Project 1 Report

CSE 572: DATA MINING

Spring 2020

Submitted to:

Professor Ayan Banerjee

Ira A. Fulton School of Engineering

Arizona State University

Submitted by:

Debarshi Roy (droy17@asu.edu)

February 10, 2020

1. Introduction

The Meal Detection Project is a part of the course requirement for Data Mining (CSE 572) for the session of Spring 2020 at Arizona State University. The goal of the project is attempting to develop a computing system that can understand variations in glucose levels corresponding to the meal intake, calculate the amount of insulin needed to keep the CGM level optimum and inject as needed. The data is collected from a continuous glucose monitoring device of five patients, which checks the glucose levels every five minutes. The data contains several time series of glucose levels with a five-minute interval for all the patients. In addition to it, it also provides information about the Bolus and Basal insulin injected at various points of time. Bolus is the amount of insulin taken by the patient himself before a meal after factoring the number of calories which he will take, and the requisite insulin amount needed to keep the CGM levels normal. Basal insulin is injected by the device during continuous monitoring. The issue with the current mechanism is that the manual injection is dangerous as erroneous prediction by a patient will lead to severe medical conditions.

2. Project Phase 1: Feature Extraction

In this phase, I have selected and implemented five feature extraction methods for the two time series data "CGMDatenumLunch" and "CGMSeriesLunch" for five patients. Before carrying on with that, I had to do some pre-processing of the data. The CGM data were collected with a MATLAB timestamp and had to be converted into Unix Timestamps. The data also had some missing and null values, they must be cleansed and estimated to the standard deviation of the respective time series. Estimation is important since replacing the values with zero will result in noise or outliers which may have a profound impact on the model. Before the feature extraction was done, data were normalized, since the values of each time series are of different scales and hence difficult to compare with.

The five feature extraction techniques that I have used are:

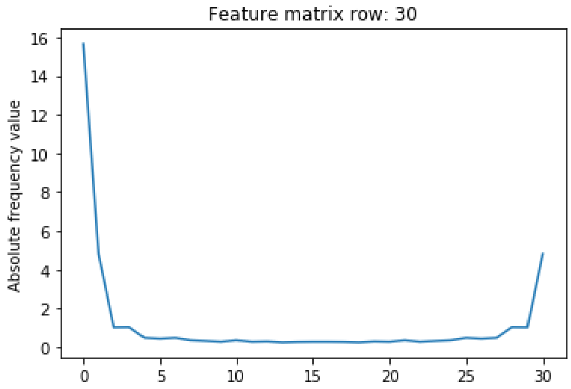
1. Fast Fourier Transform
2. Discrete Wavelet Transform
3. Coefficient of Variation
4. Windowed Entropy
5. Area Under Curve

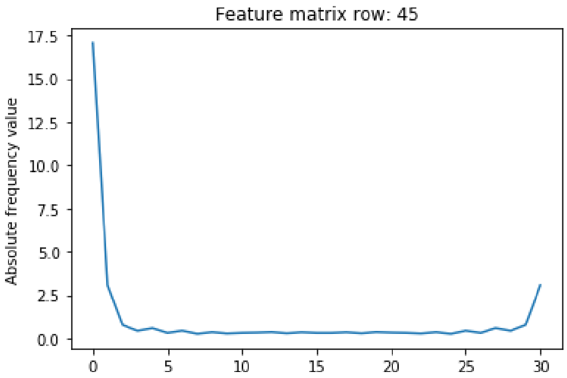
Intuition behind feature selection

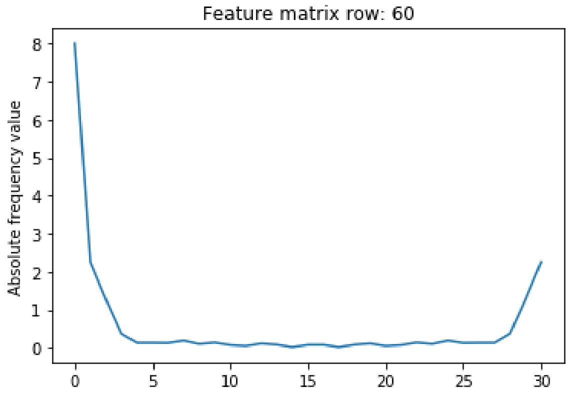
Since I am extracting features which can measure the time interval during which a meal was in took, I needed to find the spikes in glucose levels over a period. The features which I selected try to approximate the descent in the CGM Data curve. Initially, the data interpretation just by using CGM Data with respect to only timestamp was vague. However, by considering, the Insulin Basal and Insulin Bolus series plotting and comparing them with the Glucose series, we get a specific pattern depicting how glucose levels vary. Hence, I decided to deduce various ways to co-relate these levels with each other to observe specific feature patterns. I was completely aware of the fact that plotting of values for each activity along the individual feature dimension would give more suitable ground to make a reliable judgment. The contribution made by each feature extraction method and how these features are extracted are explained below.

