Home Temperature Time Series Forecasting

Energy consumption is a critical issue today. Inside homes, heating, ventilation, and air conditioning (HVAC) systems are responsible for a high amount of the home's total energy consumption, as much as 53.9%. However, we assume the premise that an HVAC system that anticipates the temperature in the future, and makes smaller adjustments to maintain the desired temperature, would use less energy than an HVAC system that's designed to simply kick on when the temperature falls above or below a certain level. In this project we have developed a predictive model to forecast indoor room temperature. We found that a tuned random forest regressor was the most effective model for predicting room temperature, with an average root mean squared error of ~2 degrees celsius.

The Data

We obtained a csv file of 2764 rows and 19 columns. This dataset came from kaggle[1]. Each row represents a set of measurements of various atmospheric data around the house, and each column represents a kind of atmospheric measurement. This data is ordered in time. A suite of measurements was taken every fifteen minutes by what is likely an automated sensor system. All data was numeric and most was continuous. These are the columns.

- Id - Each set of measurements has a single identification number. First measurement set starts at zero, and the Id's count up by one.

- Date - The date of the measurement. DD/MM/YYYY. Measurements were taken every fifteen minutes starting at 11:45 am of March 13, 2012, and ending at 6:30 am of April 11, 2012. Came initially as a string, which we converted to a datetime object.

- Time - time of the measurement in 24 hour clock format. Measurements were taken every fifteen minutes. Came as a string initially. We condensed this into the date column.

- CO2_(dinning-room) - The concentration of carbon dioxide in the air of the dining room, measured in parts per million (ppm). Vast quantities of CO_2 are known to contribute to rising temperatures, as in the context of global warming. Light from the sun passes into our atmosphere where solar radiation is reflected. The atmospheric carbon dioxide absorbs some of the energy of this reflected radiation, resulting in an insulating effect. In our present context the effect of CO_2 may be similar, albeit on a vastly smaller scale. Must have a lower limit of zero (no CO_2).

[1] https://www.kaggle.com/competitions/smart-homes-temperature-time-series-forecasting/overview

- CO2_room - The concentration of carbon dioxide in the air of the room who's temperature we're trying to predict, measured in parts per million (ppm).

- Relative_humidity_(dinning-room) - Relative humidity in the dining room, expressed as a percentage. Absolute humidity is measured as the mass of water vapor present in the air. Relative humidity is the ratio of the absolute humidity (how much water vapor there is currently) and the amount of water vapor that *could* be in the air at a given temperature. Must range between 0 and 100.

- Relative_humidity_room - Relative humidity of the target room, in %.

- Lighting_(dinning-room) - The amount of lighting in the dining room, measured in lux. Lux is the SI unit of illuminance, and measures lumens per square meter. Lumens measure the total quantity of visible light emitted by a source per unit of time. In more familiar terms, this lighting column basically tracks the intensity of light that hits or passes through a surface. Must have a lower limit of zero.

- Lighting_room - The amount of lighting in the target room, measured in lux.

- Meteo_Rain - Rain, the proportion of the last 15 minutes where rain was detected (a value in range [0,1]). A number of 0.75, for instance, means it was raining for 75% of the fifteen minute interval since the last reading.

- Meteo_Sun_dusk - Sun dusk.

- Meteo_Wind - Wind speed outside the house, in m/s.

- Meteo_Sun_light_in_west_facade - The amount of light in the west facade, in Lux. Must have a lower limit of zero.

- Meteo_Sun_light_in_east_facade - The amount of light in the east facade, in Lux.

- Meteo_Sun_light_in_south_facade - The amount of light in the south facade, in Lux.

- Meteo_Sun_irradiance - Sun irradiance, measured in watts per square meters (W/m2). Solar, or sun irradiance is the power per unit area received from the sun in the form of electromagnetic radiation. It seems to be relatively common to have small negative values at night time due to measurement error, but these should usually be positive values.

- Outdoor_relative_humidity_Sensor - Outdoor relative humidity, in %.

- Day_of_the_week - Day of the week (computed from the date), 1=Monday, 7=Sunday

Changes to the Data

The data was quite clean when it came to us. There were no missing values in the form of NaN's we needed to deal with. We confirmed that no rows were missing by observing the maximum and minimum values of the differences between each row. The differences were all 15 minute periods. We also made sure that no column had invalid data, like negative values where there shouldn't be any. The Meteo_Sun_irradiance column did have negative values, but upon cursory investigation[23], it seems to be relatively common to have small negative irradiance values. The reason for this might be because the value is recorded on a logarithmic scale, with an arbitrary zero point at, say, dawn or dusk. At night (which comprises roughly half of the measurements), the absence of sun results in low points of irradiance that fall just below the arbitrary zero point, meaning the value of the exponent variable is negative. During the day, it is vastly higher. One crucial change we made was to remove some extreme outliers in the CO2 columns. The changes to the data we made can be summarized as follows.

1) Made combined date_time columns out of the date and time columns.

2) Changed the data type of the date_time column to be a datetime object.

3) Deleted the time, date, and Id columns.

4) Set the index of our dataframe to be the combined date_time column.

5) Fixed spelling errors and in general made the column name formatting consistent and less verbose.

6) Removed outliers in both the CO2 columns. A standard definition of an outlier is 1.5 * the interquartile range. The interquartile ranges for co2_dining_room and co2_room were 10.35 ppm (200.89 to 211.25) and 10.54 ppm (202.68 to 213.22), respectively. A standard outlier on the upper end for each column, then, would be 226.77 and 229.02, respectively. As can be seen from figure 1 below, both columns had outliers in vast excess of the standard outlier value (above 400 ppm each). We replaced all outliers above the standard outlier value with the standard outlier value for each column. The plots in figure 2 show the data after this replacement.

