GreyAtom Hackathon: Predicting the Batch Performance of Machines

Team: Abhineet, Monika, Samantha & Soumya

Problem Statement

Data is being recorded from various condition monitoring sensors in a manufacturing plant. There are hundreds of such sensors which may be impacting the quality of final product which is getting manufactured, hence we can model this problem as a multivariate regression problem. Important point to note about real-life datasets is that they have various shifts since they are being recorded over the period of time and environment changes with time and further that is going to be the main focus of this challenge.

Why solve this problem?

- To predict how the machines will perform.

Objective:

- Achieve the highest possible R2 score for label y2.
- R2 Score:

  "The proportion of the variance in the dependent variable that is predictable from the independent variable(s)."

Data Overview

Input parameters are the sensors attached to a machine, which are 54 in number. In the dataset, they are named as x0, x1, x2 … x54.

Since this is a batch manufacturing, a batch is of a certain time period, say t0 is the batch start and t6 is the batch end. Sensor values being recorded at time instance t0, t1, t2 … t6.

All values are numerical (float or binary), there are no string or multiclass type features.

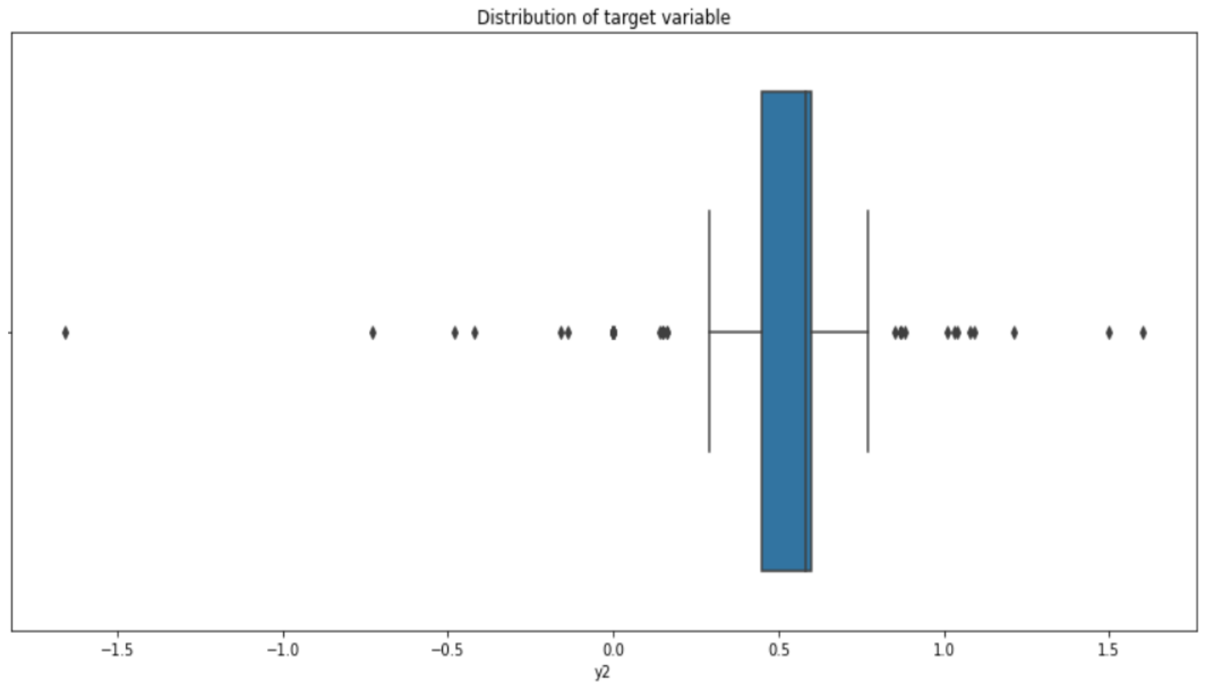
There is output parameter y2 which has to be considered independently . They describe the batch performance.

Data Overview

The given data is divided into Train and Test data sets, where Train Data consists of 794 records/rows and 387 columns including the target variable ‘y2’. The Train dataset contains 89 rows and 386 columns.

Upon Analysis it was found that there are no missing values in both the train and test data.

