THE BLUE SKY CHALLENGE - BLUE SKY BELOW

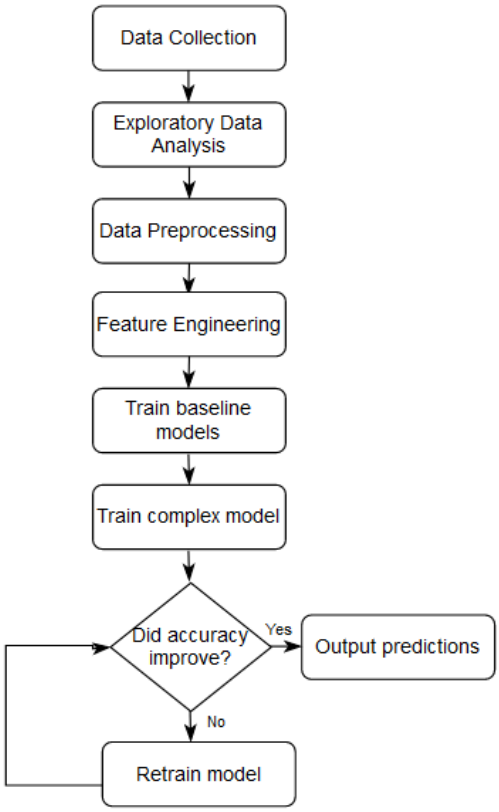
TEAM - BINARYBRAINS

Problem Statement

Analyze factors affecting quality of air using sensor data collected from air quality monitoring systems. Use this sensor data to anticipate future trends in Temperature and Carbon Monoxide (CO) concentration so as to allow appropriate preventive measures.

How we solved the problem (overview)

High-level overview of the steps followed in solving the problem.



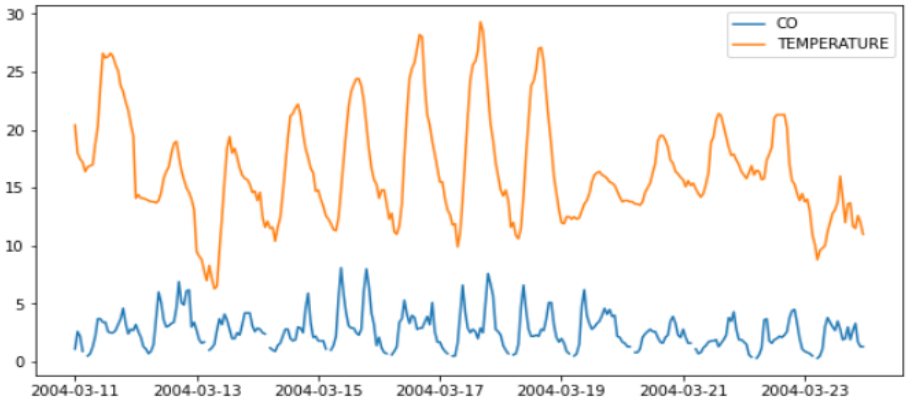
How we solved the problem (detailed steps)

1. Data Collection-

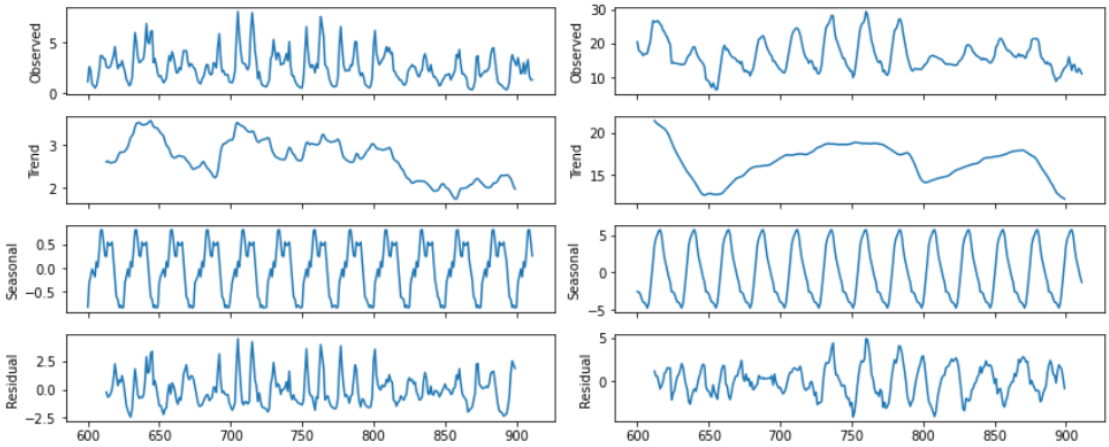
a. The 'Air Quality Dataset' is imported. Column used are: Date (DD/MM/YYYY), Time (HH.MM.SS), CO (mg/m^3), Temperature (°C)

