Time Series Modeling of Blood Glucose

1. Introduction

This project is to extract features from the given glucose data at lunch time of patients and time series when the glucose values are recorded in Continuous Glucose Monitor (CGM). For each time series, we have explained why the feature has been chosen. By the value of the features, it is validated how the feature is significant.

Once the features are extracted, we have created a feature matrix and added all the features in the feature matrix. Then we have performed Principal Component Analysis on the feature matrix to obtain the features that are orthogonal to one another. PCA will form a new feature matrix. From this new matrix we have selected top five features and plotted them for each time series.

We have plotted the PCA components in graphs and demonstrated how these are the most relevant features for our analysis.

Data for a set of four patients is given to examine our result. The data includes Glucose Level at each instance of time and Glucose Time Series as input.

Python Libraries Used

1. Numpy
2. Pandas
3. Matplotlib.pyplot
4. Scipy
5. Seaborn
6. Sklearn

2. Team Members

- Parantika Ghosh (pghosh15@asu.edu)
- Shankar Krishnamoorthy (skris106@asu.edu)
- Vashishta Harekal (vharekal@asu.edu)
- Avinash Khatwani (akhatwa1@asu.edu)

3. Project Phase 1 – Feature Extraction

This phase of the project is to extract four features from the given Glucose Level data and Glucose Time Series.

Preprocessing
Our first step was preprocessing the data to combine all the data in different files . We also removed observations with missing values to that our analysis is more effective.

We have selected four feature extraction techniques as follows:

1) Percentage Change Transformation

2) Root Mean Square

3) Fast Fourier Transform

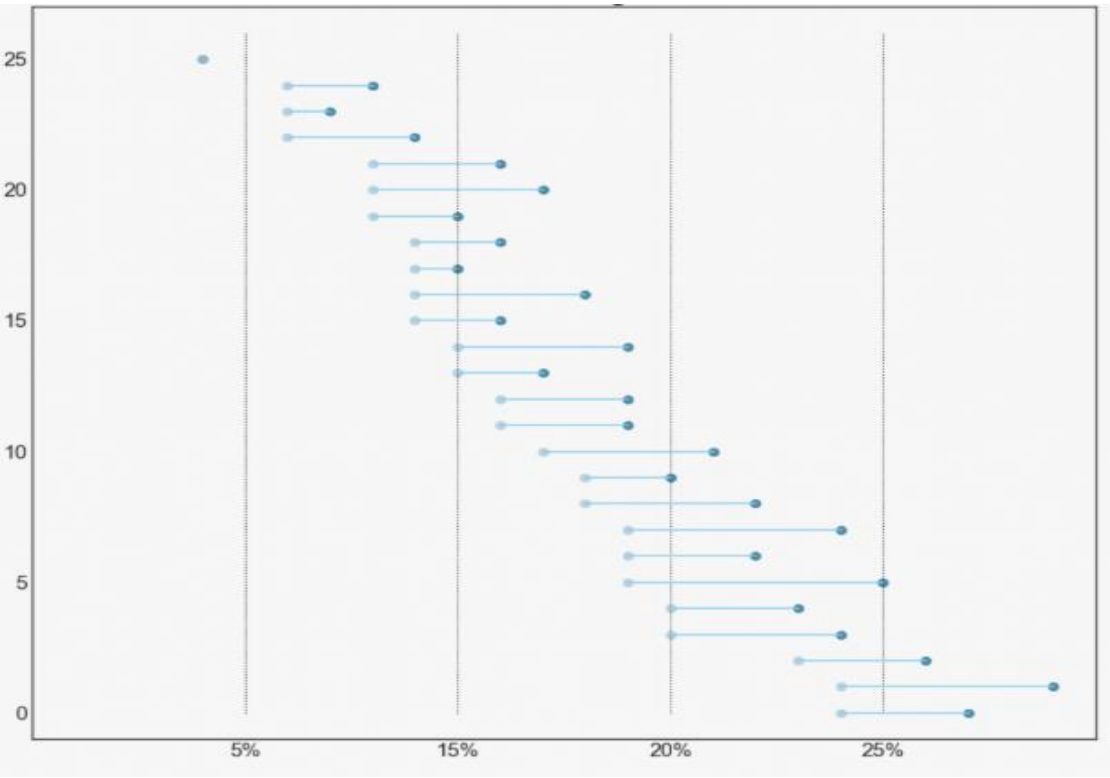
4) Interquartile Range

5) Window Mean

6) Window Standard Deviation

3.1 Percentage Change Transformation:

The PCT function returns a transformation of the provided time series using a Percentage Change transformation. We have used the inbuilt PCT function in Python for estimating the PCT value in glucose.

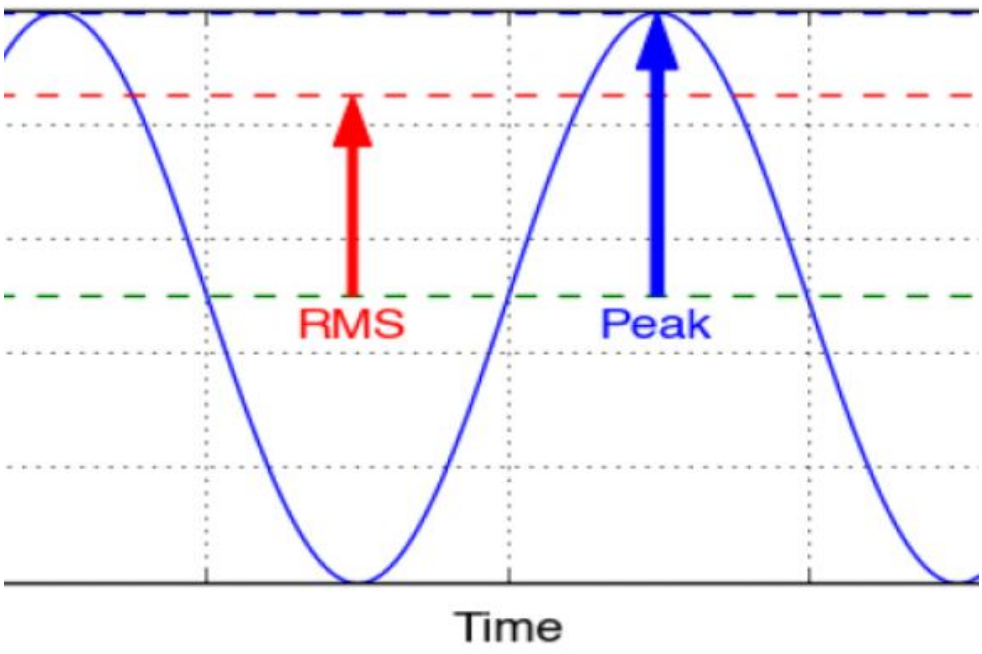


Percentage Change Transformation of Data

Reason for selection of this technique: We have used this function to see the percentage change of glucose as we proceed in the graph. This change can be positive or negative. For our feature vector we consider the max PCT value and the min PCT value.

3.2 Root Mean Square:

RMS is the square root of the arithmetic mean of the squares of the data points. It is also called as the quadratic mean. It is a measure of the magnitude of the data, having both positive and negative values.



