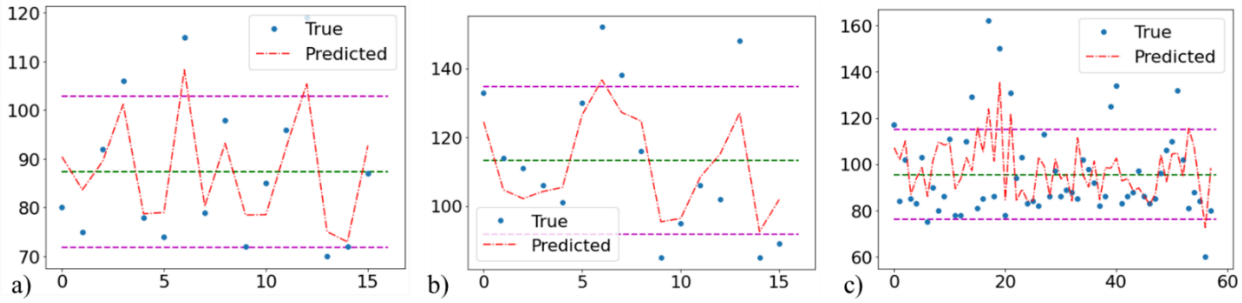


Figure 19: Plots from python pipeline

Chapter 5: Conclusions

5.1: Conclusions

This study aimed to explore the feasibility of using an array of non-invasive sensors in the prediction of blood glucose level (BGL). Self-imposed parameters required that the sensors be completely non-invasive and could realistically be included in a wrist-worn device. First, an already existing dataset (e.g., Ohio T1DM) was obtained to get a broad sense of what features would be required to accomplish this task. After eliminating all features from this dataset that would not fit into a wearable device, it was found that the non-invasive features within this dataset were not sufficient. Therefore, another dataset needed to be collected.

Since a completely new dataset was required, several sensors needed to be purchased. The sensors obtained were off-the-shelf ready to use for data collection and either well tested, or FDA approved. The signals from these sensors would then be converted into a feature set for use in regression analysis. Several commonly used machine learning models were considered and compared. The pipeline would determine the best model and present its results. This procedure was able to train several models whose performance on the test sets scored high (R^2>0.70). However, this was not true for all the subject's datasets. While the worst performing dataset underwent a different procedure, with fewer sensors, the sensors removed were bulky and likely could not be incorporated into a wrist-worn device. Furthermore, this dataset also raised the question of how whether the previous results would scale. Results for subject 1 and 2 only had 80 samples each. Therefore, the models fitted to these datasets may likely overfit due to the high dimensionality of the input feature set. This phenomenon is often called the "curse of dimensionality" and arises when an increase in dimensions causes the available target data to become sparse.

Moreover, the UofM dataset was only collected on healthy individuals, and therefore no hypo- or hyper- glycemic events were observed. These features may not maintain their predictive power outside the healthy range, not suitable for diabetic patients. A larger dataset would be required to arrive at conclusions about how these features would scale to the broader population and diabetic individuals especially. Finally, more time should be spent in pre-processing the data to reduce complexity in the system and identify the most critical features to be incorporated into a wrist-worn device.

In conclusion, although preliminary data suggests that some of the features collected could have sufficient predictive power to estimate BGLs non-invasively, the results remain inconclusive.

5.2: Future work and Recommendations

Future work on this project should focus on collecting a larger dataset, with target values that span as much of a range as possible. Even though the models trained on this dataset appeared to perform well, this dataset was very limited in size and scope. The target values' range was insufficient to get enough variability to concretely state whether this approach could scale to diabetic patients and still perform well. Furthermore, the size of the dataset makes it difficult to know whether the results are due to overfitting because of the high dimensionality of the data in proportion to the sample size. The sample size also raises the question of how well this dataset is representative of the broader population, even if diabetic individuals are not included.

To make sure this device has any commercial application, the dataset needs to have a more holistic representation of the global population. Individuals with diabetes are the most

necessary. We should increase the population range, including people from different races, ages, weights, and skin characteristics. This is critically important to avoid the failures of the PENDRA device mentioned in Section 2.2. Currently, the UofM dataset lacks any of this variability, and therefore representation, with all subjects being of the same race, nearly the same age, and all non-diabetic.

