CGM Data Peak Detection Report

DATA MINING

CSE 572: Fall 2019

Submitted to:

Professor Ayan Banerjee
Ira A. Fulton School of Engineering
Arizona State University

Submitted by:

Arijit Panda (apanda7@asu.edu)
Sunil Samal (sksamal1@asu.edu)
Amarnath Tadigadapa (atadigad@asu.edu)
Asmi Pattnaik (apattnai@asu.edu)

October 8, 2019

1. Introduction

Continuous glucose monitoring (CGM) as the name suggests is a method of continuously monitoring glucose levels in the interstitial fluid as a basis for improving metabolic control. This is helpful in reducing hyperglycemia and minimizing the occurrence of low glucose values(including symptomatic hypoglycemia)[5]. Treatments and therapies based on CGM devices associated with an insulin pump technology are rapidly becoming more common. For these methods to be truly effective, the key part is a decision support system or an artificial pancreas that can predict the glucose level in a relatively long period of time[6].

2. Team Members

The team members are:

Arijit Panda (apanda7@asu.edu)
Sunil Samal (sksamal1@asu.edu)
Amarnath Tadigadapa (atadigad@asu.edu)
Asmi Pattnaik (apattnai@asu.edu)

3. Project Description

Data was collected from a continuous glucose monitor(CGM) and provided to us by the Professor. Data given includes:

1. The first cell array has tissue glucose levels every 5 mins for 2.5 hrs during a lunch meal. The data starts from 30 mins before meal intake a continues up to 2 hrs after the start of meal consumption
2. The second cell array has timestamps of each time series in the first cell array
3. The third cell array has insulin basal infusion input time series at different times during the 2.5 hr time interval.
4. The fourth cell array has timestamps for each basal or bolus insulin delivery time series.
5. The fifth cell array has an insulin bolus infusion input time series at different times during the 2.5 hr time interval.

Phase 1 of the project expects us to find features, analyze and discuss if the features selected are appropriate and apply PCA to the feature matrix to get the top 5 features.

4. Project Task: Feature Extraction

As a part of this task, we are extracting four different types of time series features from the CGM glucose level array and CGM timestamp cell array. Before carrying on with the feature extraction, we had to perform some pre-processing of the data. Some data points have NaN (Not a Number) or were missing. To adjust for the missing data we filled them up with the average of the two adjacent glucose Levels. As we have assumed that there is not much difference in the glucose Levels at those timestamps.

The four feature extraction techniques that we have used are:

1. Auto-Correlation
2. Power Spectral Density(PSD)
3. Polyfit Regression
4. Statistical Methods and Velocity Calculations

4.1 **Auto-Correlation**

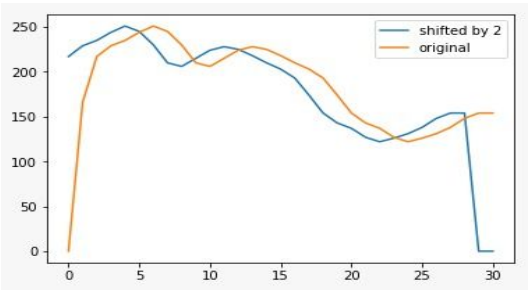
The similarity between a time series and a delayed version of the same time series over consecutive time intervals is called **Autocorrelation** [2]. We can describe **Lag** as the difference between time periods of the signal. A lag value of 1(*i.e*, n= = 1) autocorrelation is the correlation between signals one time period apart. In general a lag ***n*** autocorrelation is the correlation between signals that are *n* time periods apart.

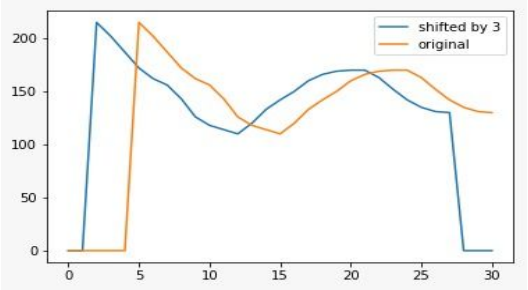
Based on the observed data we have used multiples values for the lag and computed the correlation between the data. That would help to get the peak values and also retrieve the time interval taken by glucose levels to reach the peak value from the minimum.

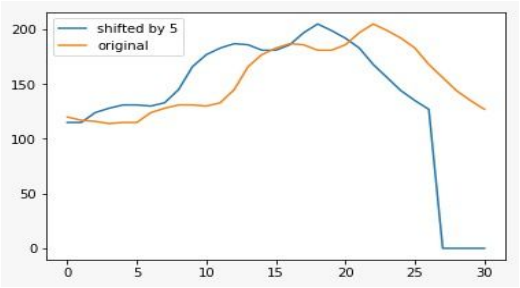
We have taken lagged forms (2,3,4,5,6,8) of the original series in order to take similarities in different time intervals and ensured that we capture the maximum dissimilar points in the signal.