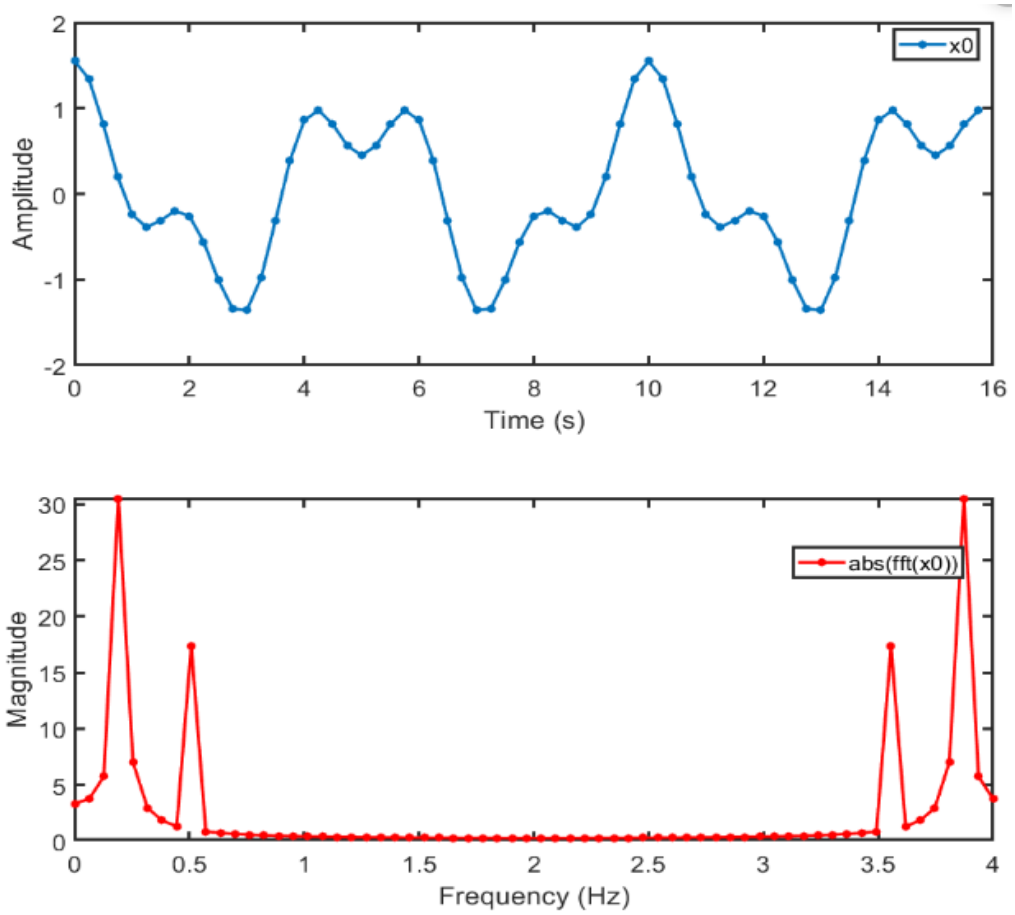
RMS value vs Peak value

Reason for selection of this technique: We have selected this technique because RMS value will give the magnitude of a set of values. In this project, we have used RMS to get the magnitude of glucose per day for any patient.

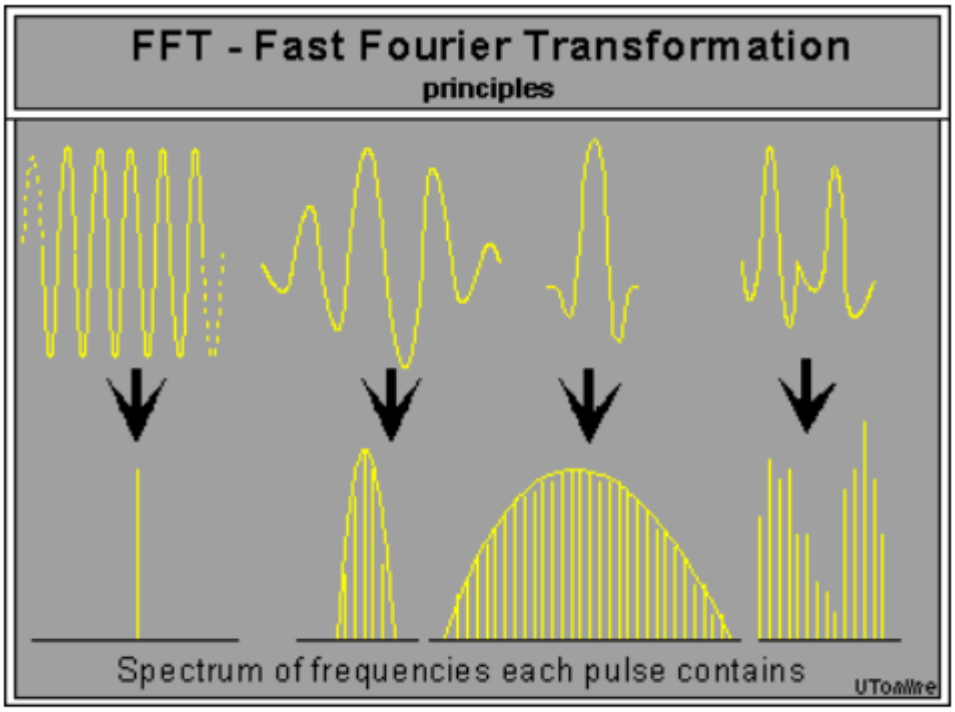
This penalizes large errors due to the squared term.

3.3 Fast Fourier Transform(FFT):

Fast Fourier Transform technique samples a signal over a period of time and decomposes it into frequency components. FFT transforms the data from time domain to frequency domain, which will help in the analysis of data.



Signal Conversion from time to frequency domain



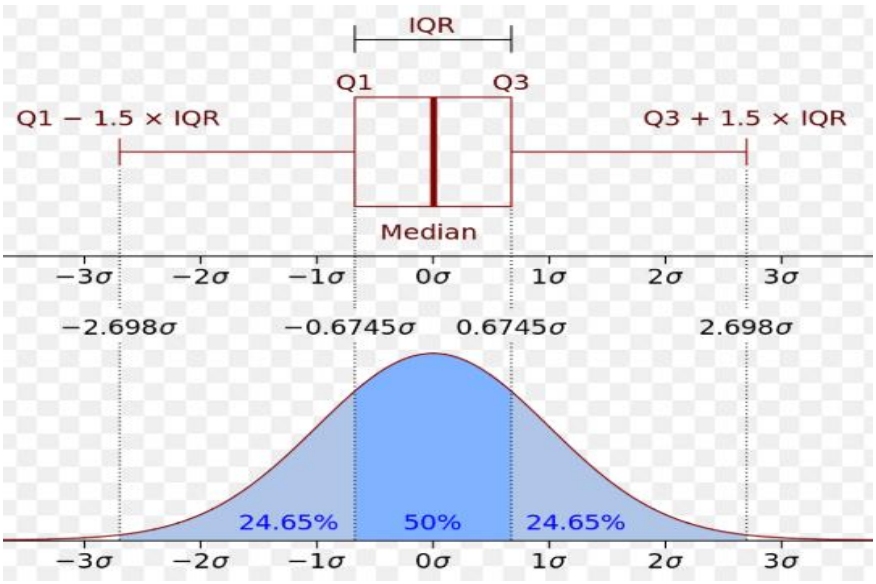
FFT

Reason for selection of this technique: The data provided is in time domain. We will convert this data to frequency domain for better understanding and analysis.

The FFT, or Fast Fourier Transform is an algorithm that essentially uses convolution techniques to efficiently find the magnitude and location of the tones that make up the signal of interest. Therefore, this will be particularly handy in knowing the data better and forecasting applications.

3.4 Interquartile Range:

Interquartile Range (IQR) is the measure of the mid-spread or middle 50% of the curve. It is a measure of variability, while dividing the data set into quartiles. It defines how spread out the middle 50% of the data is.