2. Exploratory Data Analysis-

a. EDA revealed that there are 114 rows containing all NULL values.
b. 2013 rows containing missing data for CO and/or Temperature (-200). 12 rows containing missing values for CO in the training set (2004-03-11 to 2004-03-23).
c. Training set showing moderate-high variance (2004-03-11 to 2004-03-23).



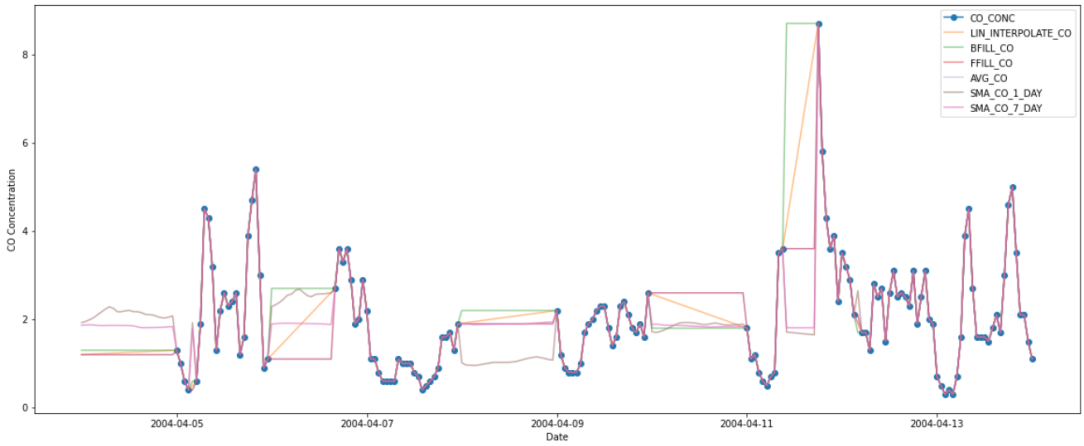
d. Seasonality found in the training data for both CO and Temperature (2004-03-11 to 2004-03-23)



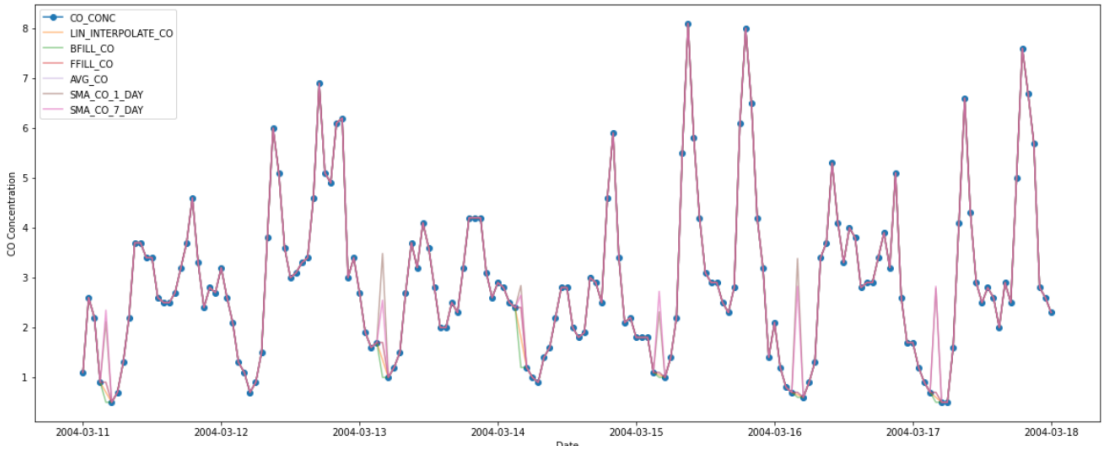
3. Data Preprocessing and Feature Engineering-

a. Drop NULL rows

b. Impute missing values (-200) using linear interpolation, forward fill, backward fill, average of one hour prior and one hour later, rolling average over 1 day, rolling average over 7 days. These methods will prove useful when the training set contains more missing values.