7) We binned the rain column into categories. The values in the rain column were described as the proportion of the last fifteen minutes it was raining. When we plotted these values and viewed the summary statistics, we saw that the vast majority of the values were 0 (no rain in the last 15 minutes), some were 1 (100% rain in the last 15 minutes) and hardly any values were not 0 or 1 (figure 3). Therefore, it made sense to bin these values into "none" (less than 50% of the last 15 minutes had rain), or "raining"

[2] https://unmethours.com/question/13731/negative-and-positive-values-of-solar-radiation-during-night/
[3] https://www.researchgate.net/post/Its_normal_to_have_negative_values_of_irradiance_in_a_database

(50% or more of the last 15 minutes had rain).

8) We also binned the sun_dusk column into categories. Much like rain, sun dusk had hardly any values between 0 and its upper half of ~611 (figure 4). We decided to bin everything below 100 as “none,” everything between 100 and 600 as “partial,” and everything above 600 as “full.”

For our preprocessing, we OneHot encoded the newly categorical rain and sun dusk columns and scaled the independent variables using a standard scaler.

Exploratory Data Analysis

We divided our visual exploration of the data into three parts: column values over time, column correlations with the dependent variable, and the dependent variable over time. Here are our observations for each part.

1) Column values over time:

    a) co2_dining_room: We see two enormous spikes, one large spike, and a few other spikes in what is otherwise a relatively mean stable distribution. The three spikes occur right around March 14, 26, and 29. These are outliers, which we removed as part of our preprocessing.
    b) co2_room: Again we see a number of spikes. The co2 levels in the room appear to have spiked on the same days as the co2 levels in the dining room, and also right around March 21 and 22. Both co2 columns have similar means at the low 200s.
    c) Humidity columns: It seems to have rained (or perhaps there was a water leak) on March 21 and April 4, as there are increases in humidity starting on those days.
    d) Lighting columns: A pair of truly periodic columns, peaking and troughing daily as the sun rises and falls. March 20 seems to have gotten almost no light (about 20 lux), as there is a massive decrease in light from the usual day time lux. Perhaps the measurement devices failed that day or something interfered with their recording.
    e) Rain: Confirming our earlier intuition, it rained on the night of March 20 and showered between April 4 and 5. The presence of rain seems to result in lower room temperature.
    f) Sun_dusk: Sun dusk jumped between zero and ~600 daily in a very periodic fashion. Sun dusk hasn't suffered the same drop on March 20 that the lighting columns did.
    g) Wind: Wind has pretty mean stable daily fluctuations with a particularly windy day on the evening of March 20 and day of April 10.
    h) Sunlight columns: Lighting on the various facades of the house rise and fall daily, as we'd expect. The south facade got considerably more light on average than the east or west facades did. This house is in the northern hemisphere. Something may have gone wrong with the sensors for March 20, where the light on all three facades was far below average.
    i) Sun irradiance seems covariant with the sunlight columns, complete with the global minimum on March 20.

j) outdoor_relative_humidity readings are a little less clear as far as tracking when it was raining, but there are maxima on the days that it did rain (March 20 and April 4-5). It was more humid outdoors compared to indoors, as we would expect.

k) indoor_temperature_room, the target variable, follows some expected behavior. It has periodic peaks and valleys with a difference of about 6 or 7 degrees celsius corresponding to day and night time. It reaches its minimum on March 21, the day after one of the heavy rain days. Then between March 21 and about April 4 we see an upward trend, followed by a slight downward trend from the rain, and then an upward trend again starting about April 7. If we were to make guesses on the forecasting of this temperature from eyeballing graphs alone we might say that the mean temperature is most affected by rainfall. Therefore, we might expect the mean temperature to fall again when the next rainfall occurs. Figure 5 shows the room temperature over time.

2) Column correlations with the dependent variable:

Nothing correlates very strongly with room temperature, positively or negatively. In our scatter plots there isn't much of a linear relationship between any variable and room temperature. The closest to a linear relationship we can see is outdoor_relative_humidity and temperature having a negative correlation.

a) Positive correlations: From our heatmap of correlations we found that, weakly, room temperature increases as wind increases (0.22 pearson correlation coefficient). We also see that room temperature increases as sunlight in the west facade increases (0.34).

b) Negative correlations: As outdoor_relative_humidity (-0.55), relative_humidity_room (-0.42), relative_humidity_dining_room (-0.28), sunlight_in_east_facade (-0.27) and rain (-0.26) increase, room temperature decreases. This somewhat confirms our earlier intuition, that rain and room temperature are negatively correlated.

c) The rest of the columns have very little correlation with room temperature.

3) Dependent variable over time:

In this subsection we looked purely at the relationship between time and our target variable. When modeling time series data, at least if you plan to use any of the SARIMAX (Seasonal Auto Regressive Integrated Moving Average eXogenous) family of models, it is important to make your data stationary. A stationary time series means three things: there is no trend, the variance is constant, and the autocorrelation is constant. For a non-stationary time series, mean, variance, and autocorrelation are changing over time, meaning there isn't one mathematical approximation function (model) that describes the system, or at least not a linear one.

Trend is the overall direction of the data. Is the mean of the data increasing or decreasing over time? Global CO2 concentration has a positive trend, as over the long run it is increasing overall even if there are month to month or year to year drops. The ARMA order components (p, d, q) of a SARIMAX model address trend. Our time series doesn't have much of a trend, as can be eyeballed in figure 5. We further confirmed this with the augmented dickey-fuller test, which tests for trend non-stationarity. The adf test assumes a null hypothesis that the time series is non-stationary as far as *trend* is concerned. If the p-value of the returned test statistic is below our significance threshold (0.05), then we reject the null hypothesis and assume, at least as far as trend is concerned, our data is stationary. The p-value of the adfuller test on our time series is very small (0.000000597), and quite a bit less than the standard 0.05 significance level. Therefore, the time series is stationary with respect to a trend.

Seasonality is the repeating cyclical patterns in a time series. Our data is clearly quite seasonal. Temperature in the room rises and falls in a predictable daily pattern. This is something that needed to be addressed with seasonal differencing, which was controlled by certain hyperparameters in our SARIMAX model.