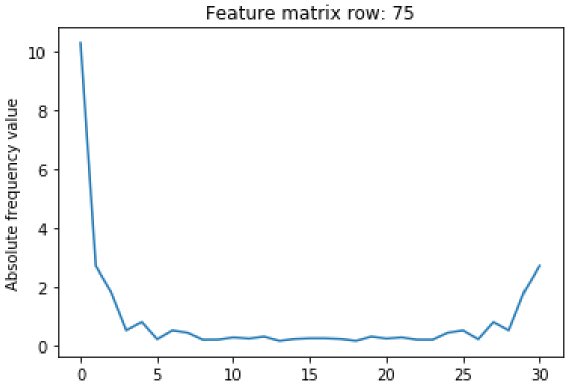
2.1. Fast Fourier Transform

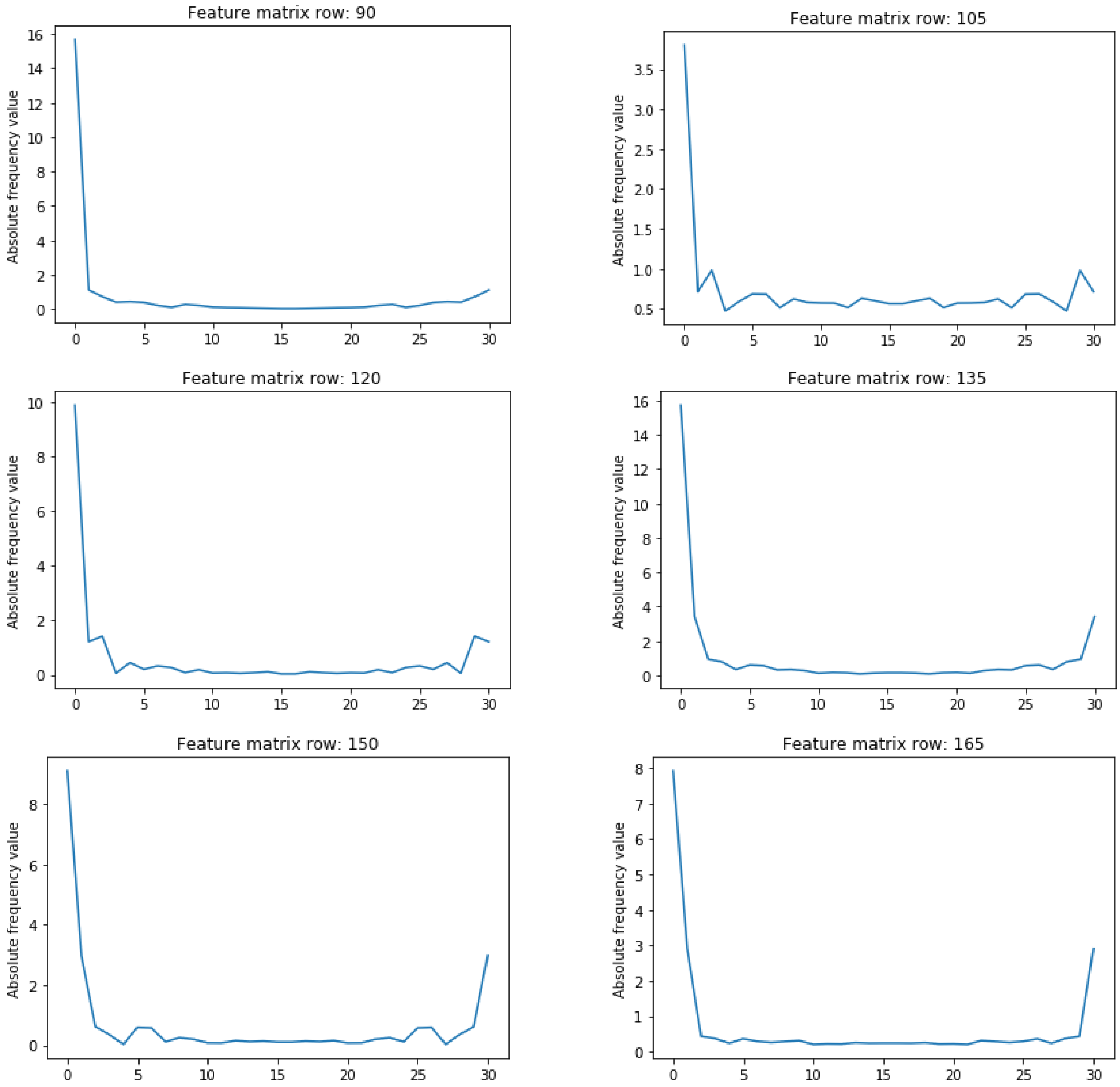
The Fast Fourier Transform is a mathematical method for transforming a function of time into a function of frequency. It deconstructs the original signal into a combination of sinusoidal functions. The glucose levels of the patients mostly follow a cyclic pattern, steadily increasing when a patient is consuming a meal and decreasing after the insulin is injected either by Bolus or Basal injection. This near cyclic pattern can be made use of by adopting the Fourier analysis in estimating the time period of meal consumption. Though Fourier Transform is a basic engineering tool to analyze signals, there have been a lot of studies about the applicability and clinical significance of the same. The harmonic decomposition performed by Fourier Transform can be used in predicting the glucose variability in absolute terms. The frequency decompositions done by FFT can help in detecting the maximum significant frequency from the given data. The idea is that we could move back and forth between the period of the wave and the frequency by using the Fourier Transform. This is extremely helpful in extracting patterns that may look like random noise. All the glucose level data have been sampled at a rate of 1 per 5 minutes for a total time period of 2.5 hours. When FFT is applied to a time series, it gives the amplitudes of the highest frequencies, which corresponds to the absolute values of the CGM data, from which the meal intake can be measured.











Feature matrix row: 90

Feature matrix row: 105

Feature matrix row: 120

Feature matrix row: 135

Feature matrix row: 150

Feature matrix row: 165

Figure 1: Fast Fourier Transform plots

2.2. Discrete Wavelet Transform

The Discrete Wavelet Transform analyses signal whose frequency varies over time. The time-domain signal is decomposed by contractions and expansions of the wavelet function by applying small windows at higher frequencies and large windows at lower frequencies. When I considered the CGM data, I found that peaks occurred at different times in 2.5-hour interval. To analyze that, I used the Discrete Wavelet Transform. The wavelet transform contains information on both the time-frequency and location of a signal and hence wavelet transform can give a better result for meal detection.