CO concentration from 2004-04-01 to 2004-04-14 where 98 missing values were present that were imputed using the above mentioned methods.



CO concentration from 2004-03-11 to 2004-03-17 (initial training set) consisting of 6 missing values.

4. Feature Engineering-

a. Since the data possess cyclical features (month, day, hour), these were cyclically encoded to provide more information to the machine learning models. The cyclical encoding involves obtaining the sin and cos of each of the features.

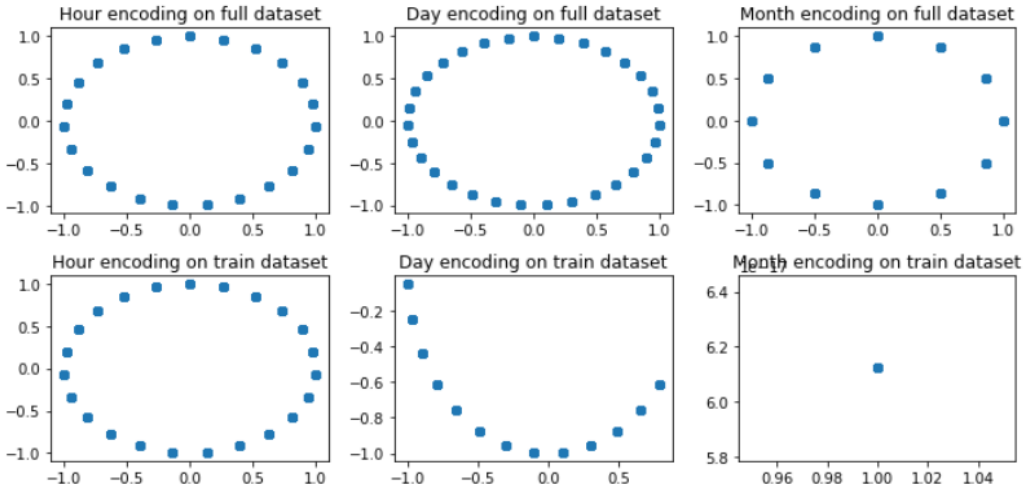
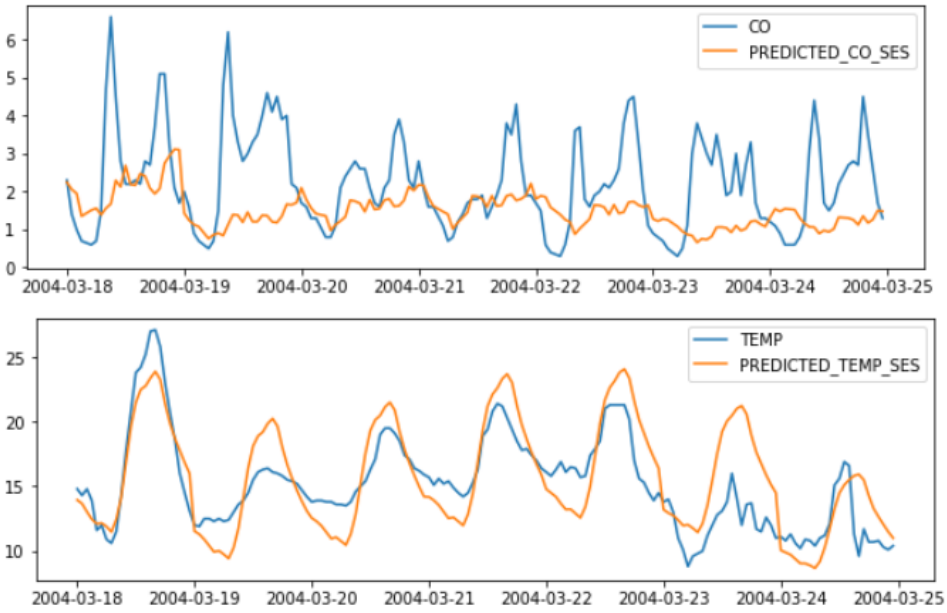