How the features are processed could also be improved, improving the pre-processing of the dataset. Instead of our simple cleaning to remove outliers and averaging over a time interval to match the timestamp to the target features, we could clean the dataset using a more automatic procedure. For example, we could use the software packages Kardia [18] and Ledalab [19]. Ledalab is a MATLAB based software package that focuses on the analysis of skin conductance (i.e., EDA). Kardia is another MATLAB based software package which is used to extract heart rate variability information from inter-beat interval information obtained from PPG/BVP signals. Both these features have been shown to be well correlated with BGL [20],[21],[22].

Finally, it would probably be worthwhile to decrease the features present in the dataset and investigate feature engineering/extraction. Some of the features, like ambient information, could be combined with other features to produce more meaningful information. Using principal component analysis, or similar dimensionality reduction techniques could also be used to transform the data of some features into a lower-dimensional space, thereby reducing the feature to sample ratio while preserving the information [23]. Thereby, we could eliminate the "curse of dimensionality" previously mentioned. We could also removed features iteratively (based on their variance, correlation to dataset, performance in models) to identify which features are the most important.

References

[1] W. V. Gonzales, A. T. Mobashsher, and A. Abbosh, "The progress of glucose monitoring—A review of invasive to minimally and non-invasive techniques, devices and sensors," *Sensors*, vol. 19, no. 4, pp. 1–45, 2019, doi: 10.3390/s19040800.

[2] N. Diabetes and D. Group, "Guide to diagnosis and classification of diabetes mellitus and other categories of glucose intolerance," *Diabetes*, vol. 28, pp. 1039–1057, 1979, doi: 10.2337/diacare.20.1.s21.

[3] W. Yang *et al.*, "Economic costs of diabetes in the U.S. in 2017," *Diabetes Care*, vol. 41, no. 5, pp. 917–928, 2018, doi: 10.2337/dci18-0007.

[4] C. Marling and R. Bunescu, "The OhioT1DM dataset for blood glucose level prediction: Update 2020," *CEUR Workshop Proc.*, vol. 2675, pp. 71–74, 2020.

[5] Burrin, J. M., & Alberti, K. G. M. M. "What is Blood Glucose: Can it be Measured?," *Diabetic Medicine*, vol 7, no. 3, pp. 199–206. doi:10.1111/j.1464-5491.1990.tb01370.x

[6] G. Cappon, M. Vettoretti, G. Sparacino, and A. Facchinetti, "Continuous glucose monitoring sensors for diabetes management: A review of technologies and applications," *Diabetes Metab. J.*, vol. 43, no. 4, pp. 383–397, 2019, doi: 10.4093/dmj.2019.0121.

[7] E. Cengiz and W. V. Tamborlane, "A tale of two compartments: Interstitial versus blood glucose monitoring," *Diabetes Technol. Ther.*, vol. 11, no. 1, 2009, doi: 10.1089/dia.2009.0002.

[8] A. Caduff, E. Hirt, Y. Feldman, Z. Ali, and L. Heinemann, "First human experiments with a novel non-invasive, non-optical continuous glucose monitoring system," *Biosens. Bioelectron.*, vol. 19, no. 3, pp. 209–217, 2003, doi: 10.1016/S0956-5663(03)00196-9.

[9] S. A. Weinzimer, "PENDRA: The once and future noninvasive continuous glucose monitoring device?," *Diabetes Technol. Ther.*, vol. 6, no. 4, pp. 442–444, 2004, doi: 10.1089/1520915041706018.

[10] J. Huang, Y. Zhang, and J. Wu, "Review of non-invasive continuous glucose monitoring based on impedance spectroscopy," *Sensors and Actuators, A Phys.*, vol. 311, pp. 1-9, 2020, doi: 10.1016/j.sna.2020.112103.

[11] A. Caduff *et al.*, "The effect of a global, subject, and device-specific model on a noninvasive glucose monitoring multisensor system," *J. Diabetes Sci. Technol.*, vol. 9, no. 4, pp. 865–872, 2015, doi: 10.1177/1932296815579459.

[12] O. K. Cho, Y. O. Kim, H. Mitsumaki, and K. Kuwa, "Noninvasive measurement of glucose by metabolic heat conformation method," *Clin. Chem.*, vol. 50, no. 10, pp. 1894–1898, 2004, doi: 10.1373/clinchem.2004.036954.

[13] I. Harman-Boehm, A. Gal, A. M. Raykhman, J. D. Zahn, E. Naidis, and Y. Mayzel, "Noninvasive glucose monitoring: A novel approach (Glucotrack)," *J. Diabetes Sci. Technol.*, vol. 3, no. 2, pp. 253–260, 2009, doi: 10.1177/193229680900300205.