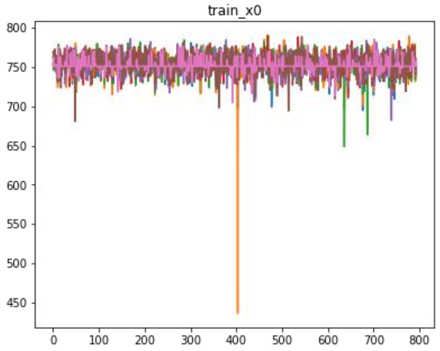
Exploratory Data Analysis: y2

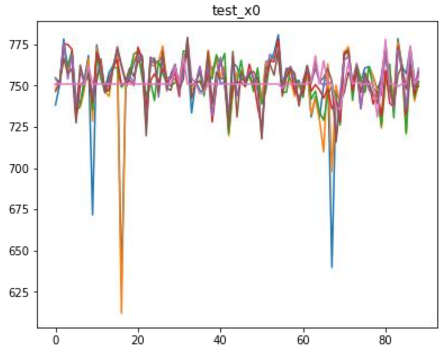


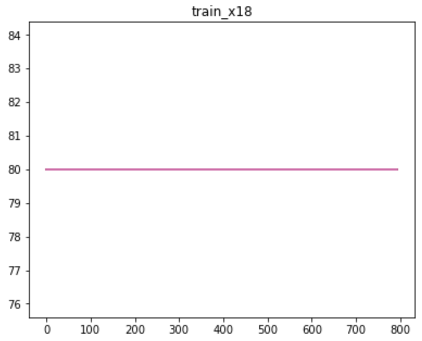
Insights:

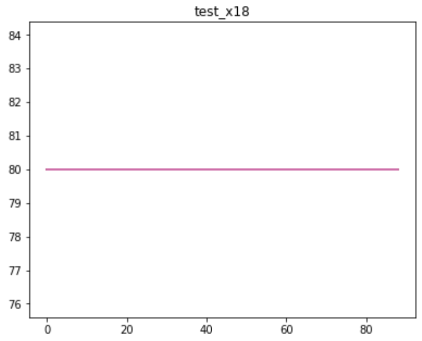
- Range of target value is (1.6, -1.66).
- Most of the data in y2 lies in (0.4, 0.6)

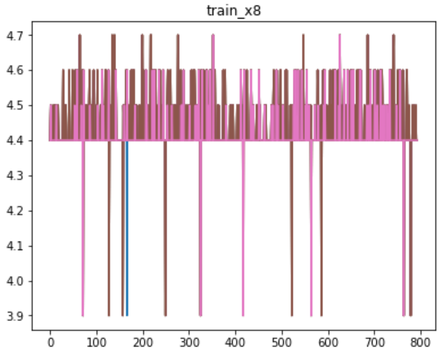
Exploratory Data Analysis: Sensor readings

