Levi Zell
12/23/2016

# NIRS Project

Machine/Deep Learning

## Table of Contents



# I. Definition

## Project Overview

Personal electronic devices are ubiquitous. These connected devices allow data to be collected and analyzed to help us do things like track our health, keep us connected with friends and family, get directions, recommend products, find reviews on products before we purchase them, and on and on. The IoT market is expected to add to the number sensors a person carries with them. Sensors that measure things like additional physiologic data.

For this project, I analyzed the raw optical light data from an small portable fitness device in order to see if I could predict heart rate, which was measured through the use of a heart rate chest strap for running. Unlike a number of electronics devices that incorporate wrist-based optical heart rates sensors, this one was not used for that. The reason for attempting this was to see if this was possible to do, what other physiologic measurements could be made using similar machine learning techniques.

## Problem Statement

The goal of this project is to be able to accurately predict heart rate from optical light reading sent from an electronic device. Data was collected and transmitted from the optical electronics device and the heart rate monitor. The heart rate values are what we want to predict (heart rate) and the optical light data will be used to make that criterion predictive value. To accomplish this, we will try different machine learning techniques that seem appropriate for this type of prediction. The algorithm that most accurately predicts the heart rate value without overfitting, will be the chosen classifier.

## Metrics

There are two metrics that will be used in this project:

1. Optical Light Data
2. Heart Rate Data

The optical light data collected from the device by sending light from the device into the user then measuring light reflected, calculating the light absorbed. This is recorded as an integer to indicate the relative amount of absorption. From this data, transmitted at every 0.5 seconds for 4 different spectrums of light. Therefore, for every half second, 8 different data points are collected.

Variance, bias, and accuracy, and F1 score will be used to measure the performance of the different models. Variance is calculated by the formula:

It was chosen along with bias, was to see both how well it accurately measures what we are trying to measure based on the sample and target data (variance) and how well it will accurately predict other data (bias). Variance measures how sensitive a model is to different training sets and bias is measured by the average error of the dataset. These are calculated using validation curves in sklearn.

Accuracy is calculated through the equation:

$$\text{accuracy} = (\text{true positive} + \text{true negatives}) / \text{dataset size}$$

This tells how many times our model can correctly predict the labeled values we have for a given data set. This will tell us how many times out of sample data sets we can predict the heart rate values we are using as accurate measures.

And finally, F1 score is used to more precisely measure the precision of a model. The best model is equal to 1 and the worst equal to 0. This will accurately reflect precision and recall of a model at the same time. It is calculated by the formula:

$$F1 = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$$

# II. Analysis

## Data Exploration

The datasets used in this project consist of four different CSV files. These were uploaded using Pandas, then the head of each was printed to show the columns, features, and data types contained within the files (see below). Next, columns with data that was determined to be either redundant or not of any significant importance was dropped. This was determined from the combination of a scatter matrix, heatmap, values not measured, and domain knowledge. These values included the following: 'mm-dd', 'Target Power', 'Speed', 'Power', 'Lap', 'Cadence', 'yyyy.mm.dd', 'Unnamed: 15', 'Unnamed: 16', and 'Temp'. Values of 0 or NaN for features 'SmO2 Averaged', 'RR Heart Rate', and 'Heart Rate' were dropped as they represented dropouts or inaccurate recordings by the devices.

| mm-dd | hh:mm:ss | SmO2 Live | SmO2 Averaged | THb | Lap | Target Power | Heart Rate | RR Heart Rate | Speed | Power | Cadence |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1-Jan | 0:00:00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Jan-00 | 0:00:01 | 0 | 81 | 12.35 | 0 | 0 | 75 | 75.01831502 | 0 | 0 | 0 |
| Jan-00 | 0:00:01 | 0 | 81 | 12.35 | 0 | 0 | 75 | 75.01831502 | 0 | 0 | 0 |
| Jan-00 | 0:00:02 | 0 | 81 | 12.39 | 0 | 0 | 75 | 75.01831502 | 0 | 0 | 0 |
| Jan-00 | 0:00:02 | 0 | 81 | 12.39 | 0 | 0 | 75 | 75.01831502 | 0 | 0 | 0 |
| Jan-00 | 0:00:02 | 0 | 82 | 12.34 | 0 | 0 | 75 | 75.01831502 | 0 | 0 | 0 |

