Gestational Diabetes Onset Estimation

By Jessica Lewis

Introduction

Context

- Gestational diabetes (GDB) is a type of diabetes that occurs during pregnancy.
- GDB is caused by mismanagement of blood sugar uptake by the placenta, which takes over hormone production for both the fetus and the host.
- GDB is typically diagnosed between 24 and 28 weeks of pregnancy.
- In March of 2021 was diagnosed I at 27 weeks and at 29 weeks I changed my diet and began taking blood sugar readings 4 times per day.

Problem Statement

Can I reverse a time series of gestational diabetes data to pinpoint the onset in my pregnancy?

A NOTE ON TERMINOLOGY

Blood glucose is the same thing as blood sugar.
Readings and records both mean rows in the dataframe.

Success Criteria

This project a success if either of these criteria succeed.

1. Make predictions for dates *before* my data starts
2. My predictions cross a threshold chosen using target blood sugar levels for pregnant people. These values are dependent on the target variable:
   - Fasting Levels > 95
   - Non-Fasting Levels > 130
   - Daily average of both > 120

Approach

The Data

Collection

Recorded by me in a mobile app called One Touch Reveal.

Saved 4 blood sugar readings and 6 carb amounts per day.

Cleaning

The data exported fairly cleanly with one record per row. I set the index and removed some nulls and duplicates.

Feature Extraction

Added columns:

- Daycount
- Subtype
- outOfRange

New dataframe for daily aggregations

Dataframes

Full

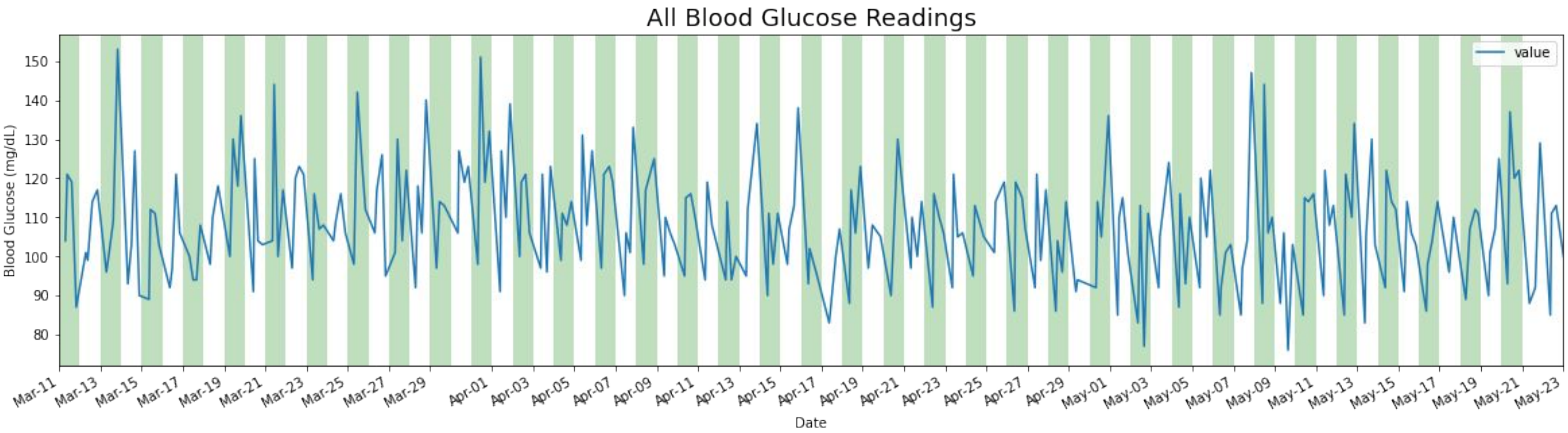
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 683 entries, 2021-03-11 07:11
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   type         683 non-null    object
 1   value        683 non-null    int64
 2   unit         683 non-null    object
 3   month        683 non-null    int64
 4   date         683 non-null    object
 5   daycount     683 non-null    int64
 6   subtype      683 non-null    object
 7   outOfRange   683 non-null    bool
dtypes: bool(1), int64(3), object(4)
memory usage: 59.5+ KB
```

Daily

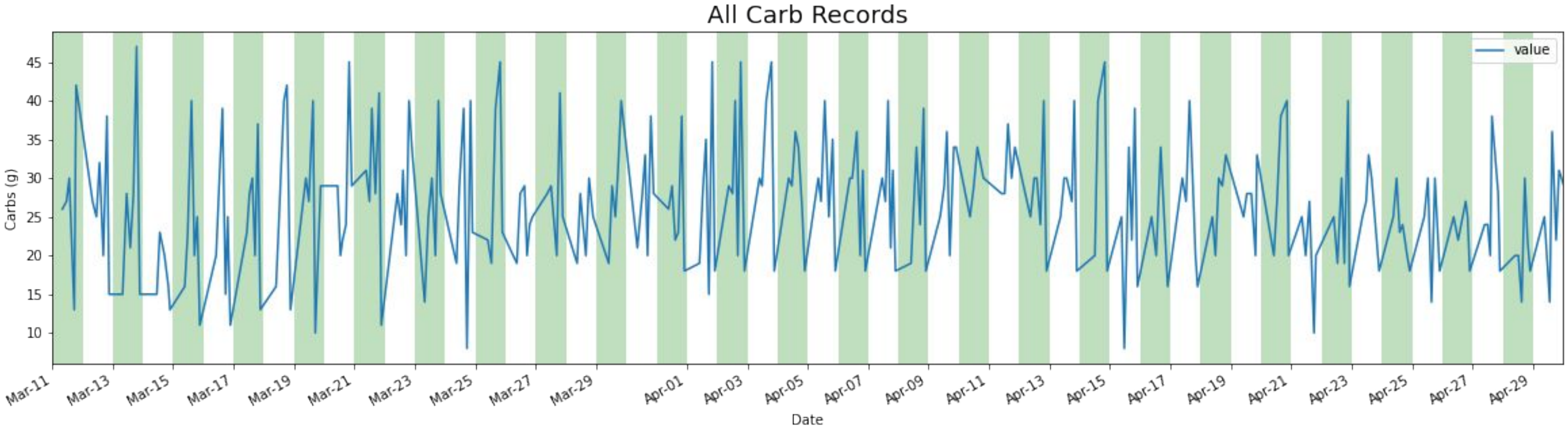
```
<class 'pandas.core.frame.DataFrame'>
Index: 73 entries, 2021-03-11 to 2021-05-2
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   daycount     73 non-null     int64
 1   bg_fasting   73 non-null     float64
 2   bg_avg       73 non-null     int64
 3   carbs_sum    73 non-null     int64
 4   meds_dose    73 non-null     int64
dtypes: float64(1), int64(4)