Variance is the distance between the data points and a zero line. Our time series did not seem to have variance non-stationarity, as the differences between high and low temperatures did not increase or decrease by much.

Autocorrelation is the correlation coefficient of a time series with a lagged copy of itself. How well can you forecast based on values at previous dates? Autocorrelation coefficients of high magnitude mean that you can forecast the series from past data points.

Our series has high autocorrelation due to the high seasonality. According to the acf and pacf functions, which show autocorrelation/partial autocorrelation at a variety of time lags, the data doesn't achieve an autocorrelation not significantly different than zero until around 40 lags. It is impossible to determine SARIMAX model orders from these plots (figure 6). It wasn't until we understood this did we figure out an appropriate differencing order to remove the autocorrelation.

Our time series has data in fifteen minute intervals. The "cycle" of our data, that is how long it takes for the repeating pattern of temperature data to repeat itself, is one day. This means that to remove autocorrelation due to seasonality, we had to difference our dataset by one cycle, or approximately 96 fifteen minute periods which fit into a 24 hour day. After taking the first difference and the seasonal difference, we were finally able to achieve stationarity in our time series. In figure 7 the acf plot shows a sharp drop in autocorrelation after the first *seasonal* lag (~96 lags), and the pacf plot shows a trailing off. Therefore, we gather that the appropriate SARIMAX model orders are probably SARIMAX(order = (0, 1, 0), seasonal_order = (0, 1, 1, 96)). Finally, in figure 8, we see a plot of the room temperature data after it has been seasonally differenced.

Modeling Results and Metrics

We have, in a sense, two sets of explanatory variables that may explain our target variable: time (how the room temperature changes over time), and the rest of the measurement data (how does co2/light/precipitation affect the room temperature). A SARIMAX model can handle both sets of explanatory variables at once by accepting a set of exogenous variables. For comparison, we also tried a random forest regressor model, which does not incorporate the element of time. Our data was split into 80% training data and 20% testing data. To do so, we manually split at the 80% index point, so as to preserve the order of the data. Our metric of choice here was root mean squared error (rmse), which is a standard for regression tasks.

With our manually determined parameters the first SARIMAX model we tried took a very long time to fit, on the order of 30 minutes. Attempts to automatically obtain optimal parameters with auto_arima resulted in consistent memory crashes. We suspect the high seasonality was causing all the slow training and memory errors.[4] We addressed this by downsampling the temperature measurement frequency to one hour, which drastically speeded up model fitting time, and even improved accuracy. Unfortunately tuning with auto_arima on our downsampled data set resulted in no improvement to rmse.

In the end a random forest regressor model, with hyperparameters tuned via randomizedsearchCV (cv = 3) scored slightly better than the SARIMAX models which we went to such great lengths to understand. The random forest regressor had a root mean squared error of ~2 degrees C while the SARIMAX models could not fall below an rmse of ~2.75 C. Figures 9 and 10 show the best SARIMAX and the random forest regressor out of sample predictions, respectively.

As is seen in figures 9 and 10, the SARIMAX model much more closely resembled the real system, with its steady peaks and valleys. Its worse rmse was a product of its forecast being offset in the positive y direction. Perhaps an ad hoc shifting would have yielded a more accurate model on new data. Though downsampling our data improved SARIMAX performance, perhaps a SARIMAX model is not an apt model for our data set.

Final Thoughts

We are pleased to have worked with SARIMAX models, and we don’t regard their worse performance as a failure, but a learning opportunity. It may seem odd that a machine learning algorithm not designed to model data in time should outperform one that it is, but again due to the high frequency of the seasonality in our data perhaps we chose the wrong time series model. In the future, we would like to use a Long Short-Term Memory (LTSM) recurrent neural network with Keras.

Our success criterion of rmse of less than 0.05 C was missed by a large margin. Perhaps this was too strict of a success criterion, but certainly more work can be done. We’re optimistic about the LTSM neural network.