*Figure 1 - File 1 Type*

| yyyy.mm.dd | hh:mm:ss | SmO2 Live | SmO2 Averaged | s680 | s720 | s760 | s800 | l680 | l720 | l760 | l800 | Good | Temp | THb | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2016.10.27 | 20:56:43 | 80 | 80 | 1842372 | 3433461 | 3035776 | 4354345 | 136292 | 278251 | 222780 | 317085 | 20 | 713 | 12.38 | 6 |
| 2016.10.27 | 20:56:43 | 80 | 80 | 1835872 | 3420182 | 3020543 | 4330833 | 134810 | 276001 | 221433 | 313725 | 20 | 713 | 12.43 | 6 |
| 2016.10.27 | 20:56:44 | 80 | 80 | 1839622 | 3428461 | 3030196 | 4349293 | 136665 | 278728 | 223080 | 316905 | 20 | 713 | 12.39 | 6 |
| 2016.10.27 | 20:56:44 | 80 | 80 | 1836259 | 3421972 | 3023095 | 4333888 | 135711 | 277520 | 222095 | 314815 | 20 | 713 | 12.42 | 6 |
| 2016.10.27 | 20:56:45 | 81 | 81 | 1836027 | 3419434 | 3019762 | 4329108 | 136325 | 277840 | 221934 | 314208 | 20 | 713 | 12.41 | 6 |
| 2016.10.27 | 20:56:45 | 81 | 81 | 1836027 | 3419434 | 3019762 | 4329108 | 136325 | 277840 | 221934 | 314208 | 20 | 713 | 12.41 | 6 |

*Figure 2 - File 2 Type*

## Exploratory Visualization

Besides the printing of the Dataframe itself, a heatmap, and scatter matrix were made using matplotlib. This helped to parse the data as described previously by giving a visual context to the importance and influence different values have. These visualizations show what types of information, whether there were any non-zero values, and what the correlation to the dependent variable (heart rate) are. The heatmap show the correlations between each feature and another feature while also presenting it with a color value. The darker the color the higher the correlation, with dark blue being nearly perfect, followed by dark red for high correlations. The lighter shades represent lower correlated values, and thus features having low relation to other features and/or the target value. The scatter matrix shows this but in a graph format where one value on the x-axis is related to another value on the y-axis. This shows directionality and “tightness” of the relationship by the shape and sharpness of the plot shape respectively. Finally, Figure 7 shows the changes in certain features, scaled on the same time value, on the x-axis. Here, the light values were overlaid and show their relationship and shape to each other which gives visualization to the differences in their magnitudes. This can be compared to our target feature – heart rate – to see if there is any prominent relationship that might be seen visually. All of these visualizations help reveal any potentially obvious relationships that could guide data exploration. Even complex relationship can be discovered when comparing values or shapes amongst one another. Although domain knowledge played a large role in determining that the optical light values were the major features used, exploring values such as the "Normalized_O2_HHb " was valuable for this dataset, as was seeing the graphed values in Figure 7 to find dropouts and other 0 values to process the data.