[14] M. Mueller, M. S. Talary, L. Falco, O. De Feo, W. A. Stahel, and A. Caduff, "Data processing for noninvasive continuous glucose monitoring with a multisensor device," *J. Diabetes Sci. Technol.*, vol. 5, no. 3, pp. 694–702, 2011, doi: 10.1177/193229681100500324.

[15] A. Bolla, R. Priefer , "Blood glucose monitoring- an overview of current and future non-invasive devices," *Diabetes and Metabolic Syndrome: Clinical Research and Reviews*, vol. 14, pp. 739-751, 2020, doi: 10.1016/j.dsx.2020.05.016

[16] Clarke, W. L., Cox, D., Gonder-Frederick, L. A., Carter, W., & Pohl, S. L., "Evaluating clinical accuracy of systems for self-monitoring of blood glucose," *Diabetes Care*, vol. 10, no. 5, pp. 622–628, 1987, doi: 10.2337/diacare.10.5.622

[17] Pedregosa, F., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830. doi: 10.1289/EHP4713

[18] Joseph, B., Haider, A., & Rhee, P., "Non-invasive hemoglobin monitoring," *International Journal of Surgery*, vol. 33, no. B, 254–257, 2016, doi: 10.1016/j.ijsu.2015.11.048

[19] Benedek, M., Kaernbach, C., "A continuous measure of phasic electrodermal activity," *Journal of Neuroscience Methods*, vol. 190, no. 1, pp. 80–91, 2010, doi: 10.1016/j.jneumeth.2010.04.028

[20] T. Forst, P. Kann, A. Pfiitzner, R. Lobmann, and J. Beyer, "Association between 'Diabetic Thick Skin Syndrome' and Neurological Disorders in Diabetes Mellitus," *Acta Diabetol*, vol. 31, pp. 73–77, 1994, doi: 10.1007/bf00570538.

[21] E. Monte-Moreno *et al*, "Non-invasive estimate of blood glucose and blood pressure from a photoplethysmography by means of machine learning techniques," *Artif. Intell. Med.*, vol. 53, no. 2, pp. 127–138, 2011, doi: 10.1016/j.artmed.2011.05.001.

[22] H. Kudat *et al.*, "Heart rate variability in diabetes patients," *J. Int. Med. Res.*, vol. 34, no. 3, pp. 291–296, 2006, doi: 10.1177/147323000603400308.

[23] M. Gusev, L. Poposka, G. Spasevski *et al*., "Noninvasive Glucose Measurements Using Machine Learning and Neural Network Methods and Correlation with Heart Rate Variability," *Journal of Sensors*, 2020, doi: 10.1155/2020/9628281