Fig. shows encoding for of cyclical column (DATE) on full dataset (top row) and training set (2004-03-11 to 2004-03-23) (bottom row)

5. Baseline models

a. Exponential Smoothing (statsmodels.tsa.holtwinters.ExponentialSmoothing)

This method produces forecasts that are weighted averages of past observations where the weights of older observations exponentially decrease. The seasonality component is handled using seasonal periods = 12.



Predictions of CO and Temperature using Exponential Smoothing

b. Auto_arima (pmdarima.arima.auto_arima)

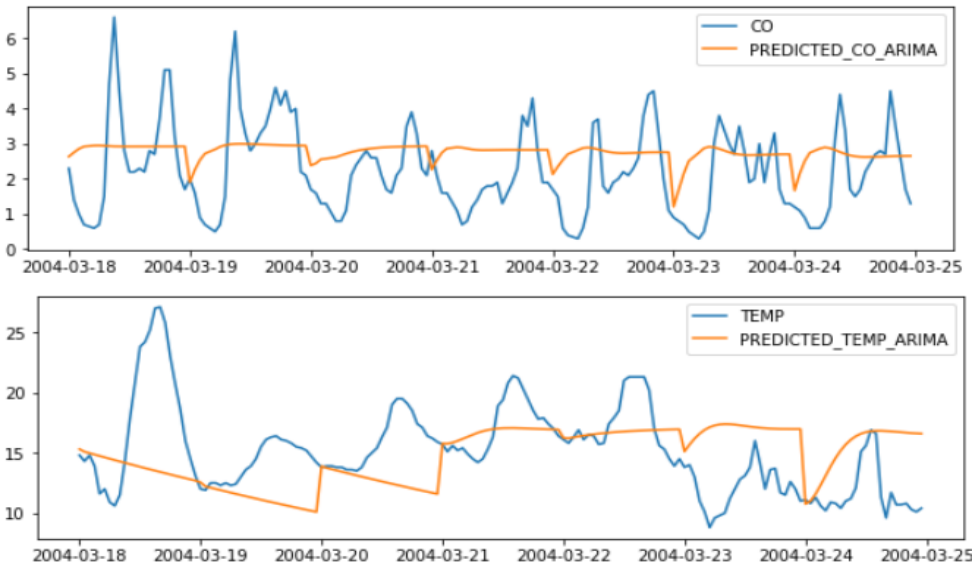
ARIMA (AutoRegressive Integrated Moving Average) is a class of models that captures a suite of different standard temporal structures in time series data.

AR: A model that uses the dependent relationship between an observation and some number of lagged observations.

I: The use of differencing of raw observations in order to make the time series stationary.

MA: A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

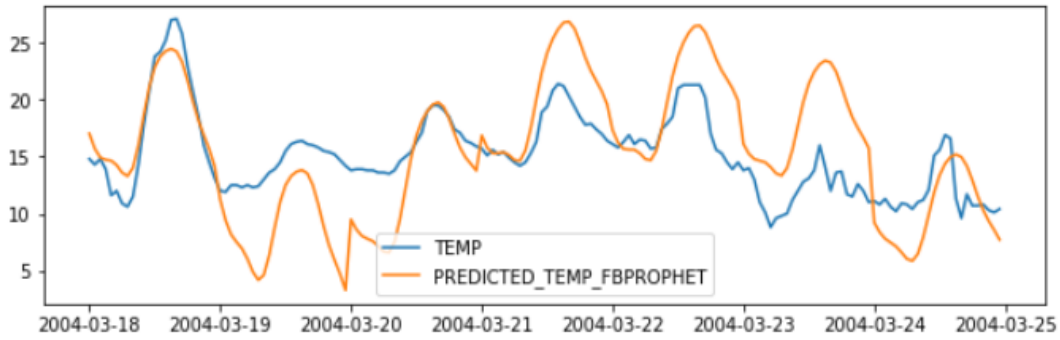
The auto_arima process seeks to identify the most optimal parameters for an ARIMA model, settling on a single fitted ARIMA model. In order to find the best model, it optimizes for a given information_criterion and returns the ARIMA which minimizes the value.



