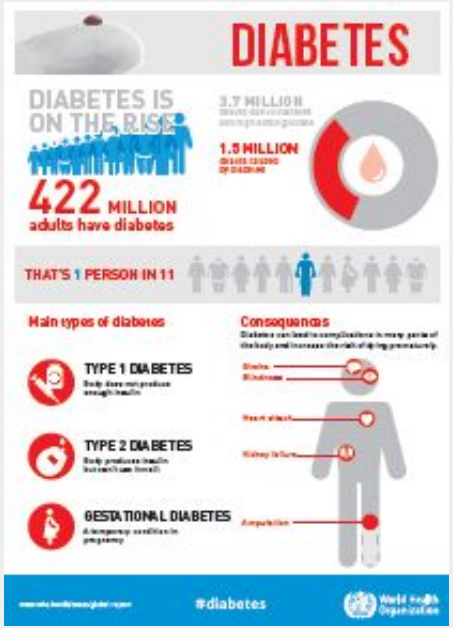
Diabetes prediction and time series forecasting

★Emily Tao <etao@hawk.iit.edu>
Yin Yang (Annie) <yyang191@hawk.iit.edu>
Jim Witkiewicz <jwitkiew@hawk.iit.edu>
Geongu Park <gpark7@hawk.iit.edu>
Woojin Choi <wchoi6@hawk.iit.edu>



"In 2016, an estimated 1.6 million deaths were directly caused by diabetes. Another 2.2 million deaths were attributable to high blood glucose in 2012."
WHO 8/6/2020

Problem Statement:

Diabetes:
A set of diseases characterized by an inability to produce or respond to insulin.

Blood glucose monitoring:
A method for diabetes management to maintain healthy levels of blood glucose.

CGM:
Continuous glucose measurement taken from bloodstream.

We create classifiers for:

(1) Diabetes prediction from the Pima dataset
(2) Blood glucose trend from the Tidepool dataset.

We model CGM time series data and forecast with:

(1) Univariate ARIMA
(2) Univariate LSTM
(3) Multivariate LSTM

Executive Summary:

Findings & Recommendations

- Predicting the trend of CGM values is a difficult problem!
- Forecasting was most effective using multivariate LSTM.
- Recommend further investigation using neural network methods.

Classifier accuracy:

(a) **Pima diabetes prediction**: 79.22%
(b) **CGM trend**: 62.6%

Forecasting effectiveness:

(1) **Univariate ARIMA** included wide confidence intervals and ‘average’ performance, making it less useful.
(2) **Univariate LSTM** was quite inaccurate.
(3) **Multivariate LSTM** had more adaptive forecasts, but no confidence intervals and could be very inaccurate.

Project Overview:

Workflow & Challenges

- Most of the group members were inexperienced in time series analysis. However, non-time series approaches had limited effectiveness.

- Independent development of methods led to very different implementations and sometimes inconsistent processing.

"More is more" approach - try everything!

Division of work:

- **Pima Classification** - Geongu

- Multivariate analysis:
  Data exploration - Emily
  Trend classification - Woojin

- Univariate forecasting models:
  ARIMA, LSTM, **Prophet** - Annie
  NNETAR - Jim*

- Multivariate forecasting models:
  LSTM - Emily

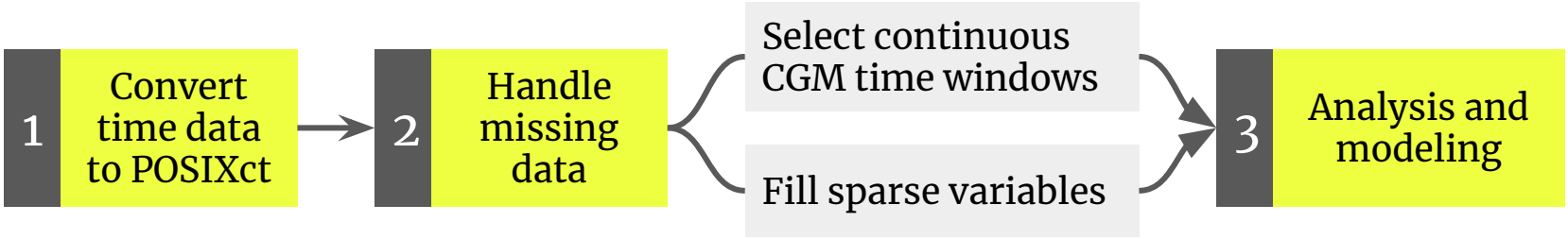
* Unfortunately, Jim was unable to complete the feed-forward nnetar model due to illness. It was omitted from our reports.

Data Exploration

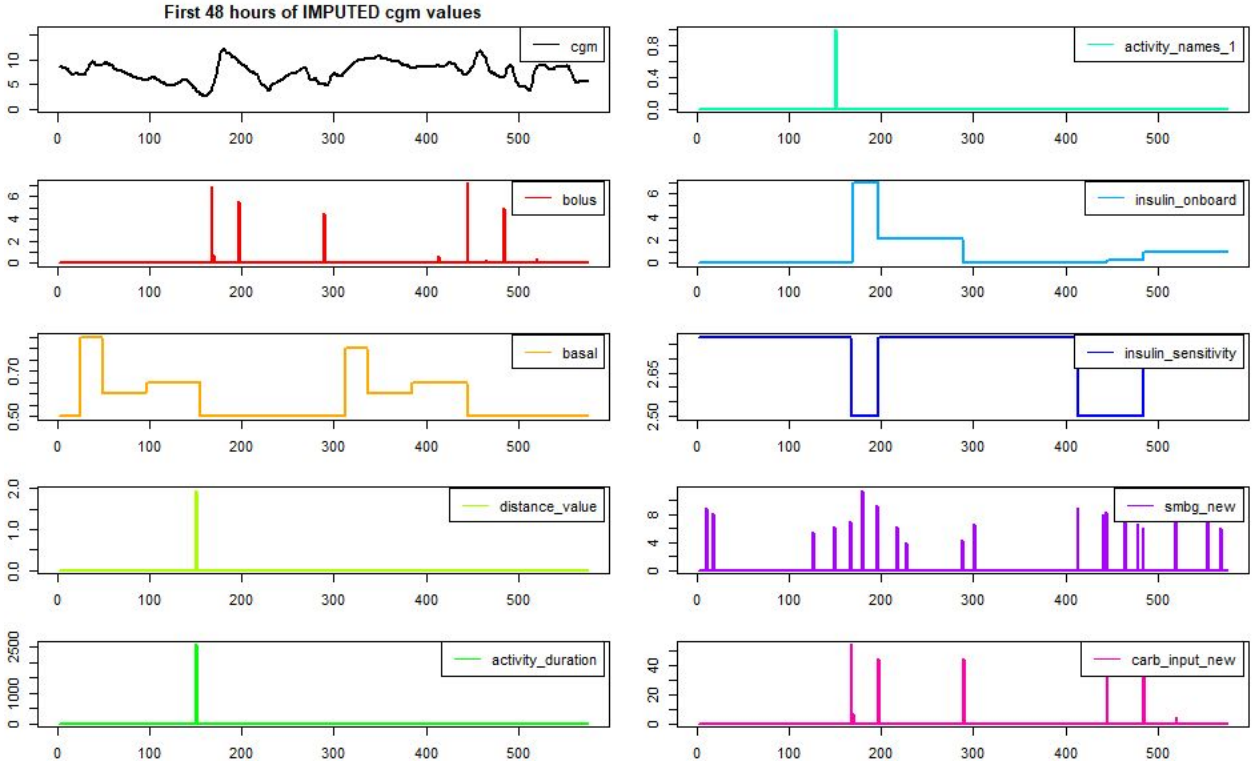
Initial Data Processing

Raw Data:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Time	IMPUTED	Denoised	DAY_WEEK	YEAR	MONTH	DAY	HOUR	MIN	SEC	BASAL	bolus	Activity_Name	Distance_value	Activity_Duration	insulinOnBoard	insulinSensitivity
2	1555286684	6.21684	5.44183454230306	2	2019	4	15	0	4	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
3	1555286984	6.10582	5.5356753437616	2	2019	4	15	0	9	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
4	1555287284	6.05032	5.67150852438383	2	2019	4	15	0	14	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
5	1555287584	6.10582	5.88543379112613	2	2019	4	15	0	19	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
6	1555287884	6.38336	6.2112278188081	2	2019	4	15	0	24	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
7	1555288184	6.82742	6.65793100081427	2	2019	4	15	0	29	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
8	1555288484	7.27148	7.21217849839891	2	2019	4	15	0	34	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
9	1555288784	7.66003	7.84528908787694	2	2019	4	15	0	39	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
10	1555289084	8.21511	8.51209819695203	2	2019	4	15	0	44	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
11	1555289384	8.8812	9.14431649682725	2	2019	4	15	0	49	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
12	1555289684	9.60279	9.67339751348056	2	2019	4	15	0	54	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
13	1555289984	10.04685	10.0482468647479	2	2019	4	15	0	59	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
14	1555290284	10.15787	10.2540516163197	2	2019	4	15	1	4	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN
15	1555290584	10.15787	10.3066426930293	2	2019	4	15	1	9	44	0.7	NaN	NaN	NaN	NaN	NaN	NaN