Appendix

Appendix A: Machine Learning Pipeline Code in Python

```
# -*- coding: utf-8 -*-
"""
Created on Mon Oct 11 16:30:00 2021

@author: Brian
"""
import warnings
warnings.filterwarnings("ignore")
#%% Load libraries
import pandas as pd
import numpy as np
from matplotlib import pyplot

from sklearn.model_selection import train_test_split,cross_validate,\
                                    KFold


from sklearn.metrics import r2_score, mean_squared_error

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.pipeline import Pipeline

from sklearn.preprocessing import StandardScaler



#%% Methods
def load_my_data(loadpath='./data/Pilot/Subject1_DP/Outputs/smallerdataset-DP.csv',):
    """Loads in datasets as specified by loadpath
        :param loadpath (list of string):
        :returns parsers (list of dataframes):
        :returns filenames (list string):
    """
    file_names = []
    parsers = []
    for file in loadpath:
        file_names.append(file.split('/')[-1])
        parsers.append(pd.read_csv(file))
    return parsers,file_names
def get_x_cols(df,target_name):
    """
    Returns a list of columns name after dropping columns that wont be used
```

```
    as variables in regression
    :params df (DataFrame):
    :params df (str or list of str):
    :returns features_df(DataFrame):
    :returns target_df(DataFrame):
    """
    x_cols = list(df.columns)
    copy = x_cols.copy()
    if not (type(target_name) == list):
        target_name = [target_name] #if not list of str, make it so
    for target in target_name:
        for col in x_cols:
            if ('Lanced' in col):
                copy.remove(col)
            if ('Time' in col):
                copy.remove(col)
            if ('Activity' in col):
                copy.remove(col)
            if ('Intervention' in col):
                copy.remove(col)
            if ('Unnamed' in col):
                copy.remove(col)

        copy.remove(target)
    features_df = df[copy].copy()
    target_df = df[target_name].copy()
    return features_df,target_df
def create_pipelines(seed, verbose=0, n_components=4):
    """
        Creates a list of pipelines with preprocessing, models and scalers.
    :param seed: Random seed for models who needs it
    :return:
    """
    models = [
            ('OLS', LinearRegression()),
            ('KNN', KNeighborsRegressor()),
            ('SVR', SVR()), #C = 1.0, epsilon=0.1
            ('DTR', DecisionTreeRegressor()),
            ('NNR', MLPRegressor(hidden_layer_sizes=(100, 100))),
                        #   tol=1e-2, max_iter=500, random_state=seed))
            ('RFR', RandomForestRegressor()),

           ]
    scalers = [
            ('StandardScaler', StandardScaler()),

           ]
    additions = [

            ]
    # Create pipelines
    pipelines = []
    for model in models:
```

```
        # Append only model
        model_name = "_" + model[0]
        pipelines.append((model_name, Pipeline([model])))

        # Append model+scaler
        for scalar in scalers:
            model_name = scalar[0] + "_" + model[0]
            pipelines.append((model_name, Pipeline([scalar, model])))

        # To easier distinguish between with and without Additions (i.e: PCA)
        # Append model+addition
        for addition in additions:
            model_name = "_" + model[0] + "_" + addition[0]
            pipelines.append((model_name, Pipeline([addition, model])))

        # Append model+scaler+addition
        for scalar in scalers:
            for addition in additions:
                model_name = scalar[0] + "_" + model[0] + "_" + addition[0]
                pipelines.append((model_name, Pipeline([scalar, addition, model])))
    return pipelines
def run_cv(X_train, y_train, X_test, y_test, target_name, pipelines, metrics, seed, num_folds,
               dataset_name, search_space,n_jobs):
    """
        Iterate over the pipelines, calculate CV mean and std scores, fit on train and predict on test.
        Return the results in a dataframe
    """

    # List that contains the rows for a dataframe
    rows_list = []

    # Lists for the pipeline results
    names = []
    # test_scores = []
    prev_clf_name = pipelines[0][0].split("_")[1]
    count = 0
    for name, model in pipelines:
        #validation bit

        kfold = KFold(n_splits=k, random_state=seed,
                              shuffle=True)
        # gs = GridSearchCV(model,this_space,scoring=metrics,refit=metrics[0],
        #                   cv=kfold,verbose=0)
        # gs.fit(X_test,y_test)
        # this_best_estimator = gs.best_estimator_

        cv_results = cross_validate(model, X_train, y_train,
                            cv=kfold, n_jobs=n_jobs,
                            scoring=metrics)
        names.append(name)
        # Add separation line if different classifier applied
        rows_list, prev_clf_name = check_seperation_line(name, prev_clf_name,
                                             rows_list)
```

```
        # Get best
        this_score = cv_results['test_r2'].mean()
        if count == 0:
            #initialize the following variables in the first iteration
            best_score = this_score
            count = 1
            best_model = model
            best_name = name
        elif this_score > best_score:
            #get best to return
            best_score = this_score
            best_model = model
            best_name = name
        # Add for final dataframe
        model.fit(X_train,y_train[target_name])
        y_pred = model.predict(X_test)
        results_dict = {"Dataset": dataset_name,
                        "Classifier_Name": name,
                        "CV_mean_R2": cv_results['test_r2'].mean(),