memory usage: 5.5+ KB
```

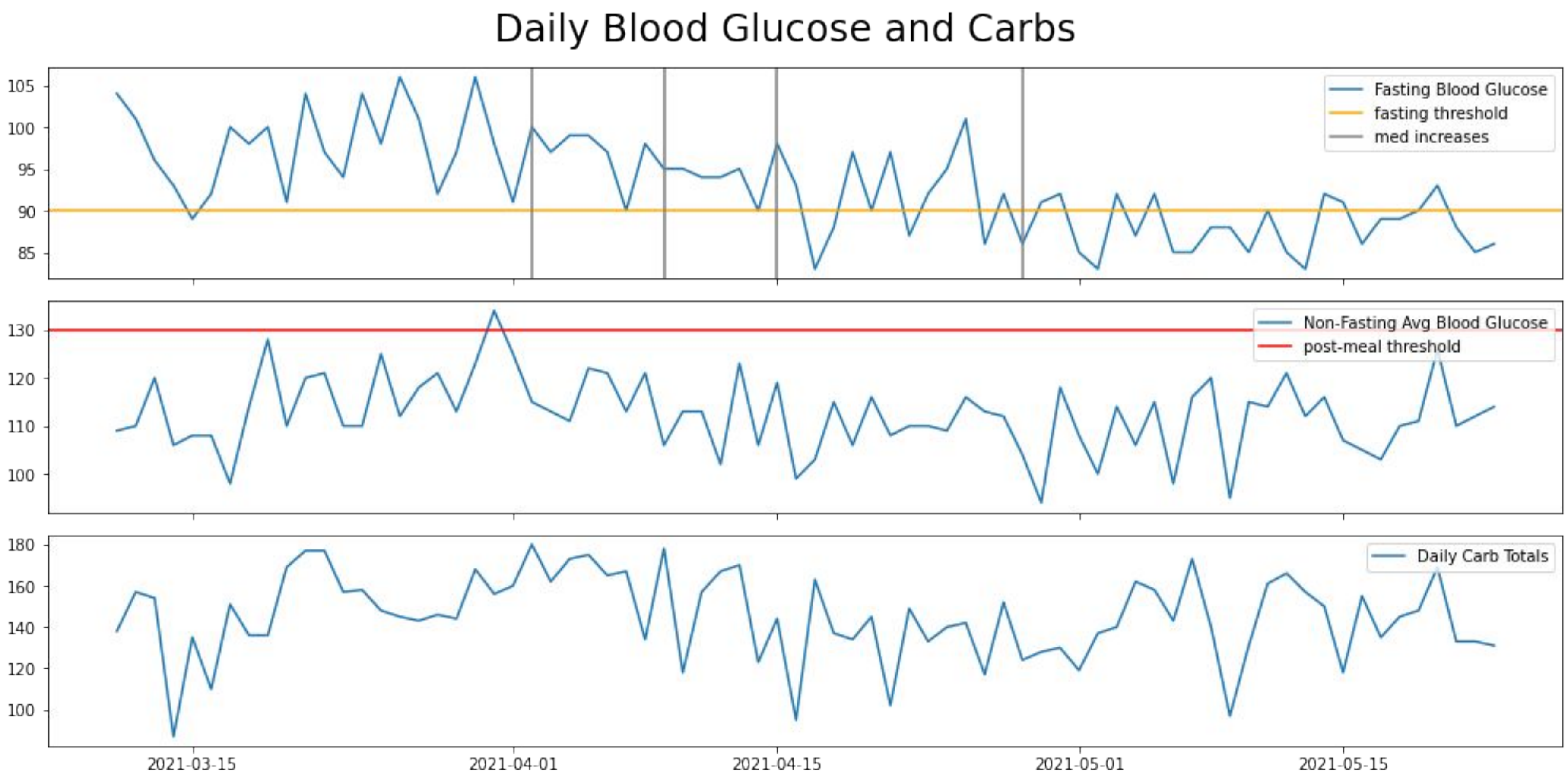
Exploratory Data Analysis

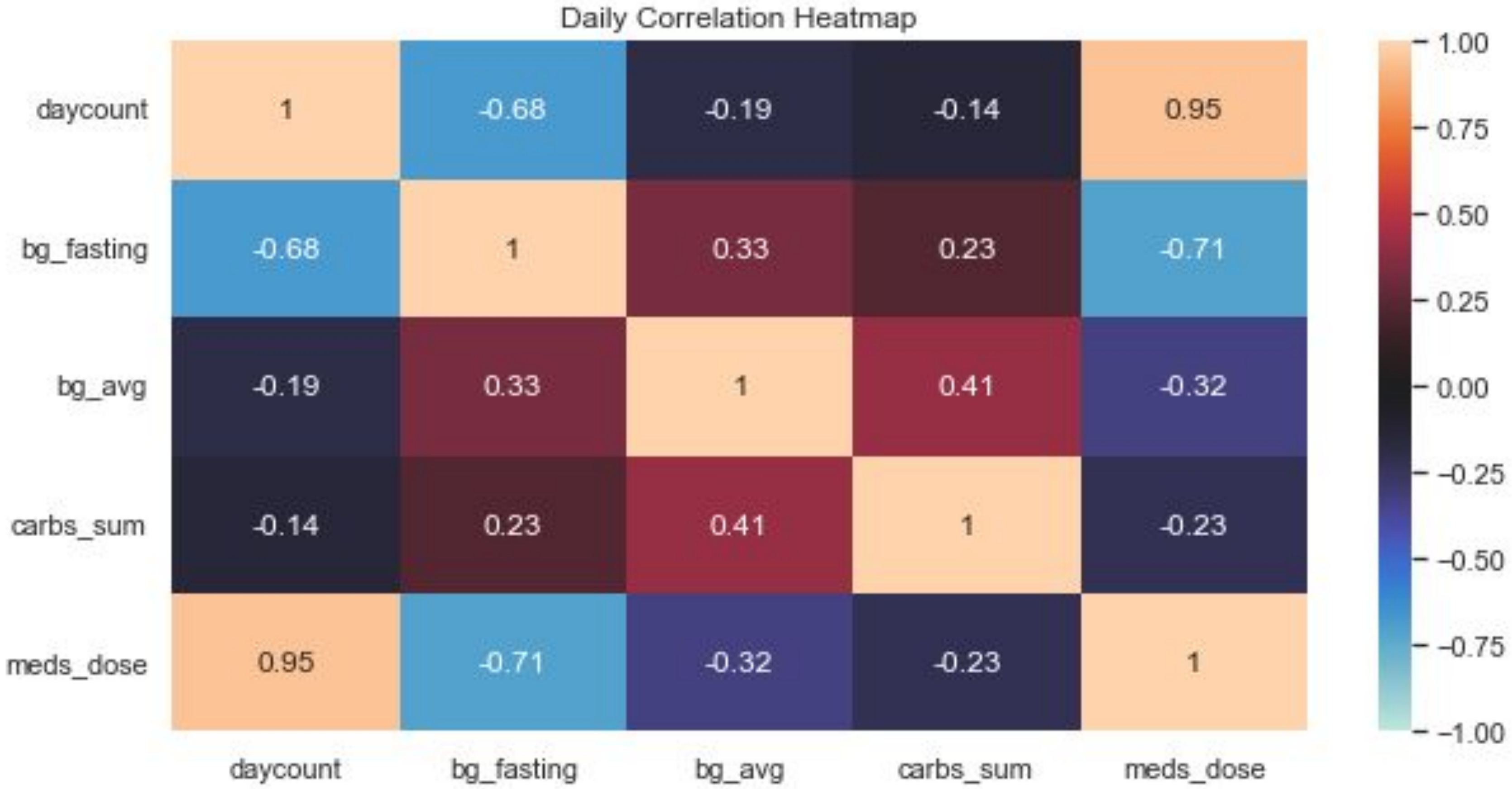
Full dataframe: Blood Sugar Readings



Full dataframe: Carb Records




Daily Blood Glucose and Carbs
Fasting Blood Glucose
fasting threshold
med increases
Non-Fasting Avg Blood Glucose
post-meal threshold
Daily Carb Totals
105
100
95
90
85
130
120
110
100
180
160
140
120
100
2021-03-15
2021-04-01
2021-04-15
2021-05-01
2021-05-15



Daily Correlation Heatmap
daycount
bg_fasting
bg_avg
carbs_sum
meds_dose
1
-0.68
-0.19
-0.14
0.95
-0.68
1
0.33
0.23
-0.71
-0.19
0.33
1
0.41
-0.32
-0.14
0.23
0.41
1
-0.23
0.95
-0.71
-0.32
-0.23
1
daycount
bg_fasting
bg_avg
carbs_sum
meds_dose
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

Pre-processing

Timeseries Reversal

Functions that required a forward monotonic series:

- seasonal _decompose

Functions that required a chronological frequency:

- plot_predict

Reversing the DateTime Index

Timeseries forecasting functions are not intended to “predict” back in time. So I knew early on that I’d have to reverse my timeseries before running it through models.