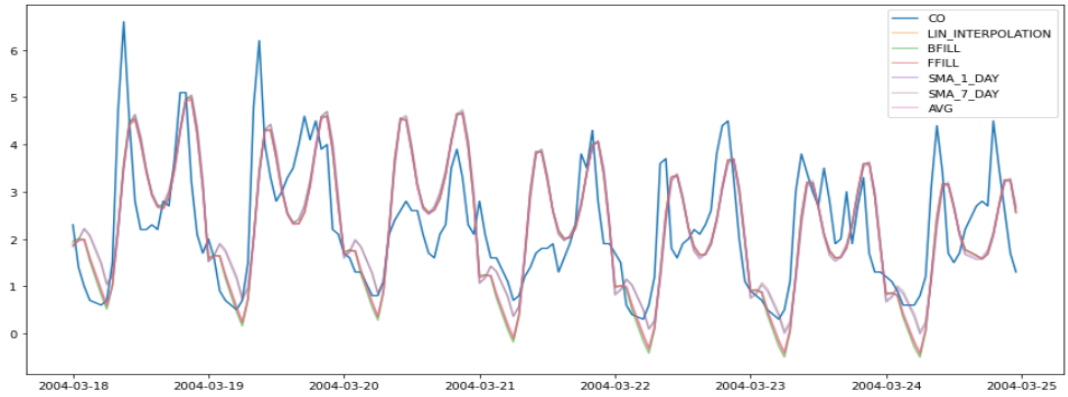
Predictions of CO and Temperature using Auto ARIMA

c. **Facebook Prophet** (fbprophet.Prophet)

It is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality. It is suitable for series that have strong seasonal effects. Prophet is robust to missing data and shifts in the trend, and handles outliers.



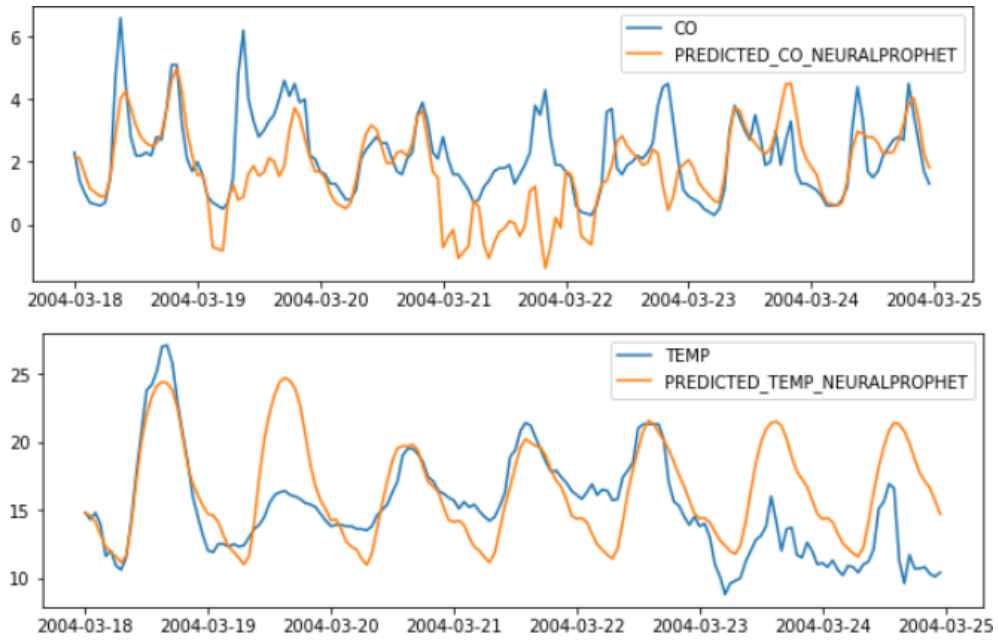
Predictions of Temperature using Facebook Prophet