                        "CV_std_R2": cv_results['test_r2'].std(),
                        "CV_mean_RMSE":
                           -cv_results['test_neg_root_mean_squared_error'].mean(),
                        "CV_std_RMSE":
                           cv_results['test_neg_root_mean_squared_error'].std(),
                        "Test_R2":
                          r2_score(y_test,y_pred),
                        "Test_RMSE":
                          mean_squared_error(y_test,y_pred,
                                    squared=False)#return RMSE
                        }
        rows_list.append(results_dict)

    all_results = pd.DataFrame(rows_list)

    return best_name,best_model,all_results
def check_seperation_line(name, prev_clf_name, rows_list):
    """
        Add empty row if different classifier ending
    """

    clf_name = name.split("_")[1]
    if prev_clf_name != clf_name:
        empty_dict = {"Dataset": "", #used to create empty rows
                    # "Classifier_Name": "",
                    # "CV_mean": "",
                    # "CV_std": ""
                    # "Test_acc": ""
                    }
        rows_list.append(empty_dict)
        prev_clf_name = clf_name
    return rows_list, prev_clf_name
def plot_true_vs_predicted(model,model_name,X_test,y_test,title):#
    """
```

```
    some useful plotting functions
    """
    # y_test.reset_index(drop=True,inplace=True)
    nsample = len(y_test)
    fig, ax = pyplot.subplots(figsize=(8,6))
    x = list(range(0, nsample))
    sorted_idx = y_test.argsort()
    ax.plot(x, np.array(y_test)[sorted_idx], 'o', label="data")
    y_pred = model.predict(X_test)
    print(mean_squared_error(y_test,y_pred,squared=False))
    ax.plot(x, (y_pred)[sorted_idx], 'r--.', label=model_name)
    fig.suptitle(title, fontsize = 12)
    ax.legend(loc='best');
def get_search_space(seed):
    """
       Create a dictionary with classifier name as a key and it's hyper parameters options as a value
    :return:
    """
    # OLS PARAMS
    ols_params = {
                "OLS__fit_intercept": [True,False]
                }

    # KNN PARAMS
    n_neighbors = [int(x) for x in np.linspace(start=1, stop=20, num=4)]
    weights = ["uniform", "distance"]
    algorithm = ["auto", "ball_tree", "kd_tree", "brute"]
    leaf_size = [int(x) for x in np.linspace(start=5, stop=50, num=5)]
    p = [int(x) for x in np.linspace(start=1, stop=4, num=4)]
    knn_params = {'KNN__n_neighbors': n_neighbors,
                'KNN__weights': weights,
                'KNN__algorithm': algorithm,
                'KNN__leaf_size': leaf_size,
                'KNN__p': p,
                }

    # SVR PARAMS
    kernel = ["linear", "poly", "rbf", "sigmoid"]
    degree = [int(x) for x in np.linspace(start=1, stop=5, num=5)]
    C = [x for x in np.arange(0.1, 2, 0.4)]
    svr_params = {'SVR__C': C,
                'SVR__degree': degree,
                'SVR__kernel': kernel,
                }


    # DecisionTreeRegressor PARAMS
    criterion = ['mse', 'mae','poisson'] #'freidman_mse',
    splitter = ['best', 'random']
    max_depth = [int(x) for x in np.linspace(10, 110, num=5)]
    max_depth.append(None)
    min_samples_split = [2, 5, 10]
    min_samples_leaf = [1, 2, 4]
```

```
    max_features = ["auto", "sqrt", "log2"]
    max_features.append(None)
    random_state = [seed]
    dtr_params = {
                'DTR__criterion': criterion,
                'DTR__splitter': splitter,
                'DTR__max_depth': max_depth,
                'DTR__min_samples_split': min_samples_split,
                'DTR__min_samples_leaf': min_samples_leaf,
                'DTR__max_features': max_features,
                'DTR__random_state':random_state
                }



    hypertuned_params = {
                        "OLS": ols_params,
                        "KNN": knn_params,
                        "SVR": svr_params,
                        "DTR": dtr_params,
                       }

    return hypertuned_params

#%% Attributes

seed = 1234
metrics = ['r2','neg_root_mean_squared_error']
k = 10
#%% Choose your Character
condition1 = 0
if(condition1 == 0):
    target_name = 'Libre (mg/dL)'
    loadpath = ['./data/Pilot/Subject1_DP/Outputs/smallerdataset-DP.csv',
                './data/Pilot/Subject2_BM/Outputs/smallerdataset-BM.csv',
                './data/Pilot/Subject3_BB/Outputs/smalldataset-BB.csv']

elif(condition1 == 1):
    target_name = 'test_glucose'
    loadpath = ['./data/Ohio/Raw/raw_559.csv']

dataframes,file_names = load_my_data(loadpath)
file_name = file_names[0]
df = dataframes[0] #just look at one at a time for now
X,y = get_x_cols(df,target_name)
#%% Choose how to split the data
condition2 = 0
if(condition2 == 0):
    X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.20,
                                            random_state=seed)

if(condition2 == 1):
    pass
```

```
#%% Main


pipe = create_pipelines(seed)
search_space = get_search_space(seed)
best_name,best_model,all_results = run_cv(X_train, y_train, X_test, y_test,
                                  target_name, pipe, metrics, seed,
                                  num_folds=k,dataset_name=file_name,
                                  search_space=search_space,
                                  n_jobs=-1)
if('Scaler' in best_name.split('_')[0]):
    scaler = best_model.steps[0][1]
    scaler.fit(X_train)
    X_test = scaler.transform(X_test)
best_model.fit(X_train,y_train[target_name])
plot_true_vs_predicted(best_model,best_name,X_test,y_test[target_name],
                   title='True vs Predicted')
print(best_model.score(X_train,y_train[target_name]))
print(best_model.score(X_test,y_test[target_name]))
print(best_model.score(X,y))
```