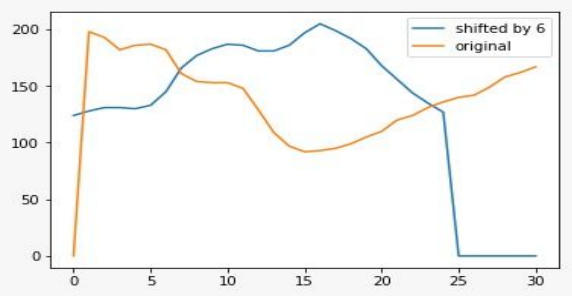
Here we can use autocorrelation values to know the relationship between past glucose levels of a person to the future glucose levels of that person. For example, if we know that a glucose level has a historically high positive autocorrelation value then we might reasonably expect the movements (the leading time series) to match those of the lagging time series and to move upward. This can help in identifying a trend in the glucose levels series data. Here we have utilized (panda) library in Python and the corresponding function (pd.Series & autocorr) to calculate the different lags.

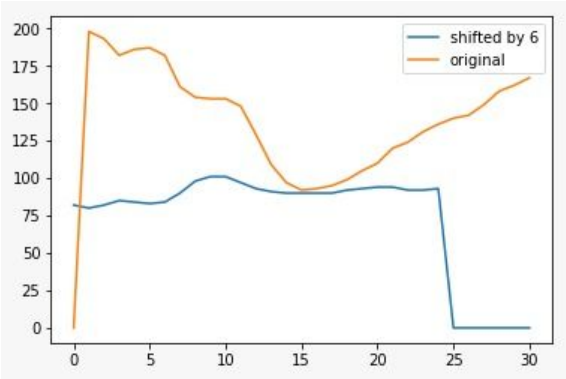
Below are correlation plots for different data points with different lagged values.

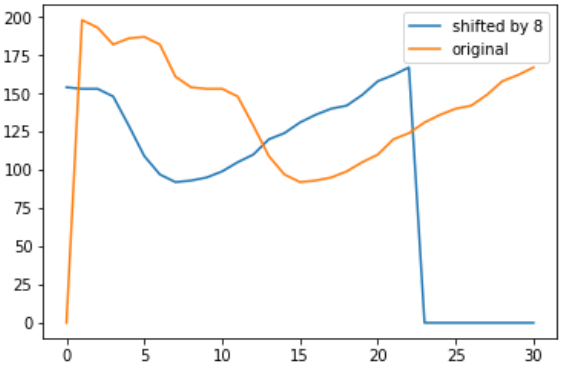










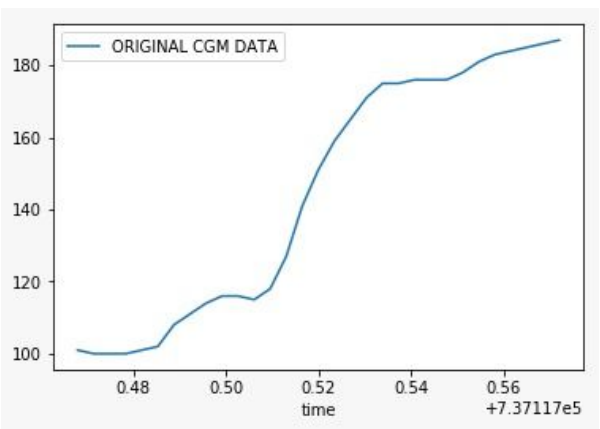


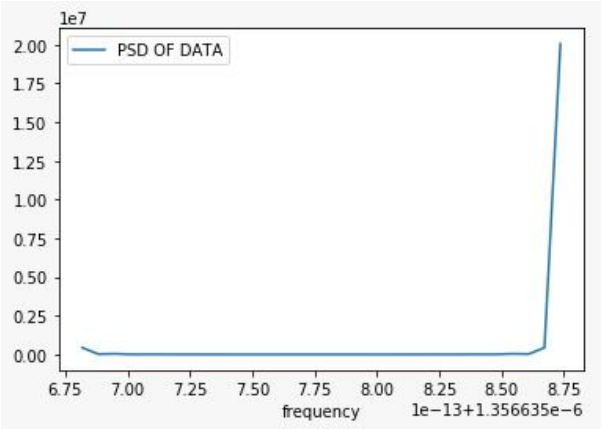
4.2 **Power Spectral Density**

Power Spectral Density gives the power of the variation as a function of frequency. It essentially shows the frequency variation that are both strong and weak. The unit of Power Spectral Density is energy or variance per frequency[1].

We have used Python's numpy function(np.fft.fft(data)) to find the Discrete Fourier Transform sample frequencies. np.abs(np.fft.fft(data))**2 was then used to calculate the power spectrum.