[4] https://github.com/statsmodels/statsmodels/issues/5727

Figures

Figure 1: CO2 Columns with Outliers

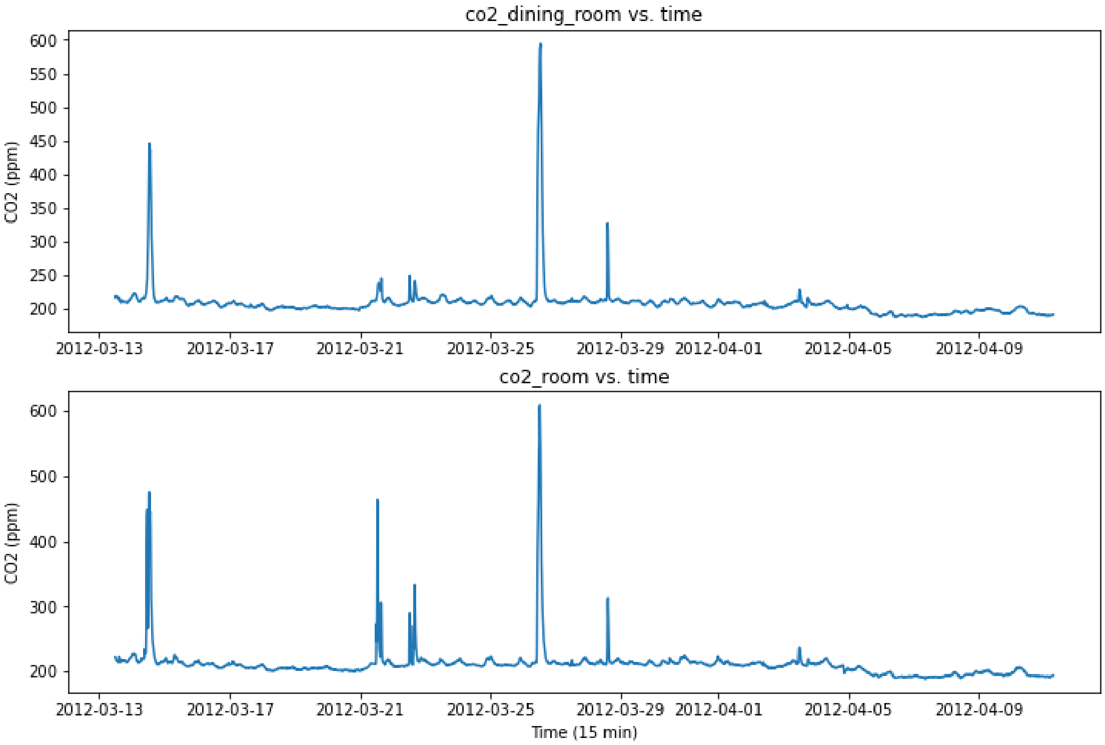