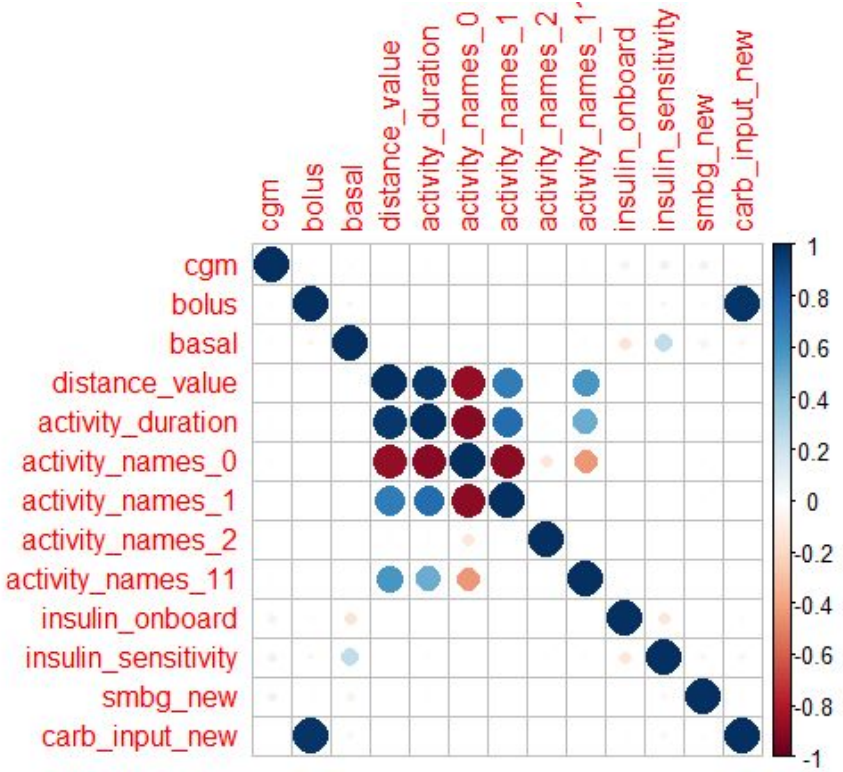


Visualizing the data



- Missing values filled with **0** (sparse) or **previous value** (insulin_onboard, insulin_sensitivity)
- IMPUTED vs Denoised ‘target variable’ (CGM)
- Some redundant, non-informative, or repetitive variables

Correlations between variables



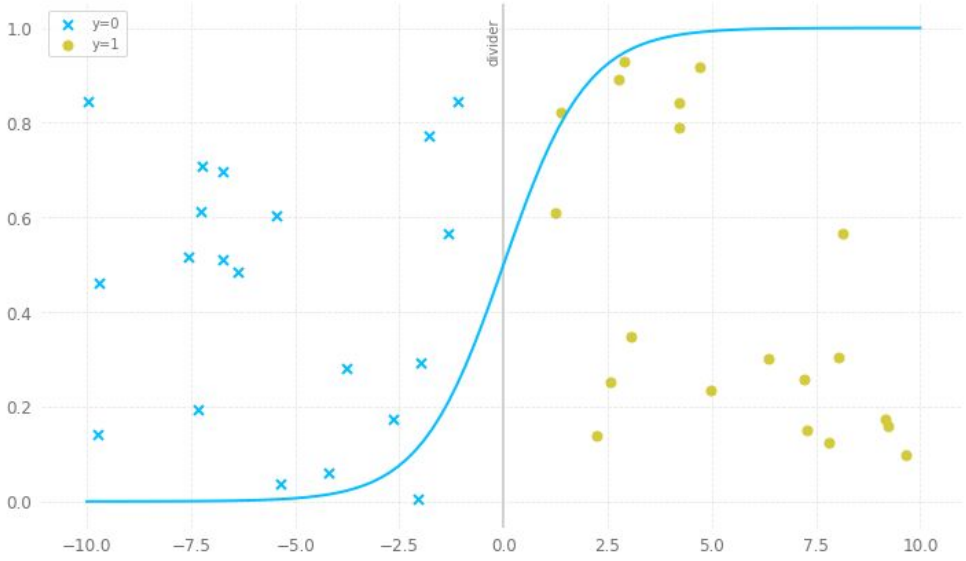
- Activity variables had a lot of overlap, and some activities had too little data - these were reduced or eliminated entirely.
- **Carb_input_new** eliminated for being highly correlated with **bolus** insulin.
- **smbg_new** added little new information, so it was eliminated.

Classification

Binary Logistic Regression

Binary Logistic Regression

- Logistic Regression allows users to classify binary classification problem.
- In this case, we have set increase or decrease in CGM as the categorical variable, and distance patient have walked, basal insulin injecting rate, amount bolus insulin was administered as independent variable.



Data Source - [Binary Classification with Logistic Regression | by Dirk Hornung | Towards Data Science]

Process

- We have tried comparing glucose value to different lag to set as a categorical variable. (Considered insulin intake takes time to take effect)

```
dia$lag5m=lag(dia$IMPUTED)
#5min the least lag for workout
dia$lag15m=lag(dia$IMPUTED,3)
#15min to bolus insulin to take effect
dia$lag1h=lag(dia$IMPUTED,12)
#1hr to bolus insulin to peak
dia[is.na(dia)]=0
```

- Split up 80%of data for training and 20% for testing.

```
set.seed(10)
index <- createDataPartition(dia1ho$ch1h,p=0.8, list=FALSE)
dia1hotr=dia1ho[index,]
dia1hot=dia1ho[-index,]
```

- Ran a regression with glm function. Omitted variables by judging from p-value and rerun.

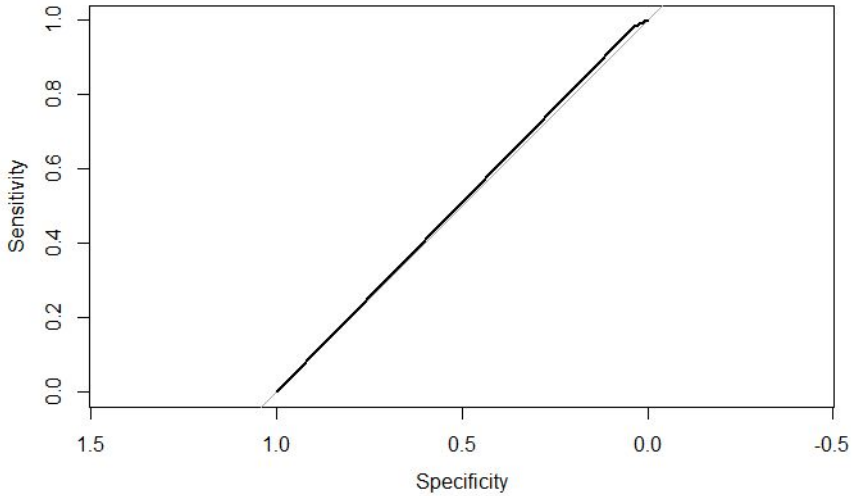
```
model1h<-glm(ch1h~BASAL+bolus+Distance_value,data=d
ia1hotr,family=binomial)
summary(model1h)
model1h<-glm(ch1h~BASAL+bolus,data=dia1hotr,family=
binomial)
summary(model1h)
```

- Compared prediction with actual data with confusion matrix.

```
pred1h <- predict(model1h, dia1hot,
type="response")
pred1ha<-ifelse(pred1h>=0.5, 1, 0)
library(e1071)
confusionMatrix(table(pred1ha,dia1hot$ch1h))
```

Results

```
pred5a    0    1
     0   14    9
     1 1544 2585

               Accuracy : 0.626
                 95% CI : (0.611, 0.6407)
    No Information Rate : 0.6248
    P-Value [Acc > NIR] : 0.4432
```



- Predicting 5 minute lag change in glucose level have highest accuracy of 0.626 compared to sub 60% accuracy.
- However, as the ROC curve shows that this model does not have much predictive power, as the ROC curve looks more of a straight line at 45 degrees.
- The simple model to classify increase and decrease of glucose level after lag does not provide valuable forecast; data given does not provide enough information.

Time series forecasting

ARIMA

Auto Regressive Integrated Moving Average

$$(1 - \phi_1 B - \cdots - \phi_p B^p) \quad (1 - B)^d y_t \quad = \quad c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