Using this technique we transform the CGM signals to the frequency domain using the Fast Fourier Transform (FFT) and mining the following spectral parameter: The maximum Power Spectral Density(PSD) and the frequency where it is located.





PSD describes how the power of the signal is distributed over the signal data and also helps us in understanding the trend in the data signal.

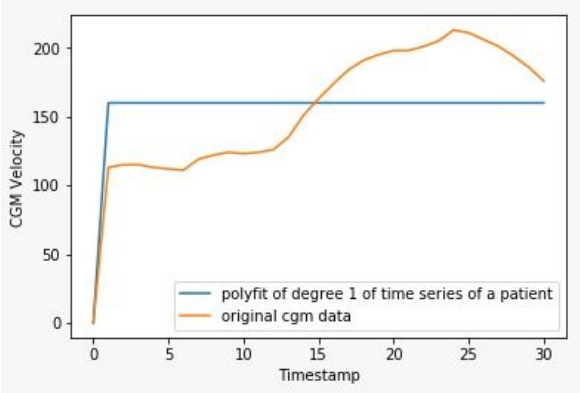
We have successfully mined the highest power spectral density along with the frequency where it is located. This will help us identify the peak glucose level and its corresponding position it has occurred. As there could be multiple peaks in the glucose levels we have picked top 8 peak values as our features that are generated using PSD.

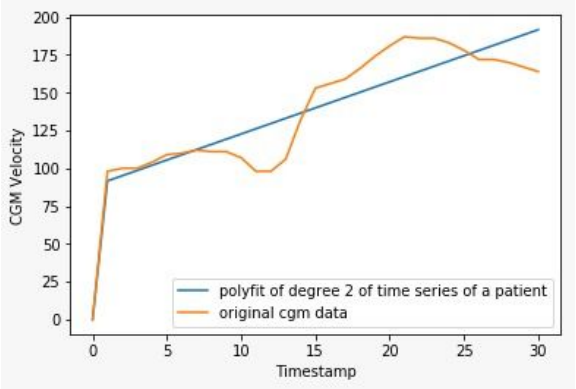
4.3 **Polyfit Regression**

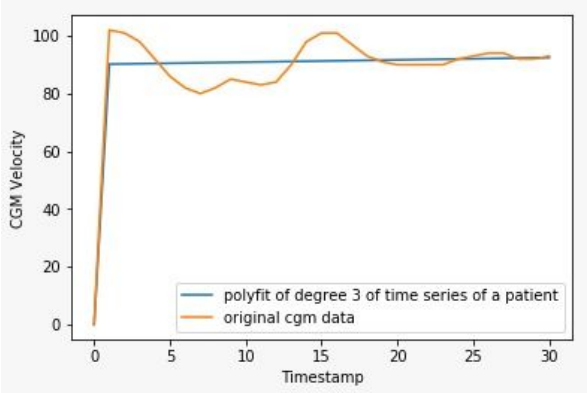
Polyfit regression is one of the non-linear regression methods available to fit data properly in cases where the data are not linear in nature(i.e we can differentiate between data points with a straight line) and can be used to study trends and variations in data. A nth degree polyfit regression can be represented as:

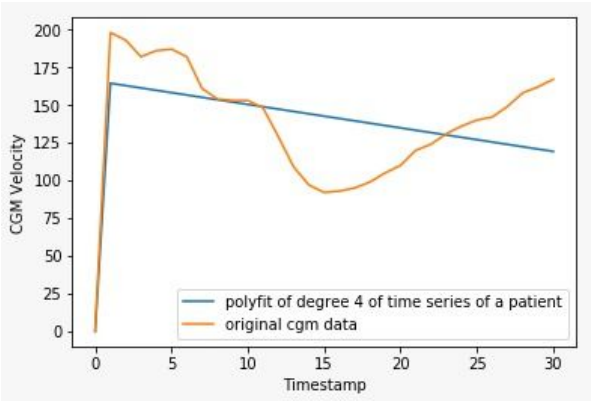
$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \varepsilon.$$

Polyfit regression employs the least-squares method to fit the data. For our case we can use polyfit regression to fit the data and study the trend in glucose levels. We can use the resulting curve to estimate the future glucose levels and in calculating the rate of change of glucose levels. We have used numpy library and polyfit function(numpy.polyfit) to derive the polynomial coefficients. Based on the signals of different data points we have implemented polyfit using multiple degrees such as first, second, third and fourth-degree polynomial fits.









