



# TIME SERIES MODELING FOR PREDICTIVE MAINTENANCE

Samuel Scovronski

April. 7<sup>th</sup>, 2022

# PROBLEM STATEMENT



Companies and Customers want to minimize the cost of ownership while maximizing the life of their products



Current method to address those needs is a prescribed preventive maintenance schedule which results in an unoptimized solution



If scheduled too frequently then the cost of ownership is not minimized



If scheduled too infrequently then a failure could occur resulting in a lower product life

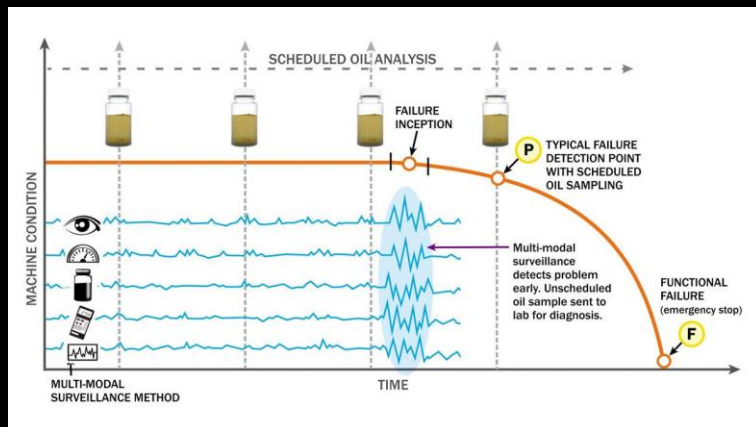
Can we reliably predict when a failure will occur with sufficient time for an intervention?

# WHY IS IT IMPORTANT?



- Minimize unplanned downtime

- Minimize maintenance costs



- Sell replacements before failure



# DATA SOURCE

---

Kaggle: Microsoft Azure Predictive Maintenance

---

Telemetry data collected hourly over a year for 100 machines

---

Includes failure, maintenance, and error logs for the machines

---

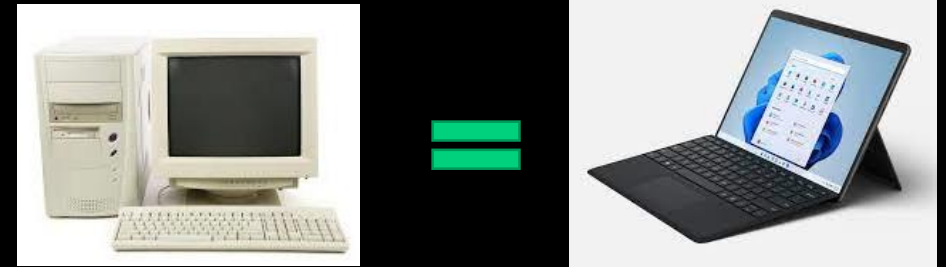
Telemetry data consists of readings from four sensors: Vibration, Rotation, Voltage, and Pressure

---

Unbalanced classes with 876,000 rows of data, and only 700 rows with failures

# DATA EXPLORATION

- Machine age makes no difference



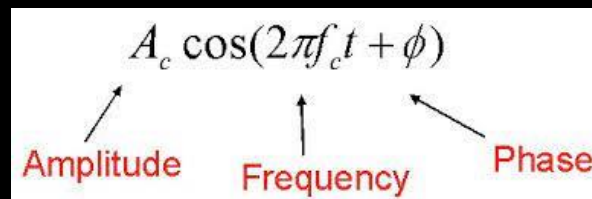
- Data spikes

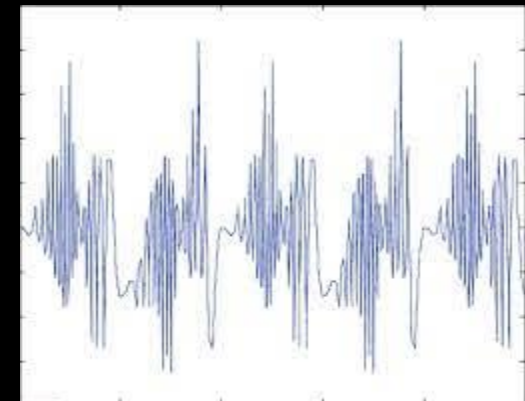


- Synthetic data

$$A_c \cos(2\pi f_c t + \phi)$$

Amplitude      Frequency      Phase

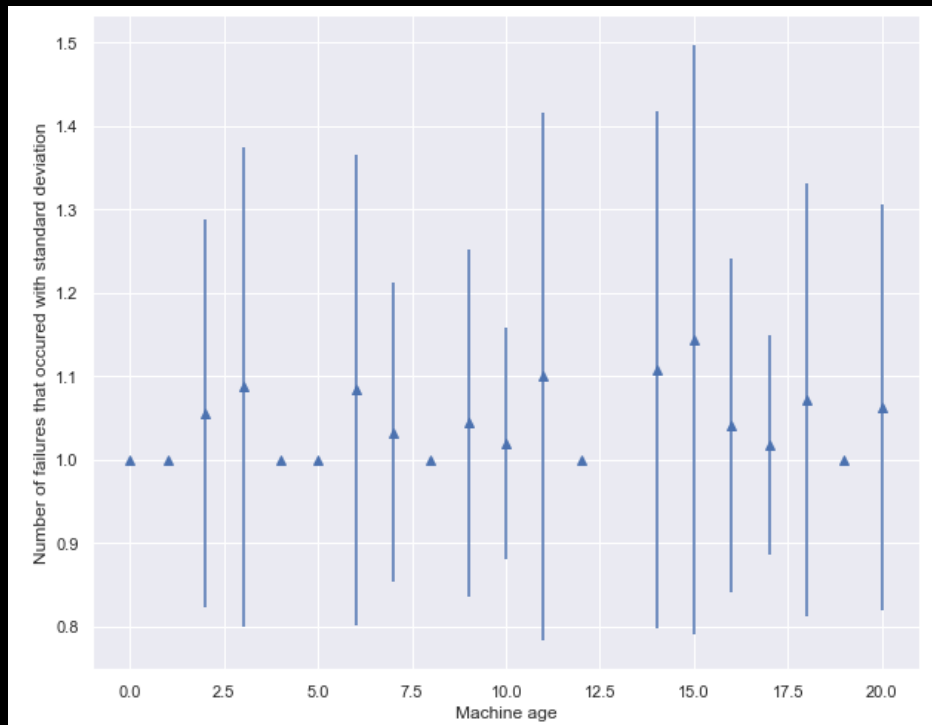
A diagram showing the equation  $A_c \cos(2\pi f_c t + \phi)$ . Below the equation, three labels are placed: 'Amplitude' with an arrow pointing to  $A_c$ , 'Frequency' with an arrow pointing to  $f_c$ , and 'Phase' with an arrow pointing to  $\phi$ .



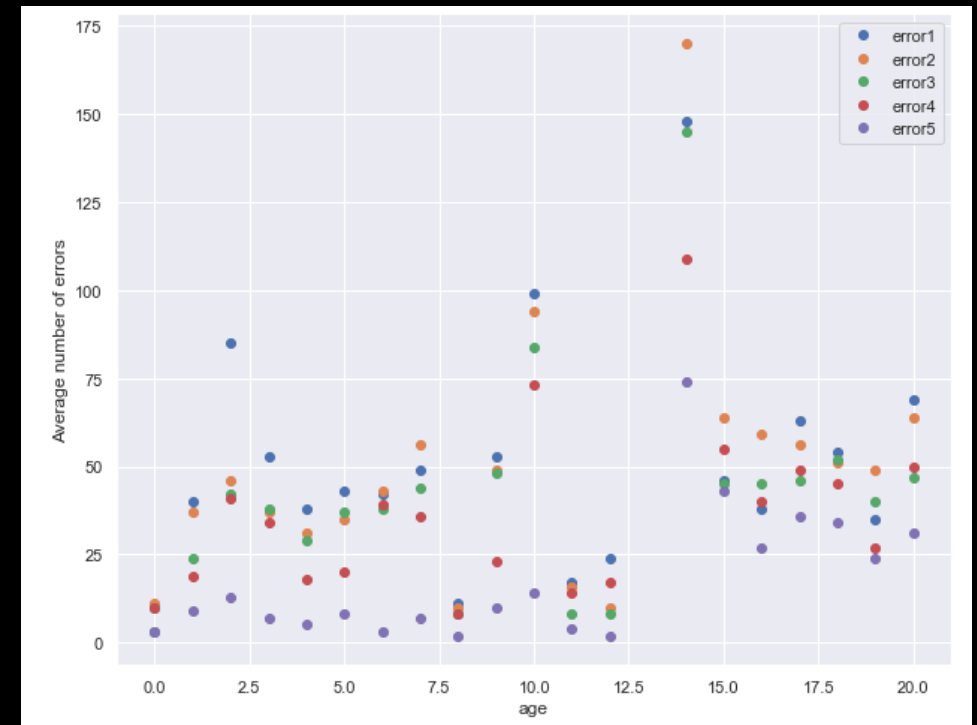


# MACHINE AGE DOES NOT IMPACT PERFORMANCE

Older machines do not experience more failures

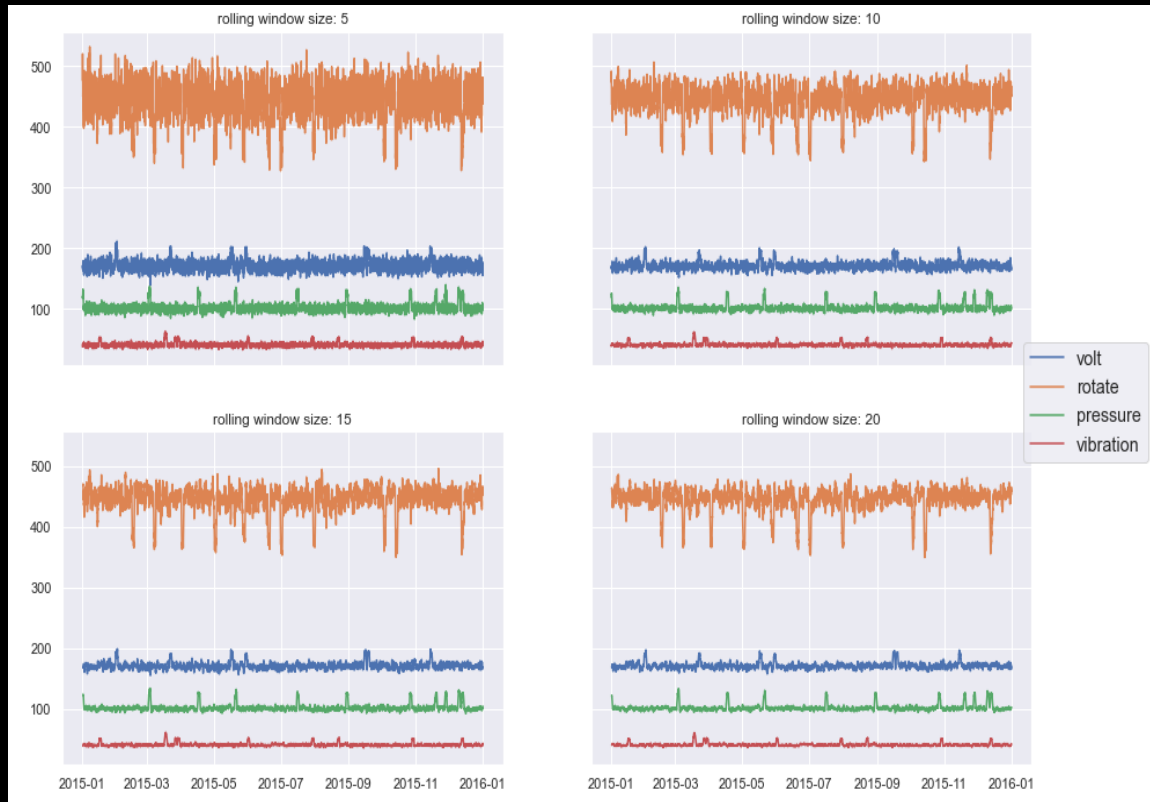


Machines experience 0.3% more errors in general per year of usage

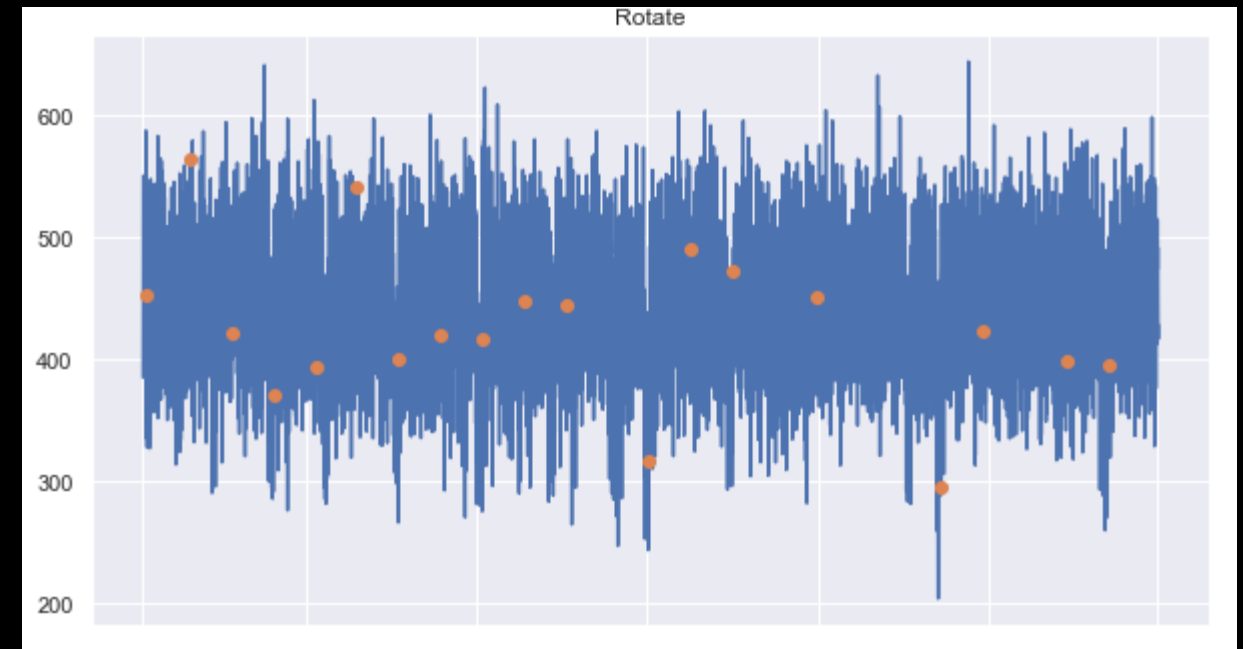


# SENSOR ANOMALIES & FAILURE PREDICTION

The moving average shows deviations from typical operations

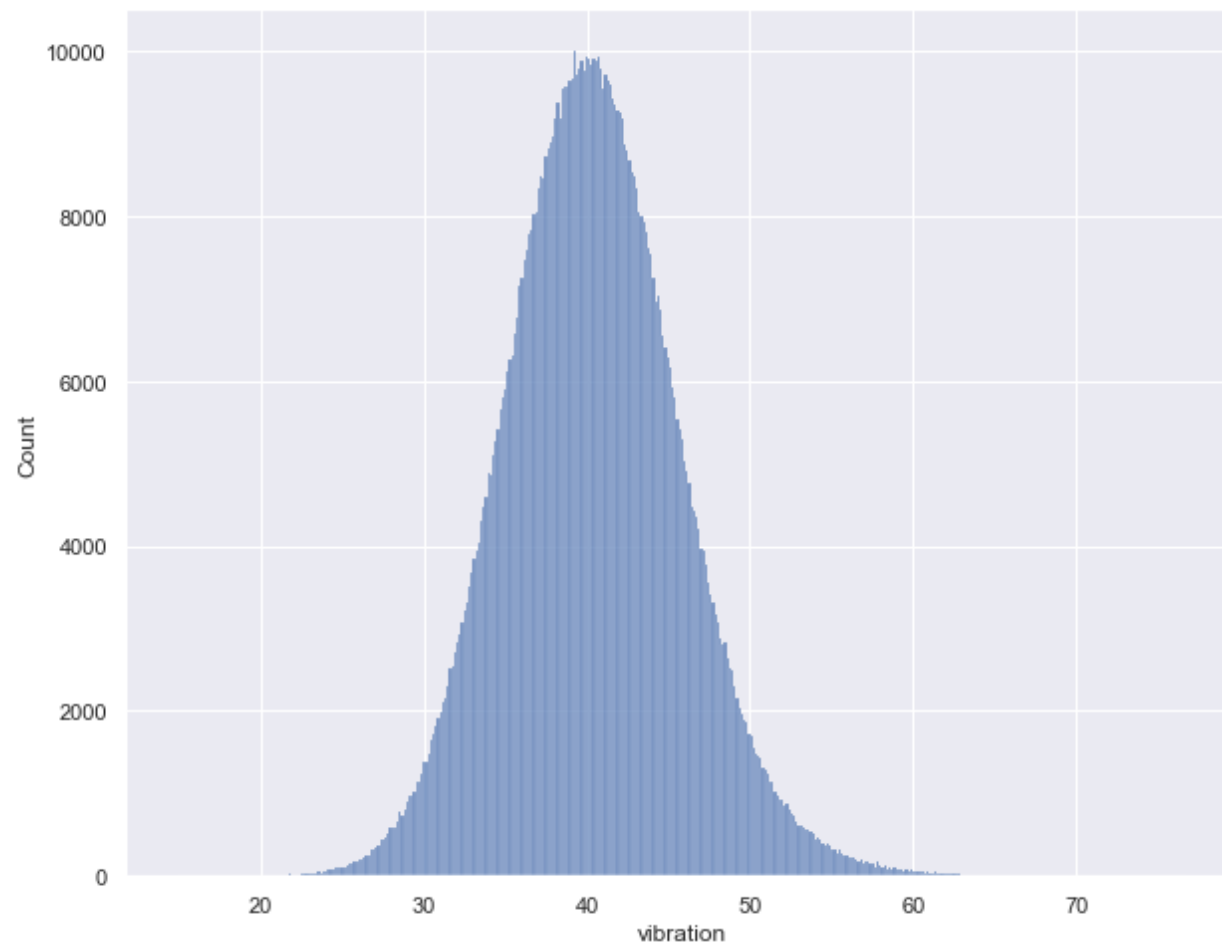


The failures that occur occasionally overlap with the deviations



# SYNTHETIC OR MEASURED DATA?

- The data owner did not state if the data was generated or collected from machines
- Although there is no way to conclusively test if the data was generated, based on the distribution the data seems like it was generated because it is almost perfectly normally distributed

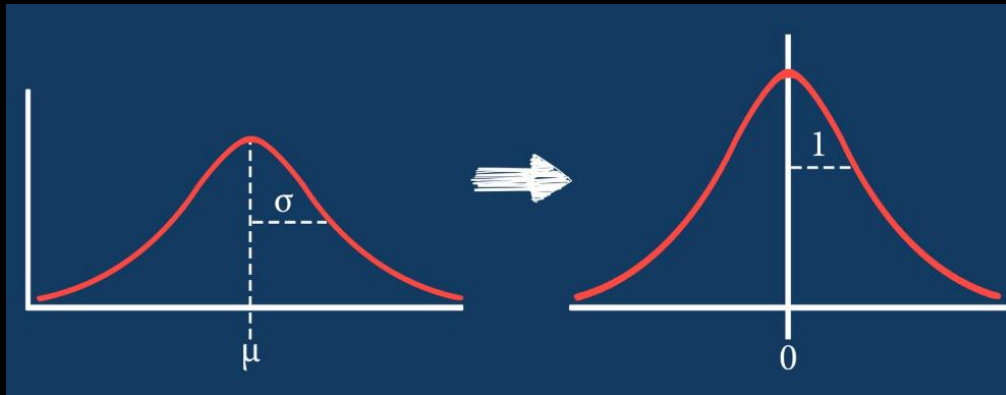
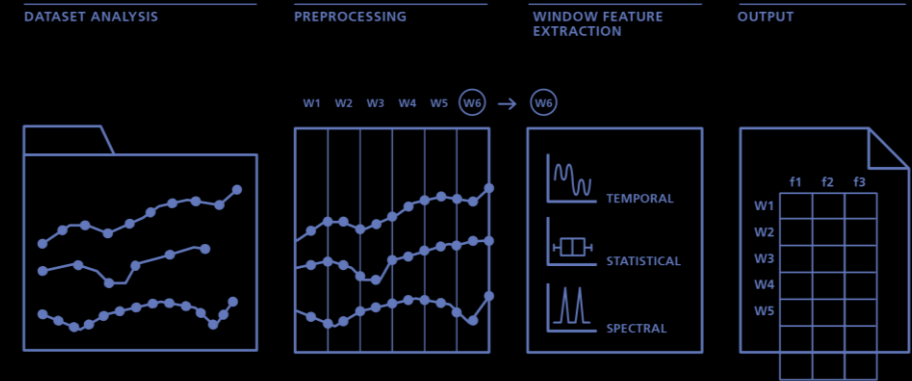




# DATA PREPARATION

## 1. Windowing Data and Feature Extraction

- The data series was grouped into time windows
- Window size was held constant and was advanced one time step between groups
- Statistical values were calculated on each window



## 2. Standardization

- Mean of 0 and standard deviation of 1
- minimize the impact on feature importance

## 3. Target Value

- Predicting failure 5 timesteps in the future
- Need to offset the failure timestamp

# EVALUATION CRITERIA & BASELINE MODEL



Predict every failure that will occur →  
maximize recall



Minimize nuisance tripping →  
maximize precision



These goals can be combined into a single value →  
maximize f1 score



A model that always predicts failure would have 100%  
recall, but 0% precision resulting in an f1 score of 0



A random guessing model would have a recall of 50%,  
and precision of 0.08% resulting in an f1 score of 0.0016  
on average

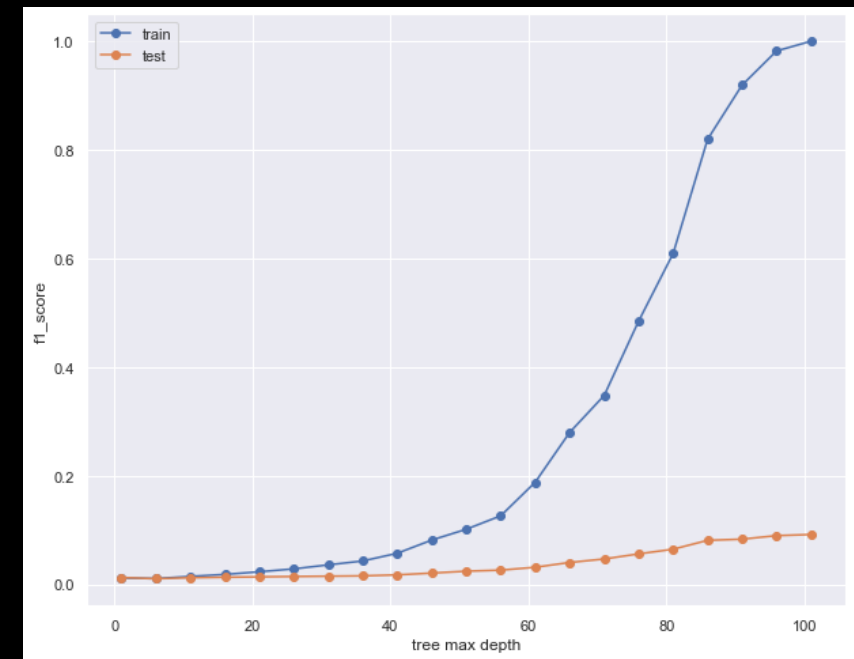
# MODEL SELECTION

All models used a weighted error during fitting due to the data imbalance

Model Type	F1 Score
Decision Tree	.0884
Random Forest	.0524
Logistic Regression	.0067
Linear SVC	.0000

Although much better performance than the baseline model, still far from useful

Conclusion: Decision Tree model performed the best when the training data was overfitted



# OVERSAMPLING



In an attempt to increase the score can use more complex models like a Neural Net, but they do not allow error weighting to compensate for the class imbalance



Used the SMOTE method for oversampling



Critical to split the data before oversampling or else risk contamination of information of training data in the other data sets

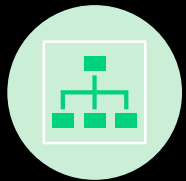
Model Results: SMOTE before splitting

Dataset	Precision	Recall	F1 Score	Accuracy
Train	.99	1.0	1.0	1.0
Validation n	.99	1.0	.99	.99

Model Results: SMOTE after splitting

Dataset	Precision	Recall	F1 Score	Accuracy
Train	.99	1.0	1.0	1.0
Validation n	.97	.21	.35	.60

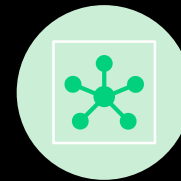
# MODEL REFINEMENT



Data split into a training, testing, and validation set



Testing set used to prevent overfitting while the model was being trained



One hidden layer with 50 nodes

Dataset	Precision	Recall	F1 Score	Accuracy
Test	.94	.83	.88	.89
Validation	.94	.86	.90	.90



# FUTURE WORK

- Create a tradeoff curve between the f1 score and the prediction horizon by creating multiple models
- Combine these models to give a probability of failure within X amount of time