AR(p) **d differences** **MA(q)**

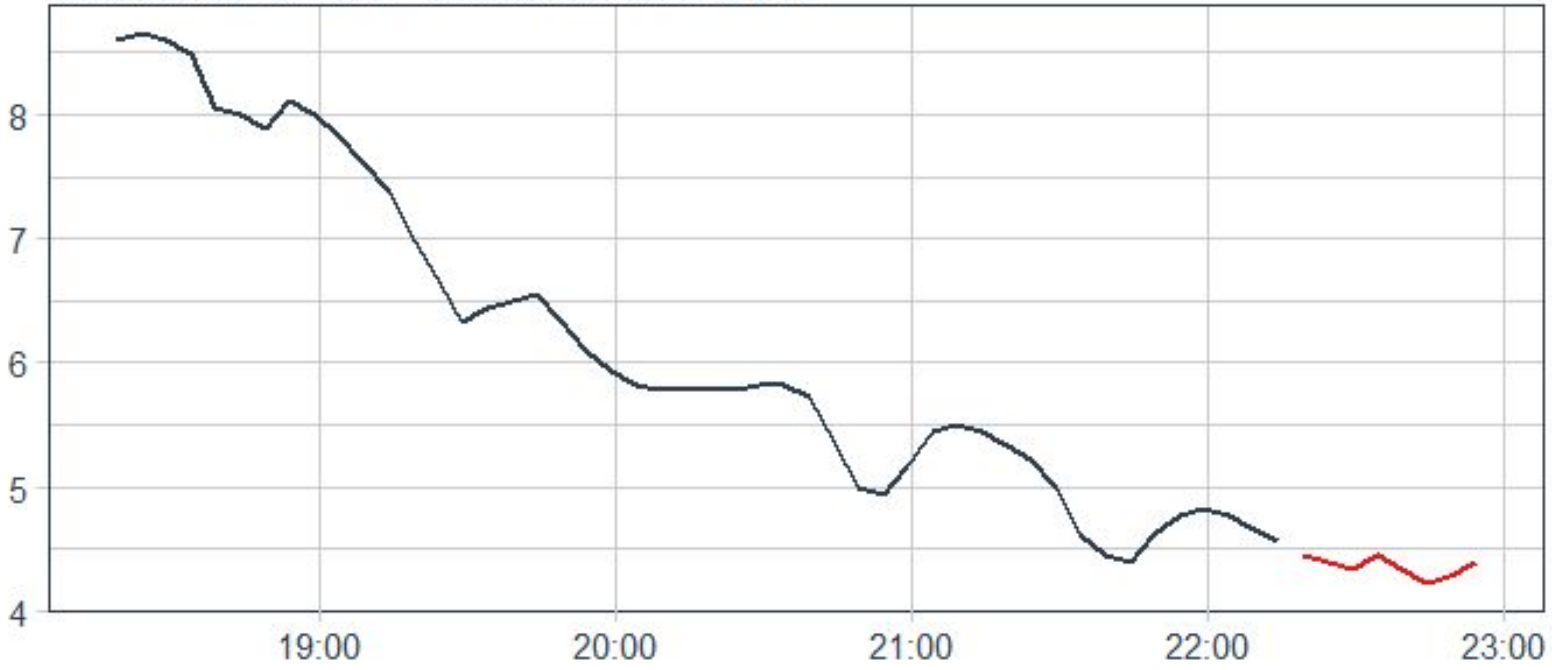
AR(p) $y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$

MA(q) $y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$

Continuous Glucose Monitoring

2019-08-08 18:19:11 ~ 2019-08-08 22:54:10

- Splitted the data to **4 hours(48 data points)** and **40 minutes(8 data points)** as the **training** and the **testing** set.



Stationary & Differencing

Augmented Dickey-Fuller Test

- When difference = 1,
  the p-value = 0.035 in the ADF test so the unit root hypothesis can be rejected at level 0.05.
  ⇒ We can reject the null hypothesis and say that it is stationary.

```
>> adf.test(diff(ts, lag=1))
```

```
data:  ts
Dickey-Fuller = -1.1926, Lag order = 3, p-value = 0.8966
alternative hypothesis: stationary


        Augmented Dickey-Fuller Test

data:  diff(ts, lag = 1)
Dickey-Fuller = -3.7009, Lag order = 3, p-value = 0.035
alternative hypothesis: stationary

p-value smaller than printed p-value
        Augmented Dickey-Fuller Test

data:  diff(ts, lag = 2)
Dickey-Fuller = -4.6068, Lag order = 3, p-value = 0.01
```

Stationary & Differencing

Kwiatkowski–Phillips–Schmidt–Shin Test

- When difference = 1,
  the p-value = 0.1 in the KPSS test so the null hypothesis will NOT be rejected at level 0.05.
  ⇒ We can NOT reject the null hypothesis that the time series is stationary.

```
>> kpss.test(diff(ts, lag=1), null = c("Level", "Trend"), lshort = TRUE)
```

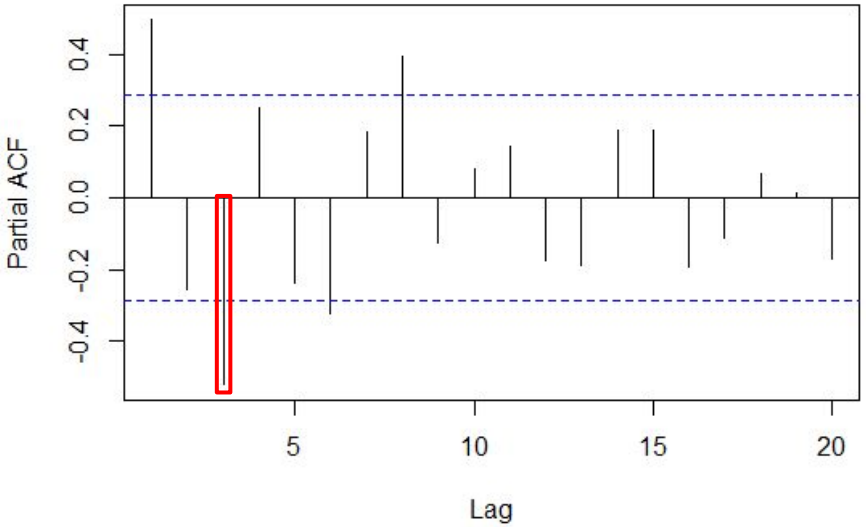
```
p-value smaller than printed p-value
        KPSS Test for Level Stationarity

data:  ts
KPSS Level = 1.1483, Truncation lag parameter = 3, p-value = 0.01

p-value greater than printed p-value
        KPSS Test for Level Stationarity

data:  diff(ts, lag = 1)
KPSS Level = 0.16703, Truncation lag parameter = 3, p-value = 0.1

p-value greater than printed p-value
        KPSS Test for Level Stationarity

data:  diff(ts, lag = 2)
KPSS Level = 0.1898, Truncation lag parameter = 3, p-value = 0.1
```

Order Selection

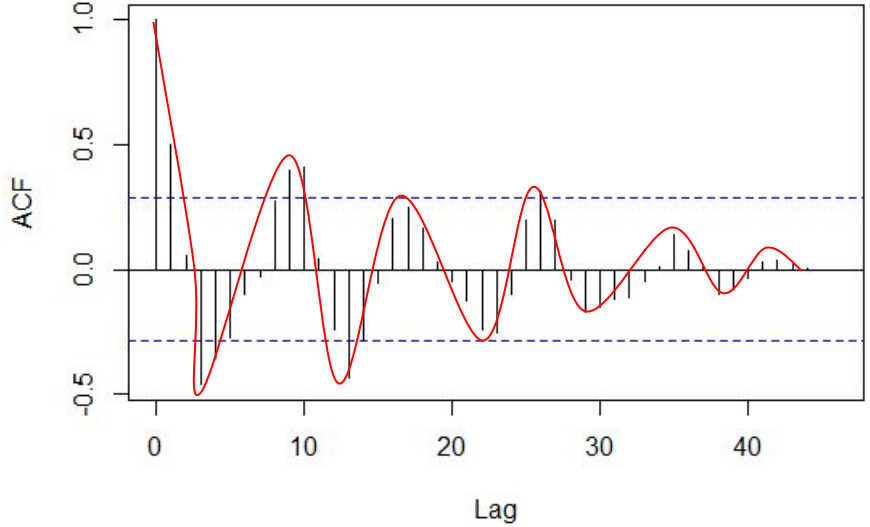
- AutoCorrelation Function
- Partial AutoCorrelation Function

- Since the ACF plot tapers to 0 in some fashion and the PACF pattern has significant values at first 3 lags; then gradually leads to zero afterward ⇒ Conclude this is as an **AR(3)** Model.

```
>> pacf(data,lag=20)
```



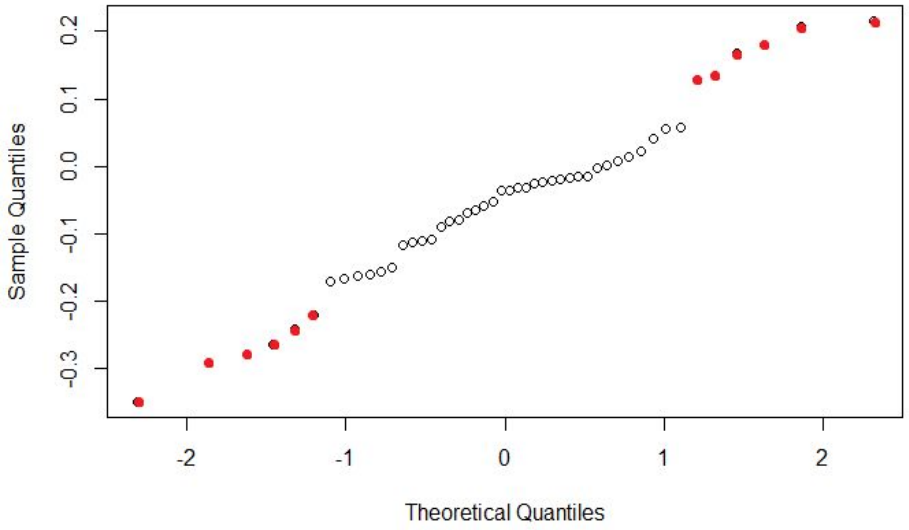

```
>> acf(data,lag=50)
```



Model Fitting & Residual Test

Quantile-Quantile Plot

```
>> qqnorm(arima.model$residuals)
```




```
>> arima(data, order=c(3, 1, 0))
```

```
Call:
arima(x = ts_trn[[20]], order = c(3, 1, 0))

Coefficients:
         ar1     ar2      ar3
      0.6357  0.0851  -0.3694
s.e.  0.1320  0.1604   0.1307

sigma^2 estimated as 0.01917:  log likelihood = 25.77,  aic = -43.54
```