What I did not expect was the struggle with monotonicity.

In the end I reversed the timeseries data but maintained the original DateTime Index. The dates were wrong, but the timeframe was correct and the data still happened in the order I needed for predictions.

Target Variable

At this point I had to consider what to use to make predictions

I identified three potential target variables that I would consider until choosing a final model:

- All Blood Sugar Values
- Daily Fasting Values
- Daily Non-fasting Averages

Testing for Stationarity

Daily Fasting

Dickey Fuller and KPSS resulted non-stationary

ACF suggests one level of differencing

I went with 1 order of differencing

Daily Avg

Dickey Fuller and KPSS results: BARELY non-stationary

ACF indicates stationarity

1 order of differencing was too much. I went with 0.

All Blood Sugar

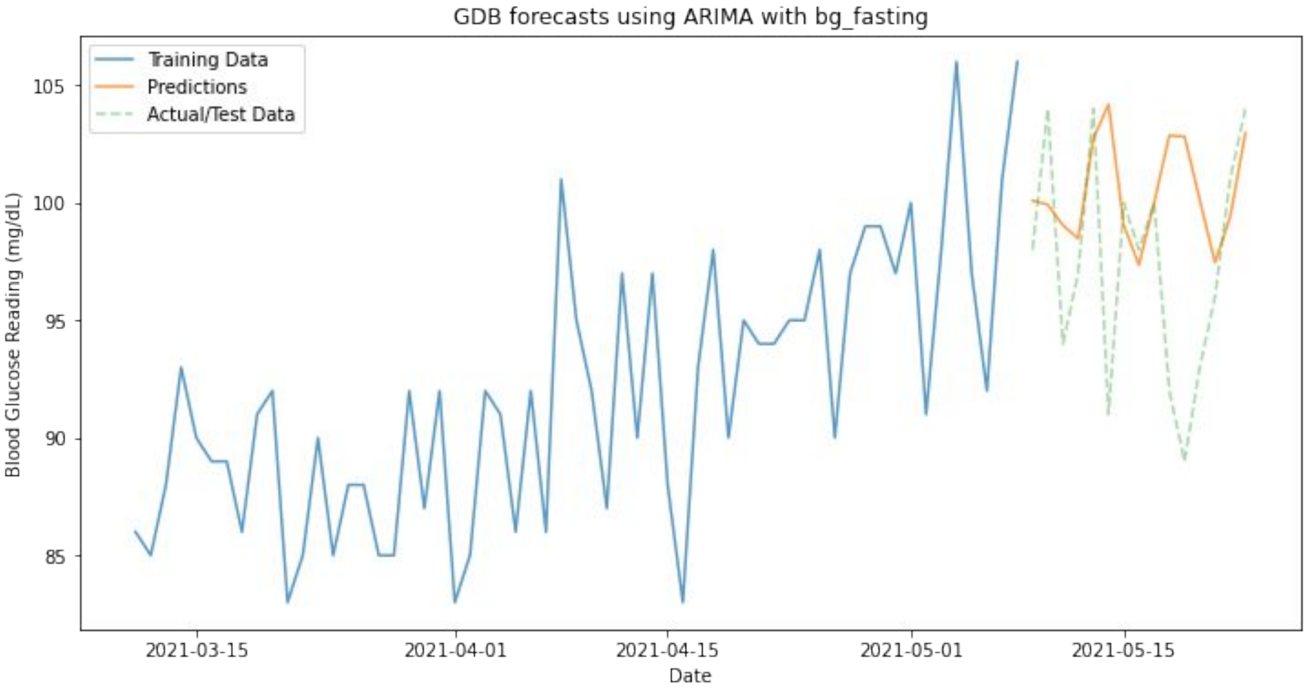
Dickey Fuller and KPSS resulted non-stationary

ACF indicates stationarity