4.4 Statistical and Velocity Calculations

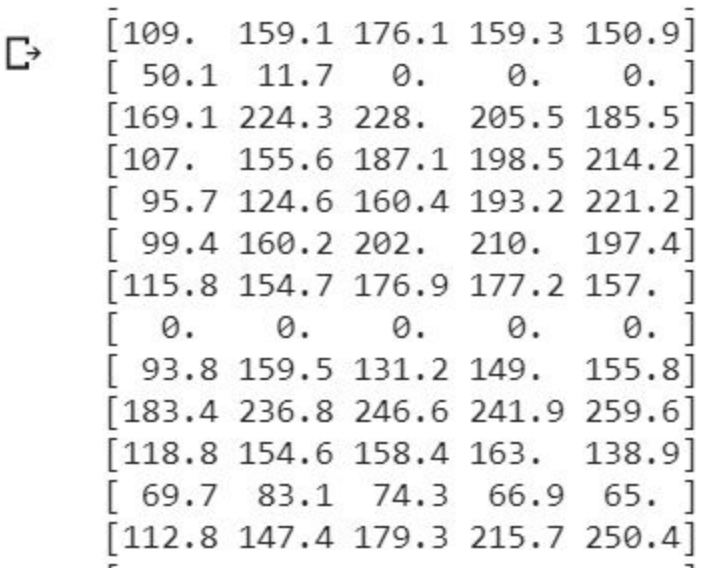
For statistical feature extraction of features, we have applied the following technique:

1. Windowed Mean
2. Windowed Variance
3. The average time taken for maximum rate of change of Glucose level
4. Average time

4.4.1 Windowed Mean

Mean is a good feature considering that all the attributes are taken into account. As one of the attributes can have extremely large or small values, mean can be biased towards that specific attribute. Basically it means that mean can be easily prone to noise. To avoid this particular issue and still take the feature we have considered windowed mean. It is a calculation of a series of averages of different subsets of the full data set. Here we have taken 50 minutes windows in 2.5 hours of time period. So there will be five means as we are sliding the window in 25 minutes (ist mean : (0,50)minutes, 2nd mean: (25,75)minutes , 3rd mean: (50,100)minutes, 4th mean : (75,125)minutes, 5th mean : (100,150)) . We used Numpy library and

mean function to compute mean. For some rows of the CSV file mean values are shown below.



```
[109.  159.1 176.1 159.3 150.9]
[ 50.1  11.7   0.    0.    0. ]
[169.1 224.3 228.  205.5 185.5]
[107.  155.6 187.1 198.5 214.2]
[ 95.7 124.6 160.4 193.2 221.2]
[ 99.4 160.2 202.  210.  197.4]
[115.8 154.7 176.9 177.2 157. ]
[  0.    0.    0.    0.    0. ]
[ 93.8 159.5 131.2 149.  155.8]
[183.4 236.8 246.6 241.9 259.6]
[118.8 154.6 158.4 163.  138.9]
[ 69.7  83.1  74.3  66.9  65. ]
[112.8 147.4 179.3 215.7 250.4]
```

In the second row 3 means are zero out of the five means. If we would have taken mean for the whole row the value would have much less than the other two means. So our intuition was correct regarding mean picking the noise.

4.4.2 **Windowed Variance**

Variance measures how far attribute values are spread out from their mean value. We want to select the features which have highly contrasting values and ignore the features which have comparatively similar values. Variance plays a big role in selecting these features. As mean is prone to noise, variance will be affected if we take the whole average and accordingly calculate the variance. In order to tackle this concern we have taken windowed variance which is basically calculated on the windowed mean. Five Windowed variance are computed in a similar way like mean and then added to the feature list. We used Numpy library and var function to compute mean. Here the same rows(as shown in Mean) of the file have been shown for mean and variance.

windowed_mean_1	windowed_mean_2	windowed_mean_3	windowed_mean_4	windowed_mean_5	windowed_var_1	windowed_var_2	windowed_var_3	windowed_var_4	windowed_var_5
109	159.1	176.1	159.3	150.9	1190.4	515.89	69.29	205.41	33.49
50.1	11.7	0	0	0	2563.89	1232.01	0	0	0
169.1	224.3	228	205.5	185.5	2153.89	303.01	234.2	228.45	107.25
107	155.6	187.1	198.5	214.2	1532.2	885.84	78.89	46.05	135.56
95.7	124.6	160.4	193.2	221.2	178.01	349.64	534.04	330.96	305.56
99.4	160.2	202	210	197.4	1551.04	1769.96	36.2	51.6	350.04
115.8	154.7	176.9	177.2	157	1709.76	250.81	87.09	85.36	207.6
0	0	0	0	0	0	0	0	0	0
93.8	159.5	131.2	149	155.8	8984.16	1100.45	229.96	436	200.36
183.4	236.8	246.6	241.9	259.6	1599.84	420.76	80.04	76.09	275.24
118.8	154.6	158.4	163	138.9	2028.96	83.44	57.64	274	1106.49
69.7	83.1	74.3	66.9	65	619.01	85.49	139.81	4.89	13.4
112.8	147.4	179.3	215.7	250.4	1502.56	333.64	393.21	514.01	289.44

In the 2nd row, if we would have taken variance for the whole row, the value would have been significantly different than the other two variance. So our intuition was

correct regarding mean picking the noise and in turn the variance getting affected by that.