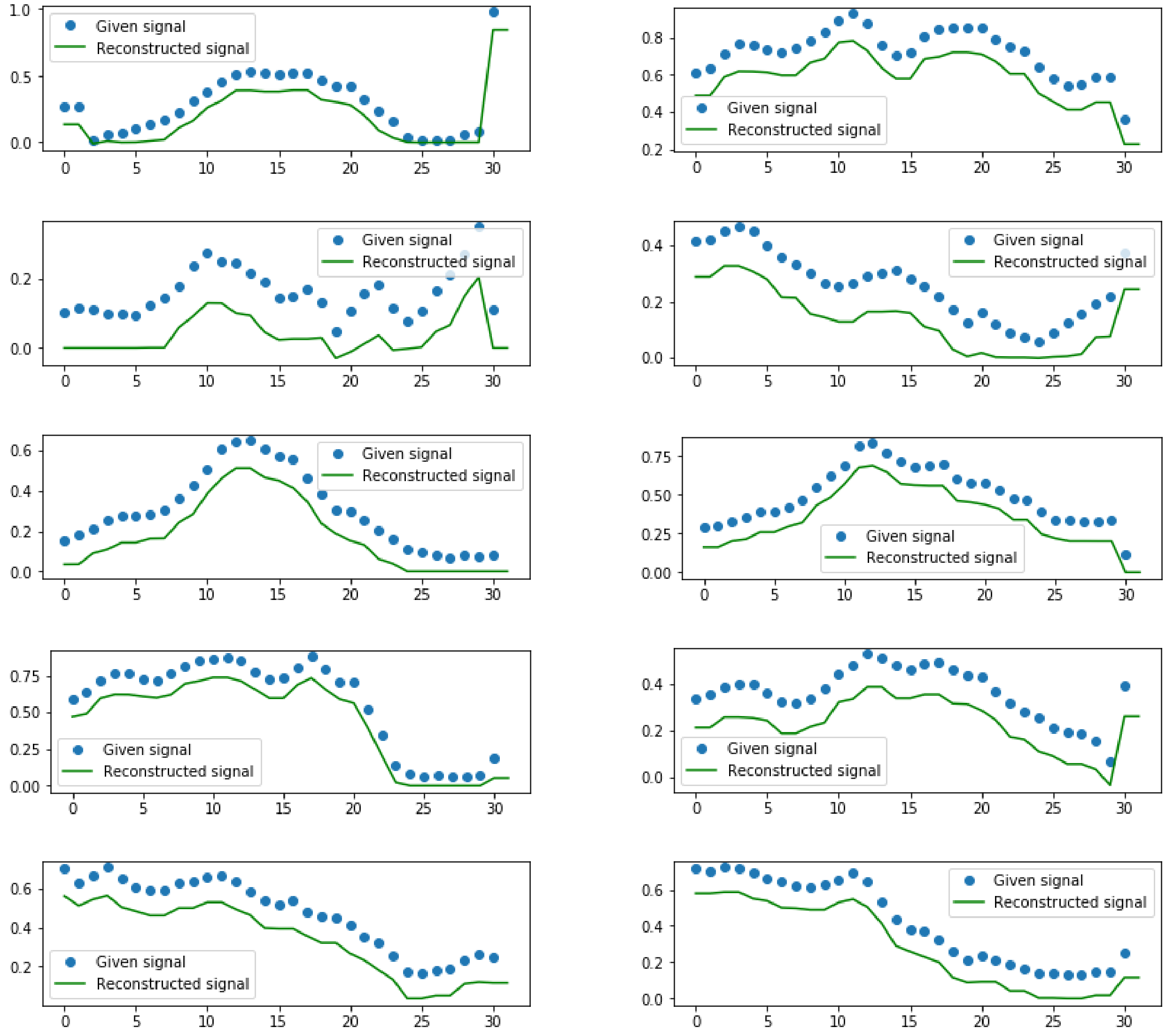


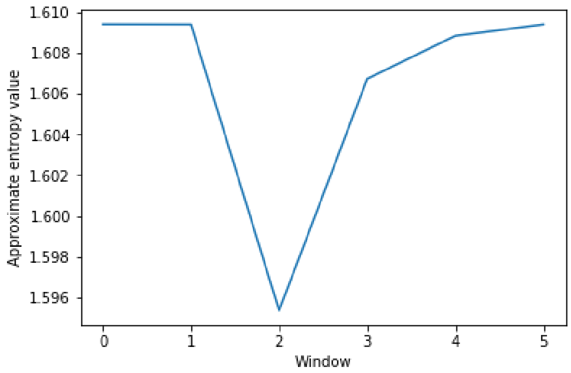
Figure 2: Discrete Wavelet Transform plots

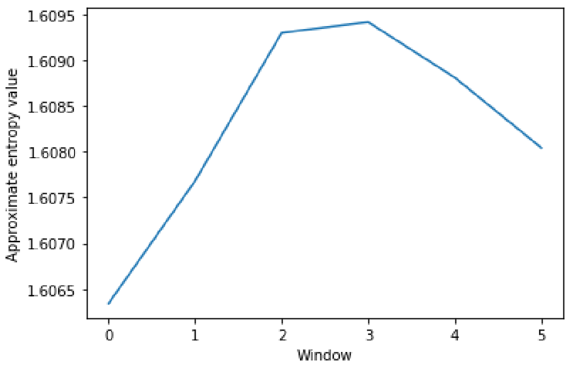
2.3. Coefficient of Variation

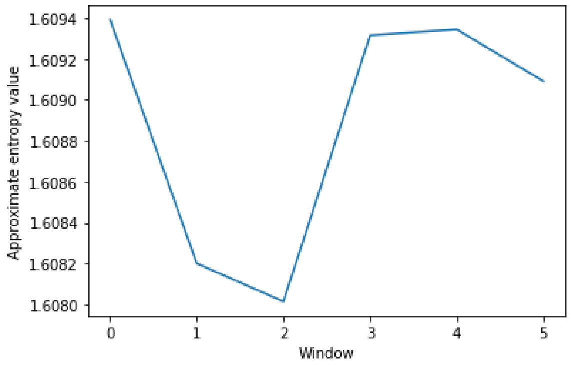
The Coefficient of Variance is a statistical measure that simplifies the interpretation of glucose level variability across patients with different means unlike standard deviation, which is proportional to the mean glucose level, i.e., someone with higher glucose will have a higher standard deviation. The Coefficient of Variance helps in normalizing the glucose level variability, paving the way to use a single variability measure that can be applied to patients with varying mean glucose levels. This feature gives us a range of values between which we can expect a patient to take his lunch. Coefficient of Variation is a single parameter for glucose level variability and hence cannot be plotted.