Interquartile Range(IQR)

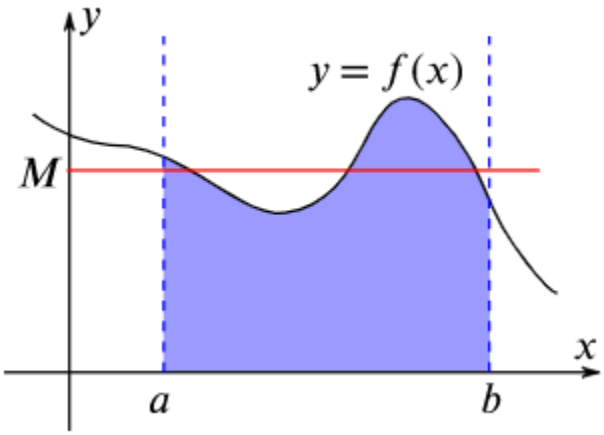
Reason for selection of this technique: In our project we will use this function as it will generate how spread out the middle 50% of each distribution curve is. Thus, when glucose level is high, it will give the time taken for each rise and fall of the glucose level.

3.5 Rolling Window Mean:

Mean(μ) is the arithmetic average of a set of data.

Window function is used to find the trends within the graph by smoothing the curve.

Window function operates on a set of data and return a single value for that set. The term window describes the set of data on which the function operates. A window function uses values from the rows in a window and return the desired output depending on the function selected.



Mean Value along Y axis for a given curve

Reason for selection of this technique: This feature will give the average of glucose value for a set of data and thus help in estimating the glucose level in a region of the curve.

Window Mean refers to applying mean function on a window of data and getting mean for that set of data. In our project, we have divided each row of data into several windows and ran mean function on each window to get the mean. Thus, the value obtained as mean of each window will act as a feature. All such features are then appended to the feature matrix, along with other features previously selected.

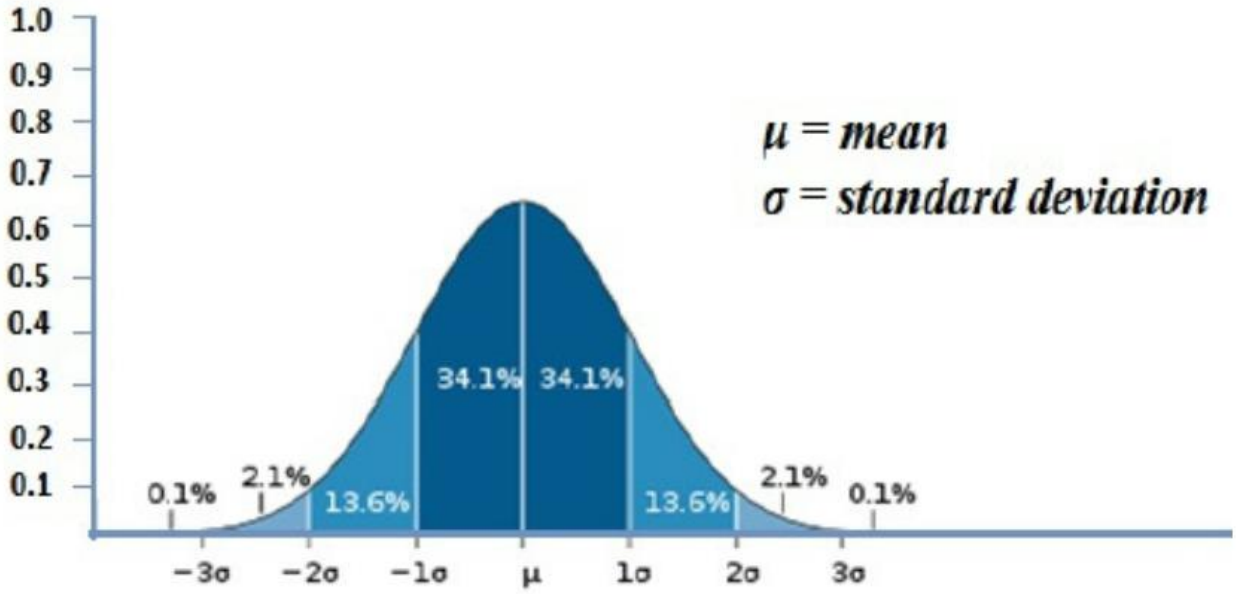
Rolling-window analysis of a time-series model assesses:

The stability of the model over time. A common time-series model assumption is that the coefficients are constant with respect to time. Checking for instability amounts to examining whether the coefficients are time-invariant. Hence, we can forecast accuracy of the model.

Therefore in our study we have chosen to analyze the mean and standard deviation in a rolling window. Calculating the mean in the window helps us measure the central tendency over time.

3.6 Rolling Window Standard Deviation:

Standard Deviation(σ) is the measure of variation in a set of data. The variation is calculated from the mean of the data. It defines how spread out the data is from the mean.



Window Standard Deviation is applying standard deviation function on a set of data or window. Each row is subdivided in a number of windows and we have applied standard deviation on each window. The output value is the standard deviation of values in each window and is a feature. This feature is then appended in the feature matrix along with other features.

Reason for selection of this technique: This feature will give the variation in glucose value for a set of data and thus help in estimating the change in glucose level with change in time in our curve.

Rolling-window analysis of a time-series model assesses:

The stability of the model over time. A common time-series model assumption is that the coefficients are constant with respect to time. Checking for instability amounts to examining whether the coefficients are time-invariant. Thus, we can forecast accuracy of the model.

Therefore in our study we have chosen to analyze the mean and standard deviation in a rolling window. Here we can measure the volatility over time using the standard deviation in a rolling window, therefore also detecting changes in trend.

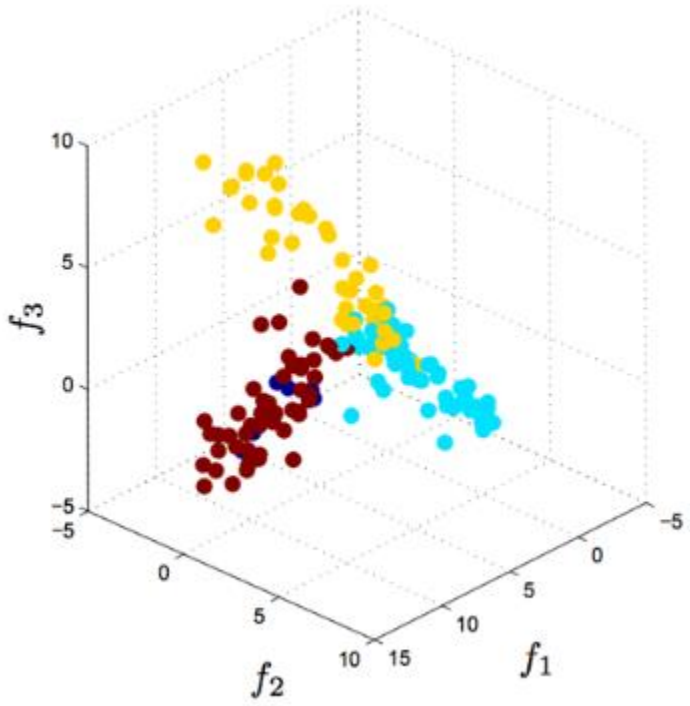
4. Project Phase 2 - Feature Matrix and PCA

4.1. Task 1: Arranging the Feature Matrix

We have extracted the features by feature selection techniques namely Percentage Change Transformation, Root Mean Square, Fast Fourier Transform, Window Mean and Window Standard Deviation. After accumulation of the features, we have arranged them in the feature matrix, where each feature is a column in the matrix.