Figure 2: CO2 Columns with Outliers Replaced

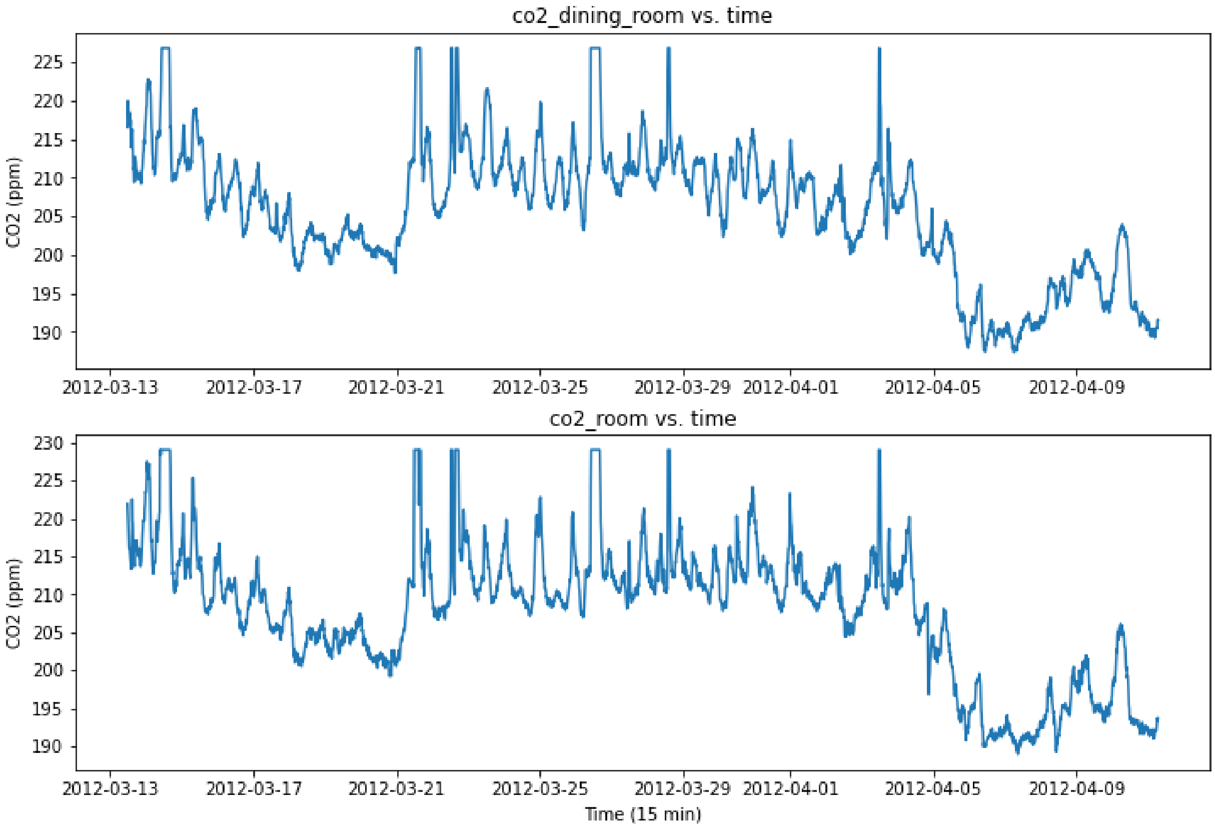
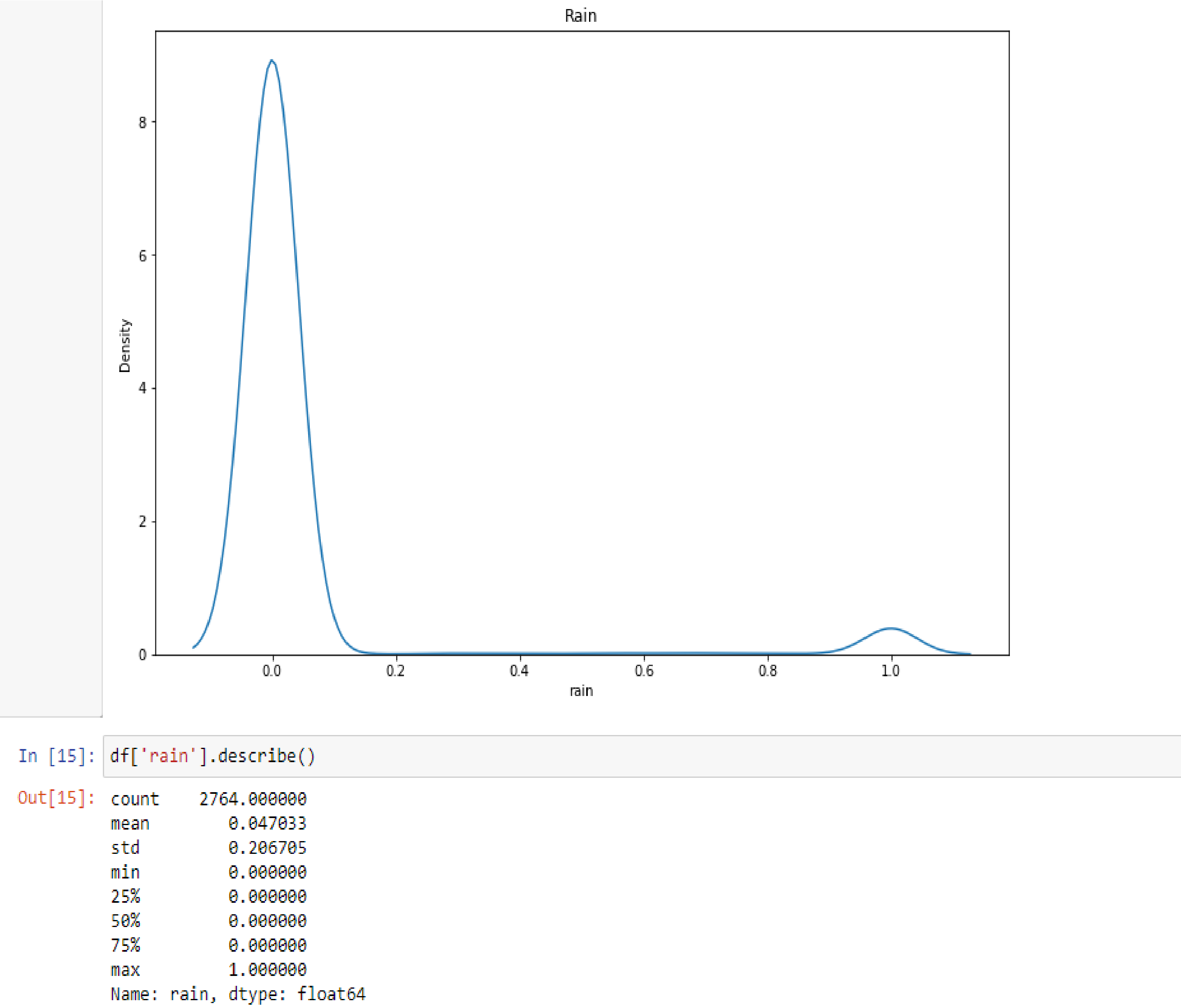


