Remaining Useful Life prediction of bearing using Self Organizing Map and Support Vector Regression

A Project Report

Submitted by

SANKET DOFE (181080022)

RISHIKESH HIRDE (181080032)

SIDDHESH BAHURUPI (181080067)

MRUNAL BELE (181080013)

T.Y.B.TECH

IN

Information Technology

AT

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE

Department of COMPUTER Engg. & I.T.

MUMBAI

March 2021

PROJECT TITLE:

Remaining useful life prediction based on Self Organising Map and Support Vector Regression.

PROBLEM STATEMENT:

Bearings are critical components of rotating machines since the failure of these may cease the functioning of the entire equipment. The damages observed due to bearing failures are huge and thus there is a need to develop an effective prognostic methodology to prevent the sudden machinery breakdown.
Accurate determination of remaining useful life of bearing is an important issue and a methodology needs to be created to address this issue that accurately predicts the Remaining Useful Life of the bearing so that when it passes a certain level, the corresponding bearings can be replaced so as to avoid failure of equipment.

DATASET:

The dataset used for this project is 'IEEE PHM 2012 Dataset' which contains the datasets for bearings experimented under 3 conditions.
Some datasets contain the observations for the bearings until failure occurs and thus these are used as the training sets for our project. Some datasets contains the observations for the bearings which are truncated before the failure occurs and for these datasets, the remaining useful life is given which is needed to be predicted and thus these are used as the test sets for our project.
The experiments were performed on the PRONOSTIA platform which gave these datasets.

<u>Dataset description:</u>
Bearings were experimented under 3 different operating conditions and there are different experiments done on separate bearings under these 3 different conditions. For a particular condition, some bearings were experimented separately. For a particular bearing, set of observations were taken out every 10 seconds. Every 10 seconds, 2560 observations of vibrations were taken out and these were exported in a csv file.
For bearings experimented until failure contains all the csv files generated every 10 seconds until failure which has been used as training set.
For bearings experimented and the observations truncated which can be used as the test set contains the csv files generated every 10 seconds before the data has been truncated.

Each csv file contains the following attributes:
1. Hour: Hour at which these observations were carried out
2. Minute: Minute at which these observations were carried out

3. Second: Second at which these observations were carried out
4. Microseconds: microsecond at which the particular observation was taken
5. Horizontal Acceleration: Vibrational signal along horizontal axis observed
6. Vertical Acceleration: Vibrational signal along horizontal axis observed

The Hour, Minute and Second for each observation in a particular csv file is common as these were taken out after every 10 seconds.

MOTIVATION:

Rolling element bearings are critical components of rotating machines since the failure of rolling element bearings may cease the functioning of the entire equipment. The performance degradation assessment and accurate determination of remaining useful life are the two key issues in prognostics of rolling element bearings.
In our project we are estimating the remaining useful life of bearings based on self-organising map and support vector regression.
The use of Self Organizing Map was itself a new concept that made us to go with project. The feature extraction especially the frequency domain features from the raw signals was a thing we noticed was different from others and thus we thought of going with this project.
The motivation for doing this project was primarily an interest in undertaking a challenging project in an interesting area of research. The opportunity to learn about a new area of machine learning apart from theoretical knowledge was appealing.

METHODOLOGY:

This project is based on the research paper 'Intelligent bearing performance degradation assessment and remaining useful life prediction based on self-organising map and support vector regression' by Akhand Rai and Sanjay H Upadhyay.
The paper has made use of the Self Organizing Map and Support Vector Regression for the learning and prediction of Remaining Useful Life of the bearings.

The steps carried out in this project to perform the Remaining Useful Life Predictions as proposed in the research paper are:

1. <u>Feature Extraction:</u>
   The input dataset is a time series data of vibrational signals. So there is a need to extract features from it to get meaningful features that represents the data more precisely.
   Number of time-domain features and frequency domain features have been identified that are extracted from the datasets.

Time domain features:

1) RMS
2) Kurtosis
3) Skewness
4) Crest Factor
5) Impulse Factor
6) Shape Factor
7) Margin Factor
8) Peak to Peak

The frequencies of the signal were obtained by the Fourier Transforms which converts the time data into its constituent frequencies.