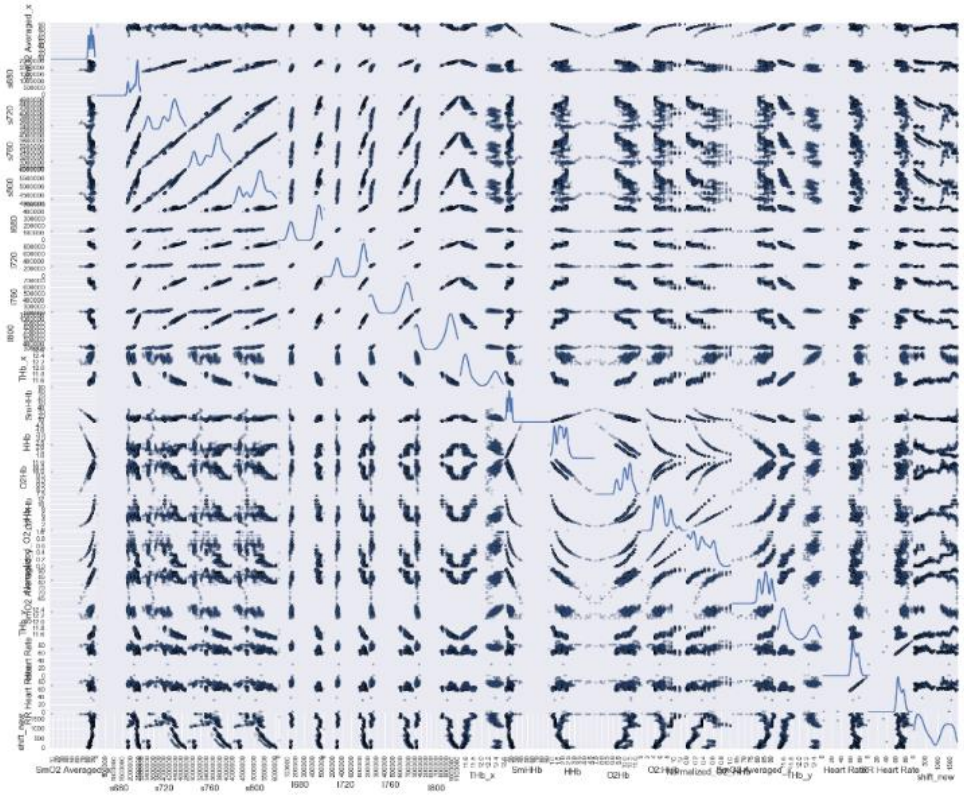

*Figure 3 – Scatter matrix*

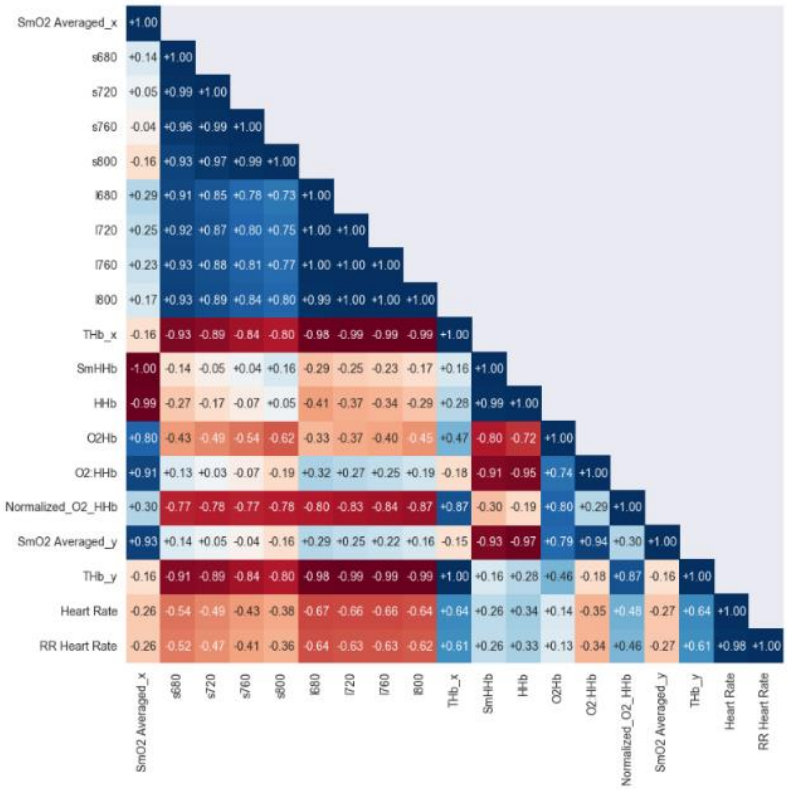


*Figure 4 - Heatmap*

## Algorithms and Techniques

It was decided to test a few different models. Taking a supervised learning approach, GuassianNB and SVC where performed. Cross-validation and GridSearch were used to properly train the models. Variance, Bias, and F Score were used to access the performance of each model. The parameters are shown in the code within the notebook.

GaussianNB was chosen because these models have fast training speeds and are relatively simple. They also perform well with smaller data sets like ours which have only a few thousand values. A disadvantage of this model is that it assumes conditional independence and therefore can't learn interactions between features.

SVC was performed because they are useful for a wide variety of problems, especially with continuous values and non-linear dependent/independent variable relationships. The downside is they can be complex and difficult to interpret while also potentially requiring large amounts of training time and computational power. However, for our problem of trying to predict non-linear values with possibly complex transformations of our data.