Figure 3: Rain



```
In [15]: df['rain'].describe()
```

```
Out[15]: count    2764.000000
         mean        0.047033
         std         0.206705
         min         0.000000
         25%         0.000000
         50%         0.000000
         75%         0.000000
         max         1.000000
         Name: rain, dtype: float64
```

Figure 4: Sun Dusk

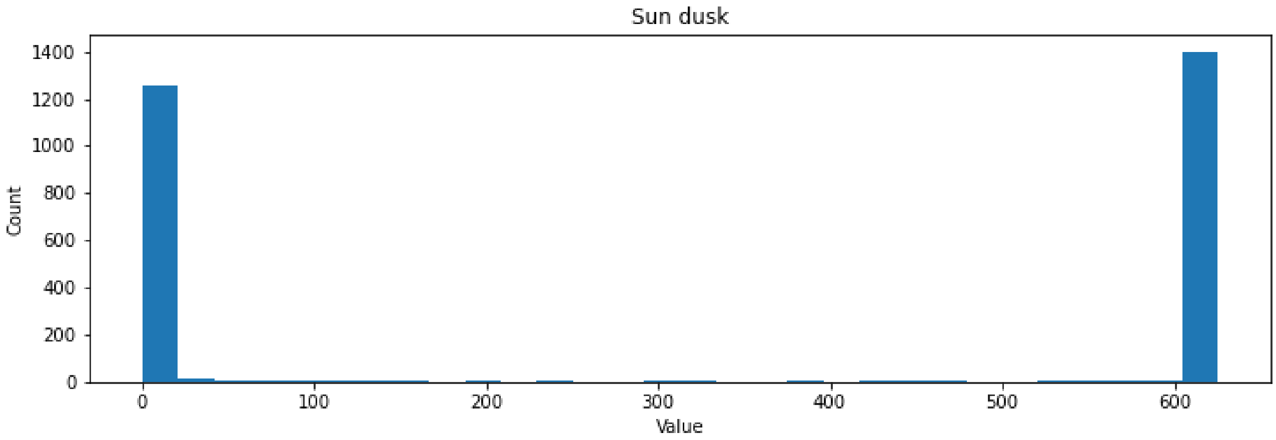


Figure 5: Room Temperature Over Time

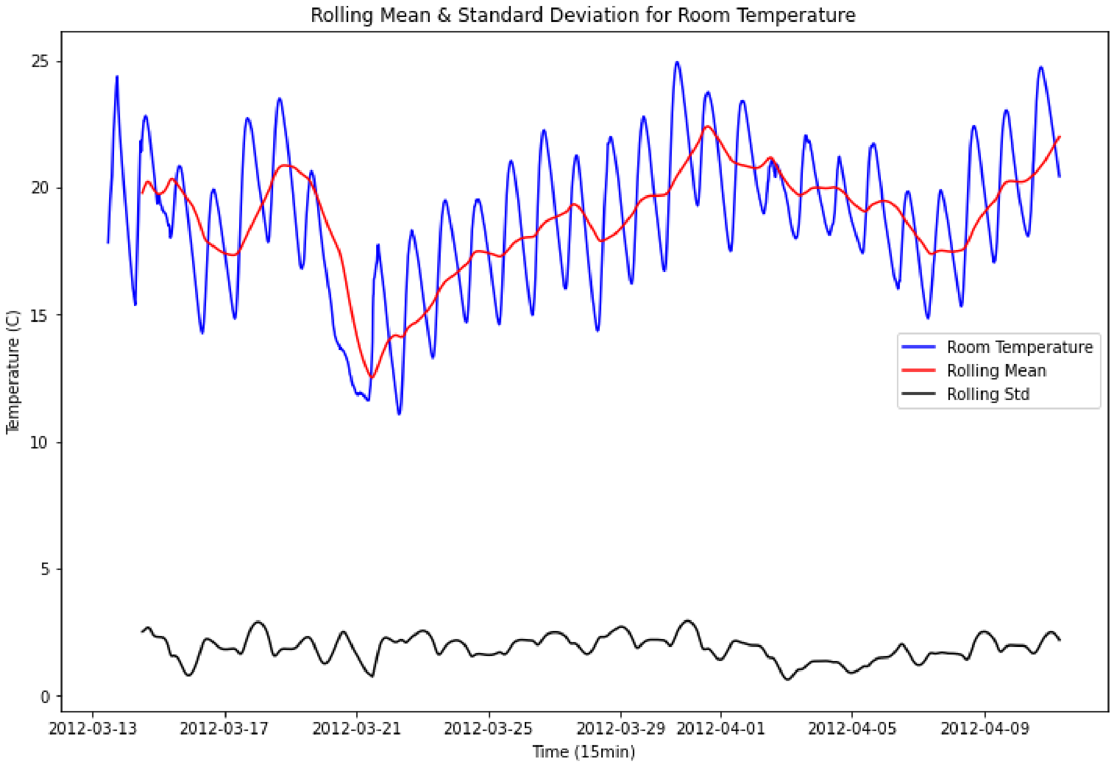


Figure 6: Default acf and pacf plots

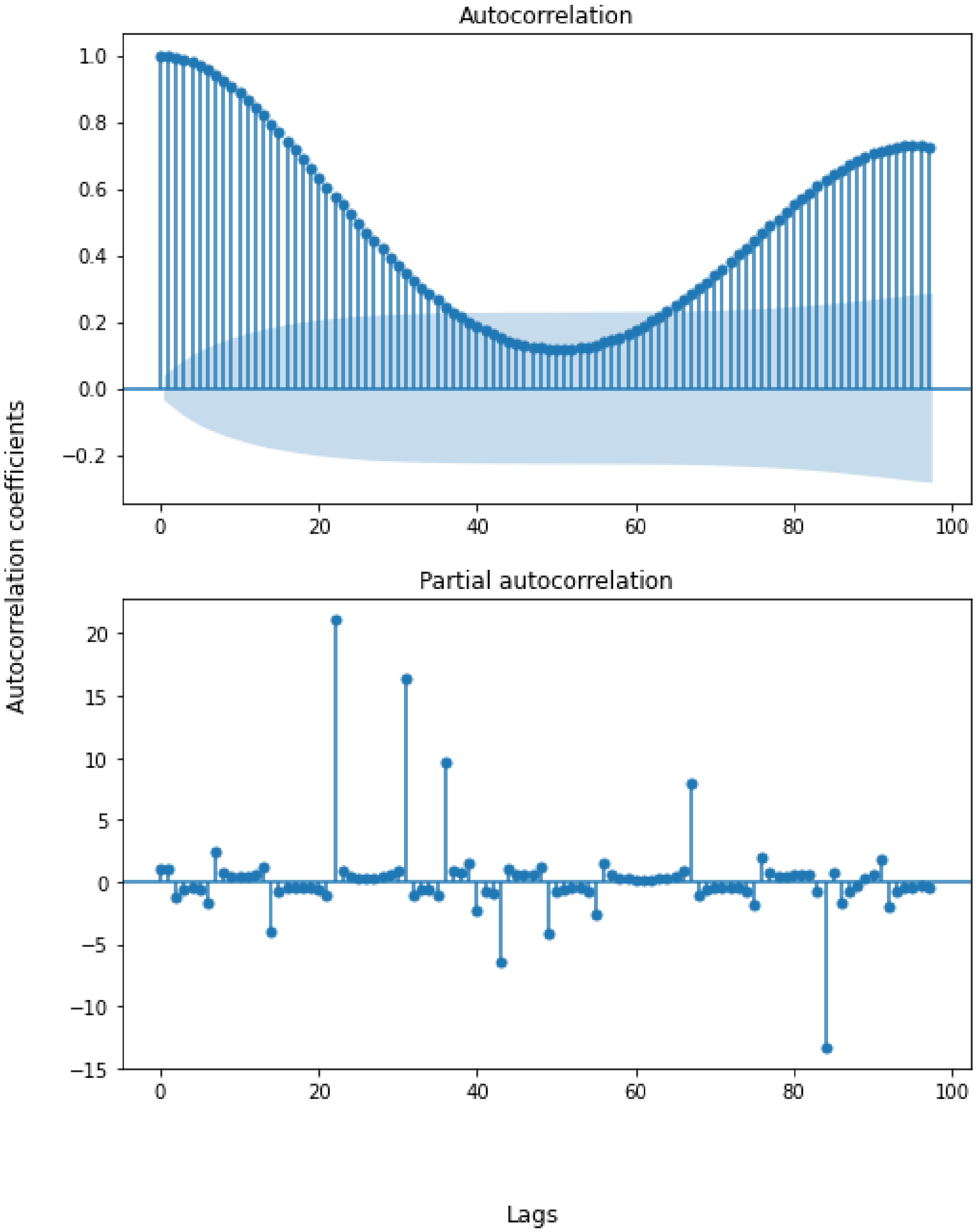


Figure 7: Acf and Pacf Plots with Appropriate Differencing

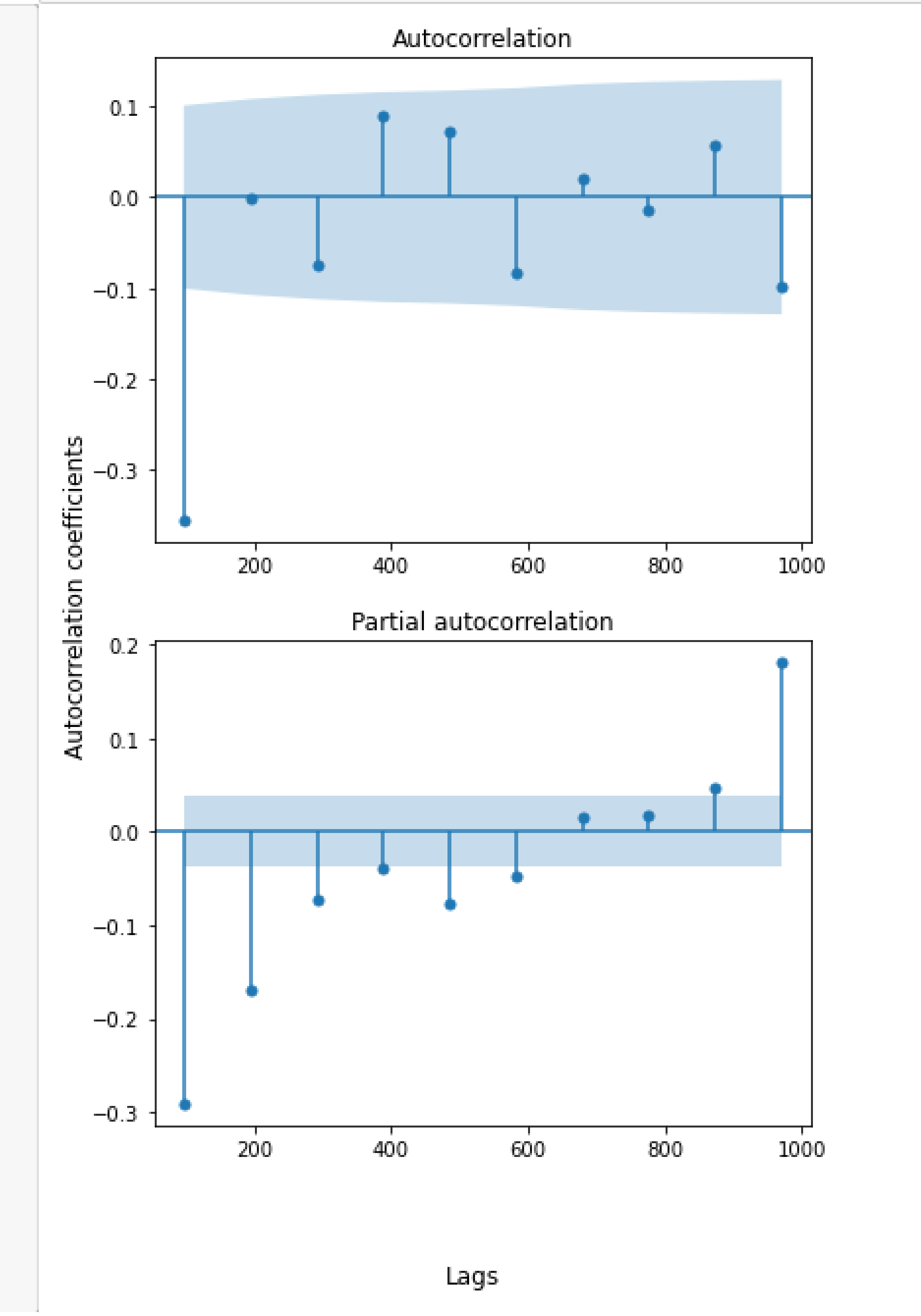


Figure 8: Differenced Time Series

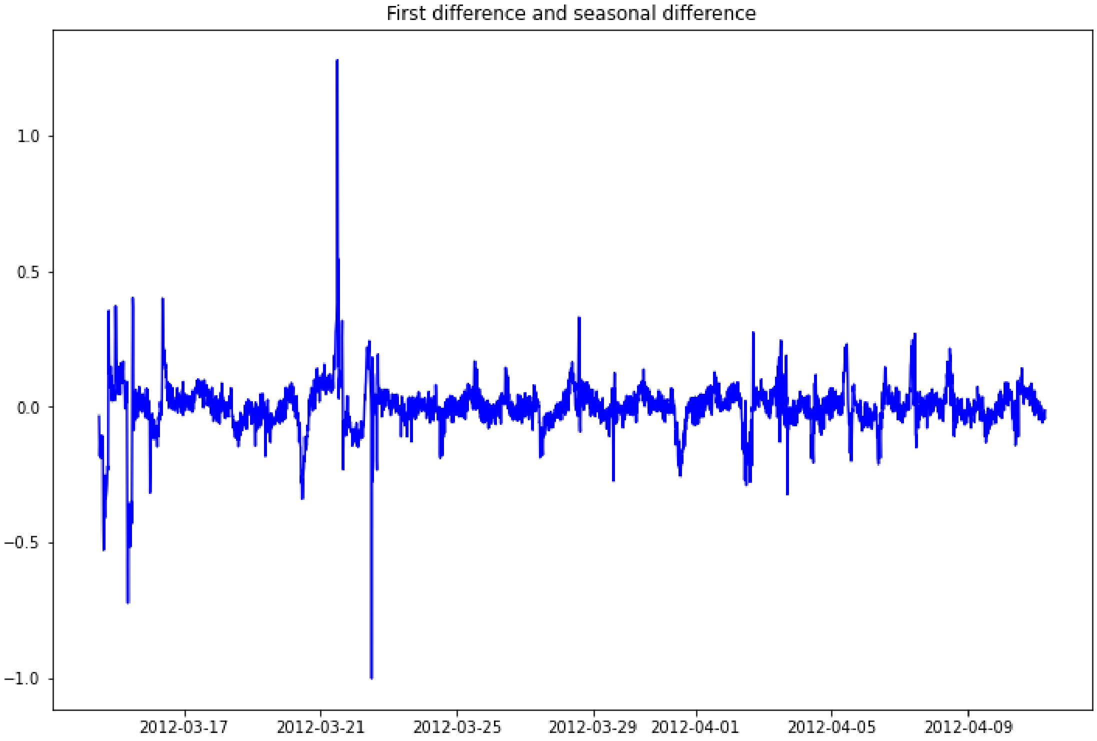