Fourier Transform is a mathematical concept that can decompose a signal into its constituent frequencies.

Frequency domain features:

1) Mean Frequency
2) Frequency Centre
3) Root Mean Square Frequency
4) Spectral Entropy of raw signal
5) Spectral Entropy of envelope signal

Each csv file corresponding to observations taken every 10 seconds is converted to a feature vector consisting of features calculated using the horizontal acceleration data in csv file.

All the csv files thus give an array of feature vectors.

The computations of each feature are given below:

RMS	$\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$	Mean frequency	$\frac{1}{N}\sum_{i=1}^{N} s_i$
Kurtosis	$\frac{1}{N}\sum_{i=1}^{N} (x_i - \overline{x})^4/\sigma^4$	Frequency centre	$\frac{\sum_{i=1}^{N} f_i s_i}{\sum_{i=1}^{N} s_i}$
Skewness	$\frac{1}{N}\sum_{i=1}^{N} (x_i - \overline{x})^3/\sigma^3$	Root mean square frequency	$\sqrt{\frac{\sum_{i=1}^{N} f_i^2 s_i}{\sum_{i=1}^{N} s_i}}$
Crest factor	$\max\lvert x_i\rvert/RMS$	Spectral entropy of raw signal	$-\sum_{i=1}^{N} p_i . \log_2(p_i)$ $\rightarrow p_i = s_i/\sum_N s_i$
Impulse factor	$\max\lvert x_i\rvert/\frac{1}{N}\sum_{i=1}^{N}\lvert x_i\rvert$	Spectral entropy of envelope signal	$-\sum_{i=1}^{N} p_i . \log_2(p_i)$ $\rightarrow p_i = h(i)/\sum_N h(i)$
Shape factor	$RMS/\frac{1}{N}\sum_{i=1}^{N}\lvert x_i\rvert$		
Margin factor	$\max\lvert x_i\rvert/\left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{\lvert x_i\rvert}\right)^2$		
Peak to Peak	x_{max}-x_{min}		

Here, the x corresponds to the acceleration, s and h denote the single-sided Fourier transforms of the original and envelope bearing signals, f denotes the frequency.

2. Self Organizing Map:
SOM is an adaptive unsupervised neural network used to classify the data by projecting the high-dimensional feature vectors onto two-dimensional topology maps. The topology maps represent the output layer of the SOM network and consists of a two dimensional array of neurons interconnected to each other. Each neuron or node has a weight vector associated to it. The dimension of the weight vector is same as the dimension of the input feature space.
It is also known as Kohonen's Map.

The training of the SOM takes place in the following steps:

i. Initially, the weight vector W_k of each neuron is loaded arbitrarily.
ii. The input vector f_i selected randomly is supplied to all the neurons at a time.
iii. The neuron most similar to the presented input vector is determined. This is achieved by minimising the Euclidean distance between the input vector and the weight vector of the individual neurons. The winning neuron is called as the best matching unit (BMU).
It can be mathematically given by

$$BMU = k \mid \min \| f_i - W_k \|$$

iv. The weight vector of BMU and its adjacent neurons are updated using the Kohonen's learning rule as follows

$$W_k(t+1) = W_k(t) + \alpha(t).h_{ck}(t).[\ f_i(t) - W_k(t)\ ]$$

where t denotes the time, $\alpha(t)$ is the learning rate at time t and h_{ck} is a montonically decreasing neighbourhood function.

v. The steps ii to iv are repeated for all the input data.

A set of healthy features that corresponds to the features that represents the bearing is in a healthy state i.e. it is not damaged is taken from the features extracted in the previous step and are the first $1/5^{th}$ feature vectors as these will always be the features corresponding to the bearing in a healthy state.

These healthy features are used to train the Self Organizing Map.