4.2. Task 2: Execution of PCA

Principal component analysis (PCA) is a method of orthogonal transformation that converts a correlated dataset into a set of values of linearly uncorrelated variables called principal components.

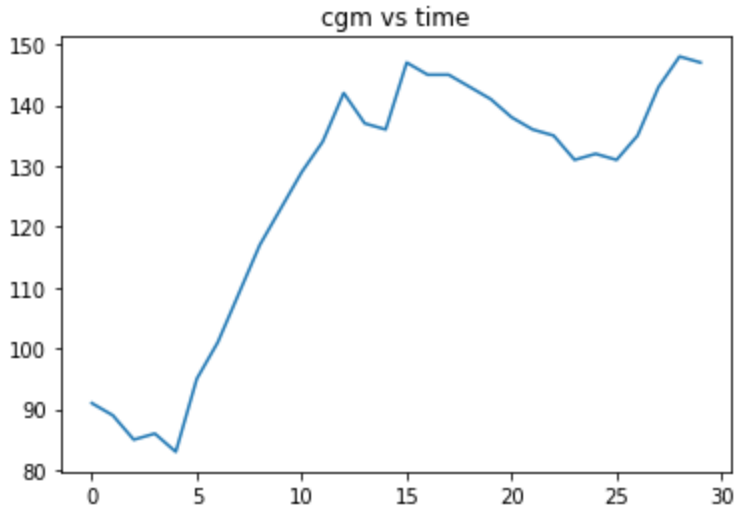


Principal Component Analysis with Three Components

We have executed Principal Component Analysis (PCA) on the feature matrix, having component size=5. PCA is a dimensionality reduction technique, which reduces the feature matrix in dimension from a large set of data points to a small set, that has most of the information of the large set.

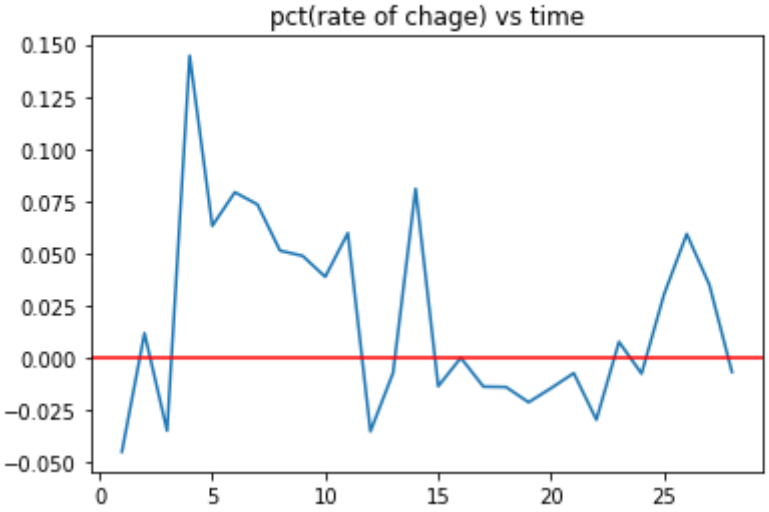
5. Project Phase 2 - **Feature Selection and Argument:**

In this step we display the outputs from our various feature extraction steps.



In the above image, we have the CGM vs Time graph, where the blue line indicates the Blood Glucose Level.

5.1) Percentage Change Transformation

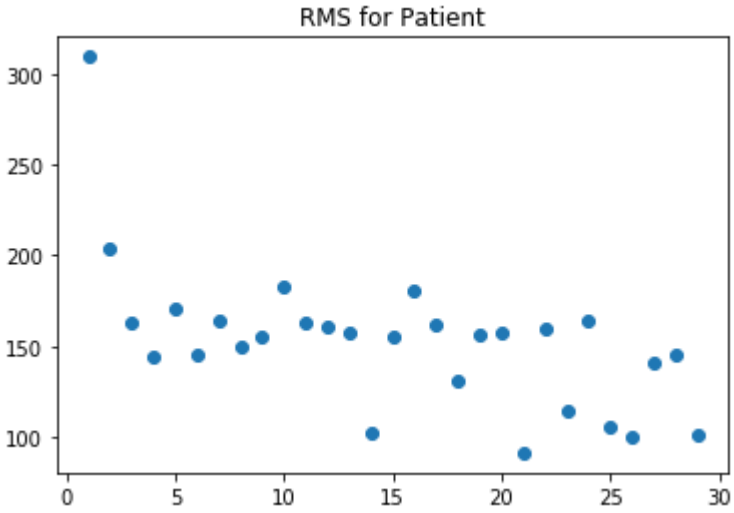


Here we can see the changes in the glucose levels captured effectively.

Here, maxPct indicates the largest change between increase between values of glucose. This can be used as an approximate index to where the person has had food as the glucose values increase the most at that point.

To make the assumptions more accurate we also take the pctZeroInd which is basically where the slope towards the maxPct starts. This value indicates where the point where the values of glucose starts increasing.

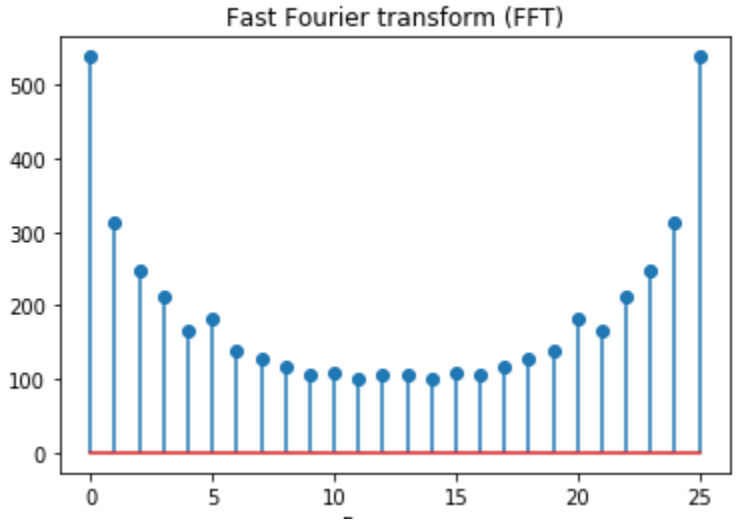
5.2) Root Mean Square



RMS Values for First 30 Data Points

This effectively explains if the data inside an observed time series varies a lot or not. Therefore, it gives us a good insight of variation in the data.

5.3) Fast Fourier Transform



Here we see the different frequency components of the data. This will give new insights to data which are otherwise hard to analyze directly from the time series. For our feature vector we have used first 6 components of the different components returned.

5.4) Interquartile Range

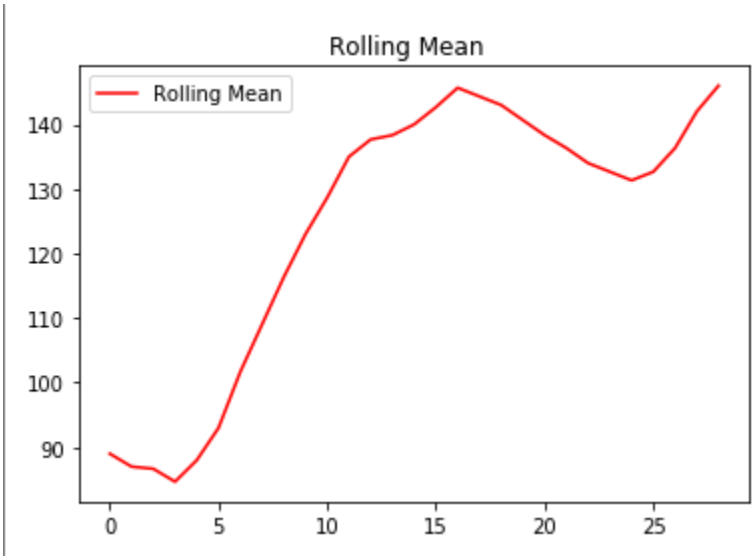
Sample Output:

For index at 0.25 207.0

For index at 0.75 236.0

Here we see that IQR is a measure of variability, based on dividing a data set into quartiles. Quartiles divide a rank-ordered data set into four equal parts. This will help us analyze how the values change across the spread of the time series.