Next a Decision Tree Regressor and PCA were performed. The PCA revealed that the set for 2 clusters had the highest Silhouette score in a K Means analysis. This model was chosen due to it's ability to use batching to increase its power and utility. Decision trees are good for using feature selection and screening to help implicitly determine which feature or set of features are most valuable. These also require little effort to setup and use with data in addition to working with non-linear data. Using PCA to combine features into new features that can be superior to the original feature, amplify a model like decision trees, which, hopefully, will result in a useful predictor.

## Benchmark

The benchmark threshold for this project was set at 85%. This was determined from searching for the relative accuracy rates for wrist-based optical heart rate sensors available for consumers. From the specific article cited here, the lowest accuracy for a similar heart rate range from the input data used, was set as the threshold. The error percentage for one watch was roughly 10% so our benchmark was set at 150% of this - 15%. Ideally, this benchmark will be surpassed despite lacking the team of professionals these companies.

https://www.cnet.com/news/how-accurate-are-wristband-heart-rate-monitors/

To improve predictability, it was determined, both to grow my personal toolbox as well as to get greater accuracy in a model, was to use an RNN with Tensorflow.

# III. Methodology

## Data Preprocessing

Four CSV files contained all the data for this project. There were two subjects used for the tests, thus two files per test. One file contains the synchronize information for the optical data and heart rate and the other contains the raw optical heart rate data used to make predictions on the labeled

heart rate dataset. In the raw optical data file, 4 wavelengths (680, 720, 760, & 800) are recorded at long and short spacing's (l or s respectively).

L680 is the sum of 20 readings at 0.5 seconds increments from the long spacing detector with the 680 LED turned on minus the average of the dark readings taken before and after it

In the raw NIRS data files, gaps in recording from the emulator were manually deleted as were data fields in the header to put the data into a easily readable format for using pandas. The correct

Following this, the time stamps in the columns with the column headers 'mm:hh:ss' were synchronized by manually finding SmO2 + THb values in a row that aligned for several consecutive rows. The data was then converted to time delta values to allow for subtraction to make the values equal, with the new time stored in the column titled 'shift'.

Further processing was done to remove NaN's and 0 values that might interfere with correct data analysis.

## Implementation

Each algorithm was implemented using the defined input features. After pre-processing, the columns that contained this values were: "s680", "s720", "s760", "s800", "l680", "l720", "l760", "l800", and "Normalized_O2_HHb ".

The column which contained the feature we are trying to predict was the last column; "Heart_Rate".

As mentioned, the data was divided into labeled test, prediction, and training sets, before being implemented into the classifiers.

The other column which held an integer value was kept in the data set because it was optimistically (or maybe naively) thought that maybe we could predict the value to a certain decimal point. It was quickly determined that with the initial error rates that this probably wasn't going to be the case. Also, the numerous 0 values that were in the data, skewed it so much that more pre-processing was needed to eliminate these. These complications led to adapting the steps and algorithms to get better results.

The decision to implement a deep learning model with Tensorflow required A LOT of additional work and added more than a week to this process. This was due to things like installing libraries, changing to Python 3, learning to process data into structures Tensorflow can use, and just learning how to build a neural network. This was worth it as it is a more relevant tool for this type of prediction as well as being a skill that will be valuable to have in the future.

## Refinement

Improvements were made by trialing different models as well as trying different types of predictive analytics. Initially, supervised ML models were used, followed by batching, and finally a deep neural network. This decision was based on research done when looking for a model that has great success with multiple features, in a time-series.

The RNN had an accuracy of, ~50% within a range of +/-2, ~73% within a range of +/-5, and 83% within a range of +/-10 of the target heart rate. This is below our target accuracy within an expected tolerance. At the beginning, it was assumed that the tolerance would be +/- 0, however +/- 1 would

be a strict tolerance itself, considering the sampling rate is just 1/s for heart rate and a rolling average is likely being used to predict each of these values anyway. Additionally, considering the metrics were recorded on two separate devices, with different sampling rates, unknown summarization algorithms, and whether these were reflected and recorded accurately by the recording software, make the accuracy of the prediction extremely difficult.

# IV. Results

## Model Evaluation and Validation

As stated in the beginning, the models would be evaluated per accuracy, bias, variance, and F1 score. The results of the SVM were extremely low, as well as the GaussianNB. For both of these models a shuffle split cross-validation was performed, separating the data into X_train, y_train, X_test, and y_test sets of varying sizes. The training sets were set at sizes of 1000, 2000, and 3000. Approximately ¼, ½, and ¾ of the total data set size respectively.