3. Minimum Quantization Error(MQE):

The Minimum Quantization Error corresponds to the Euclidian distance between the input features and its corresponding BMU.
The remaining feature vectors (apart from healthy feature vectors) are presented to the trained SOM to find the BMI for each. After getting the BMI, the corresponding MQE is calculated for each feature vector.
This gives us a new feature which represents the deviation of the inspected feature vector from the normal state of the bearing, any fluctuation in MQE value indicates either the occurrence of a fault or the presence of an outlier.

4. <u>Life Percentage:</u>

The life percentage corresponds to the percentage of the life of bearing completed at a particular instance of time. When a bearing is new, its life percentage is 0% whereas when it is damaged/failed, its life percentage is 100%. The life percentage of a bearing is calculated as the current age of the bearing divided by the total age of the bearing and expressed in terms of percentage. The benefit of using the life percentage is that there is no need to set a threshold for RUL estimation.

The life percentage in our project is calculated using the feature 'minutes' we calculated present in each feature vector. The 'minutes' feature in the last feature vector in the training set corresponds to the features when the bearing fails. So we divide all the 'minutes' features with the 'minutes' feature in the last feature which gives us the life percentage.

5. <u>BHI (Bearing Health Index):</u>

For RUL estimation purposes, the input vector must be as much monotonic as possible and must increase gradually with the increase in bearing operating time.

BHI corresponds to an index that is monotonic with the bearing operation time. It gives an approach to extract the monotonicity from MQE. It is calculated as

$$\mathrm{BHI}(m) = \exp\left(\frac{\sum_{l=1}^{m} \mathrm{MQE}(\mathrm{l})}{c \times m}\right), \quad m = 1, 2, \ldots\ldots\ldots. N$$

where BHI (m) is the value of health index BHI at the m^{th} measurement point, N is the total number of measurement points and MQE(l) denotes the MQE value at inspection point 'l'. A factor of c is introduced to adjust the scale of BHI and is helpful in maintaining similar BHI trends for different bearings working under the same conditions and eliminating the differences in BHIs under healthy states of the bearings. c is calculated as

$$c = \text{mean}\left[\frac{\sum_{l=1}^{h} \text{MQE}(l)}{\ln(\text{BHI}_0) \times h}\right]$$

where h indicates the time-index sequence for healthy MQE values only which we took as the first 1/5th features vectors of the training dataset. BHI0 denotes the starting value of BHI and can be assumed to be between 0.95 and 0.99.

6. Support Vector Regression (SVR):

Given a set of training data X = [(x_i,d_i), i=1,. . ...,N], where x_i is the input feature vector, d_i is the corresponding target vector and N is the number of training samples, the regression function to be estimated is defined as

$$y = w.\phi(x) + b$$

where ϕ(x) represents the feature of inputs, w is the weight vector and b is a constant. w and b are obtained by minimising the constrained regularised cost.

Kernel: A kernel helps us find a hyperplane in the higher dimensional space.

Hyperplane: This is a separating line between two data classes in SVM, But in Support Vector Regression, this is the line that will be used to predict the continuous output.

SVR gives us the flexibility to define how much error is acceptable in our model and will find an appropriate line (or hyperplane in higher dimensions) to fit the data.
The objective function of SVR is to minimize the coefficients — more specifically, the l2-norm of the coefficient vector — not the squared error. The error term is instead handled in the constraints, where we set the absolute error less than or equal to a specified margin, called the maximum error, ϵ (epsilon). We can tune epsilon to gain the desired accuracy of our model.
We also need to account for the possibility of errors that are larger than ϵ. The slack variables is used for any value that falls outside of ϵ, we can denote its deviation from the margin as ξ.
We know that these deviations have the potential to exist, but we would still like to minimize them as much as possible. Thus, we can add these deviations to the objective function.
Thus, the objective function can be given by,

$$\min R_{SVM}(w, \xi, \xi^*) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi + \xi^*)$$

with the following constraint:

$$\begin{aligned} &d_i - w\varphi(x_i) + b_i \leqslant \varepsilon + \xi_i \\ &w\varphi(x_i) + b_i - d_i \leqslant \varepsilon + \xi_i^* \\ &\xi_i, \xi_i^* \geqslant 0, i = 1, \ldots, .., N \end{aligned}$$