4.4.3 Rolling Velocity

Velocity describes the steepness of the curve. It is the rate of change of glucose level. In this feature, rolling velocity, we take a rolling window of twenty-five minutes (5 timestamps which are each 5 mins apart) and calculate the rate at which the glucose level increases. This gives an overall trend of the signal which can help us in predicting peak glucose levels. To calculate this, we used the following python code:

```
rolling_vel= glucose_level_data.rolling(window,axis=1).apply(lambda x: (x[-1]-x[-0])/(window*5))
```

Where window size is 5.

4.4.4 Average time taken for the maximum rate of change of Glucose Level

In this feature, we are calculating the average time taken for the maximum increase in glucose levels. This can help us in determining the approximate time that is required to reach the peak value of glucose level.

4.4.5 Average time taken to reach the peak value from the time of meal

This gives us the average time taken to reach the peak value after the meal has been detected. This can act as a feature to understand the average time that is required to reach the peak value which can be utilized to get the peak based on mealtime.

5. Creation of Feature Matrix

Feature Matrix was created by appending all the features extracted using the above mentioned four types of features. The dimension of the feature matrix is 150 X 44 which in other words means it includes 150 time-series and 44 features.

Following screenshot shows a part of the feature matrix:

	Correlation_With_Lag_2	Correlation_With_Lag_3	Correlation_With_Lag_5	polyFit_4Degree_First_Coef	polyFit_4Degree_Sec_Coef	polyFit_4Degree_Third_Coef	polyFit_4Degree_Fourth_Coef	polyFit_4Degree_Five_Coef	pol
0	0.9850410135969762	0.9683448291133888	0.9203635990987081	1.0507407835857348e-15	3.8731650879191774e-10	6.188558511193578e-11	-210.50700688798904	-310382313.49680734	0.0
1	0.7978727971686111	0.7189856258815334	0.48533027425613245	3.3605037511302606e-16	8.258087551594355e-11	-6.087979063188789e-05	-134.64516341702753	5.48647149137986 55e-08	0.0
2	0.9266609650659191	0.8977082004282066	0.8161186892095506	1.3467039969440558e-15	3.3093755050486177e-10	-0.0002439724433163979 3	-539.5816659575527	2.060772218713413e-07	0.0
3	0.9743241732795597	0.9461354313579362	0.8641084774796741	8.461570016626468e-16	3.1190007430878 48e-10	5.572523419589694e-11	-169.51346744399945	-249935948.52801776	0.0
4	0.6962452331678701	0.6604965252838617	0.5408450094243585	2.391608188326162e-16	5.876996762335 91e-11	-4.332517850925556e-05	-95.81826227689757	-2.475602990012866e-08	0.0
5	0.9728390999182748	0.9425189594932741	0.8573343708497358	8.411647266947706e-16	3.1005190900186 89e-10	5.82204069462443e-11	-168.50034027236376	-248435772.55736035	0.0
6	0.8631118327475616	0.8188585444819719	0.704119200070983	7.438618645885788e-16	1.8277697010040286e-10	-0.0001347321833266965 8	-297.94975748405074	-1.3649270539530397e-07	0.0
7	0.9517701428321715	0.9330678516467886	0.880108565569334	1.3640753955242088e-15	3.3516271941651 51e-10	-0.0002470551772921 0915	-546.3284283013201	2.997486863583146e-07	0.0
8	0.9794141531487307	0.9578599942122125	0.9066437304713821	5.237166264516628e-16	1.9302038651654992e-10	5.3105771898951856e-11	-104.87605047043245	-154611977.26579663	0.0
9	0.7036195332972671	0.6149878232262375	0.3949351762083624	1.144217331164648e-16	2.8113961429702617e-11	-2.072301187473232 2e-05	-45.8256790428536	2.843598029068832 8e-08	0.0
10	0.8891066318680289	0.8571495327953522	0.7694729956106213	1.1393889648146068e-15	2.7995082091434 98e-10	-0.0002063537114818 5796	-456.3147415394073	-4.282124090833066e-08	0.0
11	0.9629099854734988	0.9388856774722651	0.8684079615522666	1.5856967106929318e-15	3.8960918665340077e-10	-0.0002871833336710883	-635.0542202818106	1.2311106761145064e-07	0.0
12	0.8934855452316813	0.8748533383576785	0.8032993967923192	8.553779292833776e-16	2.1016743370032065e-10	-0.0001549150168081 1278	-342.56533095953495	1.1508208494113865e-07	0.0
13	0.8793943946995979	0.8360803304586827	0.7422813473965427	8.507421355191496e-16	2.0902670568740447e-10	-0.0001540729291599 9238	-340.7004299213004	1.1240575738436797e-07	0.0
14	0.7602261430542322	0.7141564494663779	0.6120472738431988	2.955129080900096e-16	7.2603479872453 41e-11	-5.351292096721196e-05	-118.3265352660711	-1.7396129119009 33e-08	0.0
15	0.9803694421178278	0.9593794135872082	0.9056066163482814	6.975727839246316e-16	2.5707532913851 55e-10	5.4091209347541405e-11	-139.65669565714032	-205869791.87799507	0.0
16	0.8503503294452021	0.8266349970679506	0.7610930915947687	8.764207222648687e-16	2.1532366144428083e-10	-0.0001587053390971 6905	-350.9240764882482	3.4524625482341594e-07	0.0
17	0.8009196368677887	0.7440468650642716	0.5935486741807058	-1.4709703374432283e-15	-3.613946272266186e-10	0.0002663673694 7106	588.981030112371	2.596037730067546e-07	-0.0
18	0.9116762548095719	0.8767353351974921	0.804530868897019	9.523277665455085e-16	2.3397192705644536e-10	-0.0001724494947042 8264	-381.3131132694862	-1.3381637783853332e-07	0.0
19	0.8819418099508687	0.8237682436559636	0.6623320472590045	6.636350409492167e-16	1.6304380895791492e-10	-0.0001201710617631 5705	-265.71581942161725	2.435458076661306e-07	0.0
20	0.872967017409009	0.7456760104134835	0.47539003709017763	3.022990417004577e-16	1.1140441952668003e-10	5.667692877407986e-11	-60.519149007315775	-89211009.55601797	0.0
21	-0.13206881899056183	-0.19809227803255355	0.013078882159152755	1.6701972194175734e-17	4.1034038586822 45e-12	-3.024307376441672e-06	-6.687209729112736	-1.7981575772052914e-09	3.0
22	0.8980129732429702	0.787577414758079	0.5212927330977171	2.988607270112625e-16	1.1013657313012501e-10	5.782971095945347e-11	-59.82959223299205	-88193940.07726723	0.0
23	0.6953644578575735	0.5606168832932659	0.32691256318889217	3.1065555202226674e-16	7.6322069539229e-11	-5.625243930085258e-05	-124.38124158957923	2.141062045416533e-08	0.0