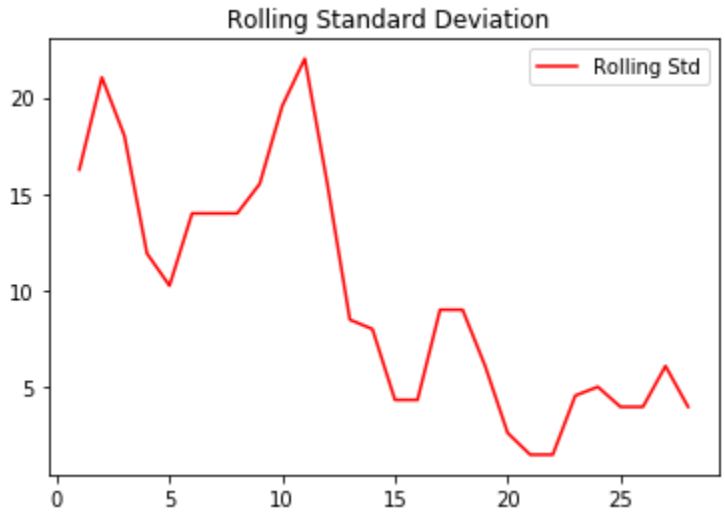
5.5) Window Mean



Here we choose windowing value of 3. Therefore, each data point is primarily compared with its neighbor to smoothen the time series. This will help us reduce noise before making any observation and inference. Calculating the mean in the window helps us measure the central tendency over time.

The vector size returned is equal to the size of the feature vector input itself. So, for each observation we have a vector of 30. And the vector effectively follows the input time series.

5.6) Window Standard Deviation



Rolling-window analysis of a time-series model assesses:

The stability of the model over time. A common time-series model assumption is that the coefficients are constant with respect to time. Checking for instability amounts to examining whether the coefficients are time-invariant. Therefore, we can forecast accuracy of the model.

Here we can measure the volatility over time using the standard deviation in a rolling window, therefore also detecting changes in trend. The vector size returned is equal to the size of the feature vector input itself. So, for each observation we have a vector of 30. Here, we can see that the vectors capture the changing trends in the time series effectively.

6. Final Feature Vector

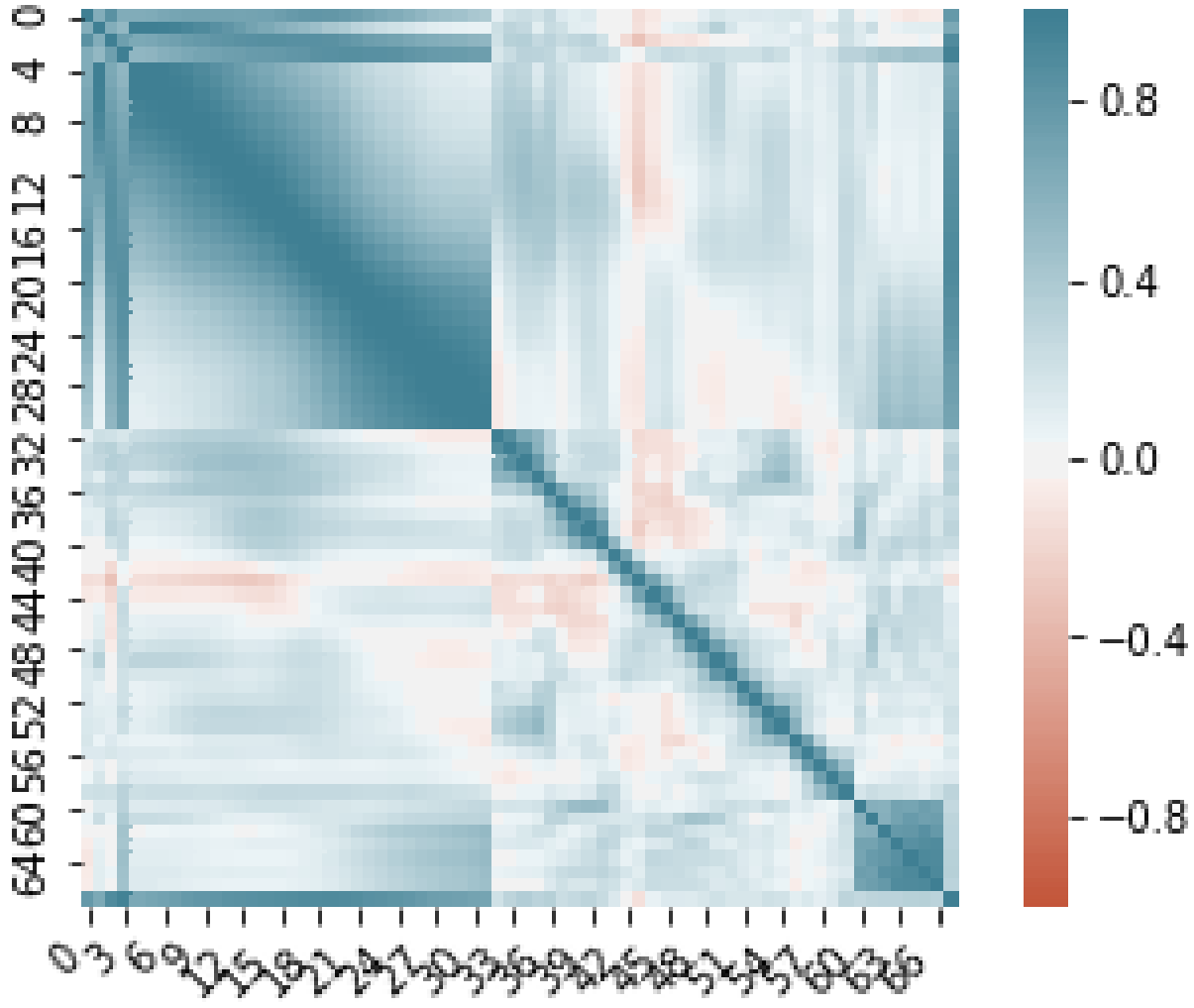
We see here that for each data point after concatinating the different features, we get 68 dimensional feature vectors for each observation in our dataset.

```
[131.         90.         99.        241.         90.5
  90.66666667  90.66666667  93.         95.66666667  98.
  98.66666667  98.66666667 100.        104.66666667 115.33333333
 130.        146.66666667 162.66666667 178.66666667 193.33333333
 204.66666667 213.66666667 220.66666667 228.33333333 235.
 241.        246.33333333 250.66666667 254.66666667 257.33333333
 258.66666667 258.          0.70710678   0.57735027   0.57735027
   3.46410162   4.163332     1.          0.57735027   0.57735027
   1.73205081   7.3711148   14.6401275   16.52271164  16.01041328
  16.50252506  15.50268794  14.0118997    9.71253486   7.02376917
   7.5055535    7.5055535    6.55743852   5.          5.50757055
   4.163332     3.05505046   3.05505046   1.15470054   2.
```

319.75235784 294.36285241 216.05141798 159.78914677 127.56938722
121.7164718 99.60520132 183.06091882]

6.1) Correlation Matrix:

The relationships between each other is indicated with the help of correlation matrix below. We can see there are a good number of features that are not correlated with each other.