For the SVM data set the F1 score was high (~0.99) it was low for the test sets (<0.1). This shows a high bias and low variance. This suggests that this model might lack robustness for making predictions for other training sets. Although different types of SVMs have strengths that can account for various types of data, in addition to having tunable hyperparameters, limitations in the quality of the data discussed in the next section lead us to continue pursuing other types of models.

The GaussianNB showed extremely low scores for both the training and test sets, < 0.10. The reason for this may be that conditional independence may be a poor assumption for this data set. Instead, prediction requires feature dependence. Although useful for providing this insight, this model is not a good fit for this type of data.

For the Decision Tree Regressor a preprocessing module in sklearn along with a visualization was used to explore the max_depth, variance and bias scores for the training and test sets, shown below.


*Figure 5 - Variance, Bias, max_depth visualization*

The optimal depth was found to be 11 using this preprocessing step with the Grid Search function. Our performance score for this optimized model was 0.287. This surpassed our previous models and with what we learned about from the GaussianNB model, performing PCA and implementing dimensionality reduction, which combines features into more powerful ones. A useful visualization that help see the explained variance is shown in Figure 7.

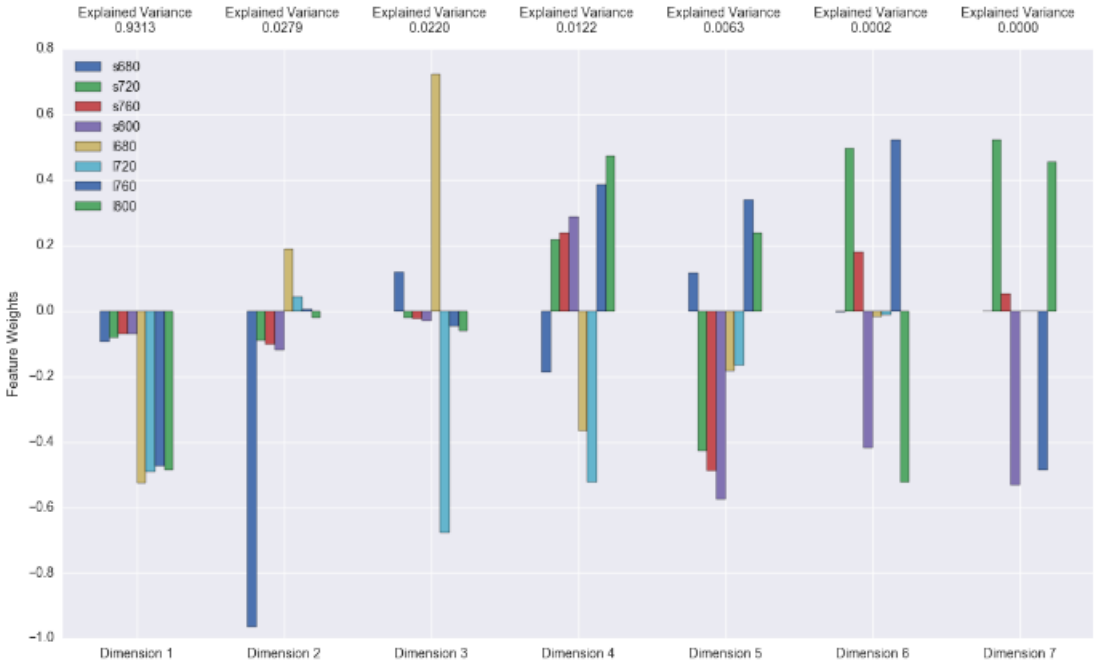


*Figure 6 - Explained Variance*

Finally, our RNN model appears to provide the greatest accuracy once trained. This varied based on the accepted variability mentioned previously and is restated here: approximately 50% within a range of +/-2, 73% within a range of +/-5, and 83% within a range of +/-10 of the target heart rate

It terms of speed and complexity it might not be the right choice, but in terms of accuracy it wins. The parameters were tuned to optimize the model for each accepted range and included altering batch size, learning rate, mu, and epoch. Learning rate and batch size seemed to have the greatest impact on this model, and was different for each range.