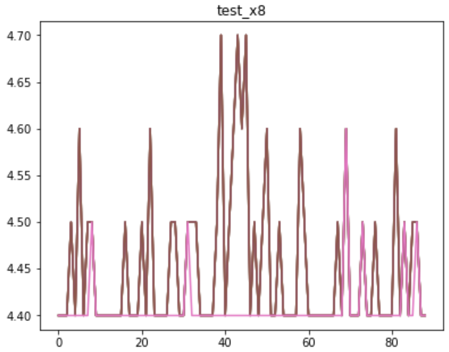


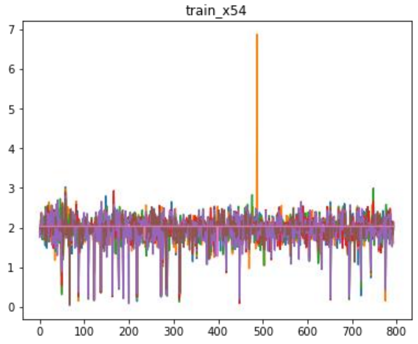


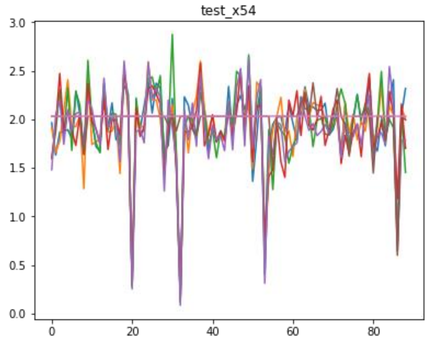












Dimensionality Reduction techniques

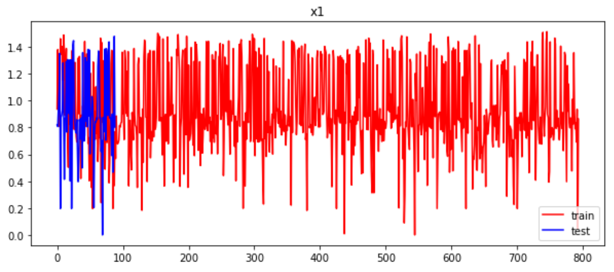
- Created a new dataframe with average of sensor data readings for each sensor.

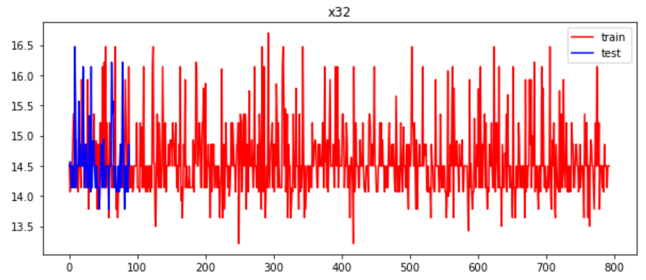
  e.g x0_t0, x0_t1, x0_t2, x0_t3, x0_t4, x0_t5, x0_t6 becomes x0 and so on.

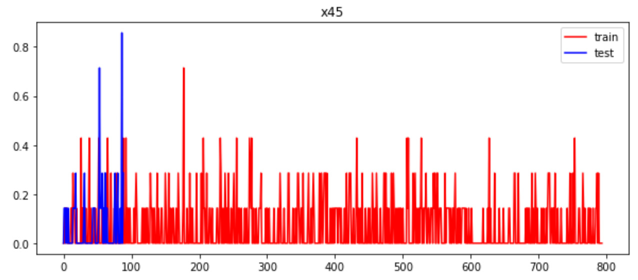
  The new train dataset obtained after taking the mean of readings for each sensor contains 57 columns in which there is an Id column, a target variable y2 and 55 columns each corresponding to one sensor x_i where $0 <= i <= 54$.

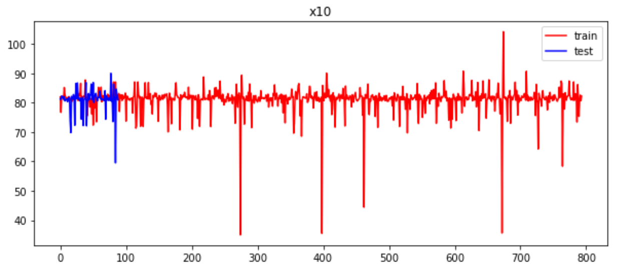
- **PCA (Principal Component Analysis)-** It is a method that uses simple matrix operations from linear algebra and statistics to calculate a projection of the original data into the same number or fewer dimensions.

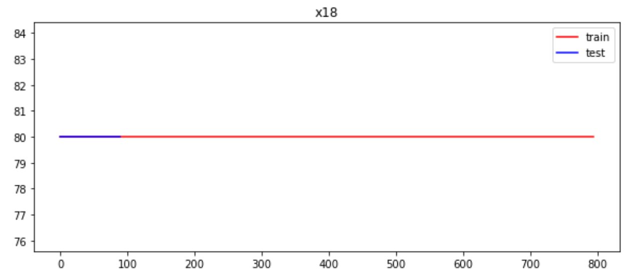
Visualizing the averaged sensor data from train and test dataframes