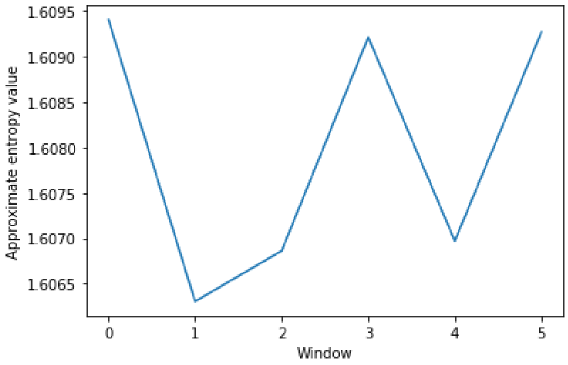
2.4. Windowed Entropy

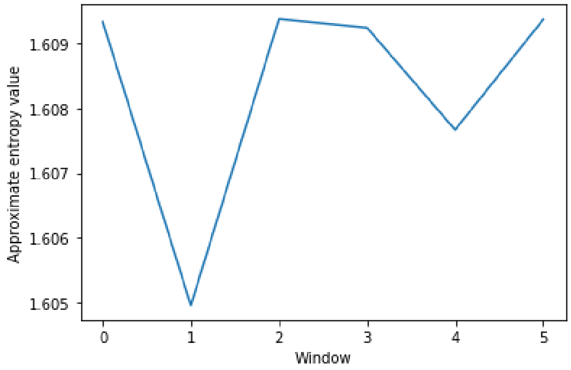
In statistics, approximate entropy is a technique used to quantify the amount of regularity and the unpredictability of fluctuations over time-series data. Measures such as mean and variance only tell us about the distribution of data in general. Hence, we need a measure like entropy to measure the randomness which is helpful in forecasting based on time-series data. Applying entropy over a series will give a valued measure of the randomness. I adopted the same and split the entire series into different windows and calculated the entropy for every window. The window which shows the maximum randomness corresponds to the period of meal intake. With a window size of 5, I calculated that the lowest entropy value correlated with the window in which the patient took a meal. The results can be seen below.

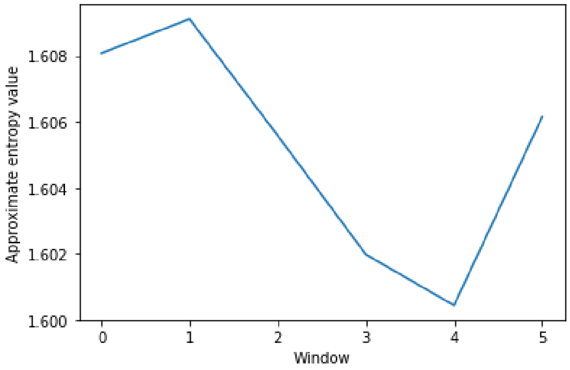












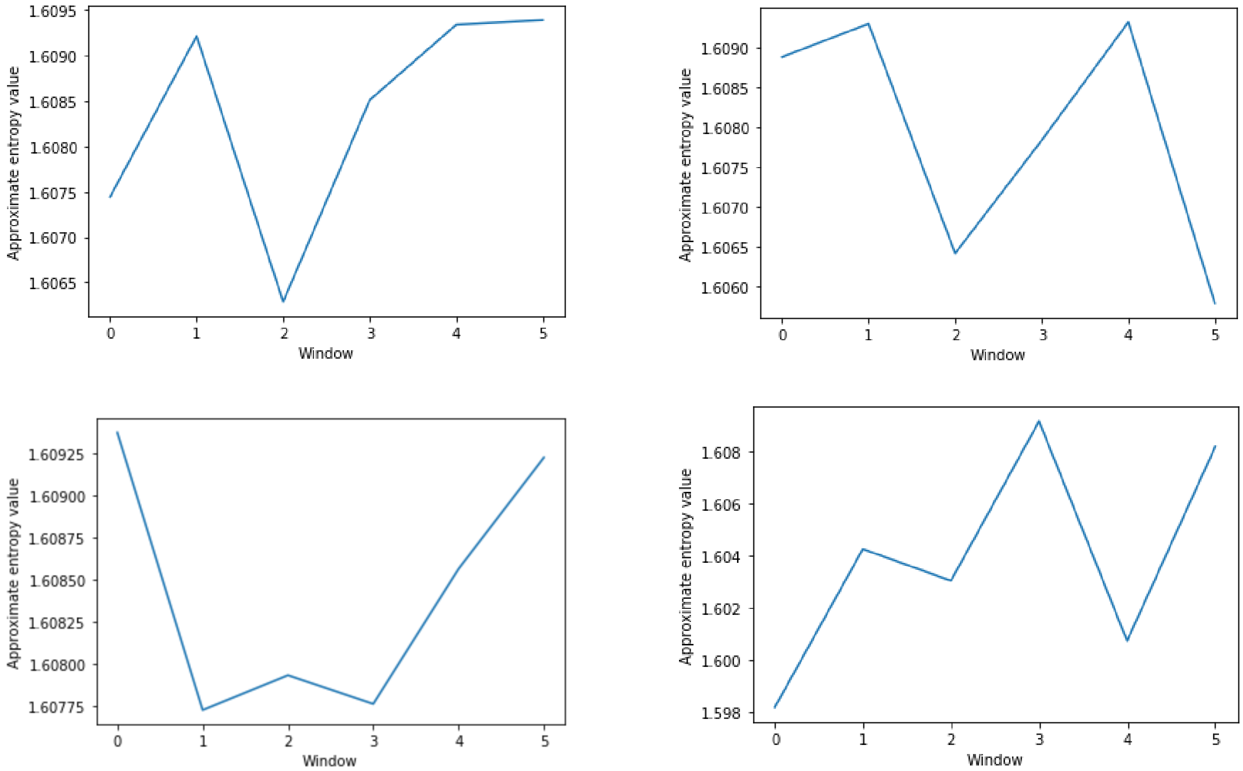
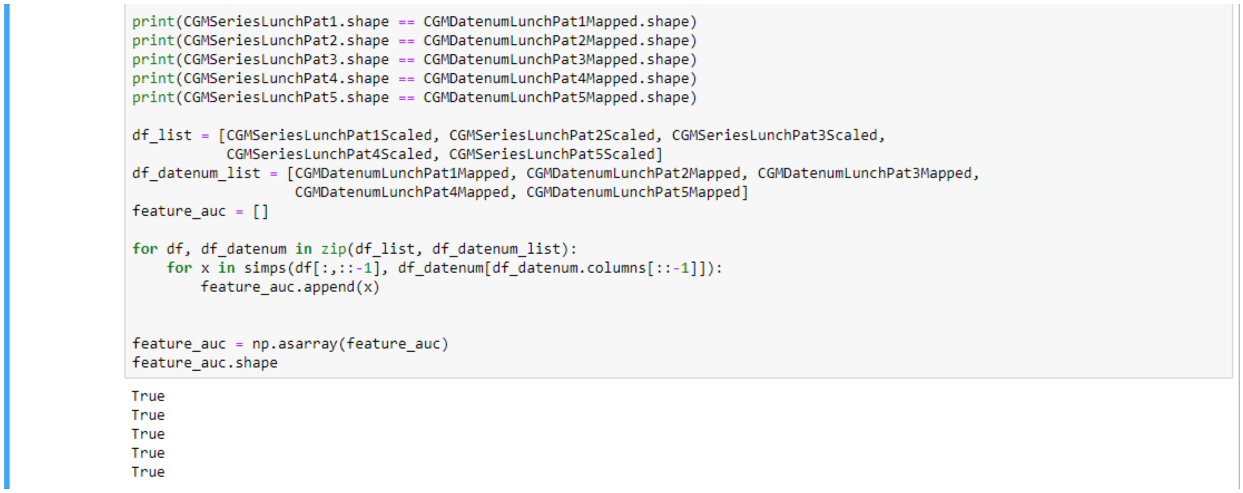