Predictions of CO using Facebook Prophet (using all the feature engineering methods to account for missing values)

d. Neural Prophet (neuralprophet.NeuralProphet)

NeuralProphet is a python library for modeling time-series data based on neural networks. It's inspired by Facebook Prophet and AR-Net libraries. It has some additional features that help with better predictions. The neural prophet hyperparameters are tuned using validation sets (last 2 days of training set) for each of the forecast days (eg. for prediction for 2004-03-18, training set is 2004-03-11 to 2004-03-15, and validation set is 2004-03-16 and 2004-03-17). The optimal parameters are then saved and used to make the future predictions.



Predictions of CO and Temperature using Neural Prophet

MAPE values for Temperature using the above mentioned baseline models.

	Temp_ExponentialSmoothing	Temp_AutoArima	Temp_FacebookProphet	Temp_NeuralProphet
2004-03-18	0.082678	0.265608	0.099684	0.051531
2004-03-19	0.139071	0.203722	0.393012	0.241133
2004-03-20	0.117455	0.183328	0.211996	0.086763
2004-03-21	0.098078	0.091781	0.169529	0.087637
2004-03-22	0.153658	0.098580	0.205373	0.127125
2004-03-23	0.310662	0.425929	0.473194	0.346471
2004-03-24	0.175360	0.342143	0.250745	0.403168
Average	0.153852	0.230156	0.257648	0.191975

MAPE values for CO using the above mentioned baseline models.

	CO_ExponentialSmoothing	CO_AutoArima	CO_FacebookProphet	CO_NeuralProphet
2004-03-18	0.479754	0.835220	0.632948	0.253505
2004-03-19	0.529232	0.783793	0.498033	0.343960
2004-03-20	0.273057	0.587450	0.371505	0.186616
2004-03-21	0.230828	0.797220	0.679878	0.463026
2004-03-22	0.734041	1.297360	0.387852	0.352109
2004-03-23	0.696812	1.178952	0.544219	0.578151
2004-03-24	0.566951	0.891762	0.387670	0.256816
Average	0.501525	0.910251	0.500301	0.347740

From among the baseline models, the best predictions for:
Temperature is given by the Exponential Smoothing model (MAPE - 0.15)
CO is obtained using Neural Prophet (MAPE - 0.34).
*Note - This is a snapshot of one run. Results vary with every run of the model.

For our final model, the aim is to achieve MAPE values <= 0.15 for Temperature, and <= 0.34 for CO. We use the LSTM (Long Short Term Memory) to overcome the challenges faced in the baseline models and make more accurate predictions.

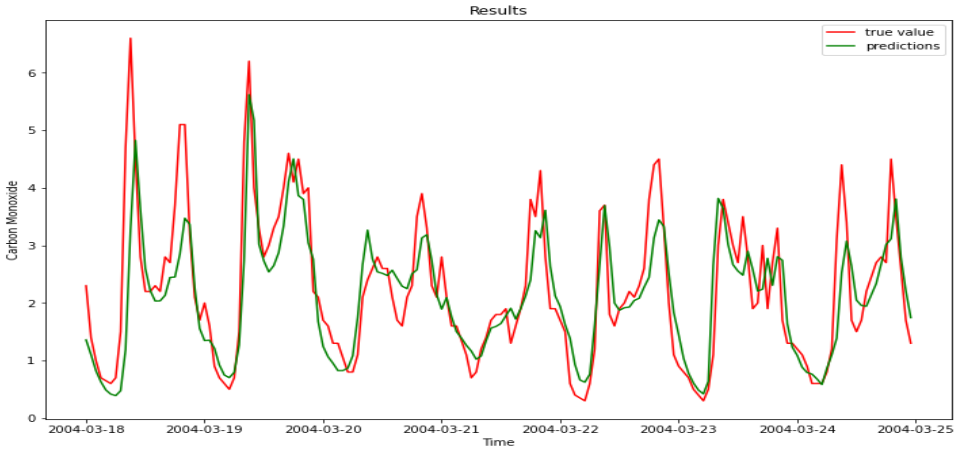
6. Final Model (LSTM - Long Short Term Memory)