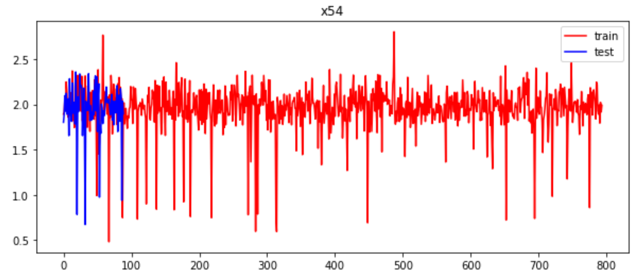




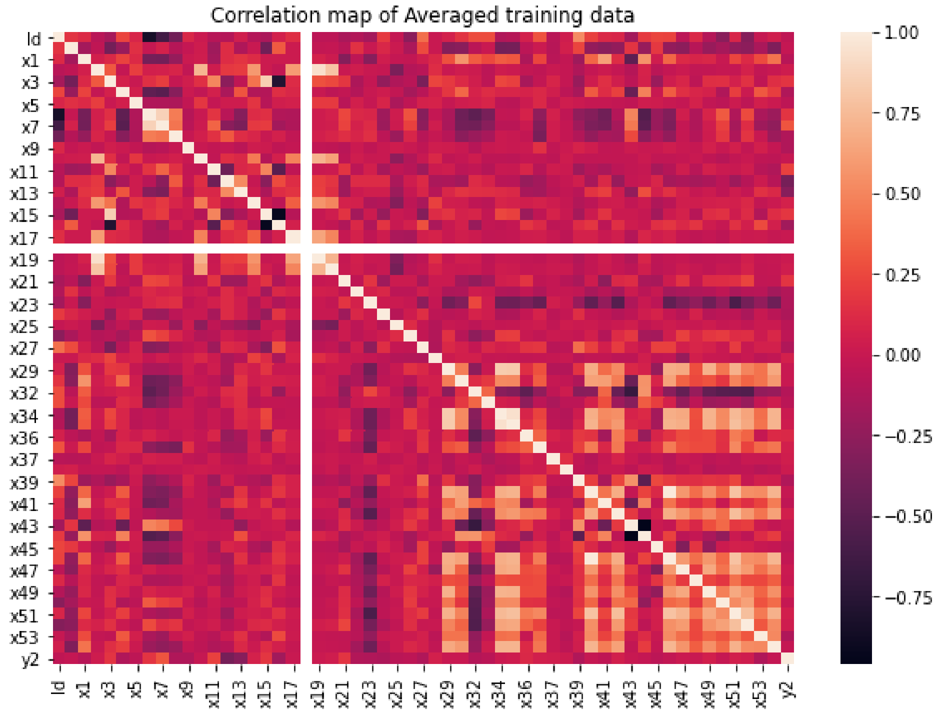








Correlation of the new averaged sensor data



Hypothesis Testing

Test 1: To check if the distribution of y2 is normal or not.

The **Shapiro-Wilk test** tests the null hypothesis that the data was drawn from a normal distribution.

- H0: y2 is Normal distribution data
- H1: y2 is not Normally distributed data

Result: stat=0.729, p=0.000 : Null Hypothesis Rejected, y2 is not normally distributed.

Hypothesis Testing

Test 2: To check which features are correlated with target feature y2

The Spearman rank-order correlation coefficient is a nonparametric measure of the monotonicity of the relationship between two datasets. Like other correlation coefficients, this one varies between -1 and +1 with 0 implying no correlation. Correlations of -1 or +1 imply an exact monotonic relationship

- H0: the feature is independent/uncorrelated with y2.
- H1: there is a dependency/correlation between the feature and y2..

RESULT: The features that are correlated with y2 are: x3, x4, x5, x6, x9, x11, x12, x13, x14, x15, x16, x21, x22, x23, x24, x27, x30, x31, x32, x33, x34, x36, x41, x43, x44, x46, x49, x51, x53.

Hypothesis Testing

Test 3: To check whether the features of the train and test datasets come from the same population.

Using the 2 sample Kolmogorov-Smirnov test on all the features to see whether they follow the same distribution or not.

- H0: The two samples follow the same distribution.
- H1: The two samples do not follow the same distribution.

Result: All the features follow the same distribution in train and test data.