Figure 3: Windowed Entropy plots

2.5. Area Under Curve

Area Under Curve is a performance measurement for classification problems at various threshold settings. It can be used as a single figure for measuring performance as well as classifying the data as to whether meal has been taken or not. Data with glucose level peak has an AUC that is different from the data without glucose peak i.e. when the graph is stationary. The area can also be used to derive true-positive rate and false-positive rate which can be used to evaluate the model.



```
print(CGMSeriesLunchPat1.shape == CGMDatenumLunchPat1Mapped.shape)
print(CGMSeriesLunchPat2.shape == CGMDatenumLunchPat2Mapped.shape)
print(CGMSeriesLunchPat3.shape == CGMDatenumLunchPat3Mapped.shape)
print(CGMSeriesLunchPat4.shape == CGMDatenumLunchPat4Mapped.shape)
print(CGMSeriesLunchPat5.shape == CGMDatenumLunchPat5Mapped.shape)

df_list = [CGMSeriesLunchPat1Scaled, CGMSeriesLunchPat2Scaled, CGMSeriesLunchPat3Scaled,
           CGMSeriesLunchPat4Scaled, CGMSeriesLunchPat5Scaled]
df_datenum_list = [CGMDatenumLunchPat1Mapped, CGMDatenumLunchPat2Mapped, CGMDatenumLunchPat3Mapped,
                   CGMDatenumLunchPat4Mapped, CGMDatenumLunchPat5Mapped]
feature_auc = []

for df, df_datenum in zip(df_list, df_datenum_list):
    for x in simps(df[:,::-1], df_datenum[df_datenum.columns[::-1]]):
        feature_auc.append(x)


feature_auc = np.asarray(feature_auc)
feature_auc.shape
```

```
True
True
True
True
True
```

Figure 4: Area Under Curve all true positives

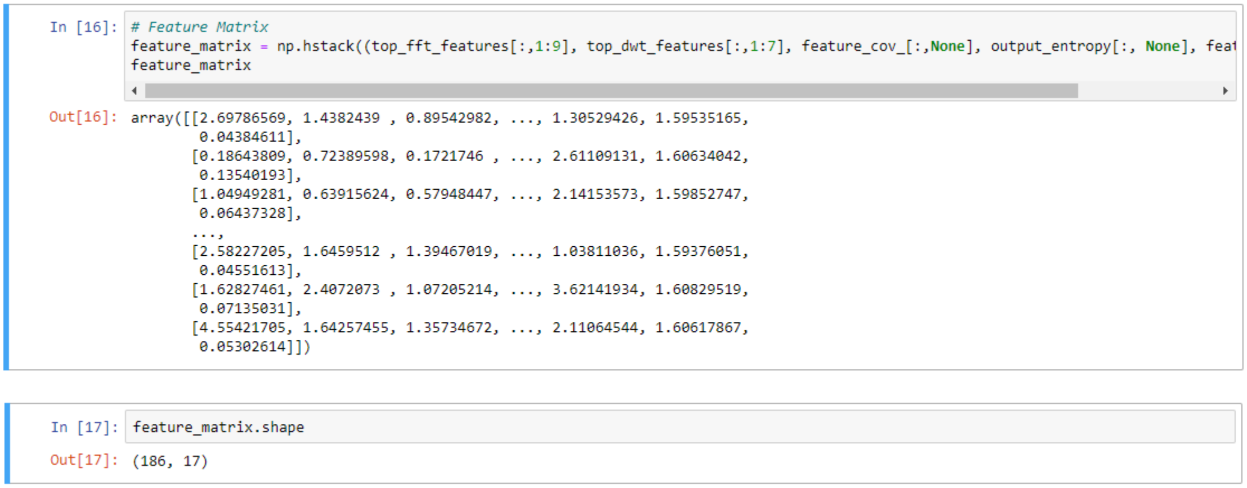
2.6. Intuition about each feature and their outcomes

I performed these five different feature extraction techniques namely, Fast Fourier Transform, Discrete Wavelet Transform, Coefficient of Variation, Windowed Entropy and Area Under Curve. My intuition was that all these features would provide a proper distinction amongst the CGM time-series data. And as we observe all the outputs, we can conclude that we have maximum distinction in Fast Fourier Transform (FFT) and Discrete Wavelet Transform (DWT) out of all the feature extraction techniques.