where
ξ_i and ξ_i^* denote the positive slack variables, C is called as the error penalty factor and ε is called as the tube size of SVR.
The introduction of Lagrange multipliers and optimality constraints leads to a regression function of the form

$$f(x) = \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^*\right) K(x_i, x) + b$$

where α_i and α_i^* are the Lagrange multipliers and $K(x_i, x)$ is the kernel function.
The kernel function used in the project is the Gaussian given by

$$K(x_i, x) = \exp(-||x_i - x||^2 / 2\sigma^2)$$

The BHI values calculated in the previous step are supplied as inputs to the SVR and the outputs is taken as the life percentages calculated.
Then, the SVR model is trained using the input and output data and the hyperparameters are tuned to obtain the SVR model giving the best performance.

7. <u>Predicting RUL:</u>
Once the SOM and the SVR model has been trained, these can be used to predict the RUL on test sets.

First, the feature extraction is performed on the test set data. After getting the feature vectors, the corresponding BMU are identified using the trained SOM model and the corresponding MQE values are calculated.
Once the MQE values are obtained, these are converted to the BHI. The BHI value for the last observation so obtained is then input to the trained SVR model which gives the normalized life percentage (i.e. life percentage/100) as the output.
Once, we get the life percentage and we already have the minutes features which corresponds to the time at which this the observations were taken which were used to calculate the features. We can use them to get the minutes i.e. the time at which the bearing is about to fail as predicted by our model's output which corresponds to the 100% of life percentage. So to get the minutes, we will us the following

minutes at 100% = 100 * (minutes at the last observations / predicted life percentage for last observation)

Once, we get the minutes at 100% which corresponds to the minutes when the bearing is predicted to be failed, we can get the RUL by subtracting thee last observation's minutes from the minutes at 100% which corresponds to the time left for the bearing to reach 100%.

CONTRIBUTION:

As the input data from the PHM dataset consisted of time series of vibrations, frequency domain features required it to be converted to frequencies to calculate these features. For this, A mathematical concept called Fourier Tranforms was used to get the frequencies from a time series.
As the input data contained raw signals, and as there was a need to get the envelope signals so as to calculate the entropy of envelope signal, it was obtained by using the Hilbert.
As the MQE was needed to be calculated, it was calculated as the mean of the absolute difference between the BMU and Feature vector.
A different feature was created to calculate the Life Percentage which was required as the target to train the SVR. The feature 'minutes' which corresponds to the minutes for which the bearing is running and the instance when the observations were taken which were used for feature extraction was created for each feature vector. It was then converted to a percentage for each feature vector by dividing it with the last feature vector's minutes which corresponds to the instance when the bearing failed.

PSEUDO CODE:

The program mainly consists of 3 functions:

1. getDataset(folderpath)
   This function is used to load all the datasets corresponding to a particular bearing and then calculate the features for each such dataset and returns an array of feature vectors.
   The folderpath corresponds to the path of the bearing's folder which contains its datasets.

   The following steps are performed in this function:
   1) An empty features array is initialized first.
   2) The following steps are performed for each csv file in the folderpath mentioned in the parameters corresponding to a bearing. For each such csv file:
      i. First, the file is loaded into a dataframe.
      ii. Then, the feature 'minutes' is calculated using the attributes in the dataframe which is common for a csv file as is it generated every 10 seconds.
      iii. Then, the time domain features - rms, kurtosis, skewness, crestfactor, impulsefactor, shapefactor, marginfactor, peaktopeak are calculated using all the horizontal acceleration values present in the dataframe.
      iv. Using the fourier transform function, the frequencies are calculated using all the horizontal acceleration values present in the dataframe. Only the

first half frequencies are used for further calculations as the next half are just the negatives of these frequencies.

v. The envelope signal is obtained using the Hilbert function by applying on the horizontal acceleration values present in the dataframe which is then passed to the Fourier transform function to obtain the frequencies of the envelope signal.

vi. Then, the frequency domain features - meanfrequency, frequencycentre, rmsfrequency, entropyraw, entropyenvelope are calculated using the fourier transform frequencies obtained in step iv.

vii. All these features are then added into an array and then this array is appended to the features array.