Figure 9: SARIMAX Forecast

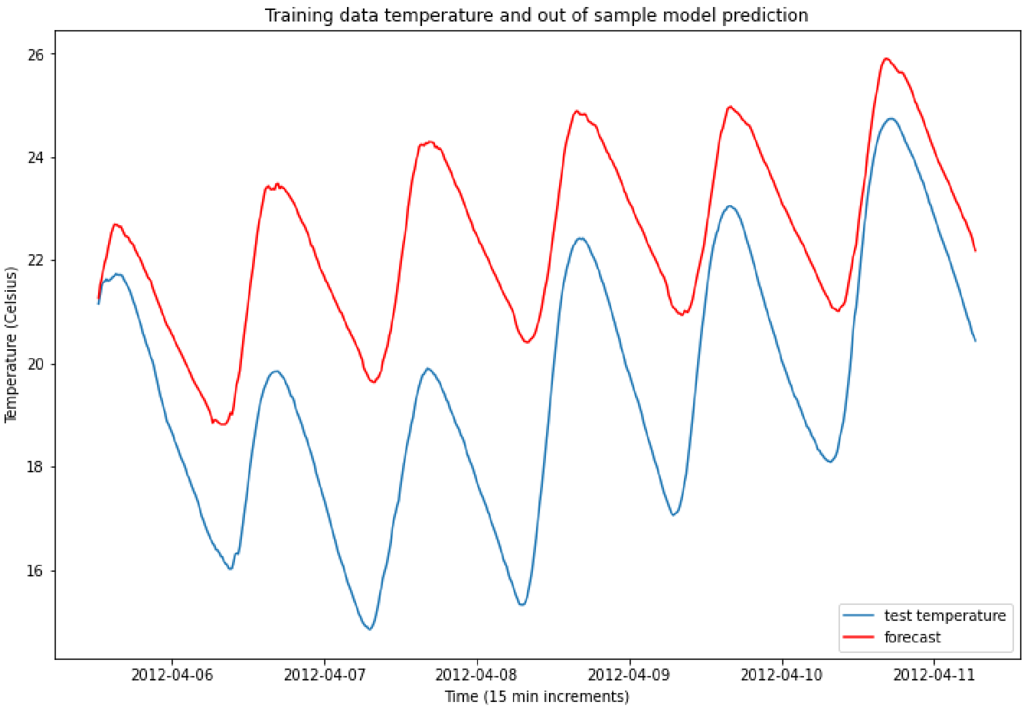
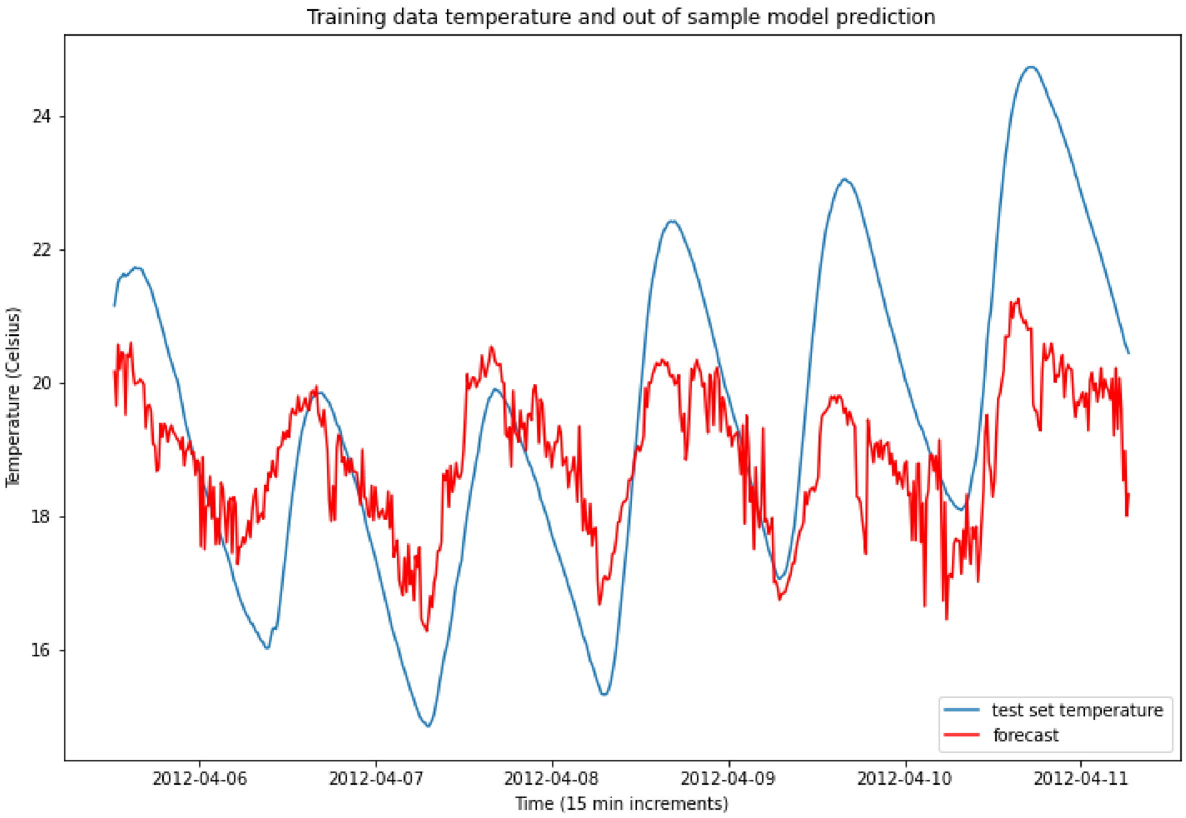


Figure 10: Random Forest Regressor Forecast


Training data temperature and out of sample model prediction
Temperature (Celsius)
24
22
20
18
16
2012-04-06
2012-04-07
2012-04-08
2012-04-09
2012-04-10
2012-04-11
Time (15 min increments)
test set temperature
forecast