This model was chosen because it may be the best choice for training on time-series data, based on research when looking for models that would work best for time-series data. With built in gradient descent and back propagation, it was hypothesized to powerful predictive potential. Further developing this to an LSTM, although more difficult, could prove to be even more powerful.

Further testing should be done to increase the trust in the model working and verify its strength. However, again I will mention that having better quality data will improve this model and its robustness for being useful to use with other data sets.

## Justification

The different models performances failed to match the benchmark of 85% accuracy established earlier in the project. As already discussed, initially a range for acceptable heart rate values was not

set, however a range of +/- 2 from the labeled result seems to be reasonable. Despite this added leniency, none of the models matched the benchmark.

The significance of these results is not sufficient for the benchmark threshold, as the benchmark was established from a project were similar lengths of time/data were used to establish the accuracy of the optical device. Additionally, the accuracy of the RNN for the data may become too biased. Even an accuracy of 90%+ would be questioned for the reasons discussed more in the next section.

Based on these results the model should not be strong enough for this problem. This doesn't mean it is the fault of the model itself, rather the due to the inferiority of the data. I would hypothesize that collected data in a better way, in the same software, as well as having more data points would aid in training a model that would be more extensible to other data sets. Also, as discussed, building an LSTM RNN may do an even better job.

# V. Conclusion

## Free-Form Visualization

The visualization below shows the relevant problem of the data set - the different features along with the value that is value to predict. The plot overlays each feature, scaled to display all on the same axis with their relevant measures, scaled on the y axis and all plotted per the same time measurement they were recorded on, on the x axis.