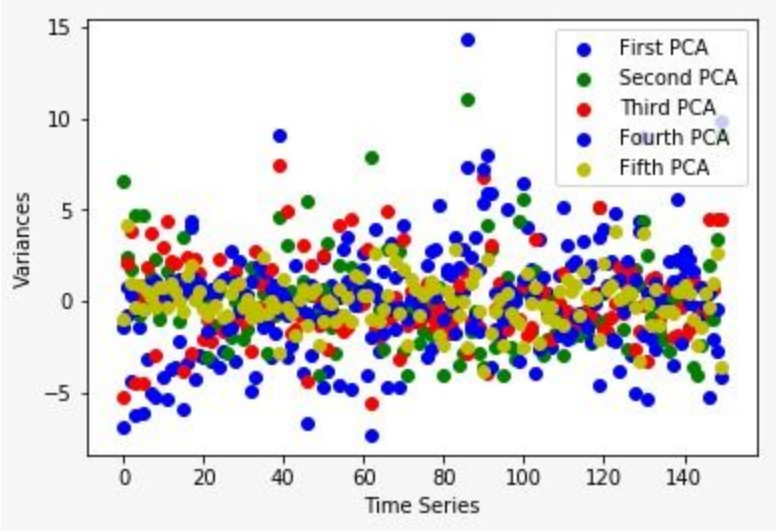
6. PCA

Principal Component Analysis or PCA acts as a tool for dimensionality-reduction. It is used to reduce a large set of features or variables into a small set that still contains most of the information of the original feature set or in other words, PCA is used to emphasize variation in data and bring out strong patterns in a dataset. Generally, the feature/covariance matrix and the top k latent features we want to figure out are provided as input to the PCA algorithm. It decomposes the feature matrix into 3 matrices which are 1.U:Left Factor Matrix, 2.S: Core Matrix(latent feature values stored in decreasing order), 3.VT: Right Factor Matrix. Here S will contain the square root of the eigenvalues in decreasing order along its diagonal. All the other elements in this matrix will be zero. According to the k input value to PCA, it will pick the top k largest values in the diagonal and change the order of all other matrices accordingly. PCA picks top k elements as these are the most important features as these new features have more variance than others. PCA picks the latent features in such a way that the variance in these features will be the largest and the features will be independent of each other. As eigenvectors are orthonormal to each other, they are independent of each other. That's why eigenvectors were used in PCA. After selecting the features we are normalizing the values to make sure that the magnitude of the attributes

The entire feature matrix (44 features) is fed and k as five is fed to PCA and the top 5 latent features are picked.