Correlation of Features in Heatmap

6.2) Principal component analysis (PCA):

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

To reduce the high dimensional data of 68 features extracted, we are using PCA to reduce it to a 5 dimensional space while we preserve the maximum variance in the data for reproducibility.

Thus, on decomposing we get the following components.

PCA Components:

Here we can see that each of the 5 identified latent semantic is expressed in terms of the 68 features in the original data space.

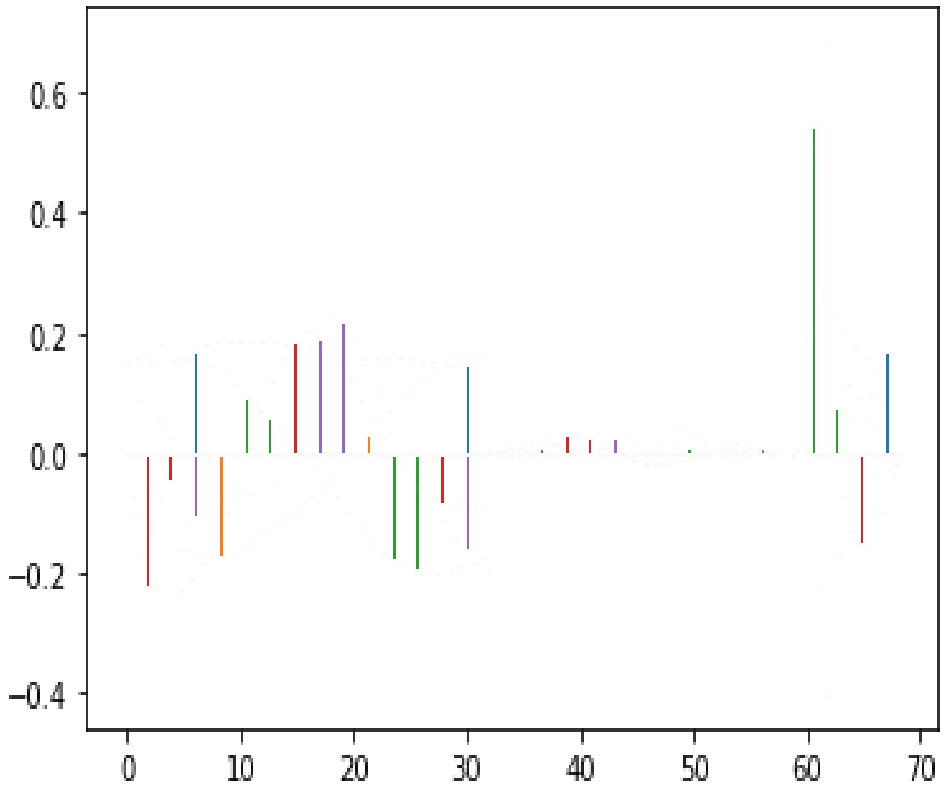
```
[[ 1.46438372e-01  1.57798803e-01  1.51540967e-01  1.83945109e-01
   1.56573119e-01  1.61739000e-01  1.71920907e-01  1.80382716e-01
   1.85166297e-01  1.86623250e-01  1.85807758e-01  1.85095215e-01
   1.84497030e-01  1.84847761e-01  1.83963569e-01  1.81747103e-01
   1.78020279e-01  1.74334601e-01  1.70427530e-01  1.66141028e-01
   1.62303583e-01  1.58952666e-01  1.58342494e-01  1.59711315e-01
   1.61350224e-01  1.59487728e-01  1.55754646e-01  1.52624329e-01
   1.51057963e-01  1.49706125e-01  1.47902609e-01  1.45088175e-01
   2.99419989e-03  6.17489822e-03  7.07432976e-03  7.07761059e-03
   8.66970098e-03  6.50927699e-03  6.79618335e-03  8.00492833e-03
   5.43908466e-03  2.95272765e-03 -2.10712038e-04 -2.86899072e-03
   1.08222709e-03  2.06593446e-03  2.08356661e-03  2.44187968e-03
   3.32989968e-03  4.02947725e-03  2.85651952e-03  2.58809445e-03
   3.16457008e-03  4.14086981e-03  3.13856803e-03  1.43104199e-03
   2.07813856e-03  1.20876454e-03  1.83991947e-03  4.55992237e-03
   1.95717223e-01  1.24372023e-01  7.75920643e-02  6.99785107e-02
   6.11735325e-02  5.83630074e-02  4.66357829e-02  1.68640808e-01]
 [-9.76202039e-02 -1.69628466e-01 -6.62438318e-02  3.56323294e-02
  -1.60726676e-01 -1.63054389e-01 -1.67736053e-01 -1.70231916e-01
  -1.70994133e-01 -1.66596306e-01 -1.58746280e-01 -1.46108581e-01
  -1.32984394e-01 -1.19231068e-01 -1.05343684e-01 -8.95954758e-02
  -7.25064480e-02 -5.45359080e-02 -3.52650046e-02 -1.39868595e-02
   8.72927772e-03  3.14260002e-02  5.26000087e-02  7.27697005e-02
   9.18844369e-02  1.10694265e-01  1.27535411e-01  1.41461693e-01
   1.50337136e-01  1.57118095e-01  1.63099315e-01  1.68464094e-01
  -3.25156285e-03 -4.48115848e-03 -3.76559068e-03 -1.87340529e-03
  -1.20803649e-03  1.91940924e-03  4.56344923e-03  6.97064407e-03
   7.55488032e-03  4.81258162e-03  4.46934437e-03  6.54911356e-03
   9.10017651e-03  5.94107225e-03  3.32462205e-03  2.75897369e-03
   7.22655373e-04 -1.04796547e-03  1.32090108e-03  1.20413035e-03
   2.09882637e-05 -8.86346767e-06 -8.88113314e-04 -5.27892655e-04
  -5.96305034e-05 -2.77754042e-05  8.11460830e-05  1.48911854e-03
   5.11995246e-01  2.77884986e-01  2.34405863e-01  2.06240636e-01
   1.77678786e-01  1.62330084e-01  1.36603242e-01 -8.16947275e-03]
 [-7.12338148e-02  1.72172553e-01 -5.20295856e-02 -4.20080228e-02
   1.61080438e-01  1.59422996e-01  1.54065043e-01  1.47757813e-01
   1.32988920e-01  1.15985701e-01  9.52344983e-02  7.87773798e-02
   6.17054039e-02  4.29184859e-02  2.29677166e-02  1.31575233e-03
```