3. Feature Matrix and Dimensionality Reduction

3.1. Arranging the feature matrix

Principal Component Analysis (PCA) takes in a matrix of dimensions ‘o*f’ where ‘o’ corresponds to the number of data samples and ‘f’ corresponds to the number of features or the combined length of the feature vector. I constructed a feature matrix of dimensions 186*17, the total number of samples was 186 and the combined length of all five features amounted to 17. This has a higher dimension; hence it is necessary to reduce the dimensions for better matching and prediction.



```
In [16]: # Feature Matrix
         feature_matrix = np.hstack((top_fft_features[:,1:9], top_dwt_features[:,1:7], feature_cov_[:,None], output_entropy[:, None], feat
         feature_matrix

Out[16]: array([[2.69786569, 1.4382439 , 0.89542982, ..., 1.30529426, 1.59535165,
                 0.04384611],
                [0.18643809, 0.72389598, 0.1721746 , ..., 2.61109131, 1.60634042,
                 0.13540193],
                [1.04949281, 0.63915624, 0.57948447, ..., 2.14153573, 1.59852747,
                 0.06437328],
                ...,
                [2.58227205, 1.6459512 , 1.39467019, ..., 1.03811036, 1.59376051,
                 0.04551613],
                [1.62827461, 2.4072073 , 1.07205214, ..., 3.62141934, 1.60829519,
                 0.07135031],
                [4.55421705, 1.64257455, 1.35734672, ..., 2.11064544, 1.60617867,
                 0.05302614]])
```

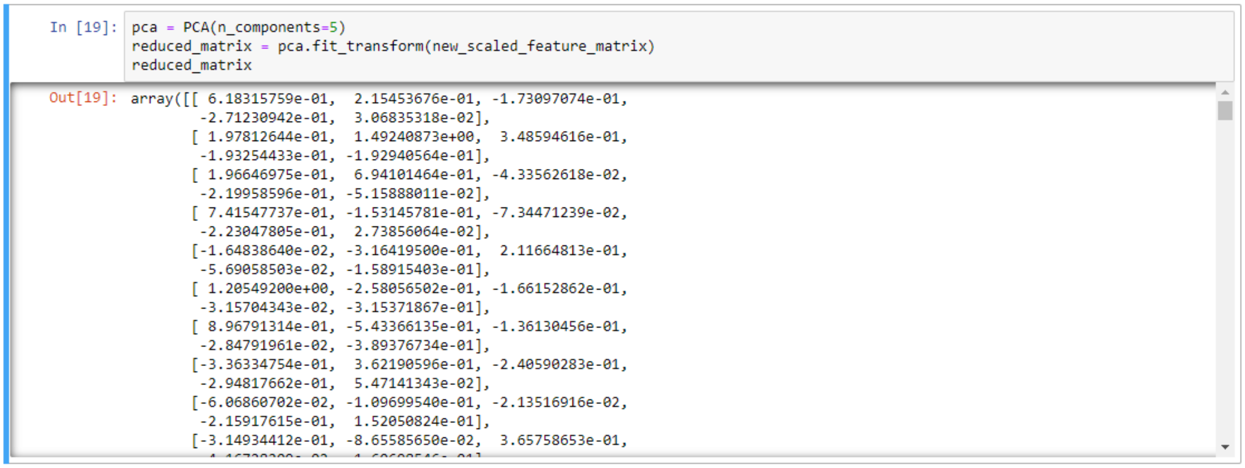
```
In [17]: feature_matrix.shape

Out[17]: (186, 17)
```

Figure 5: Feature Matrix

3.2. Execution of PCA

Principal Component Analysis is a method that is used to prevent the curse of dimensionality by extracting only features that show high variance in the data set. Highly correlated variables are abstracted from the feature vector to form a significant set of variables known as principal components which define the variance of the dataset. PCA reduces attribute space from a larger number of variables to a smaller number of factors and as such is a ‘non-dependent’ procedure (i.e., it does not assume a dependent variable is specified). PCA gets a data matrix of dimensions ‘o*f’ (‘o’ is the number of data objects, ‘f’ is the number of features or attributes corresponding to each object) as input. It then computes the covariance matrix of dimensions ‘f*f’, which tells how every feature correlates with each other. If two or more features are correlated, then they are redundant. PCA tries to decompose this matrix using Singular valued Decomposition or Eigen Decomposition. It computes the Eigenvalues and Eigenvectors for the covariance matrix, which forms the new ortho-normal basis and can be called as principal components or latent space. Eigenvalues are arranged in descending order of value and the top ‘n’ values and its corresponding vectors are chosen. They represent reduced principal dimensions. By multiplying the original data matrix with it, we project the data on to the latent space, which can conveniently represent the data with fewer dimensions and retaining maximum variance among them.



```
In [19]: pca = PCA(n_components=5)
         reduced_matrix = pca.fit_transform(new_scaled_feature_matrix)
         reduced_matrix

Out[19]: array([[ 6.18315759e-01,  2.15453676e-01, -1.73097074e-01,
                 -2.71230942e-01,  3.06835318e-02],
                [ 1.97812644e-01,  1.49240873e+00,  3.48594616e-01,
                 -1.93254433e-01, -1.92940564e-01],
                [ 1.96646975e-01,  6.94101464e-01, -4.33562618e-02,
                 -2.19958596e-01, -5.15888011e-02],
                [ 7.41547737e-01, -1.53145781e-01, -7.34471239e-02,
                 -2.23047805e-01,  2.73856064e-02],
                [-1.64838640e-02, -3.16419500e-01,  2.11664813e-01,
                 -5.69058503e-02, -1.58915403e-01],
                [ 1.20549200e+00, -2.58056502e-01, -1.66152862e-01,
                 -3.15704343e-02, -3.15371867e-01],
                [ 8.96791314e-01, -5.43366135e-01, -1.36130456e-01,
                 -2.84791961e-02, -3.89376734e-01],
                [-3.36334754e-01,  3.62190596e-01, -2.40590283e-01,
                 -2.94817662e-01,  5.47141343e-02],
                [-6.06860702e-02, -1.09699540e-01, -2.13516916e-02,
                 -2.15917615e-01,  1.52050824e-01],
                [-3.14934412e-01, -8.65585650e-02,  3.65758653e-01,
```