PCA Top 5 Features:

First Feature	Second Feature	Third feature	Fourth Feature	Fifth Feature
POLYFIT_4DEGREE_FOURTH_COEF	PSD_SEC_PEAK	POLYFIT_1DEGREE_SEC_COEF	CORRELATION_WITH_LAG_5	PSD_FIRST_PEAK

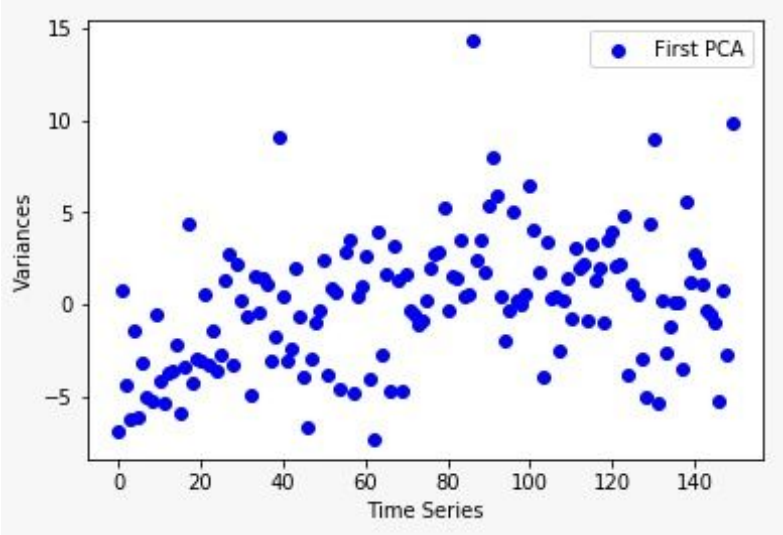


Scatter plot for top 5 features

6.1 **POLYFIT 4DEGREE FOURTH COEF**

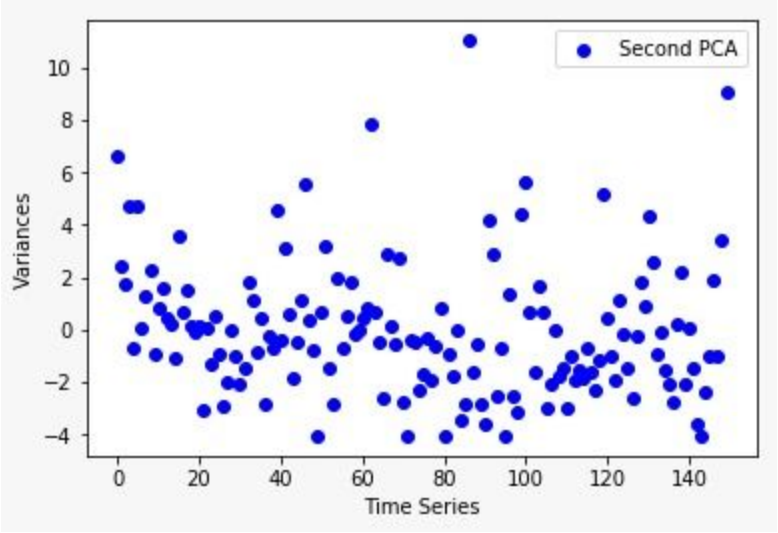
As explained above PCA selects the features which have more variance so that it can classify any new test record to a certain class. According to the data provided to us Polyfit 4th degree 4th Coefficient has comparatively more variance than other features. To visualize it more clearly the variance vs time series plot is presented below.

We have observed that 4th degree polynomial fit curves best fit the time series and has shown the maximum variance and got picked by PCA as the Top Feature that can help in distinguishing the different time series data points.



6.2 **PSD SEC PEAK**

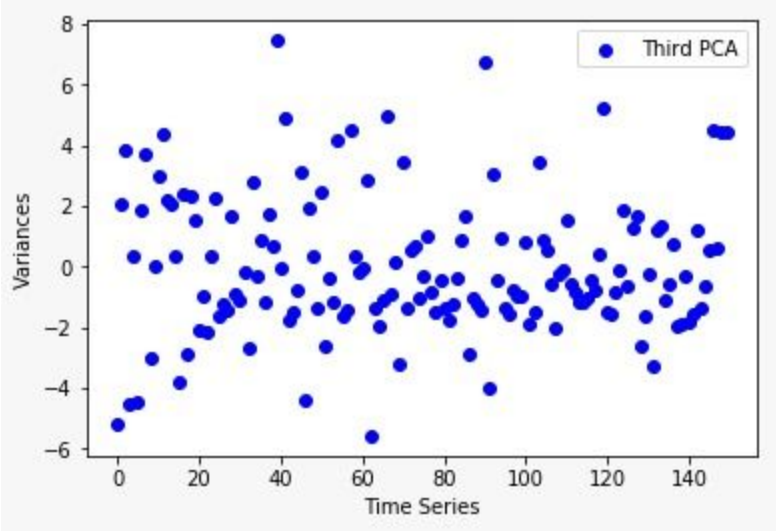
The second peak values of time series have the second highest variance among other features as plotted below we can observe that the variance of the the column varies between -4 and 10.