Overall inconclusive. I went with 0 orders of differencing

Model Selection

Daily Fasting



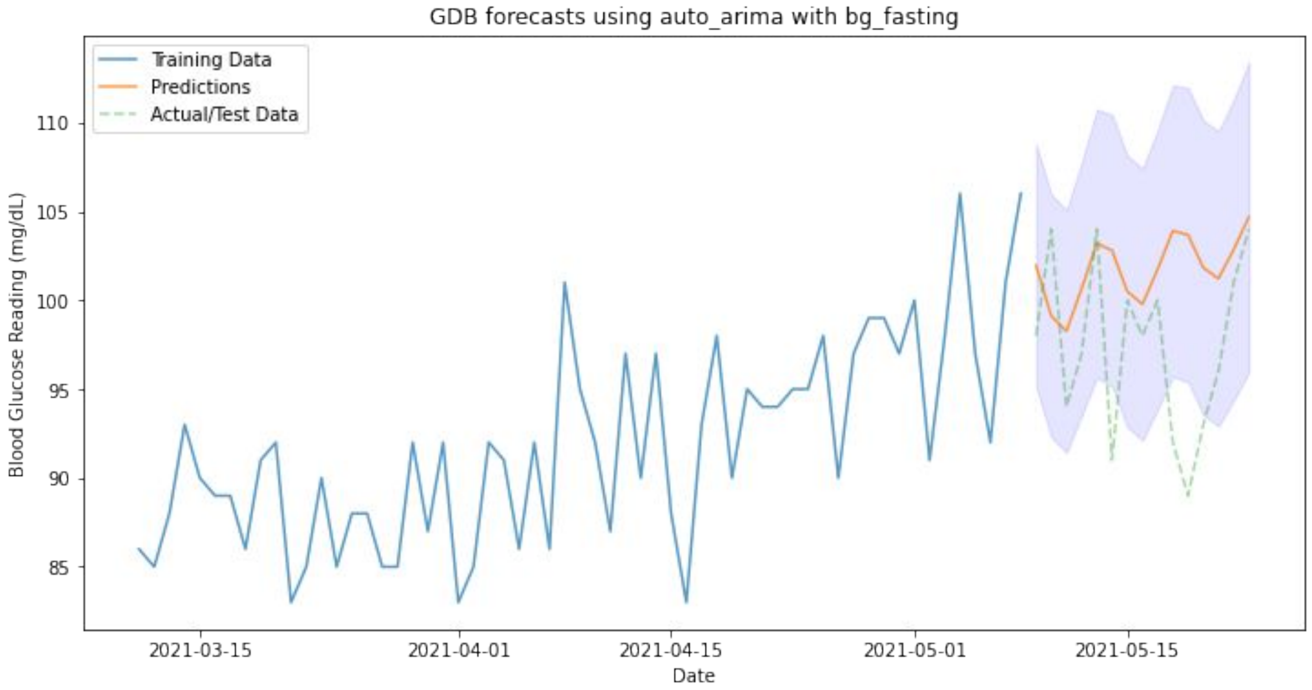
ARIMA

Order Selection*:

1. (5, 1, 3)
2. (5, 1, 4)
3. (4, 1, 7)

*Based on AIC values

Daily Fasting



auto_arima

Model Order:
(2, 1, 3)

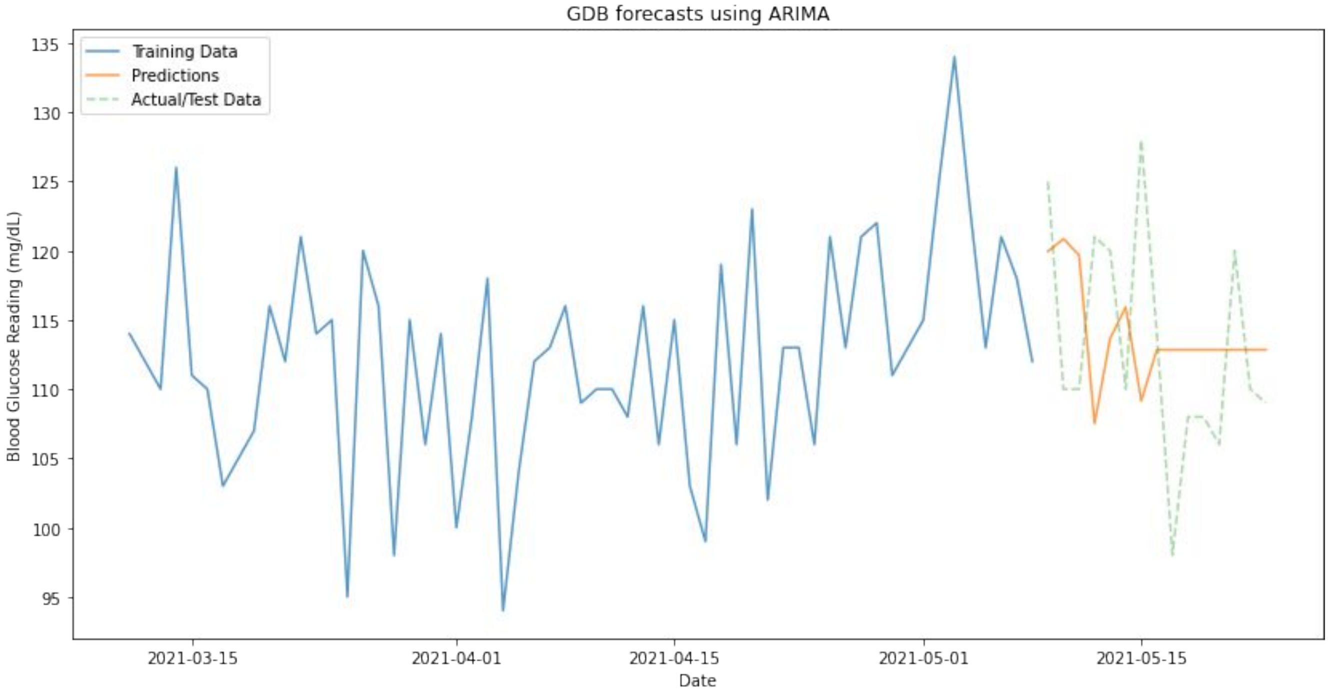
Model Metrics: Daily Fasting

	RMSE	Standardized
ARIMA	6.2781	1.3353
Auto Arima	6.7511	1.4359

The ARIMA model is performing better than the auto-arima.

I’m not sure I’m seeing the downward trend I’d like to see at the very end of the predictions; there might be a slight decrease in the ARIMA model, which would account for the better RMSE.

Daily Blood Sugar Average



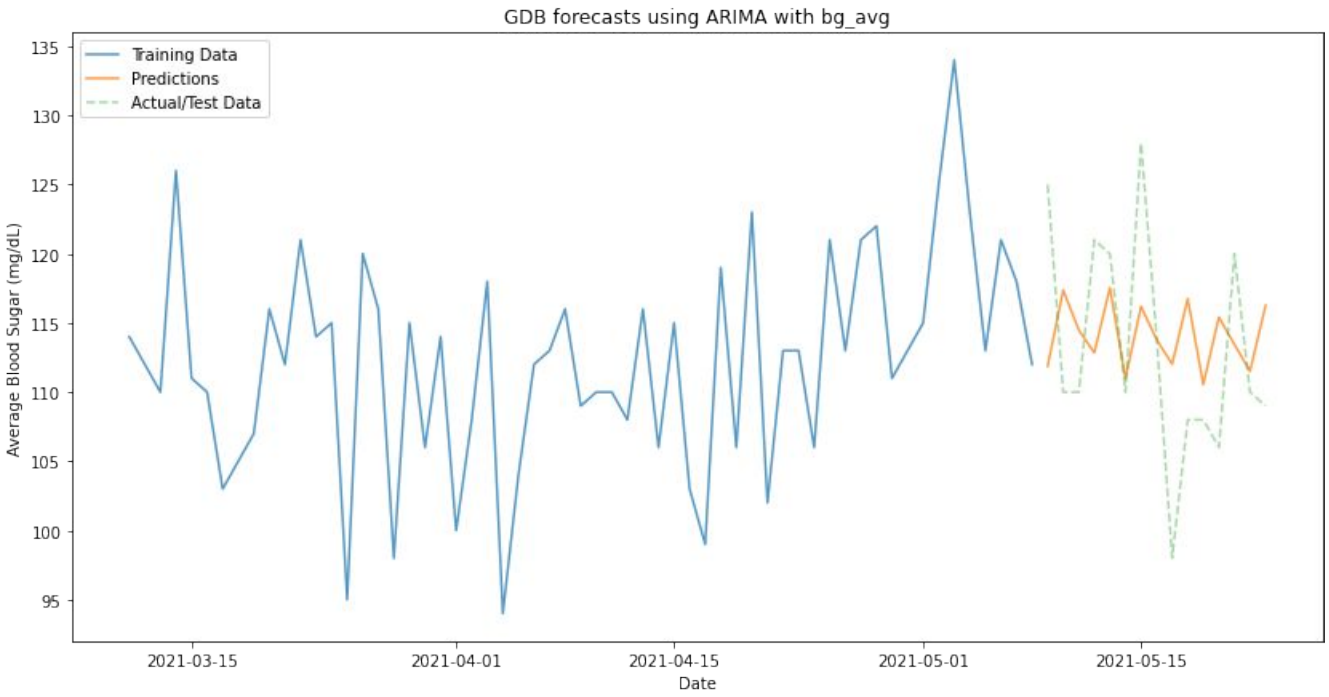
ARIMA

Order Selection*:

1. (0, 0, 7)
2. (3, 0, 3)
3. (0, 0, 0)

*Based on AIC values

Daily Blood Sugar Average: Take 2



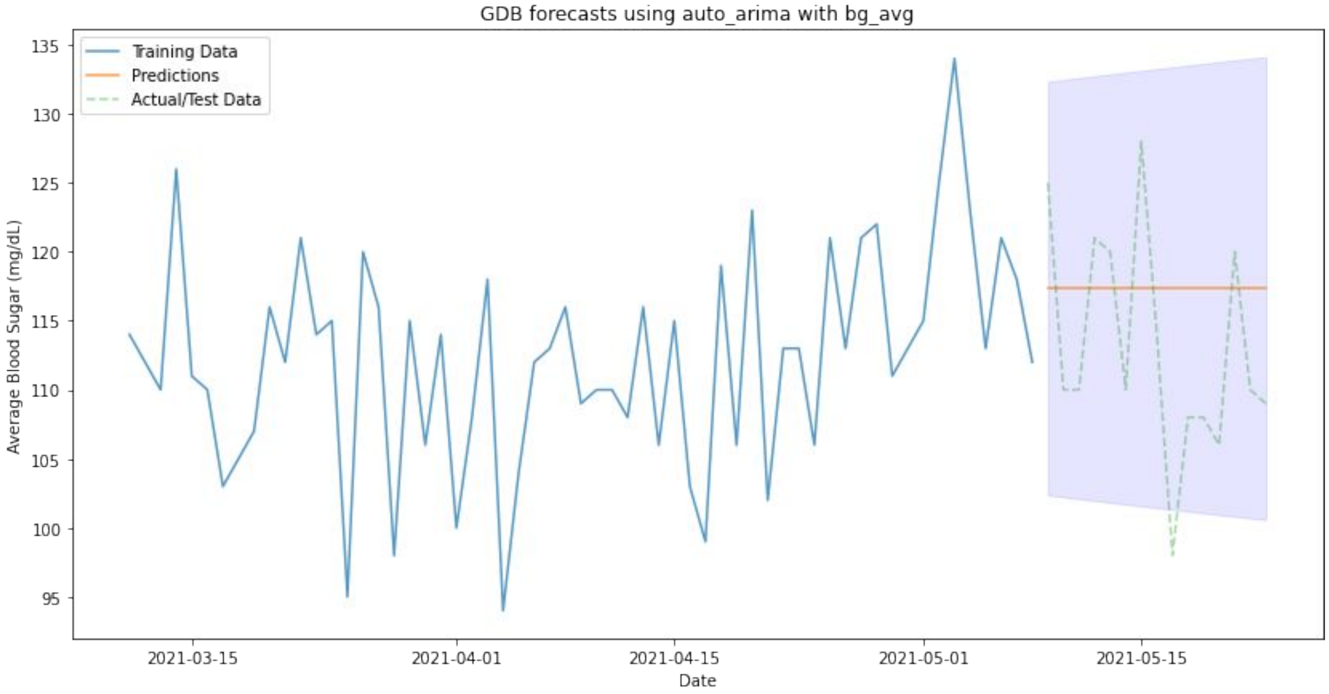
ARIMA

Order Selection*:

1. (0, 0, 7)
2. (3, 0, 3)
3. (0, 0, 0)

*Based on AIC values

Daily Blood Sugar Average



auto_arima

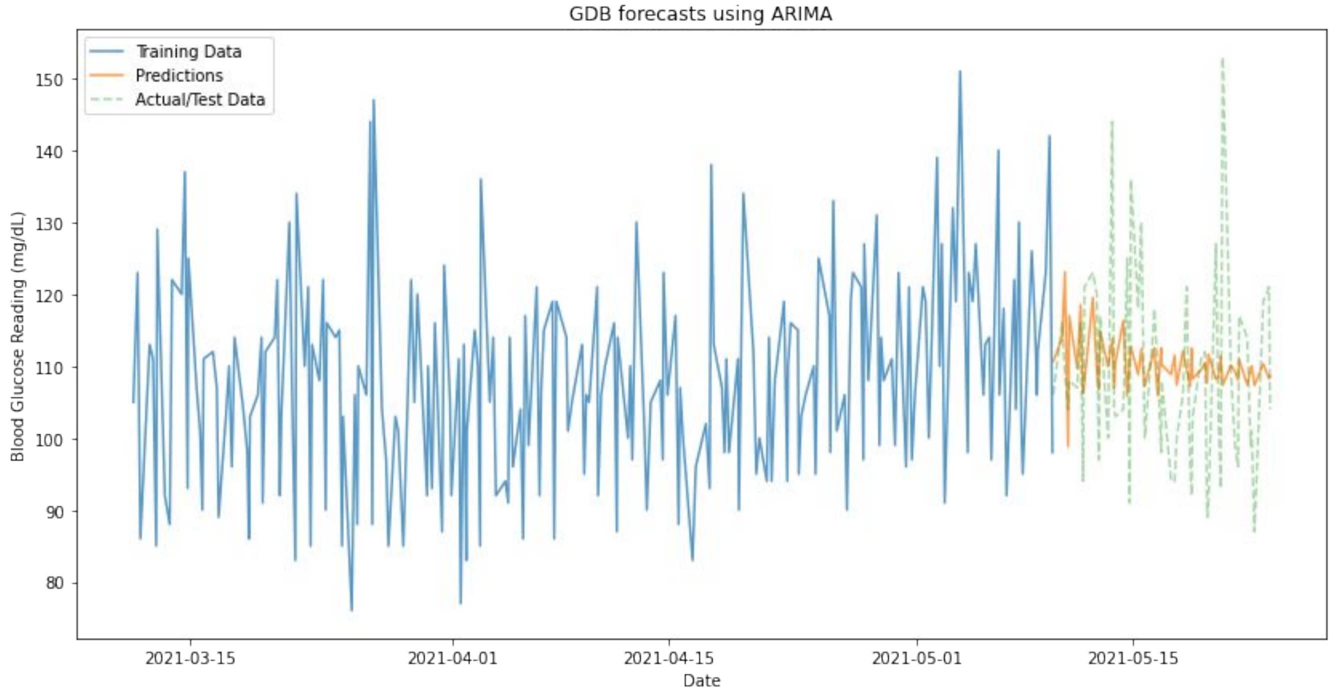
Model Order:
(0, 1, 1)

Model Metrics: Daily Averages

	RMSE	Standardized
ARIMA (3, 0, 3)	7.8574	1.0101
Auto Arima	8.8481	1.1374
ARIMA (0, 0, 7)	9.0899	1.1685

These RMSE results mirrors my opinion that the second ARIMA model performed the best of the three.

All Blood Sugar Readings



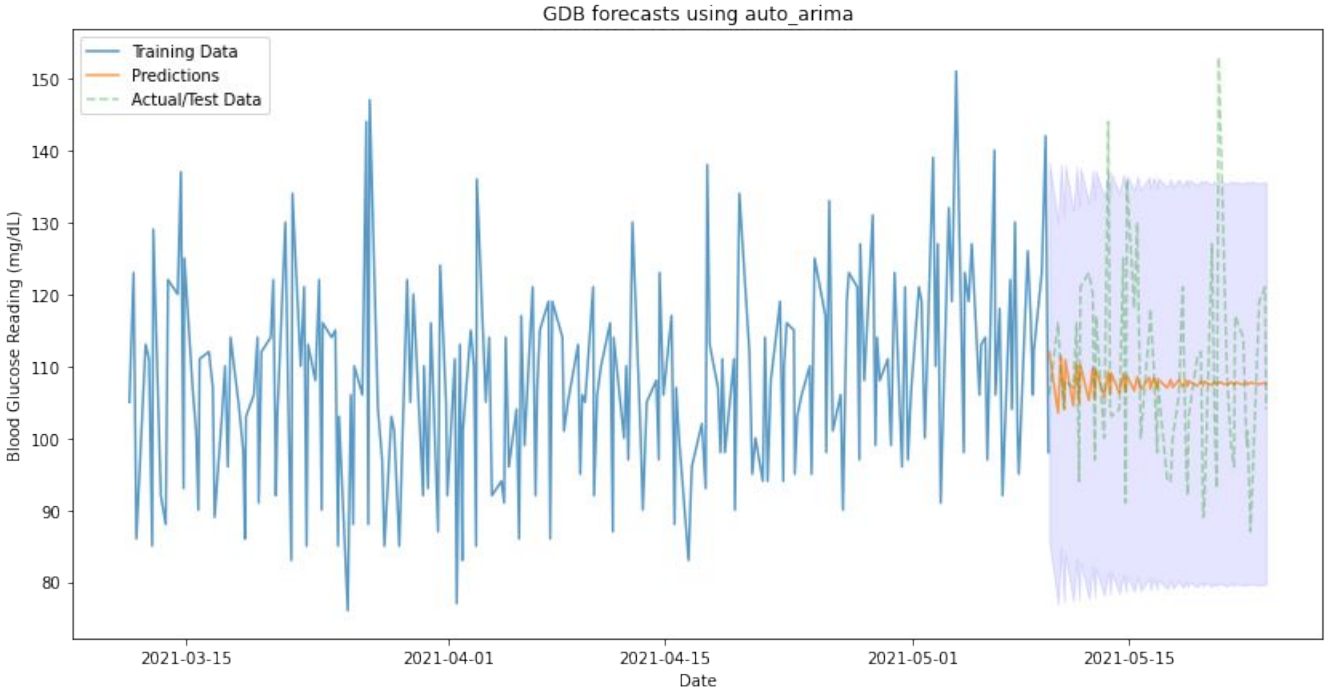
ARIMA

Order Selection*:

1. (7, 0, 5)
2. (4, 0, 4)
3. (5, 0, 5)

*Based on AIC values

All Blood Sugar Readings



auto_arima

Model Order:
(1, 0, 1)

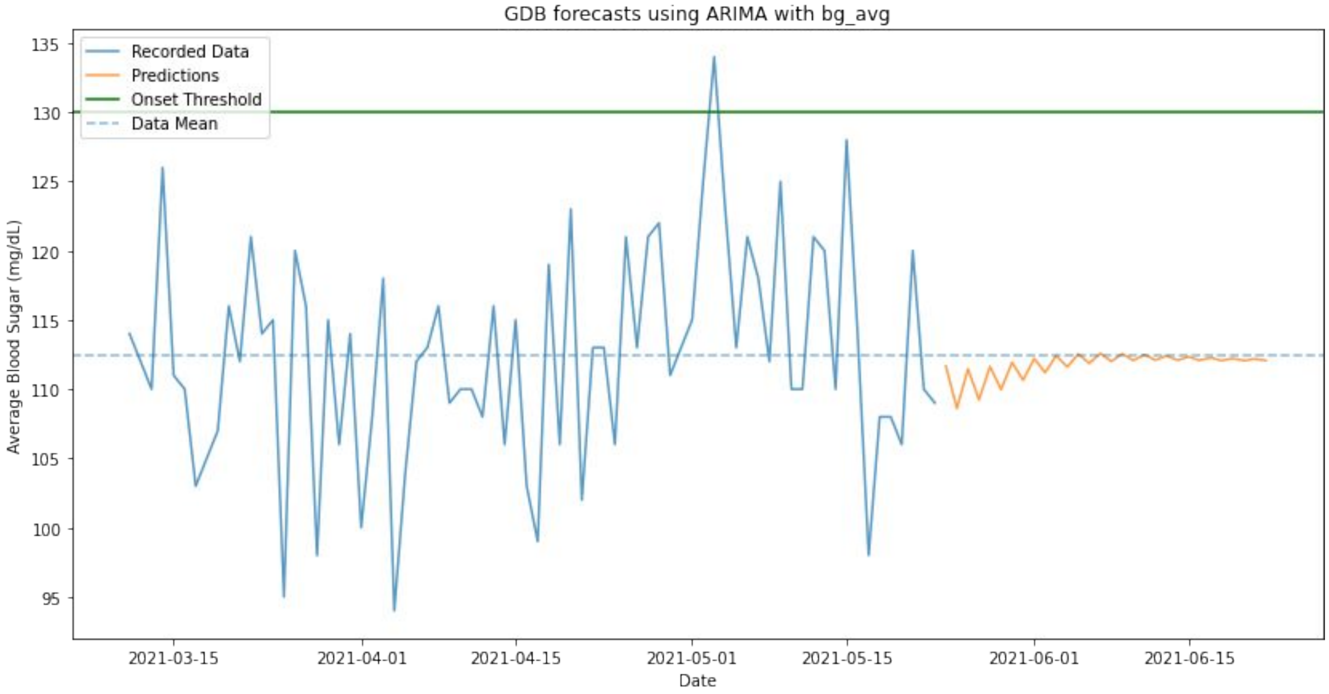
Model Metrics: All Blood Sugar

	RMSE	Standardized
Auto Arima	13.3582	0.9837
ARIMA (7, 0, 5)	13.5269	0.9961

Despite the lower RMSE of auto_arima, I think ARIMA technically scored better because the standardized value (RMSE / Standard Deviation of y_test) is closer to 1. It just happens to be below 1 instead of over 1 like the other models.

Final Model

ARIMA (3, 0, 3) using Daily Averages



Notes

The trend changed once I trained on the full dataset

The target threshold I chose for success is too high. It's the diagnosis value, not a realistic daily average