Prediction & Validation

- **Prediction** with 95% confidence interval

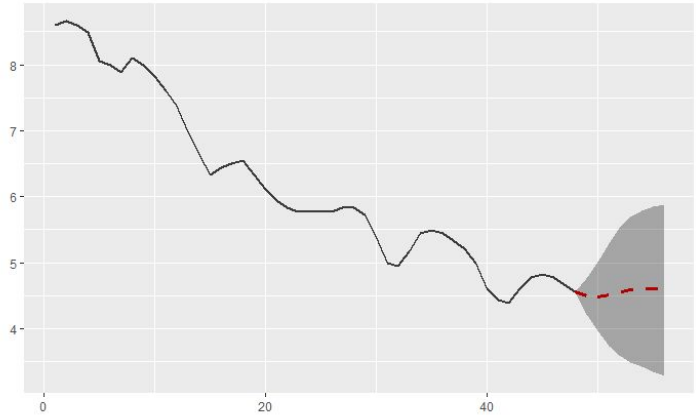
```
>> forecast(arima.Model, level = c(95), length(data))
```

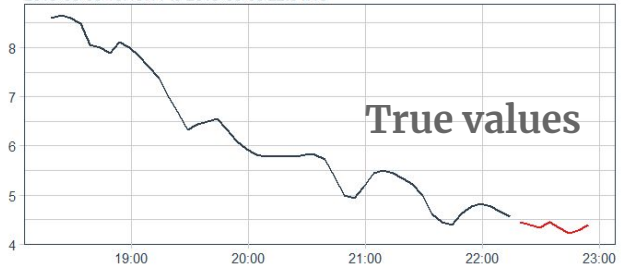
Point Forecast <dbl>	Lo 95 <dbl>	Hi 95 <dbl>
4.492098	4.220729	4.763467
4.485832	3.965577	5.006087
4.517800	3.741166	5.294433
4.559574	3.592990	5.526158
4.591164	3.492518	5.689811

- **Validation** - MAE/RSS/MSE/RMSE

Mean Absolute Error(**MAE**)	0.2912914
Residual Sum of Squares(**RSS**)	1.03907
Mean Squared Error(**MSE**)	0.1298837
Root Mean Square Error (**RMSE**)	0.3603939

- **Forecasting Plot**



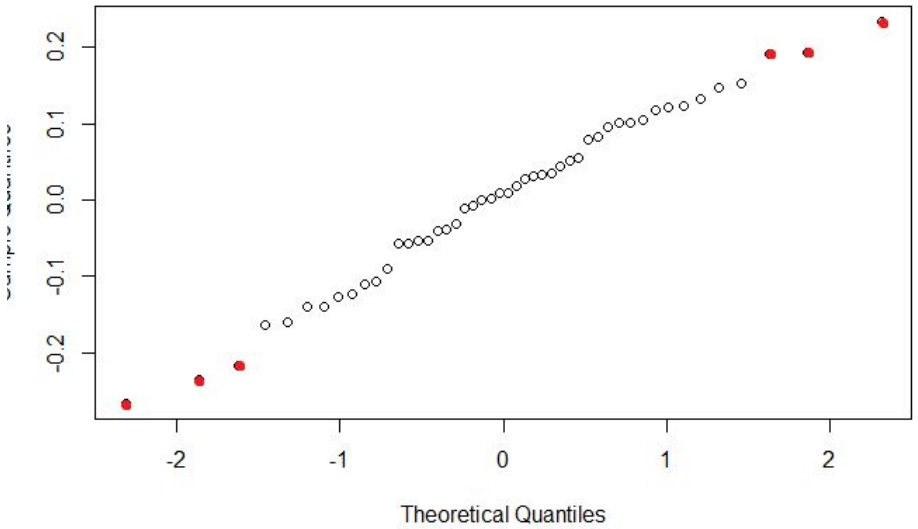


Comparison - *auto.arima()*

- Compare the above result with the model produced by ***auto.arima()*** function.

Quantile-**Q**uantile Plot

```
>> qqnorm(auto.arima.model$residuals)
```




```
>> auto.arima(data)
```

```
Series: ts_trn[[20]]
ARIMA(3,1,1) with drift

Coefficients:
          ar1     ar2      ar3     ma1     drift
      0.1762  0.2661  -0.5893  0.4482  -0.0873
s.e.  0.1937  0.1350   0.1111  0.2402   0.0219

sigma^2 estimated as 0.01534:  log likelihood=33.33
AIC=-54.65   AICc=-52.55   BIC=-43.55
```

Comparison - *auto.arima()*

- **Prediction** with 95% confidence interval

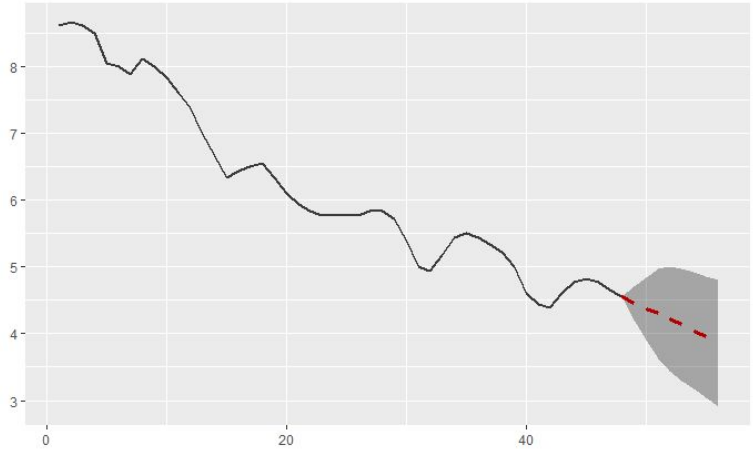
```
>> forecast(auto.arima.Model, level = c(95), length(data))
```

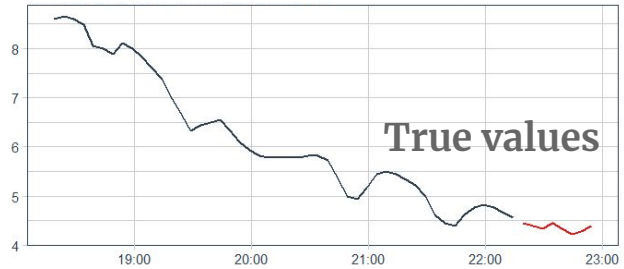
Point Forecast <dbl>	Lo 95 <dbl>	Hi 95 <dbl>
4.454722	4.211935	4.697509
4.373450	3.910339	4.836560
4.298695	3.627606	4.969784
4.220915	3.440156	5.001675
4.135136	3.291820	4.978451

- **Validation** - MAE/RSS/MSE/RMSE

Mean Absolute Error(**MAE**)	0.2381593
Residual Sum of Squares(**RSS**)	1.6756753
Mean Squared Error(**MSE**)	0.08445942
Root Mean Square Error (**RMSE**)	0.290619

- **Forecasting Plot**

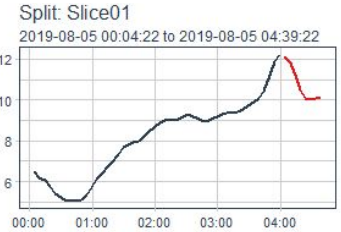


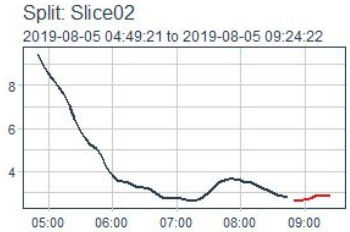


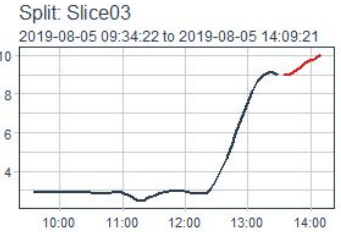
Measure the Overall Performance

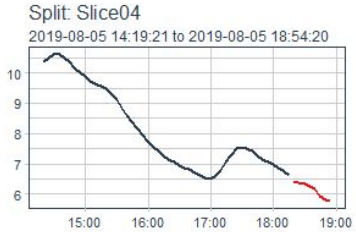
- Use the CGM data(monitoring the blood glucose level per 5 minute) with a duration of 4 days (2019-08-05 00:00:00 ~ 2019-08-09 00:00:00).
- Splitted the data to **4 hours(48 data points)** and **40 minutes(8 data points)** as the **training** and the **testing** set.
- And the **skip time** during this period (five days) is equal to a total **56 data points**.

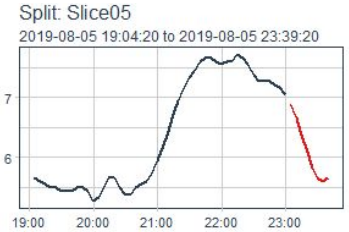
⇒ Total **20** time slices can be obtained.