6.3 **POLYFIT 1DEGREE SEC COEF**

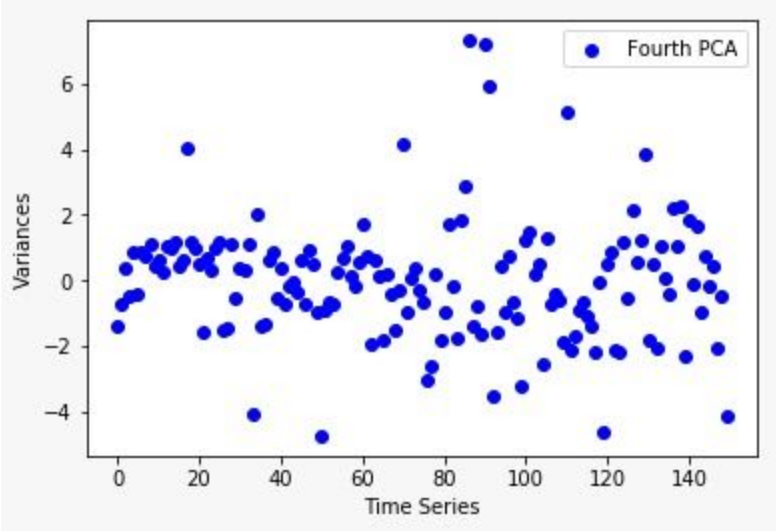
Polynomial Fit happens to be an important feature as two of the top five features are getting selected by PCA. According to data the 4th degree and 1st degree polynomial curves best fit

the data and therefore have more variance.



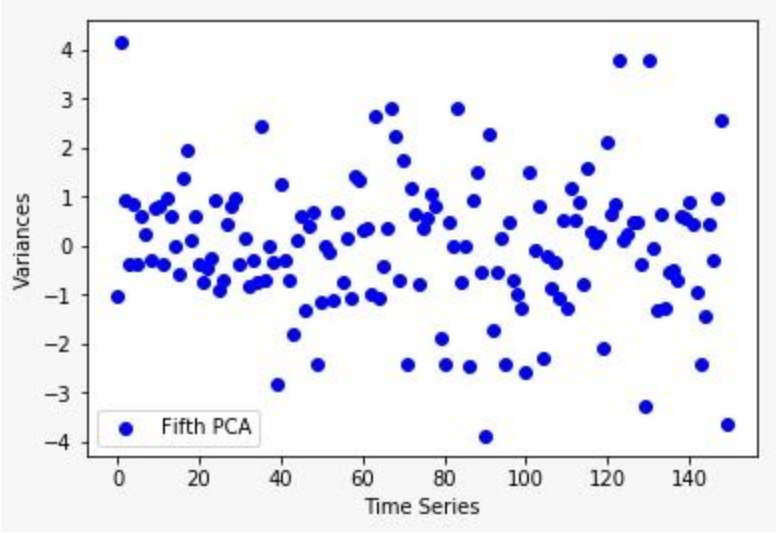
6.4 CORRELATION WITH LAG 5

This gives the auto-correlation between signals that have a time lag of 5 time intervals and gives the similarity between the signals that differ by 5 time intervals which is 25 minutes in our case. Based on the data we observe good amount of variance in this feature and due to that it was picked as a fourth feature in PCA.



6.5 PSD FIRST PEAK

The second peak of PSD has higher variance compared to first peak of PSD and this shows that the time series data differ more by location of second peak compared to the first peak.



7. References

[1]https://www.cygres.com/OcnPageE/Glosry/SpecE.html
[2]https://www.investopedia.com/terms/a/autocorrelation.asp
[3]https://blogs.oracle.com/datascience/7-effective-methods-for-fitting-a-linear-model-in-python
[4]https://newonlinecourses.science.psu.edu/stat462/node/188/
[5]Petrie, John R., et al. "Improving the clinical value and utility of CGM systems: issues and recommendations." *Diabetologia* 60.12 (2017): 2319-2328.
[6]Contreras I, Oviedo S, Vettoretti M, Visentin R, Vehí J (2017) Personalized blood glucose prediction: A hybrid approach using grammatical evolution and physiological models. PLoS ONE 12(11): e0187754. https://doi.org/10.1371/journal.pone.0187754
[7]Zheng, Hui, David M. Nathan, and David A. Schoenfeld. "Using a multi-level B-spline model to analyze and compare patient glucose profiles based on continuous monitoring data." *Diabetes technology & therapeutics* 13.6 (2011): 675-682.