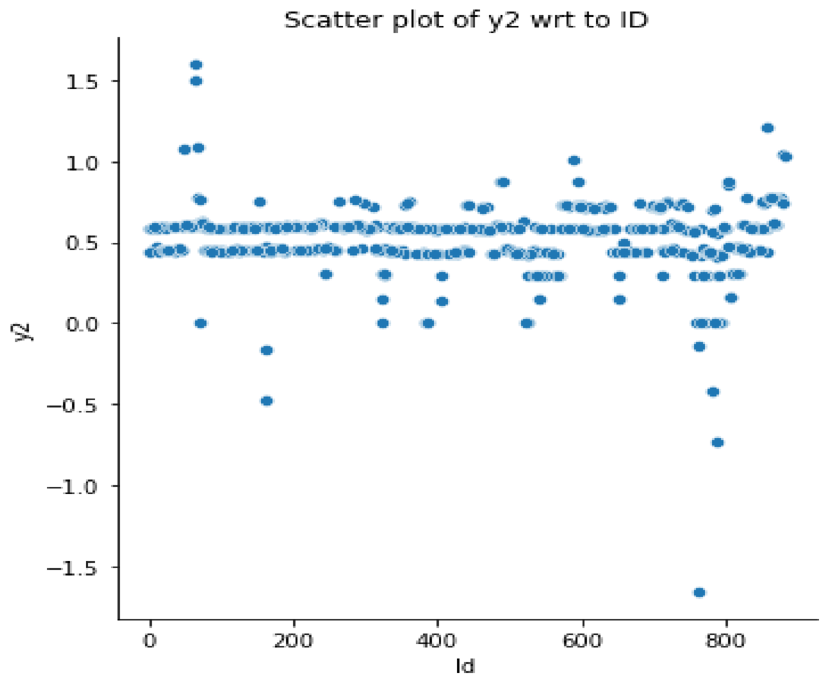
Testing for Covariance Shift

The basic steps that we have followed are:

- Creating a random sample of the training and test data separately and adding a new feature origin which has value train or test depending on whether the observation comes from the training dataset or the test dataset.
- Combining these random samples into a single dataset. Note that the shape of both the samples of training and test dataset should be nearly equal, otherwise it can be a case of an unbalanced dataset.
- Creating a model taking one feature at a time while having 'origin' as the target variable on a part of the dataset (say ~75%).
- Predict on the rest part(~25%) of the dataset and calculate the value of AUC-ROC.
- Now if the value of AUC-ROC for a particular feature is greater than 0.80, we classify that feature as drifting.

RESULT: There was no feature with a covariance shift in the train and test dataset.

Relationship between Id and y2



Correlation Coeffeicient: -0.036

Models and Approaches

Models	R2 score
Linear Regression	0.20
Linear Regression with RFE	0.22
Random Forest Regressor	0.52
Random Forest Regressor With RFE (30 features)	0.49
Random Forest Regressor with Hyperparamter Tuning	0.34
Gradient Boosting Regressor	0.49
Voting Regressor (Ensembling)	0.48
Random Forest Regressor with PCA	-0.34

Different Techniques to improve the Random Forest model's performance (R2 score)

1) Differencing the rows

It is given that the row numbers are important as they are sorted time-wise. Which means that the 1st row represents the 1st readings taken from the sensors for a batch of machines, the 2nd row represents the readings taken after that and so on. So we tried differencing the rows on 1st order to see if it improves the performance of the model.

RESULT: R2 Score: -0.027

2) Differencing the columns

Each sensor has 7 readings taken at 7 different time stamps t0, t1,...,t6. Assuming that t0 is the first time stamp and the time progresses from t0 to t6, we try differencing the columns for each sensor, to drop the (t0)th and retain the difference for the remaining on original train data.

RESULT: R2 score: 0.093 (on differenced columns before averaging)

R2 score: 0.132 (on differenced columns after averaging)

Different Techniques to improve the Random Forest model’s performance (R2 score)

3) Using the original dataframe without any feature engineering
Result: R2 score: 0.349

4) Rebuilding the model with only important features.
Using the feature_importances_ method of the random forest model to select the most important features and rebuild the model using the features.
In this method, the importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature.

Result: The top 25 features with importance value greater than 0.1 are [x11, x5, x8, x6, x2, x12, x30, x4, x13, x20, x25, x36, x10, x15, x21, x22, x26, x28, x32, x38, x45, x48, x51, x52, x53]
R2 score of random forest regressor model built using the above features: 0.562

Evaluation & Results

Random Forest Regressor model with selected important features is giving the best performance among the models considered and is used for the final prediction.

Thank you!