Figure 6: PCA Reduced Matrix

3.3. Results of PCA

I chose the top 5 eigenvalues from the PCA and calculated their ratio and plotted them on a graph as shown below. It shows that the top 5 latent features obtained contributes to approximately 89% of the total variance of the data. Hence, it is enough to represent the data as a combination of 5 features rather than 17.

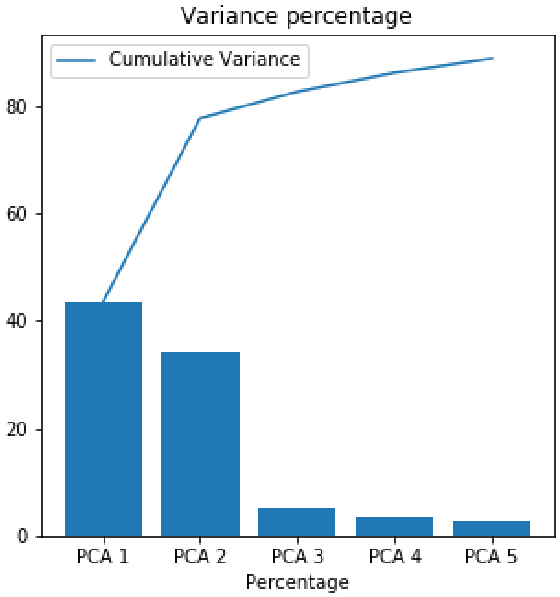


Figure 7: Variance percentage

3.4. Explanation for choosing the top 5 features

The weightage of each feature after PCA can be visualized from the below histograms. As we can see, the histograms show 5 different peak bars in each plot that corresponds to the maximum weighted feature that can be used as a feature. In graph 1 below, the 6th feature of FFT has maximum weightage and can be used as an important feature for meal detection. Likewise, in graph 2, the 5th feature of DWT tops. In graph 3, AUC is the maximum. In graph 4, the 6th feature of DWT can be used. Finally, in graph 5, the 1st feature of FFT is the maximum.