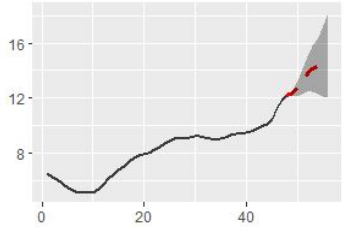


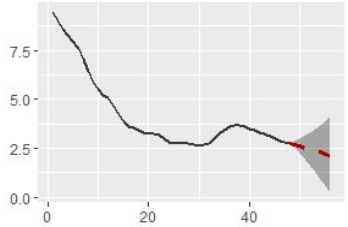


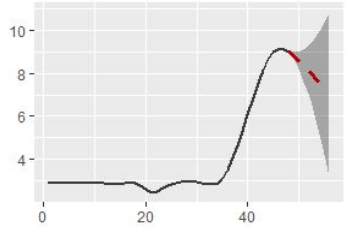


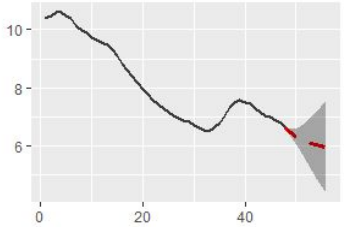


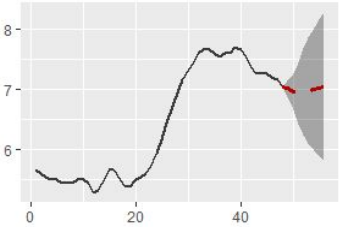
- **Forecasting Plots –** even the blood glucose level trend can NOT be predicted properly.











Time series forecasting

Facebook Prophet

Facebook Prophet

- More suitable for business behavior data with **obvious internal laws**.

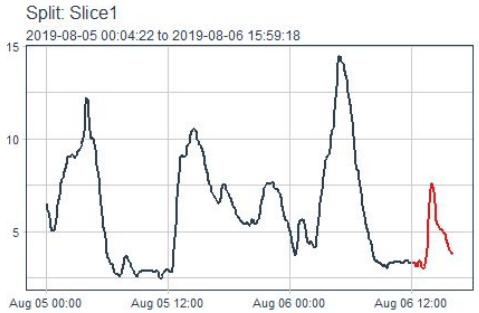
$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

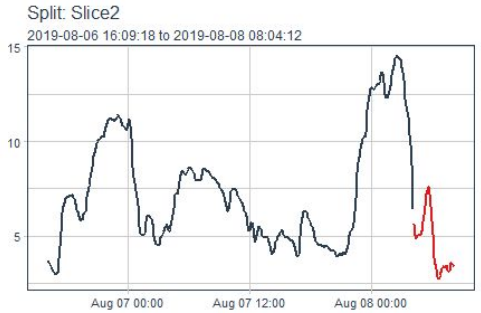
g(t)	Non-periodic changes (trend)
s(t)	Periodic change (eg weekly / annual seasonality)
h(t)	Irregular holiday effects
ε_t	Error term (Normal distribution)

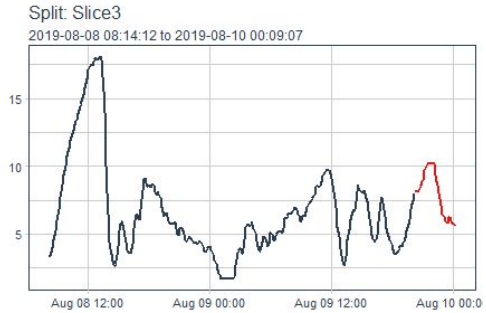
Continuous Glucose Monitoring

2019-08-05 00:00:00 ~ 2019-08-15 00:00:00

- Use the CGM data(monitoring the blood glucose level per 5 minute) with a duration of 10 days. (2019-08-05 00:00:00 ~ 2019-08-15 00:00:00).
- Splitted the data to **1.5 day(12*36 data points)** and **4 hours(48 data points)** as the **training** and the **testing** set.
- And the **skip time** during this period (five days) is equal to a total **12*40 data points**. ⇒ Total **6** time slices.







Facebook Prophet

- **Input data format**

	ds	y
1	2019-08-05 00:04:22	6.49438
2	2019-08-05 00:09:22	6.21684
3	2019-08-05 00:14:23	6.05032
4	2019-08-05 00:19:22	5.77278
5	2019-08-05 00:24:22	5.43973
6	2019-08-05 00:29:22	5.21770
7	2019-08-05 00:34:23	5.05118
8	2019-08-05 00:39:22	5.05118
9	2019-08-05 00:44:22	5.05118
10	2019-08-05 00:49:22	5.10669
11	2019-08-05 00:54:22	5.32872
12	2019-08-05 00:59:22	5.71727

- **Prediction**

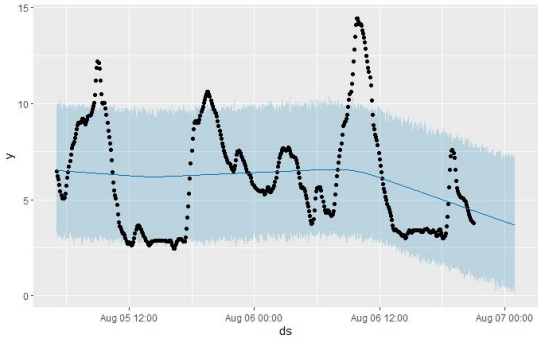
ds <S3: POSIXct>	trend <dbl>	zeros <dbl>	zeros_lower <dbl>	zeros_upper <dbl>	additive_terms <dbl>	additive_terms_lower <dbl>
-05 05:04:22	6.520441	0	0	0	0	0
-05 05:09:22	6.517154	0	0	0	0	0
-05 05:14:23	6.513856	0	0	0	0	0

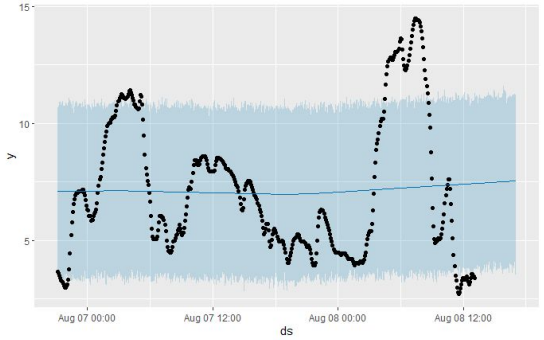
additive_terms_upper <dbl>	multiplicative_terms <dbl>	multiplicative_terms <dbl>	multiplicative_terms_lower <dbl>
0	0	0	0
0	0	0	0
0	0	0	0

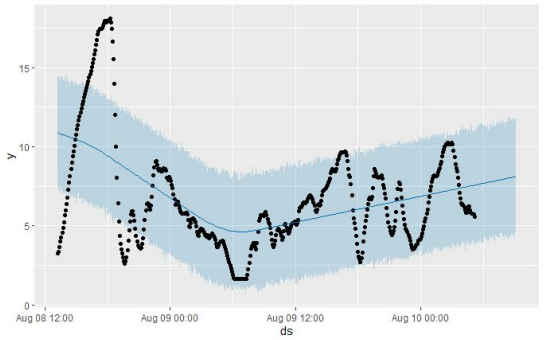
multiplicative_terms_upper <dbl>	yhat_lower <dbl>	yhat_upper <dbl>	trend_lower <dbl>	trend_upper <dbl>	yhat <dbl>
0	3.3089941	10.010278	6.520441	6.520441	6.520441
0	2.9471449	9.941756	6.517154	6.517154	6.517154
0	3.1553788	10.084434	6.513856	6.513856	6.513856

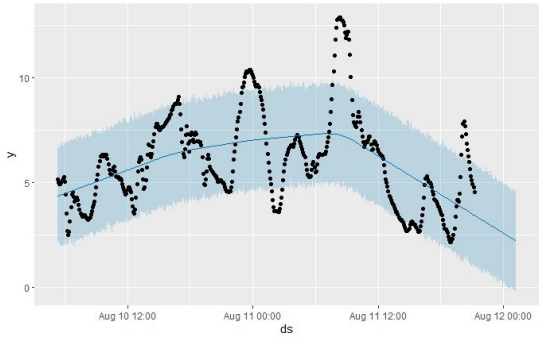
Facebook Prophet

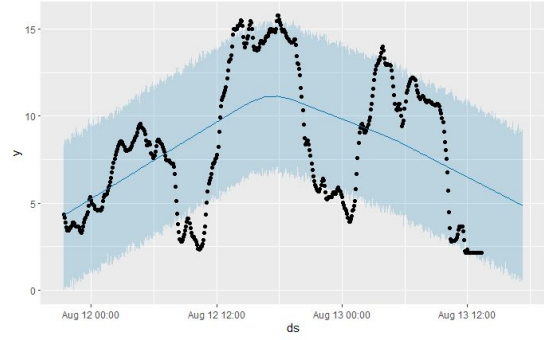
- **Predicting Plots**











- **Overall Validation** - average 5 results

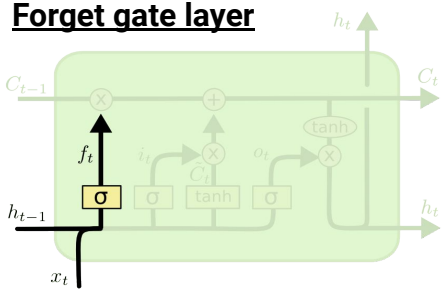
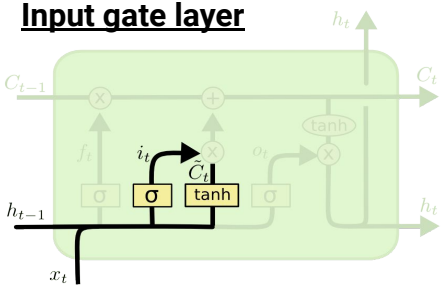
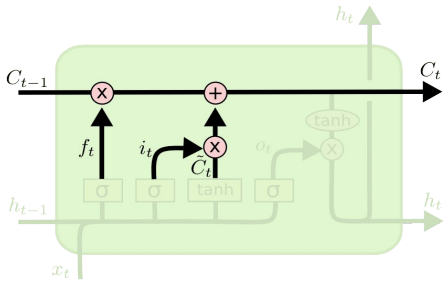
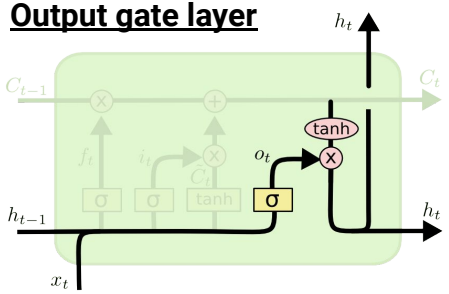
Mean Absolute Error(**MAE**)	10.24718
Residual Sum of Squares(**RSS**)	1426.864
Mean Squared Error(**MSE**)	29.72633
Root Mean Square Error (**RMSE**)	11.77535

Time series forecasting

Univariate LSTM

Long Short-Term Memory

▶️ LSTM is a special kind of **Recurrent Neural Network(RNN)**, capable of learning **long-term dependencies**.