The predictions from the above-mentioned models (Exponential Smoothing, Auto ARIMA, FB Prophet and Neural Prophet, provided us with a baseline.
Our final predictions, which gave the least MAPE values on the test set, have been made using LSTM (Long Short Term Memory).
The primary reason for opting for this neural network is due to its ability to remember patterns for longer durations (overcomes the limitation of short term dependency of RNN). Thus, historical patterns in CO and Temperature are taken into account while making future predictions. LSTMs achieve this with the help of gates (Forget Gate, Input Gate, Output Gate).

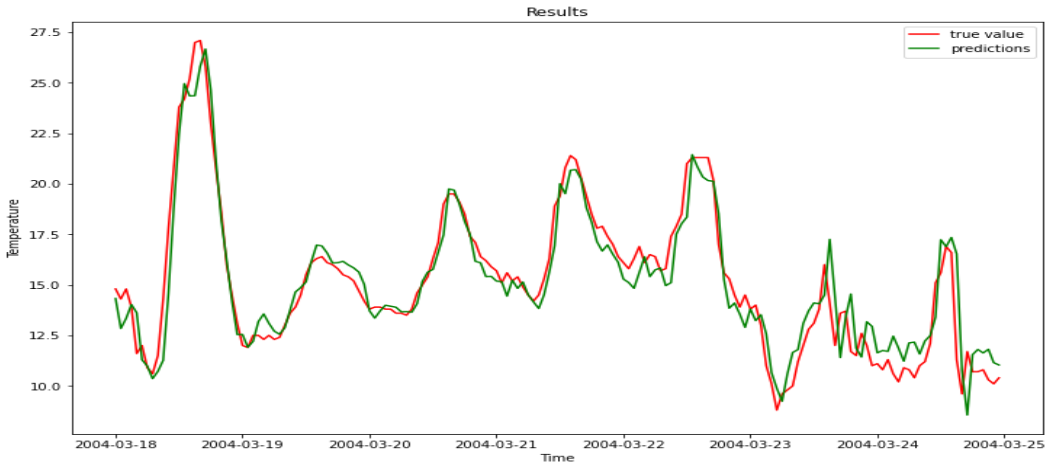
After fine tuning parameters, we decided to generate an hour of prediction based on the values starting from 15 hours before.
As for hyperparameters, they are tuned using validation sets (last day of training set) for each of the forecast days (eg. for prediction for 2004-03-18, training set is 2004-03-11 to 2004-03-16, and validation set is 2004-03-17). The optimal parameters are then saved and used to make future predictions.

Architecture of the LSTM network, using tensorflow keras library.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm_1 (LSTM)                (None, 50)                10400

 dropout_1 (Dropout)          (None, 50)                0

 dense_1 (Dense)              (None, 1)                 51

=================================================================
Total params: 10,451
Trainable params: 10,451
Non-trainable params: 0
_________________________________________________________________
```



Predictions of CO using LSTM model



Predictions of Temperature using LSTM model

Below are the MAPE values using LSTM.

	Date	MAPE
0	2004-03-18	0.259470
1	2004-03-19	0.198947
2	2004-03-20	0.220511
3	2004-03-21	0.176077
4	2004-03-22	0.369441
5	2004-03-23	0.285708
6	2004-03-24	0.200177
7	Average	0.244333

Carbon Monoxide

	Date	MAPE
0	2004-03-18	0.064732
1	2004-03-19	0.034104
2	2004-03-20	0.022608
3	2004-03-21	0.033040
4	2004-03-22	0.052091
5	2004-03-23	0.098298
6	2004-03-24	0.112446
7	Average	0.059617

Temperature

Tech Stack

1. Python
   Libraries in python - numpy, pandas, matplotlib, sklearn, statsmodel, statistics, pmdarima, fbprophet, neuralprophet, tensorflow keras, flask
2. Cloud application - Heroku

Results

Model used - Long Short Term Memory (LSTM) network
Average MAPE for CO - 0.244
Average MAPE for Temp - 0.059