-2.15346421e-02 -4.66554335e-02 -7.30036511e-02 -9.97098872e-02
-1.24646220e-01 -1.48123035e-01 -1.66111382e-01 -1.78809032e-01
-1.85708569e-01 -1.91784913e-01 -1.96222712e-01 -1.98097559e-01
-1.95953206e-01 -1.90160662e-01 -1.82614317e-01 -1.72464940e-01
6.22443709e-03 7.38076153e-03 5.25555191e-03 4.71454429e-03
1.06255337e-02 1.34926486e-02 1.06319959e-02 8.63044254e-03
6.92431196e-03 6.06526939e-03 5.00914012e-03 1.71973866e-03
-1.48852554e-03 -1.39243570e-03 2.64164247e-03 6.61169838e-03
8.77047925e-03 9.88981928e-03 6.27164180e-03 3.77328058e-03
3.53029520e-03 5.87116187e-03 4.94919958e-03 6.39797406e-04
8.98836508e-04 8.34096483e-04 1.07382016e-03 1.42587363e-03
5.43745235e-01 2.59945278e-01 7.75670207e-02 1.16773774e-01
9.10251747e-02 9.91173158e-02 8.11544227e-02 -3.14432814e-02]
[-6.92792904e-03 -2.23169845e-01 6.09156154e-02 -4.24525184e-02
-2.25659027e-01 -2.11946602e-01 -1.84359924e-01 -1.48130974e-01
-1.01348636e-01 -4.61151010e-02 1.18767696e-02 6.98576229e-02
1.18988033e-01 1.59307955e-01 1.87808412e-01 2.03793436e-01
2.00363755e-01 1.79962769e-01 1.52548510e-01 1.20887245e-01
8.79164253e-02 5.66279637e-02 2.80392638e-02 -1.21974693e-03
-3.09525069e-02 -5.69148360e-02 -7.42814212e-02 -8.10590291e-02
-8.40609648e-02 -8.56492587e-02 -9.00207095e-02 -9.09996257e-02
7.53149066e-03 9.39465494e-03 1.10298346e-02 1.86080532e-02
1.15119884e-02 1.61689784e-02 3.18573561e-02 3.25063909e-02
2.44447340e-02 1.20712253e-02 5.67178546e-03 -4.95981233e-03
-1.13496826e-02 -1.56069953e-02 -1.60666192e-02 -1.48298139e-02
-9.44871097e-03 -5.61764623e-03 3.59802847e-03 6.59879231e-03
1.16306966e-02 1.46429139e-02 9.98892010e-03 6.25522575e-03
3.38705007e-03 1.15846709e-03 -3.80059825e-03 -2.27133609e-03
4.83822673e-01 -4.03611276e-01 -1.70511403e-01 -1.25949201e-01
-1.50710483e-01 -9.55485846e-02 -1.02305345e-01 -1.50226467e-02]
[8.76077586e-02 -1.19619412e-01 -7.92638788e-02 2.78039319e-02
-1.15209038e-01 -1.07148798e-01 -9.01435094e-02 -7.69704883e-02
-7.02056638e-02 -6.73740687e-02 -6.53457766e-02 -5.66933268e-02
-4.22982478e-02 -6.21920165e-03 5.61808385e-02 1.33276287e-01
1.94542936e-01 2.25086709e-01 2.23108588e-01 1.92837294e-01
1.48706034e-01 1.04007307e-01 7.09283280e-02 4.04801482e-02
5.53493834e-03 -3.49881413e-02 -7.68669935e-02 -1.13120369e-01
-1.40548685e-01 -1.62797414e-01 -1.80639669e-01 -1.96670469e-01
-4.60624249e-03 -3.70496809e-03 1.65090479e-03 1.94419754e-02
2.64012009e-02 1.71896522e-02 4.94868232e-03 -2.32049392e-03
-3.31617847e-03 9.31260578e-03 2.76729405e-02 3.28011978e-02
1.76591747e-02 4.01504920e-03 1.79378002e-02 4.08471014e-02
4.26702660e-02 2.98384952e-02 1.42269287e-02 9.86386127e-03
8.49913905e-04 2.88194333e-03 1.25380405e-02 8.41843613e-03
2.67179638e-03 -3.37639182e-03 -1.99491643e-04 5.80637191e-03
-2.23012219e-01 6.88707644e-01 -7.04857939e-02 -2.08095733e-02
3.74741319e-03 -3.64705107e-02 -2.46694930e-03 -7.44011730e-03]]

PCA variance:

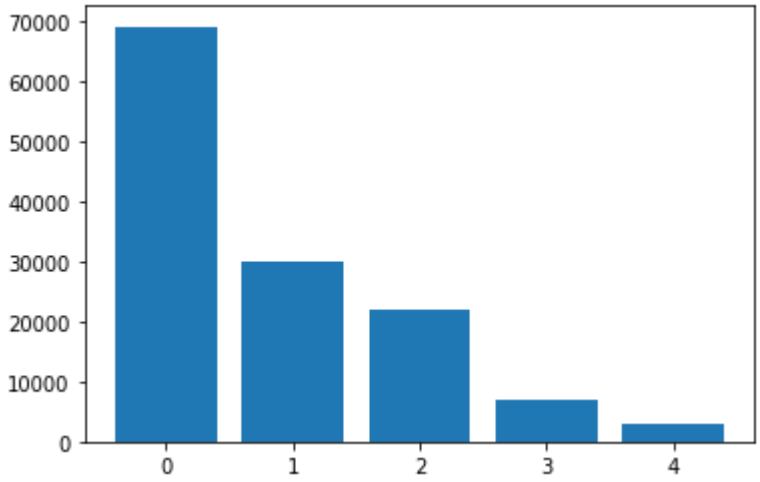
[69250.52493166 29978.68783969 21925.73688341 6792.45907642
3048.962887]

EigenVectors:



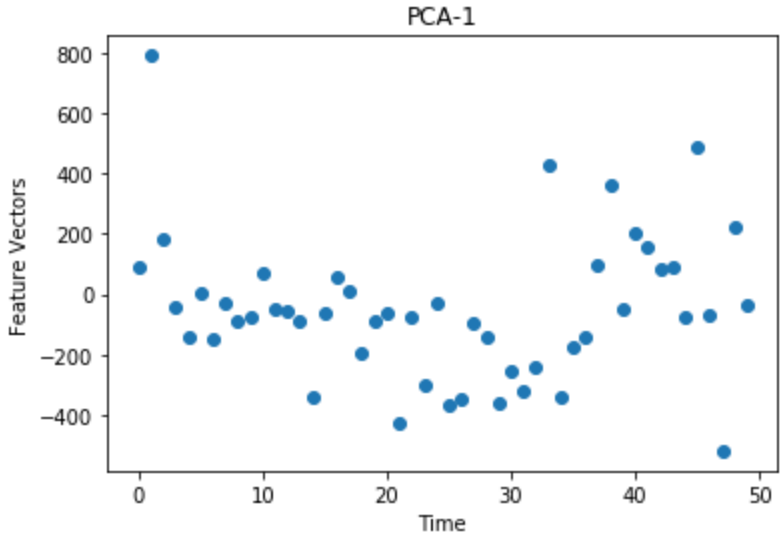
Eigen Values:

Here, we see the different eigen values visualized.

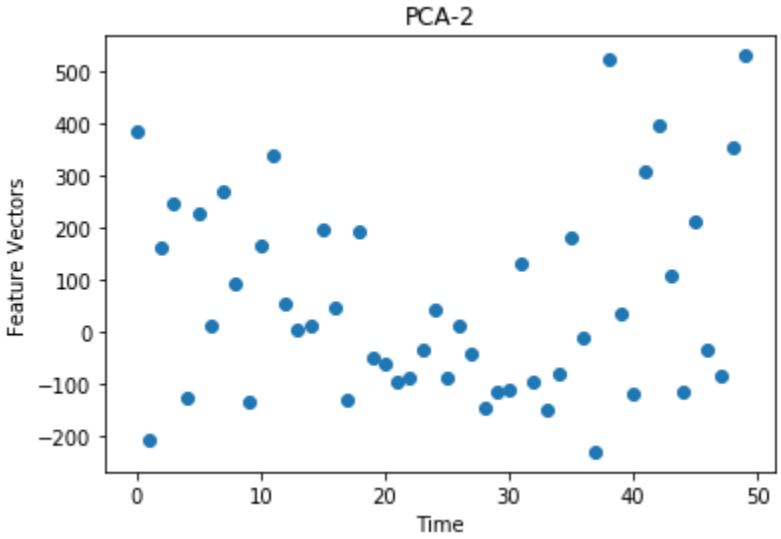


Here we see how the amount of information captured is the highest for the first vector. After that the increase in the actual learning is lesser.