Room for Improvement

Ideas for Improvement

- Optimize ARIMA order by RMSE instead of AIC or BIC
  - This could lead to overfitting, but since I have so little data, and only a small trend at the very end I want to focus on, I'd like to see if that actually helps with the prediction.
- Focus on data before meds, either by isolating that data or by giving it extra weight
  - Duplicate the data to give it more weight
  - I'd have to fudge the dates a bit, but perhaps it's worth it
  - Look into Facebook prophet model
- Split up training/test sets better
  - I split my data without shuffling, but because the most important trend in my data ended up in the test set I suspect my final prediction results suffered
  - I'd like to try using TimeSeriesSplit, or making 5 or so splits manually, and then averaging out the prediction results between the models

Conclusion

Challenges

Challenge 1

Limited Data

I only have data for the time after my GDB diagnosis.

I had already started making diet changes when I started tracking.

Challenge 2

Monotonicity

Some functions required the time series to be monotonic forward.

My Daily dataframe is monotonic backward.

My Full dataframe is not monotonic.

Challenge 3

Inconsistency

My diagnostic test flagged my non-fasting levels as high, but my at-home tests identified my fasting values to be problematic.

Which do I use as my target variable?

Success Criteria

1 of my 2 goals was successful

1. Make predictions for dates *before* my data starts

SUCCESS

2. My predictions cross a threshold chosen using target blood sugar levels for pregnant people.

FAILURE

Final Thoughts

Despite the failed predictions of the final model I still call this experiment a success.

The goal was to see if I could reverse a time series to make predictions in the past and I found a way to do that.

I very much enjoyed diving into this data, as well as experimenting with reversing a timeseries. It's reaffirming to look back at this data and see how my blood sugar levels improved; being diagnosed with GDB was not the happy pregnancy I wanted.

It all turned out okay in the end!