Ventilator Pressure Prediction

GA DSI Capstone Project by Suma Karanam, 25 October 2021

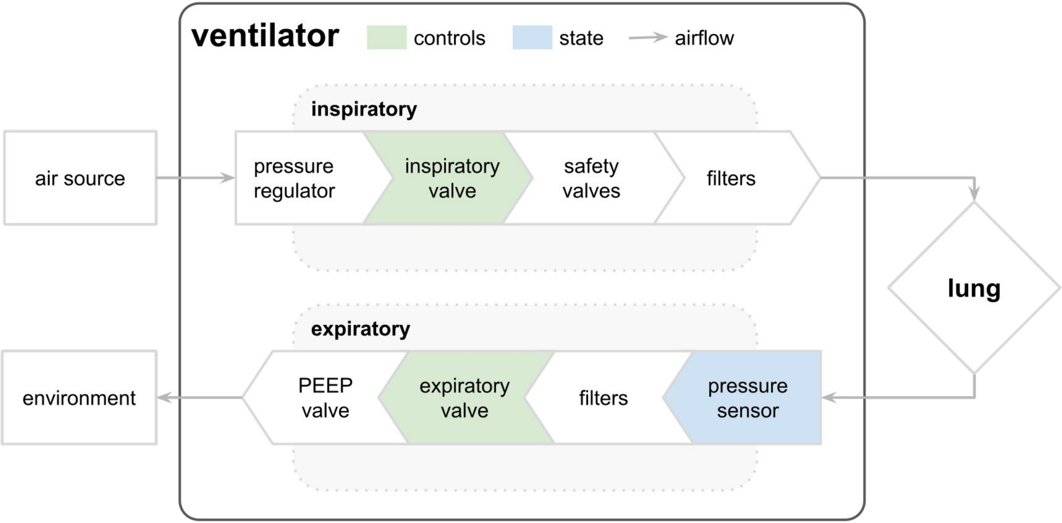
Introduction - The People's Ventilator Project (PVP)

- COVID-19 pandemic has highlighted the need for a low-cost, rapidly-deployable ventilator
- PVP:
  - Open-source
  - Low-cost
  - Resistant to supply chain shortages

source 1: PVP homepage

source 2: medRxiv Preprint

Setup and Operation



1. Ventilator circuit attached to an artificial test lung
2. O_2/Air mixture applied to the system via a hospital gas blender
3. Clinician sets
   a. Desired peak inspiratory pressure (PIP)
   b. Desired expiratory pressure
   c. Target respiratory rate
4. Clinician can modify these parameters in real-time as needed

[Diagram from Kaggle](#)

Problem Statement

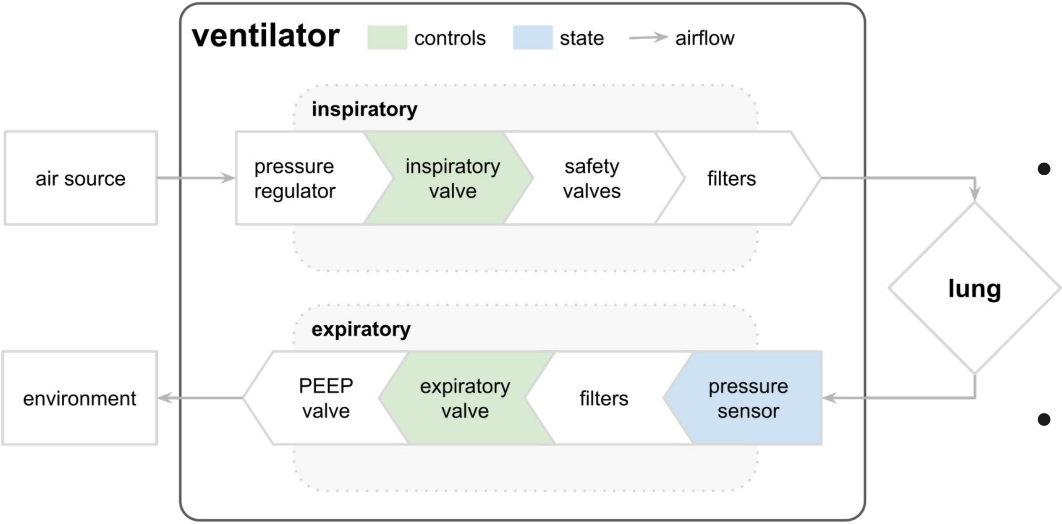


Diagram from Kaggle

- Predict the airway pressure highlighted in blue in the diagram
  - This prediction will be used as an input to inform modifications to the design of the circuit
  - Minimize mean absolute error
- The control variables
  - u_in: extent to which the inspiratory valve is open. Continuous variable between 0 to 100
  - u_out: A binary variable which indicates if the expiratory valve is open or not
- Lung Attributes
  - R: Resistance, higher values of R -> more difficult to increase pressure
  - C: Compliance, higher values of C -> easier to increase pressure

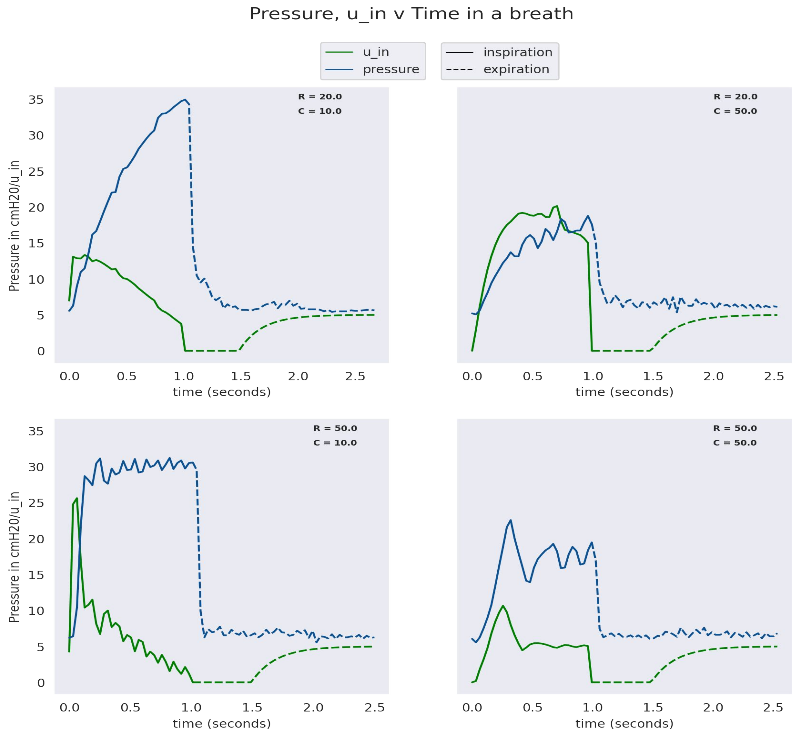
Data

- 75,000 time series of breaths
- Each time series represents a 3 second breath
- Each time series has 80 observations at different time steps between 0s and 3s
- For every time step, we have the following data
    - u_in
    - u_out
    - Lung attributes R and C (which are constant for a given breath)
    - Pressure in the airway measured in cmH_2O (the target variable)

Data cleaning and EDA

- Dropped less than 0.1% of breaths that had negative pressure recorded at some time step
- Analyzing a few sample time series of breaths

Sample breath time series

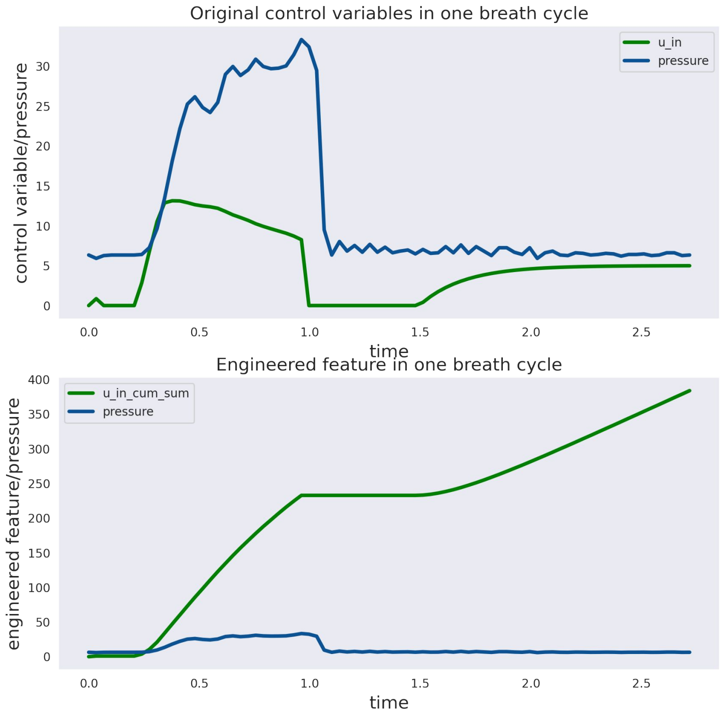


- Inspiratory phase: airway pressure increases, reaches a peak inspiratory pressure (PIP)
- Expiratory phase: airway pressure decreases until it reaches the end expiratory pressure

Approach

- Used tree based models
- Feature engineered ‘breath-level’ features
- Tuned the hyperparameters of the models

Feature engineering: Breath-level features

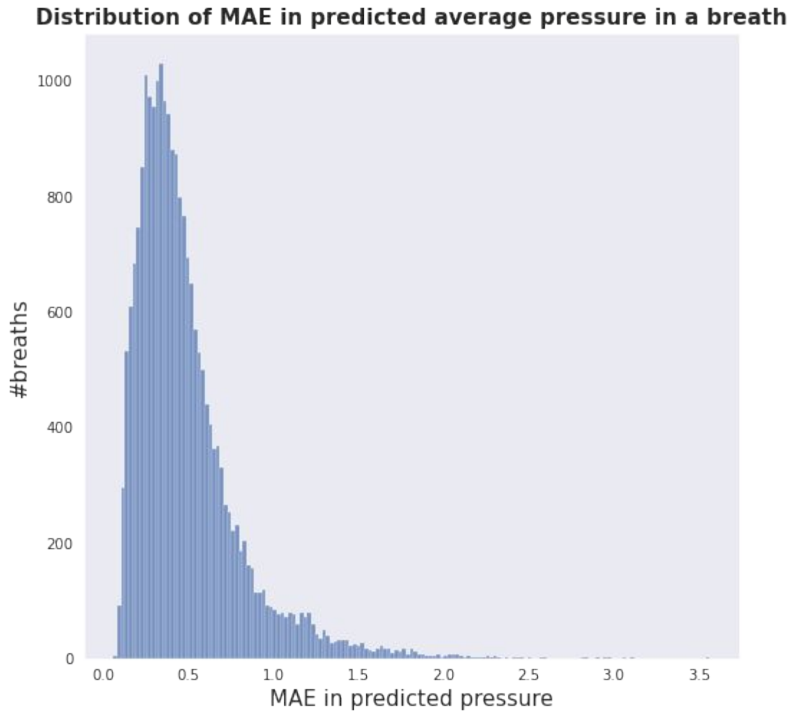


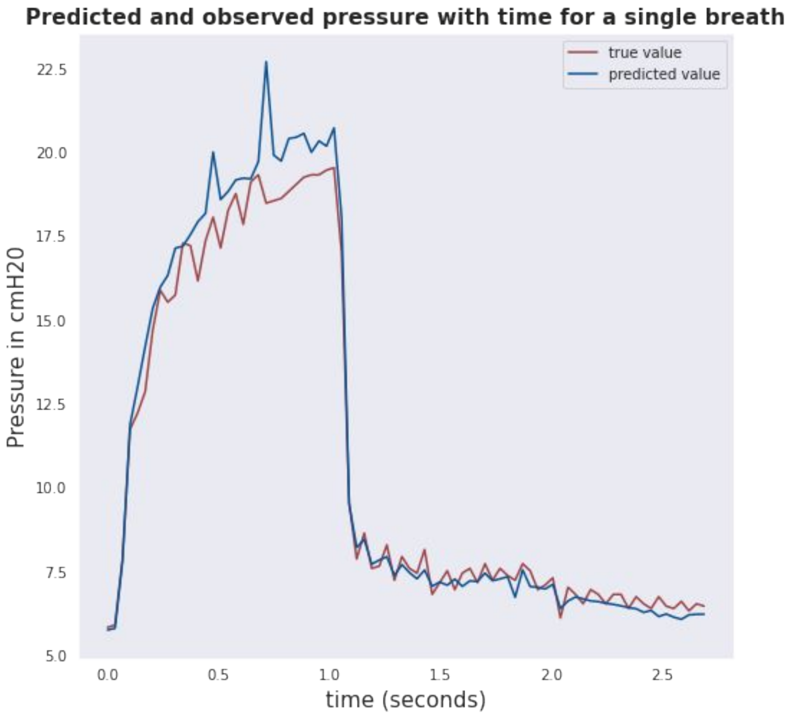
- Cumulative sum of u_in and u_out
- Average of u_in and u_out
- Rolling sum (over 2 time steps) of u_in and u_out
- Lags (over 1 and 2 time steps) of u_in and u_out
- Difference in the time steps
- Polynomial interaction features, $(u\_in)^a \times (time\_step)^b$ for various values of a and b

Model Performance

	Mean absolute error in pressure (cmH_2O)	
	Training data	Test data
Null model	6.22	6.18
Random forest regressor (without engineered features)	2.05	2.08
Random forest regressor (with engineered features)	0.36	0.49
Gradient boosting regressor	0.14	0.45

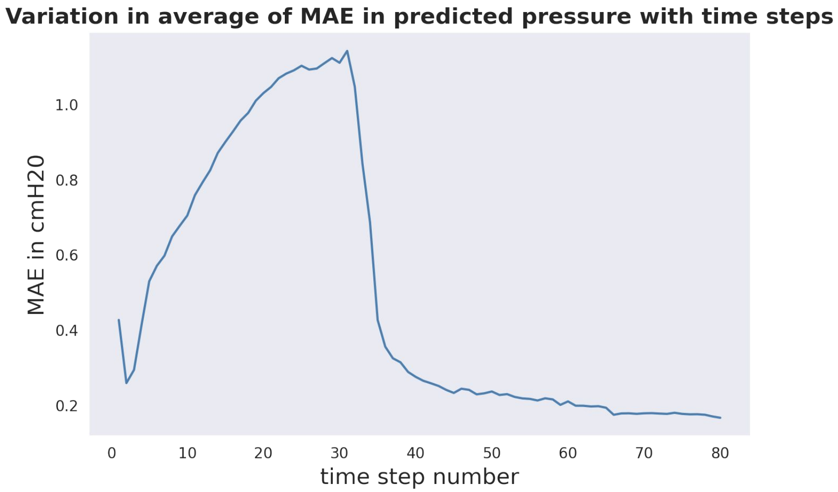
Error analysis





Error analysis

Error in the inspiration phase is much higher than the error in the expiration phase



	Inspiration phase	Expiration phase
Mean absolute error in predicted pressure (cmH20)	0.84	0.27

Conclusions

- **Current recommendation:**
  - Use the random forest regressor with a mean absolute error of 0.49 as this model was less over fit than the best performing model and will likely generalize better
- **TODO**
  - Use the current model to seed the first few pressure values in every new breath cycle and use a time series model to predict the pressure for the remaining time steps
  - Use deep learning techniques such as RNNs which are expected to do well with data that has time dependencies)
  - More hyperparameter tuning and feature engineering to reduce the mean absolute error