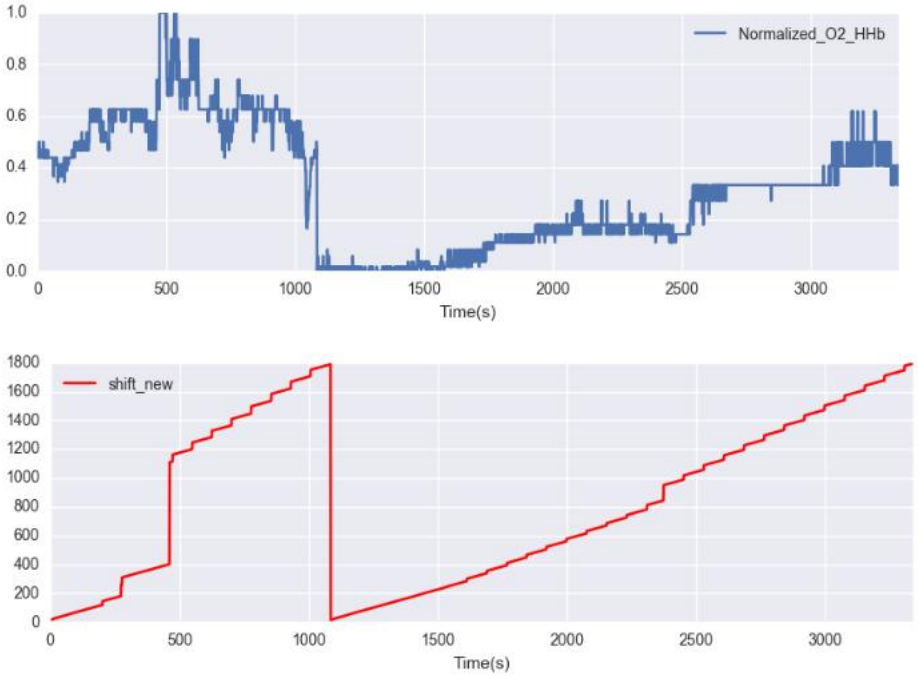


*Figure 7 - Scatter Plot*

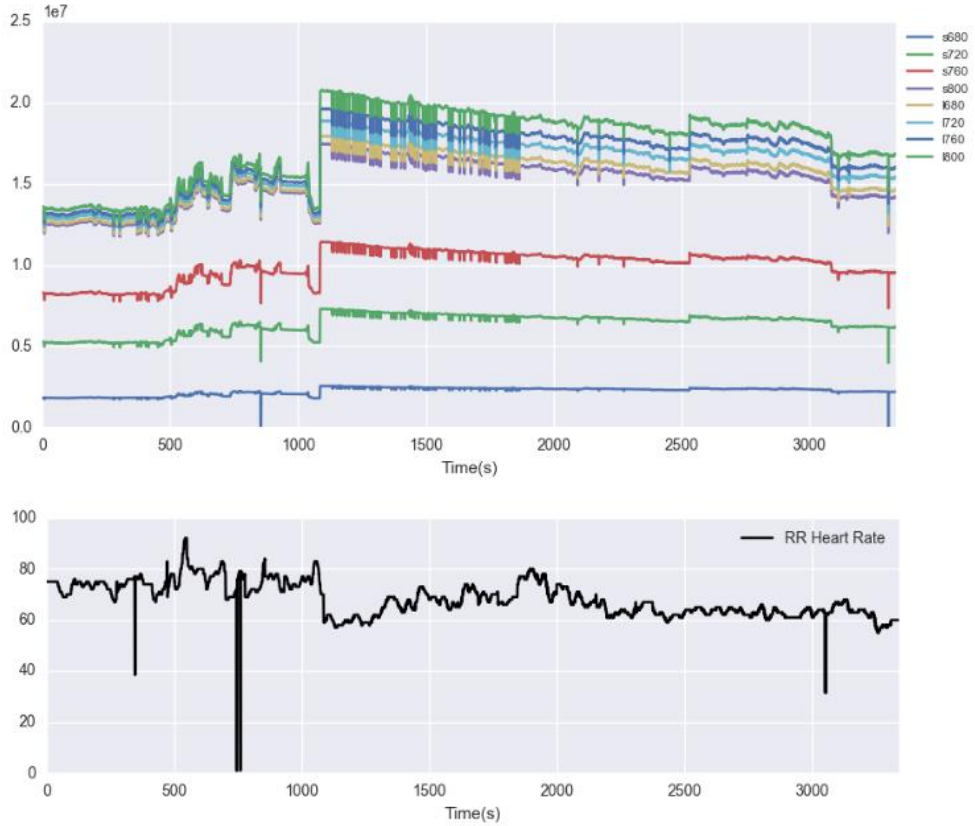


*Figure 7 - Scatter Plot cont.*

From this visualization, you can see how some of the features relate to one another and the relative shape of the variable we are trying to predict. You can see that the optical data is the same, however the magnitude is what varies. Therefore, the relevance in its predictability likely lies in these differences in magnitude.

## Reflection

Our solution was achieved by determining the following:

1. **What were we trying to predict?**
   This was our HR value recorded along with our optical data
2. **What do we need to be able to predict this?**
   This was the optical data and normalized value we measured and calculated above.
3. **What methods can we use to train a model to be able to accurately predict this?**
   We did this by experimenting with different Machine Learning techniques and measuring their accuracy. A baseline value was determined by information collected about what similar technologies can achieve.
4. **Have we achieved our goal?**
   Processing the data and tuning our different algorithms has shown that we failed to achieve our goal of surpassing our benchmark.

The limitations of this project were that we were unable to accomplish our benchmark, for a variety of reasons. The first is we may have not perfectly tuned our models to optimize our accuracy. It may be possible to adjust models and hyperparameters, however this may and did increase the bias. As

mentioned, the general tactic and models of this project were valuable. Even when a model didn't work, it helped explain why this might be – lacking conditionally independent features. And an RNN may be a powerful tool to accomplish our benchmark, especially by implementing an LSTM, if the problem occurred in the data collection step. Although the benchmark was failed to be reached, the project as a whole was a success in exploring the overall process, and learning about the numerous ways problems can arise in the training of a model, including: collection, parsing, training, tuning, and execution.

## Improvement

The implementation could be improved by increasing the amount of data fed into the model. Deep neural networks are said to be improved, and even require extremely large datasets to make them useful. A key limitation of this study seems to be that the features used to predict our target – heart rate, were measured by separate devices and software. The merging of these two datasets was difficult because the values did not match identically. At each time step where the devices were determined to be in sync, the values for SmO2, which was measured on both software's, did not exactly match, varying by 1-4 values in certain places. Therefore, to develop a successful model, the process needs to be repeated starting with the initial step of collecting the data properly.

The project itself was successful overall in terms of learning. Although redoing it all over again might result is passing the benchmark threshold, in real life that isn't always the case either. Identifying the problem and modifying the method of action is probably something that occurs at least some of the time. Making these necessary fixes could make the next iteration a success by passing our benchmark threshold.