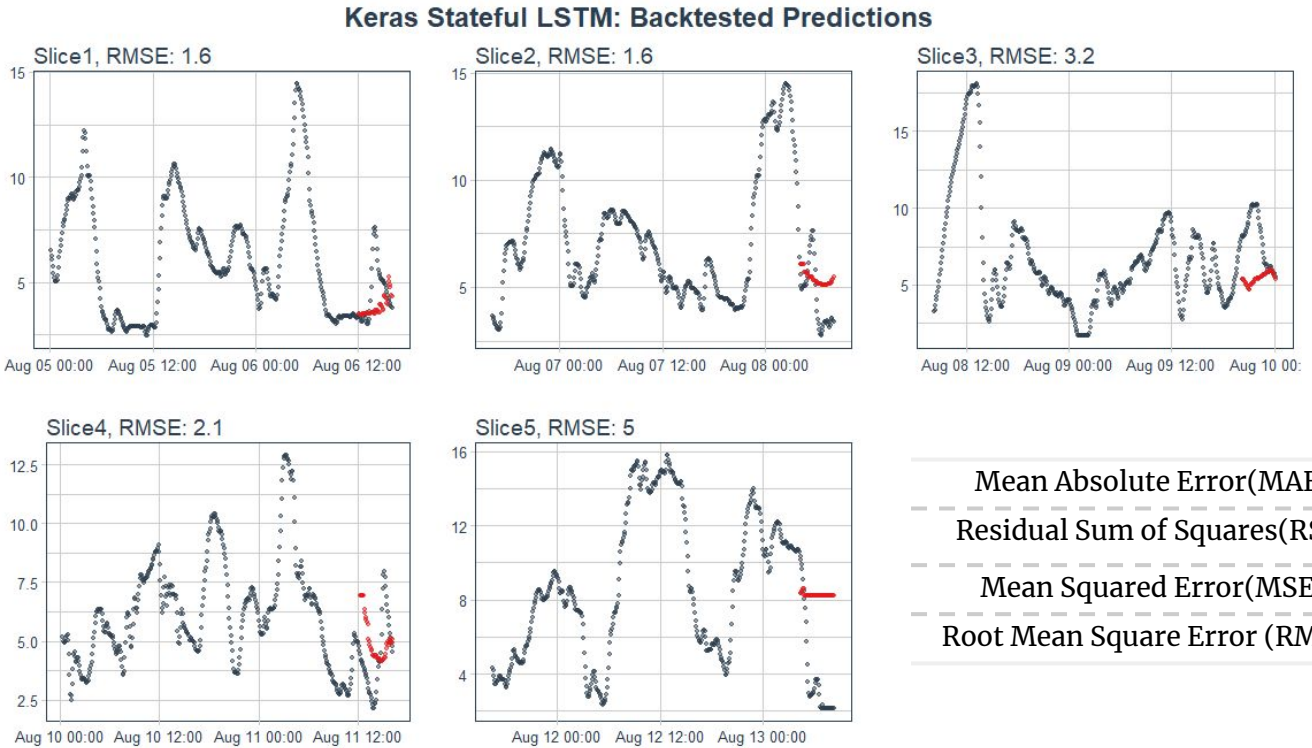
Forget gate layer	**Input gate layer**		**Output gate layer**
			
• Decide what information we're going to ignore from the cell state. $f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$	• Decide what information we're going to store in the cell state. $i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$ $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$		• Decide what we're going to output. $o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$ $h_t = o_t * \tanh\left(C_t\right)$

• Update the old cell state, C_{t-1}.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Dara Source : https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-Term Memory



Mean Absolute Error(MAE)	2.334635
Residual Sum of Squares(RSS)	430.5573
Mean Squared Error(MSE)	8.969945
Root Mean Square Error (RMSE)	2.707877

Time series forecasting

Multivariate LSTM

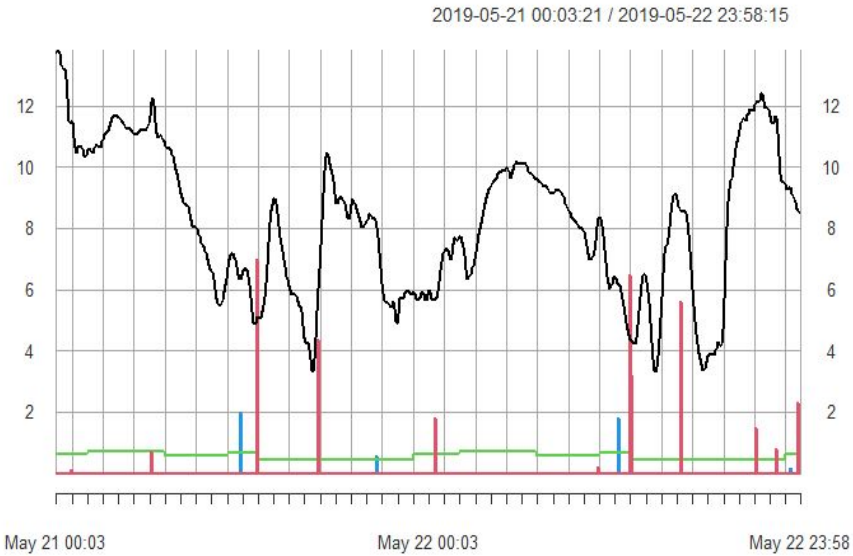
Why Multivariate LSTM?

- Many of the variables included in the data are **sparse**, meaning that the rows contain many 0 values and the overall data matrix is **rank-deficient**. This is bad for regression, as we are unable to solve for the solution of a rank deficient matrix.

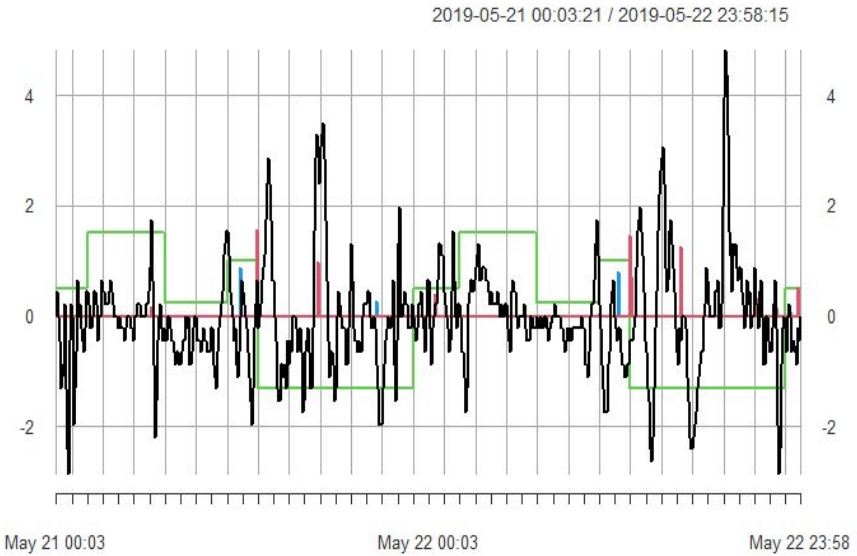
- Therefore we don’t use vector ARIMA, and instead go directly to neural network models.