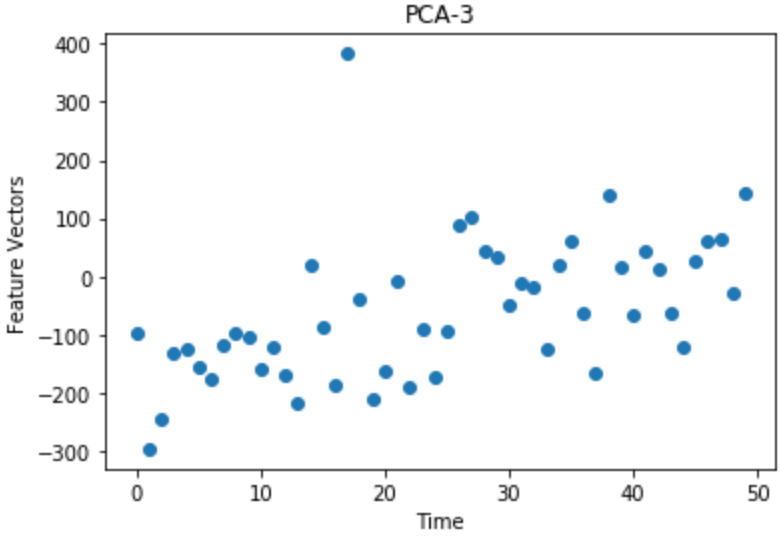
After feeding the original feature matrix to the PCA model generated we get the transformed matrix. This is visualized below in terms of vector. The variance of 5 PCA feature vectors is indicated in the diagrams below. We can see that each vector captures good variance captured in them.



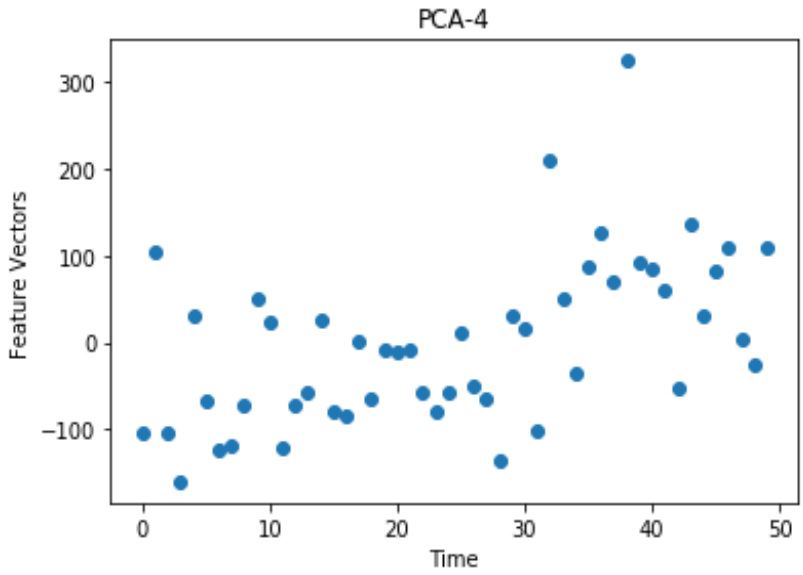
Principal Component - 1 Vs Time



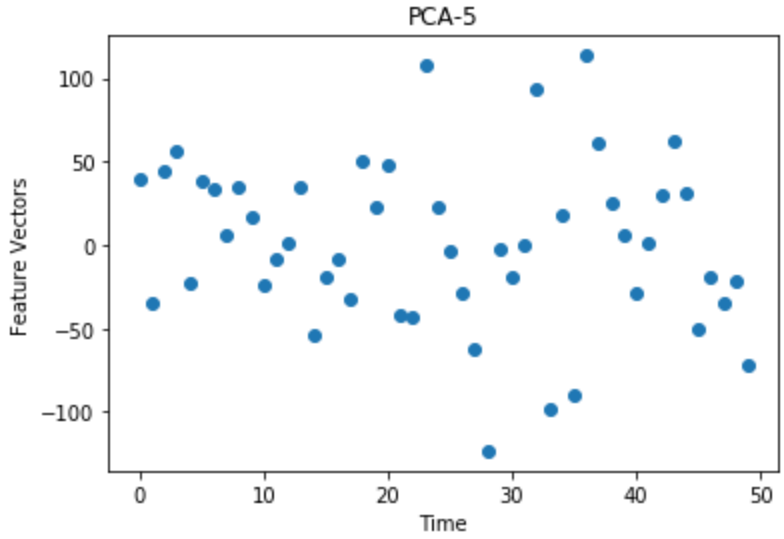
Principal Component - 2 Vs Time



Principal Component - 3 Vs Time

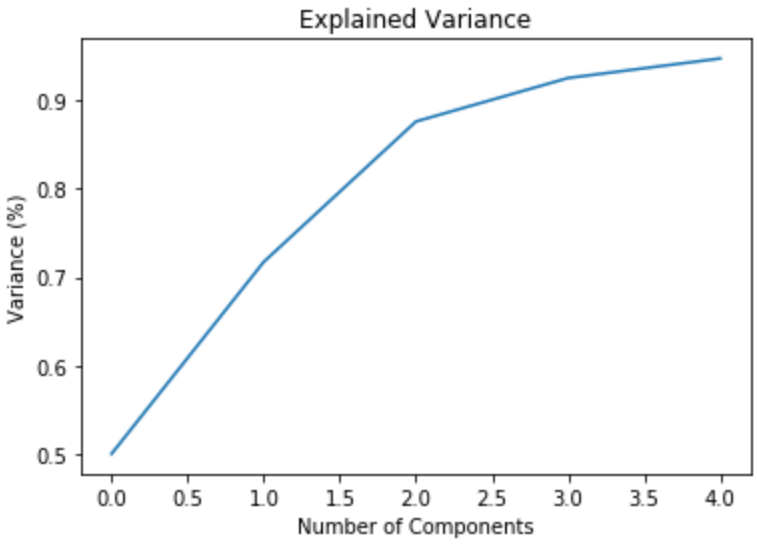


Principal Component - 4 Vs Time



Principal Component - 5 Vs Time

All these indicate that the 5 components picked through PCA are very effective in capturing the maximum information while reducing the dimension from 68 to 5.



In PCA, we order the eigenvalues from largest to smallest so that it gives us the components in order or significance to effectively reduce dimensionality. If we have a dataset with n variables, then we have the corresponding n eigenvalues and eigenvectors. It turns out that the eigenvector corresponding to the highest eigenvalue is the principal component of the dataset. Here, to reduce the dimensions, we choose the first 5 eigenvalues and ignore the rest. We do lose out some information in the process, but if the eigenvalues are small, we do not lose much.

This plot tells us that selecting 5 components we can preserve something around 98.8% or 99% of the total variance of the data. The first vector itself has almost 90% of the information preserved.

Dimensionality reduction:

The higher the number of features, the harder it gets to visualize the training set and then work on it. Many of these features are correlated, and hence contains the same information. PCA helps to tackle this by obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

7. References:

1. Time Series Analysis:

https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b

2. Feature Selection techniques:

https://www.statisticssolutions.com/time-series-analysis/

3. Principal component Analysis:

https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

https://www.mathworks.com/help/econ/rolling-window-estimation-of-state-space-models.html

https://en.wikipedia.org/wiki/Principal_component_analysis

https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial