

# AIHack2020

Chris Chia, Sherman Khoo, Erwan Delorme

March 1, 2020

## Code

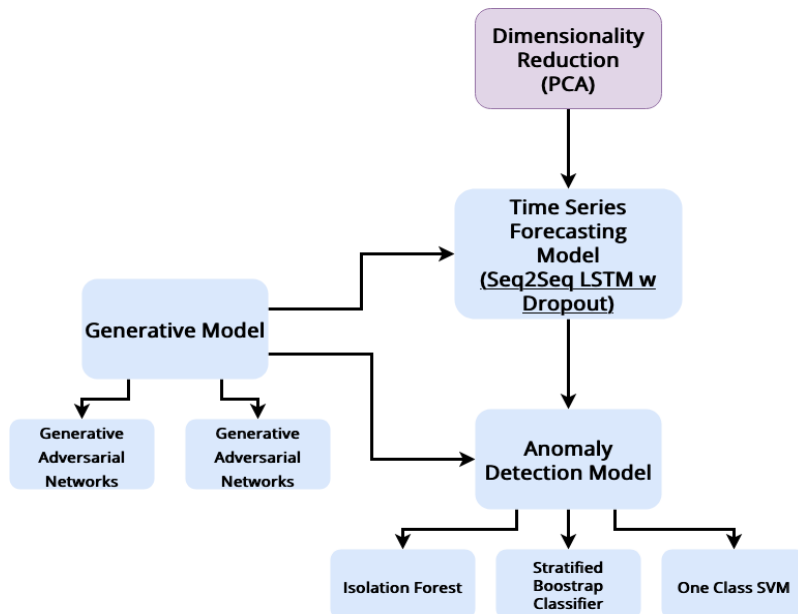
A notebook containing our code/visualisations is available [here](#) and our Github repo is available [here](#)

## Motivation and Solution

The aim our solution is to create a pipeline of models that can "understand" the data generating process (using GANs or Autoencoders), produce a probabilistic forecast of future measurements (Forecasting using Seq2Seq LSTM with Dropout) , and at each step produce a probability of an anomaly (An ensemble of anomaly detection methods).

By being able to understand latent structure of the data and what conditions that are more likely to be associated with *anomalies* advance planning can be done, leading to significant reductions in maintenance costs and time, and also increasing work safety.

This solution would take the form of a dashboard built in Flask and Plotly, in which the user can generate synthetic data and a corresponding forecast. Given the time constraints, we were not able to complete this, but have made progress on the underlying models.



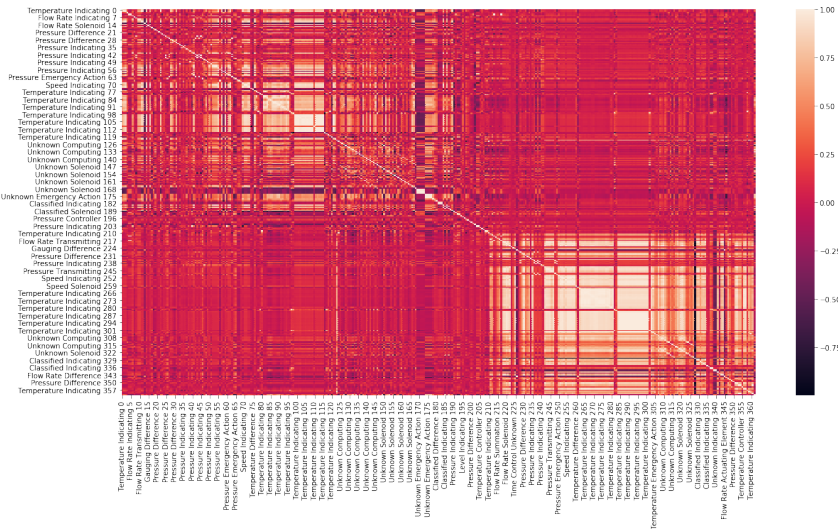
## Data Description

We are given a preprocessed dataset *clean\_data.csv*, consisting of 106710 observations with 362 features, corresponding to sensor readings for a single *asset*, a "low pressure compressor". 9 indices corresponding to 9 automatic overrides were also provided, associated with the raw dataset *raw\_data.csv*

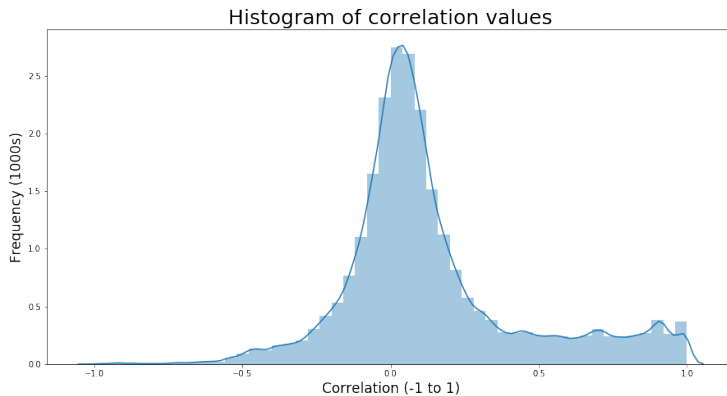
The Preprocessing that has been done for us: datapoints for which "Temperature too low (below 80)" and "Exit pressure too high (above 58)" are removed without smoothing, and features with more than 4 hours missing continuous data, more than 5% total data missing are dropped.

## Data Exploration

The data is too high in dimensionality for individual plots; we plot a correlation matrix and observe that there is significant correlation for certain features.



A histogram of correlation values shows that this is indeed the case; The distribution of correlations roughly follows a normal with  $\mu = 0.1347$ ,  $\sigma = 0.3067$  bounded in  $[-1, 1]$



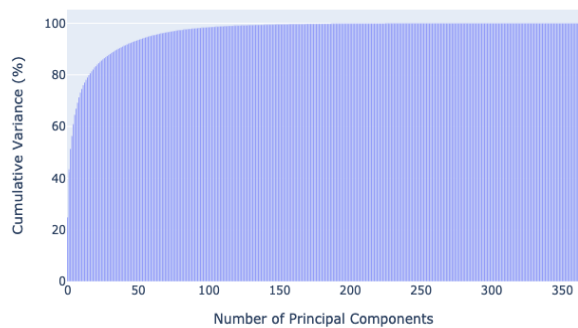
# Dimensionality Reduction

The high dimensionality of the data, coupled with the correlation between features suggested that dimensionality reduction was a good first step, in order to both reduce complexity for use in further models, and also for feature extraction/ to model a latent space. We attempted two principal approaches: Principal Components Analysis and Autoencoders.

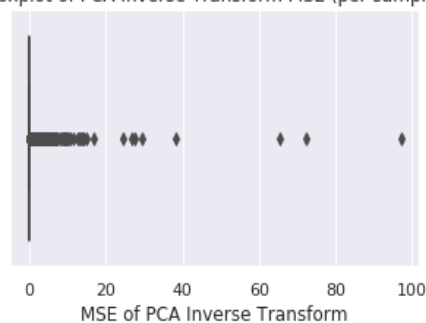
## Principal Components Analysis

We first experimented with dimensionality reduction using PCA in sklearn. The data is standardised using sklearn's StandardScaler, with missing values zero-filled; this makes the assumption that each feature is Gaussian and Stationary, which may not necessarily be correct. Feature imputation methods could also be considered as a next step but is costly in terms of time.

Cumulative Variance against Principal Components

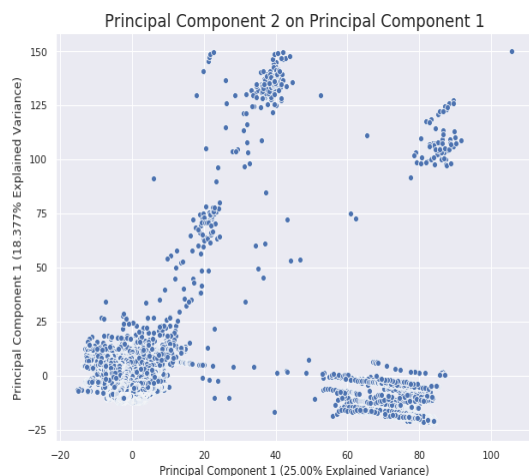


Boxplot of PCA Inverse Transform MSE (per sample)

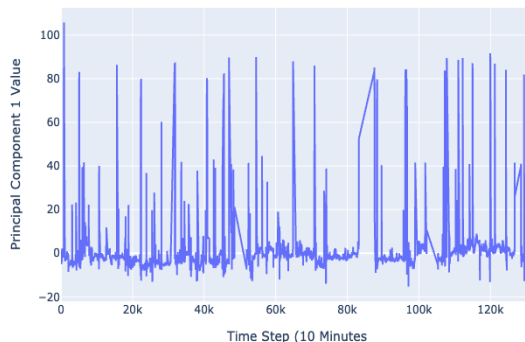


80% and 90% of the explained variance is given by the top 16 and 36 principal components, respectively. With 36 principal components, the mean MSE of an inverse transform (assuming all features are equally important) is 0.101288 with 0.5% of samples exceeding a MSE of 1. These 'outliers' could be potentially be further utilised in the anomaly detection.

We can visualise the latent structure of the data:



Principal Component 1 over time



## **Other Dimensionality Reduction**

We also attempted but discarded approaches using Kernel PCA, due to its high time complexity, and Agglomerate Clustering, as without more explicit engineering it would be limited to merging the features linearly.

## **Autoencoders**

We compare PCA with dimensionality reduction with Autoencoders.

## **Generative Models**

### **Variational Autoencoders**

We can adapt the autoencoder to generate samples.

### **Generative Adversarial Networks**

## Time Series Forecasting

We built a Sequence-to-Sequence LSTM as a way to forecast the next 6 (corresponding to next hour) measurements. We added Dropout layers as a way to create confidence intervals.

## Anomaly Detection

With only 9 indices known to correspond to anomalies (automatic overrides), our initial intention was to use unsupervised outlier detection methods to identify any other potential anomalies and take an *ensemble*:

### Isolation Forest

We fit a Isolation Forest to the standardised *clean\_data.csv*, which we shall term  $X$ , with the following parameters: **100 estimators, subsampling 70% of features, with bootstrap**. The model detects 3351 outliers, or 3.14% of the data

The advantage of Isolation Forests are that it can learn non-linear structures and multi-modalities, the disadvantages is that this method is not interpretable.

### One Class SVM

We fit a linear One Class SVM, first on the standardised data  $X$ , and then on the PCA transformed data  $X_{pca}$ . This makes the assumption that anomalies are linearly seperable.

The advantage of this is that it is interpretable (given its linear decision boundary), but the running time ( $O(n^3)$ ) takes too long to render it practical.

### Imbalanced Classification

Given the relative limitations of the two unsupervised methods, we discarded the previous methods and experimented with treating the problem as an imbalanced classification problem.

We use a stratified bootstrap method; we undersample non-anomalies and keep all anomalies and concatenate them into training set  $df_i$  with target variable  $y$  (whether it is an anomaly).

For each  $df_i$ , we fit a classifier  $f_i : df_i \rightarrow y_i$ . We require a classifier that can return probabilistic estimates such as Logistic Regression (sklearn), Gradient Boosted Decision Tree (CatBoost), or Bayesian Networks. We repeat this for  $N = 100$  rounds, and take the average of the feature importances  $I \in R^{|F|}$ .  $\hat{I} = \frac{1}{N} \sum I$ , . For Logistic Regression this could be the magnitude of coefficients, CatBoost the in built feature importance and Bayesian Networks Shapley Values. This allows us to determine the association between each feature and an anomaly occurring. We can further take an ensemble of these classifiers.

In this case we undersampled non-anomalies, another avenue to explore could be to use SMOTE to oversample anomalies.

## Further Development

Our models (apart from the time series forecasting) in general, i.e. those for dimensionality reduction and anomaly detection have made little use of the temporal structure of the data; this could be an avenue to be further explored in the future, however in the context of AIHack we were heavily constrained by time and our lack of knowledge

## Bibliography

- 1.
- 2.
- 3.



Is this a hot dog or not a hot dog?