Data processing for multivariate analysis

CGM / basal / bolus / distance_value

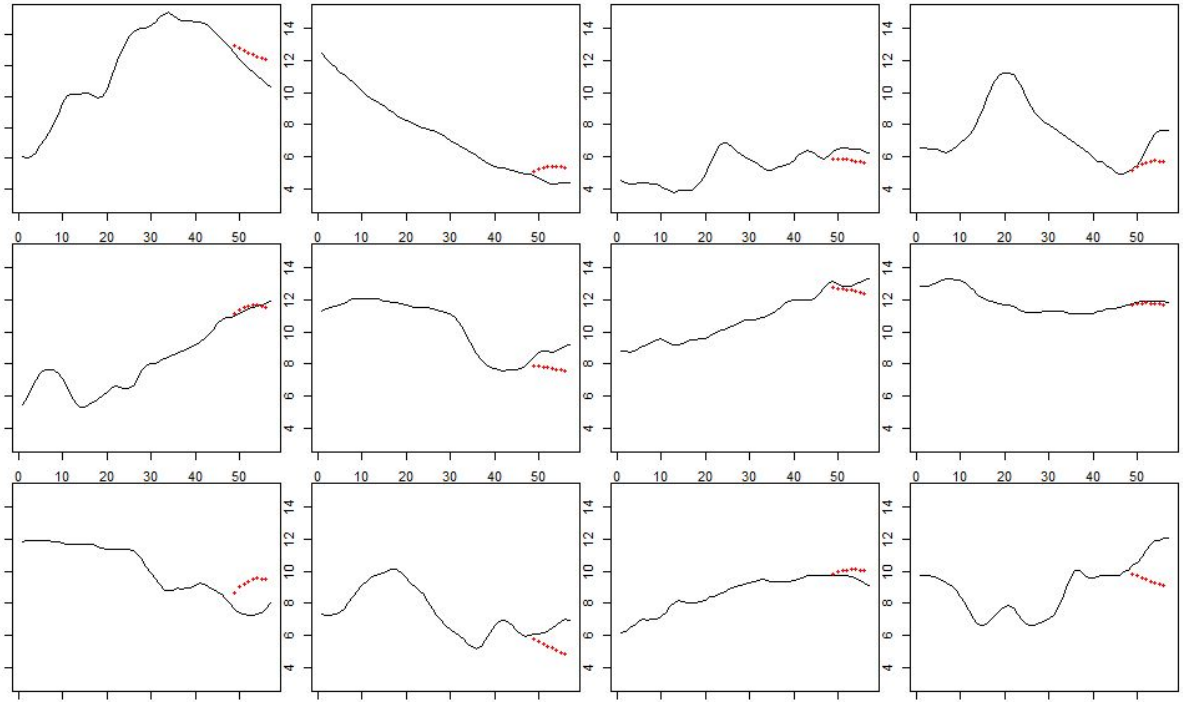


Data before differencing and rescaling



Data after differencing and rescaling

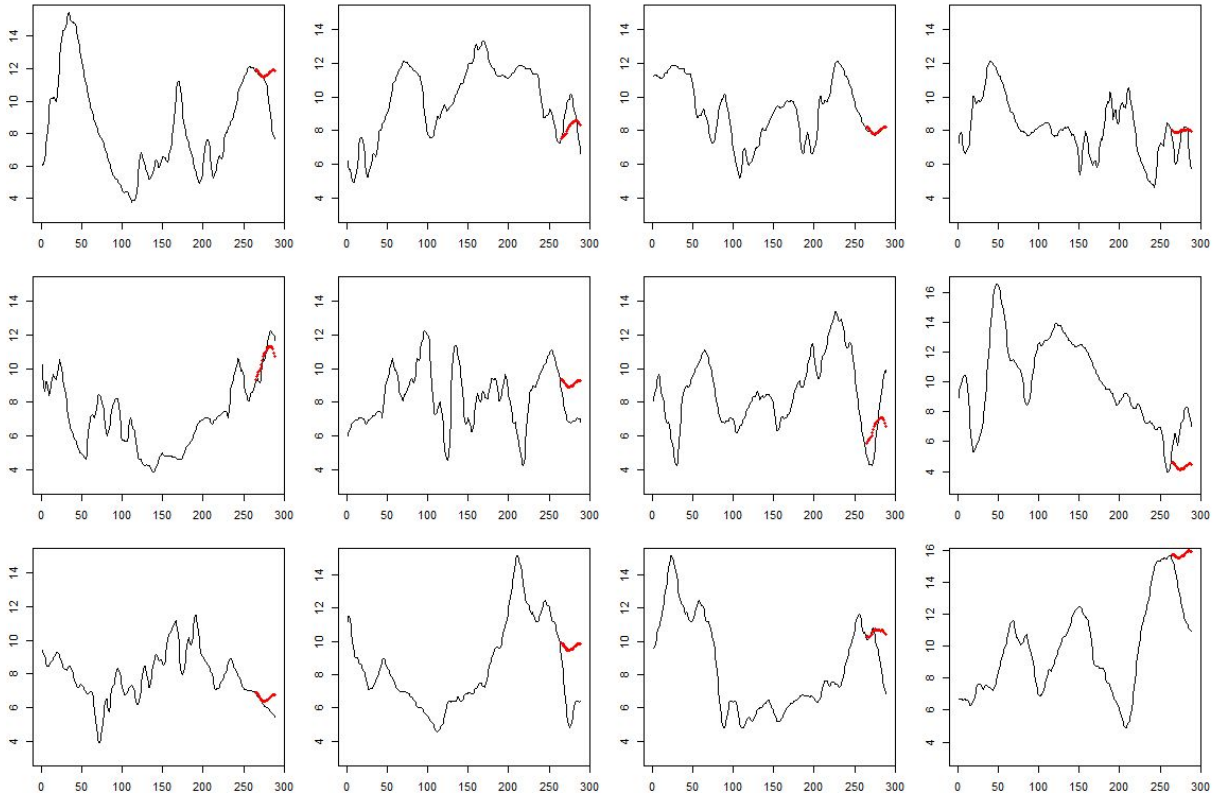
Forecasting



- 4 hours(48 data points) input matrix
- 40 minutes(8 data points) forecasting output.

MAE	0.7371
MSE	1.2566
RMSE	1.1210

Forecasting

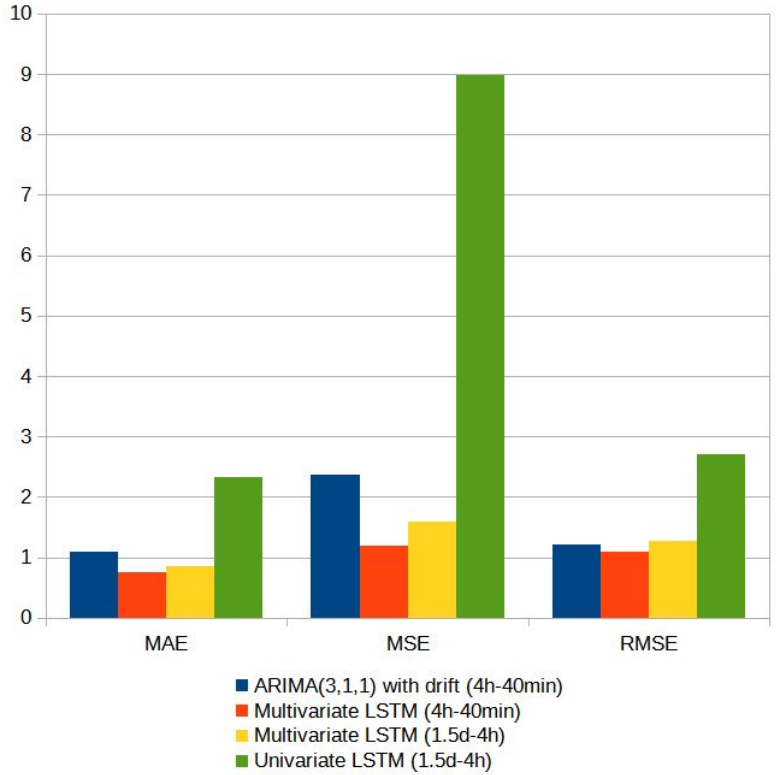


- 22 hours (12*22 data points) input matrix
- 2 hours (24 data points) forecasting output.

MAE	0.8546
MSE	1.6038
RMSE	1.2664

Time Series Forecasting Conclusion

- Our metrics: **MAE, MSE, RMSE =** average differences between forecasted and true values
- **Multivariate LSTM** performed the best out of the three methods to forecast CGM up to 4 hours
- How good is it really?
  Further investigation is recommended using neural network methods and multivariate analysis.



Pima Indian Diabetes Dataset

Pima Indian Diabetes Dataset

Column	# Observations	Missing
Pregnancies	768	0
Glucose	763	5
BloodPressure	733	35
SkinThickness	541	227
Insulin	394	374
BMI	757	11
DiabetesPedigreeFunction	768	0
Age	768	0
Outcome	768	0

- 8 features
- 1 binary outcome
- Total Observations:768

Dealing with Missing Data

```
#2.2 Replace 0 with mean of the each feature
imputer = SimpleImputer(missing_values=0,strategy='mean')
imputer = imputer.fit(X[:, 1:6])
X[:,1:6] = imputer.transform(X[:,1:6])
```

Column	# Observations	Missing
Pregnancies	768	0
Glucose	768	0
BloodPressure	768	0
SkinThickness	768	0
Insulin	768	0
BMI	768	0
DiabetesPedigreeFunction	768	0
Age	768	0
Outcome	768	0

Note: Pregnencies feature was not affected by this step due to having 0 pregnencies does make sense.

Data Normalization

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.352941	0.670968	0.489796	0.304348	0.170130	0.314928	0.234415	0.483333	1
1	0.058824	0.264516	0.428571	0.239130	0.170130	0.171779	0.116567	0.166667	0
2	0.470588	0.896774	0.408163	0.240798	0.170130	0.104294	0.253629	0.183333	1
3	0.058824	0.290323	0.428571	0.173913	0.096154	0.202454	0.038002	0.000000	0
4	0.000000	0.600000	0.163265	0.304348	0.185096	0.509202	0.943638	0.200000	1

Model/Feature Selection

	VIF Factor	features
0	35.0	Intercept
1	1.4	Pregnancies
2	1.3	Glucose
3	1.2	BloodPressure
4	1.5	SkinThickness
5	1.4	Insulin
6	1.3	BMI
7	1.1	DiabetesPedigreeFunction
8	1.6	Age

The VIF value of each feature was calculated to check the existence of multicollinearity.

- None of the features were bigger than 10.
- No Multicollinearity.

Outlier/Leverage Detection/Drop

- Outlier Detection: 350
  - Studentized Deleted Residual
- Influential Points Detection: 229
  - Cook's Distance
  - DFBETAS
  - DFFITS

- Leverage Points (Rule of Thumb):

5 9 14 46 50 59 194 229 248 333 358 371 372 446 454 538 580 585 685 707

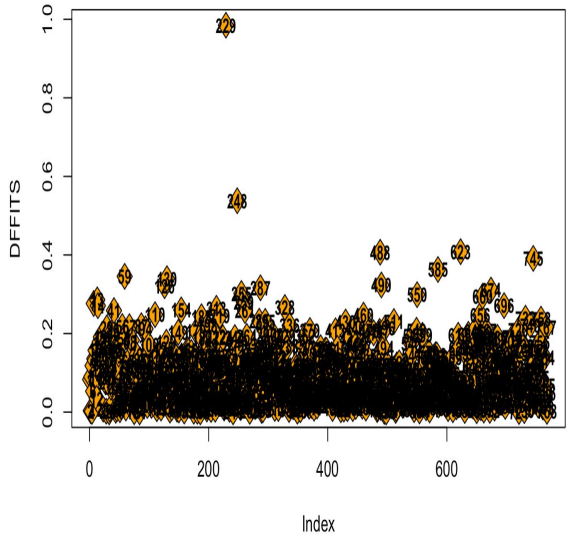
```
#Extreme X values and using Rule of Thumb
hv$warn <- ifelse(hv[,'hat_val']>3*hatMean,"extreme","non_extreme")
extremes <- subset(hv, warn=="extreme")
ext_x_val <- data[c(rownames(extremes)),]
extremes
as.numeric(rownames(extremes))
```

	cooksd <dbl>	cd_threshold <chr>	abs.dffits. <dbl>	dff_threshold <chr>	X.Intercept. <dbl>	Pregnancies <dbl>	Glucose <dbl>	BloodPressure <dbl>
229	0.06650009	X	0.7766899	Consider influential	0.07567425	0.07881857	0.00586766	0.004898616

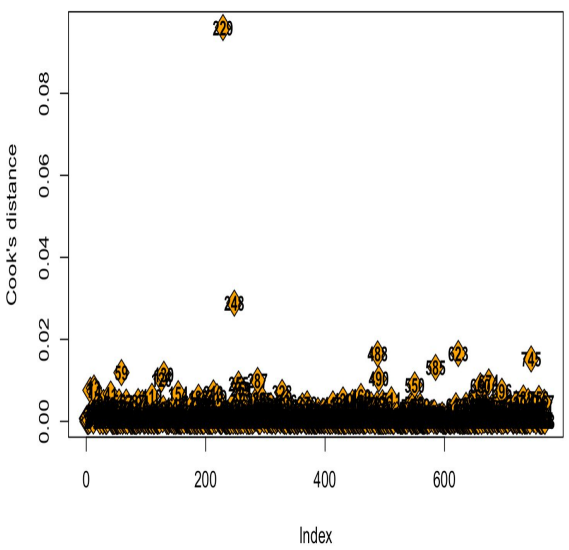
	studres(model) <dbl>	check <chr>
350	3.16618	outlier

Influential Points Detection

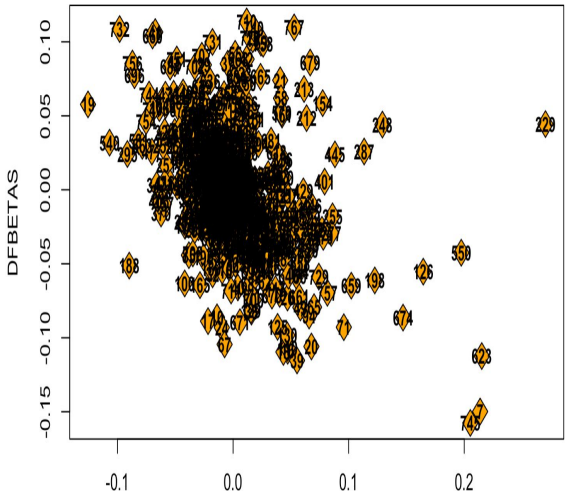
DFFITS



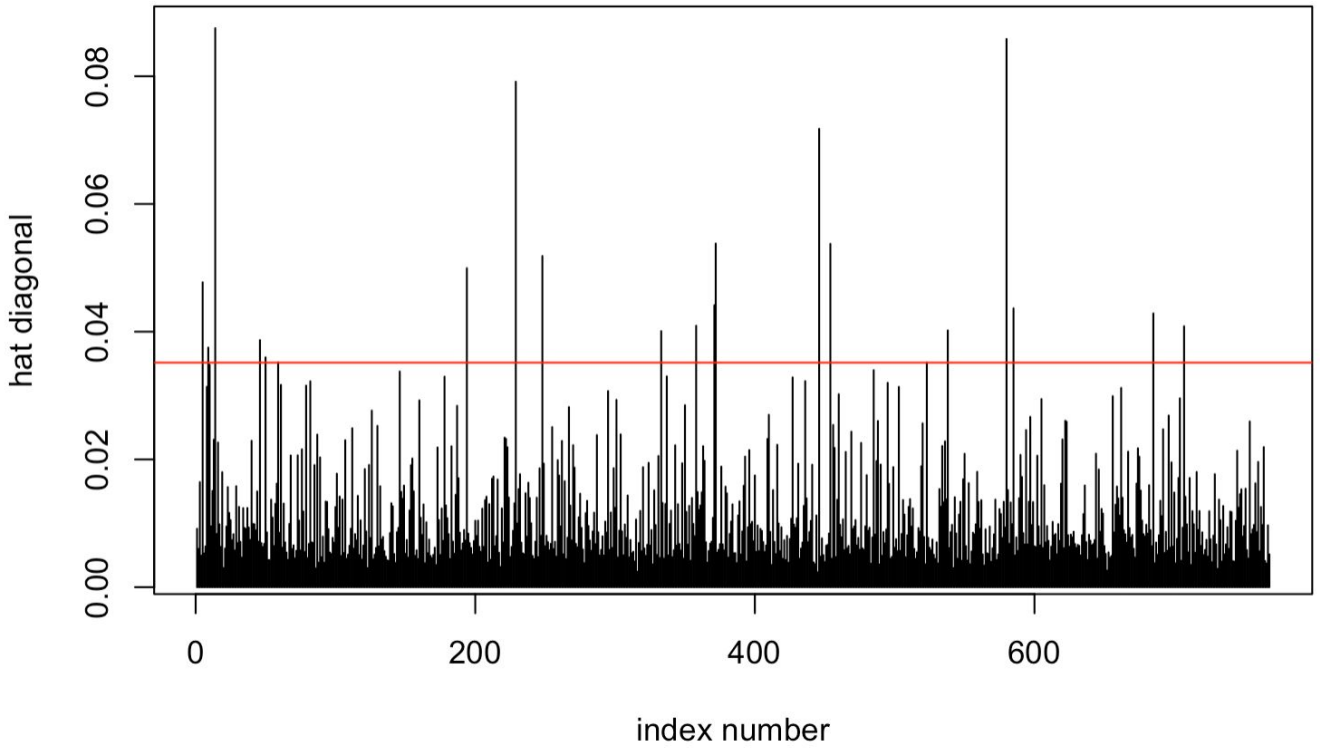
Cook's Distance



DFBETAS



Leverage Detection



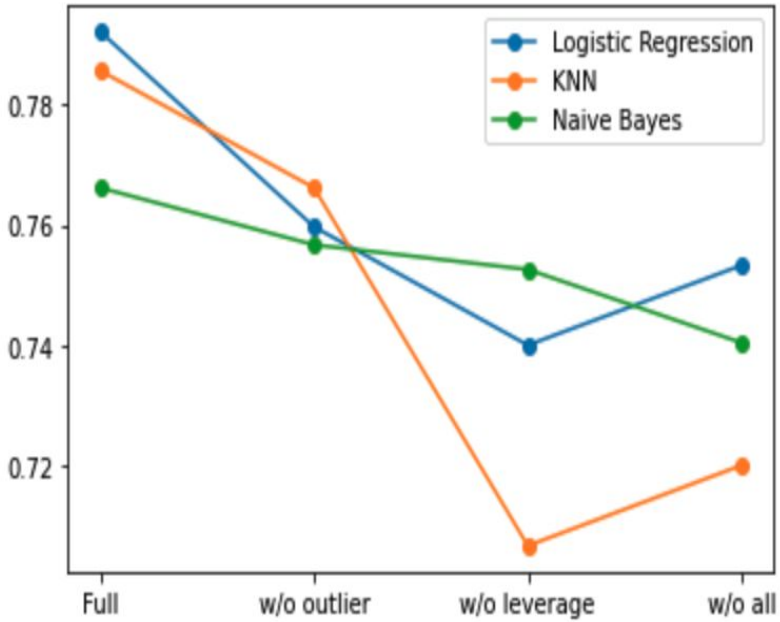
LR vs KNN vs NB

LR		Full data	w/o outlier	w/o leverage	w/o All
	Test Accuracy	0.7922	0.7597	0.7400	0.7533

KNN		Full data	w/o outlier	w/o leverage	w/o All
	Test Accuracy	0.7857	0.7662	0.7066	0.72

NB		Full data	w/o outlier	w/o leverage	w/o All
	Test Accuracy	0.7662	0.7567	0.7526	0.7404

Comparison/Conclusion



LR > NB > KNN