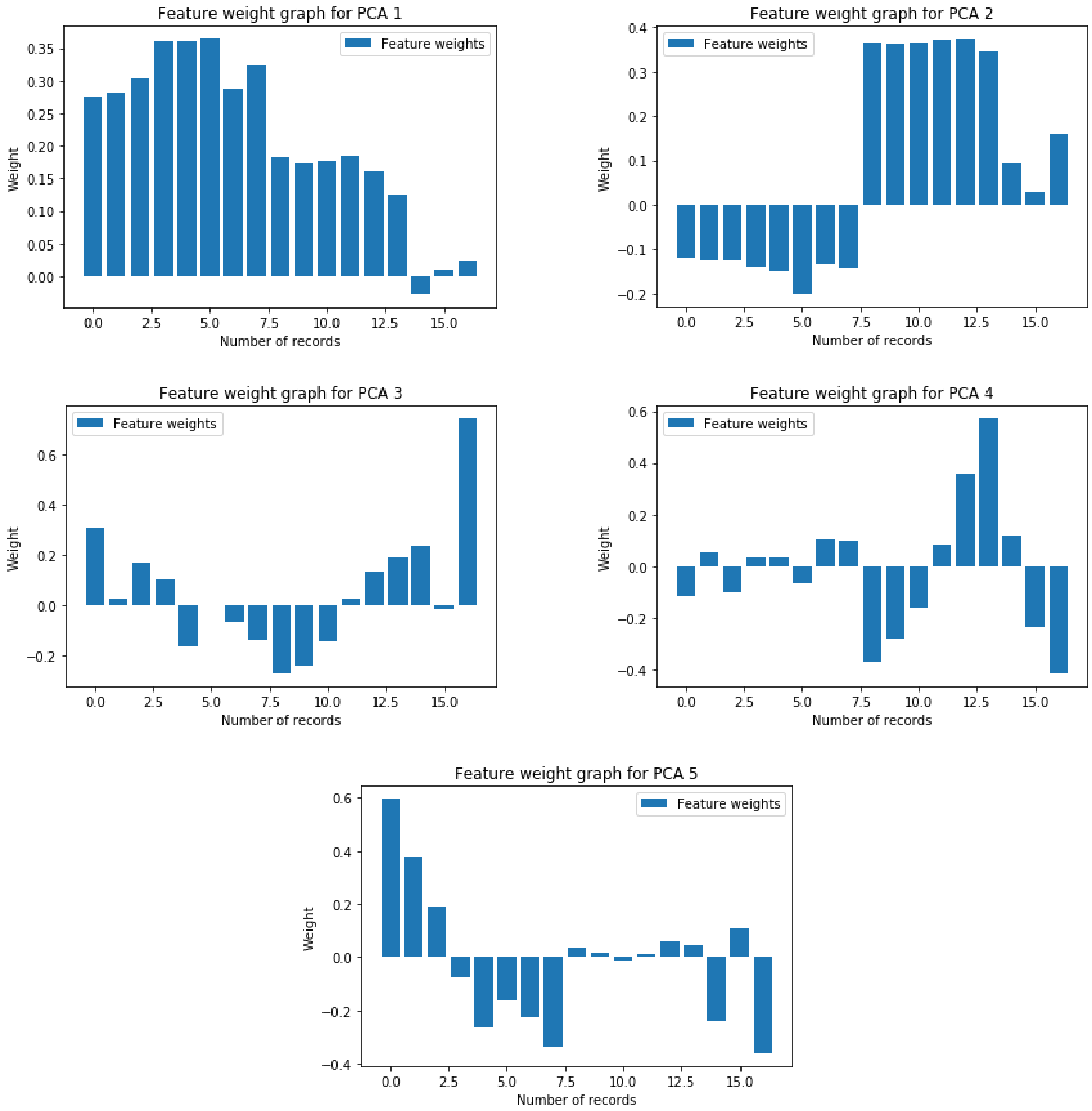


Figure 8: Feature weight graphs for PCA top 5 features

The principal components contribute to the bulk of variance in the entire dataset. I tried to visualize it by plotting the spread of data points along a principal component on the x-axis against all other principal components for all patients. As seen in the below graphs, the principal components ‘PCA 1’ and ‘PCA 2’ have the maximum data spread, indicating high variance and high discrimination power. Though the variance decreases as we move down to other principal components, the distribution of data among the five principal components is varied enough to have higher discrimination power and hence can result in better analysis and prediction without errors.

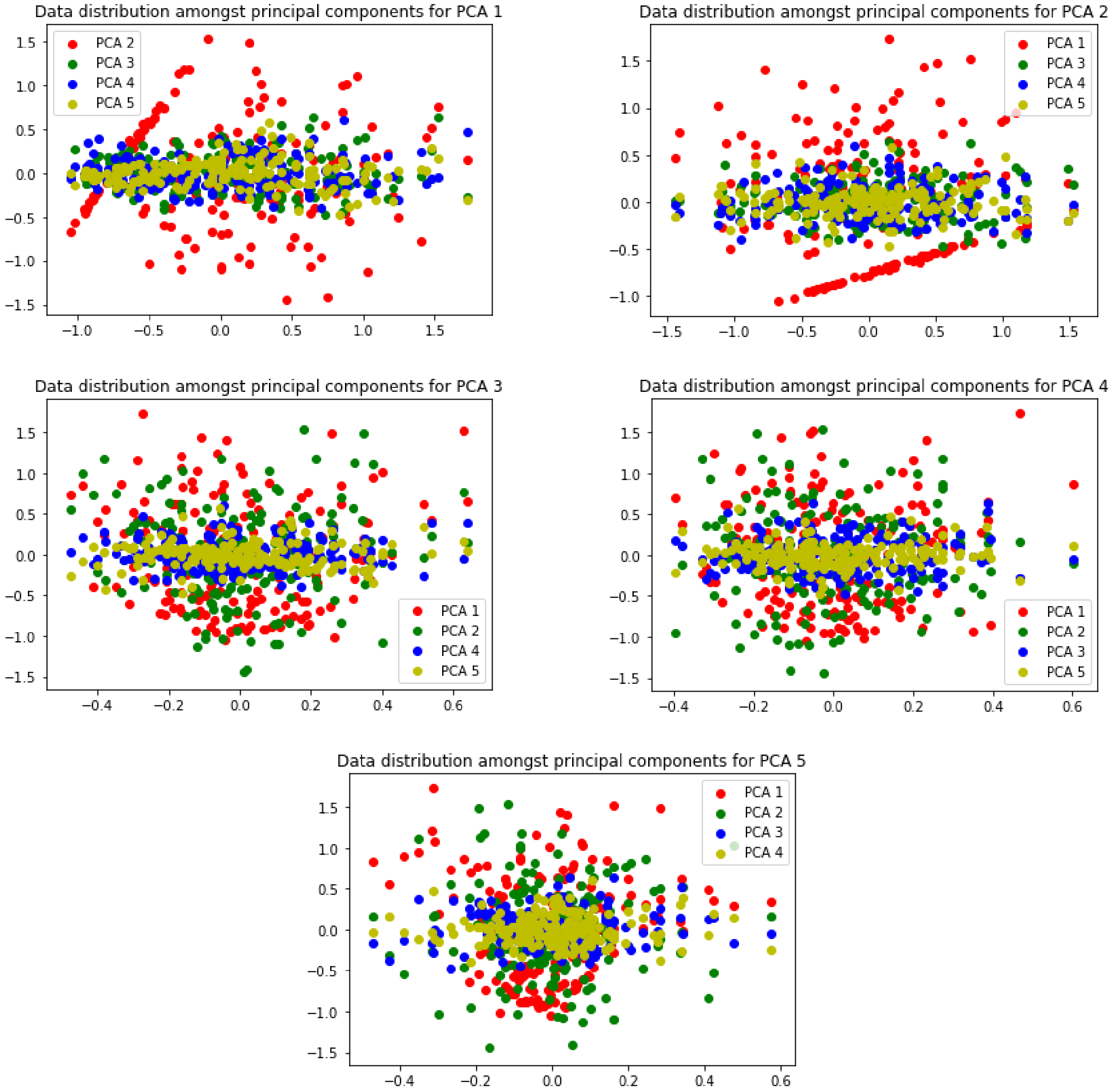


Figure 9: Data distribution amongst principal components for PCA top 5 features

References:

1. The Comprehensive Glucose Pentagon: A Glucose-Centric Composite Metric for Assessing Glycemic Control in Persons with Diabetes. Robert A. Vigersky, John Shin, Boyi Jiang, Thorsten Siegmund, Chantal McMahon, and Andreas Thomas.
2. Feature selection: stability, algorithms, and evaluation, Doctoral thesis. Pavel Krizek.
3. MinMaxScalar – Python, https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
4. Fast Fourier Transform (FFT) – Python, https://docs.scipy.org/doc/numpy/reference/generated/numpy.fft.fft.html
5. Discrete Wavelet Transform (DWT) – Python, https://pywavelets.readthedocs.io/en/latest/ref/dwt-discrete-wavelet-transform.html
6. Principal Component Analysis (PCA) – Python, https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
7. Example Report Group20_Assignment1.zip, https://canvas.asu.edu/courses/46639/modules