3) The features array containing feature vectors is then normalized to have zero mean and standard deviation as 1 except the minutes features.
4) The features array thus obtained is returned.

2. runModel(bearing, testsetlast):

This function is used to create and train the SOM and then obtain MQE and the corresponding BHI and the Life Percentage and train the SVR using these obtained BHI and Life Percentage. Finally, the trained SOM is then used to get the MQE and then BHI and then this BHI is input to the SVR to obtained the Life Percentage which is then converted to the corresponding minutes which tells the time at which the failure may occur and the difference obtained from the calculated minute and the last observation's minutes feature gives the Remaining useful life.

This function takes in the condition number of the bearing and the corresponding last test set's dataset number of a bearing under given condition and gives two arrays of RUL of the test sets for the given condition corresponding to RULs predicted by two different models built on two different dataset.

The following steps are performed in this function:

1) For the given conditions, the corresponding first bearing's dataset features extracted are obtained by calling the function getDataset on it and are stored.
2) Similarly, getDataset is called for the second bearing's dataset and the features obtained are stored.
3) The features extracted and obtained for both the bearings are plotted as a function of time.
4) Then, two Self Organizing Maps, one for bearing 1 and bearing 2 is initialized and trained on the healthy features of the corresponding bearing's features set.
5) Using the trained SOM for the first bearing, the BMU's (Best Matching Unit) for each features of the first bearing's healthy features are obtained and the corresponding MQE (Minimum Quantization Error) is calculated.
6) Similarly, the above step is performed for the remaining feature set (apart from healthy features) of the bearing 1.

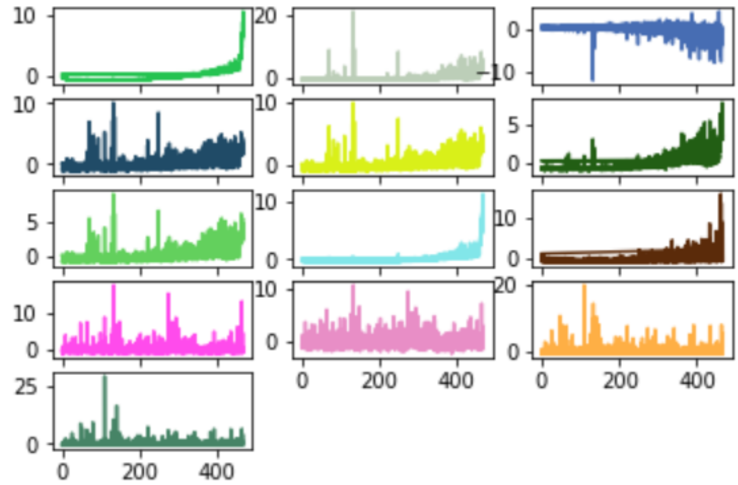
7) The steps 5 and 6 are similarly done for the bearing 2 feature set.
8) Using the MQE values obtained from the healthy features of the bearing 1, the corresponding c value needed to scale the BHI (Bearing Health Index) is calculated.
9) Similarly, the step 8 is done for bearing 2.
10) Then, the BHI for the bearing 1 MQE is calculated and similarly done for bearing 2.
11) The life percentages are calculated for both the bearing 1 and bearing 2 using the feature 'minutes'.
12) Then, two separate Support Vector Regression Models are initialized and trained using input as the BHI and output as Life percentage for the corresponding model built for the corresponding bearing.
13) The SOM and SVR models are trained for each of the datasets.
14) Now, the test sets corresponding to the given condition are now taken. For each such test set, the following steps are performed:
   i. The function getDataset is called for the test set and the features returned are stored.
   ii. The BMU for the healthy feature vectors as well as for the remaining feature vectors are obtained by using trained SOM model of bearing 1.
   iii. The corresponding BMU obtained for both healthy and remaining feature vectors and the corresponding feature vectors are used to calculate the corresponding MQE values.
   iv. The MQE values of the healthy feature vectors are used to calculate the value of c needed to scale the BHI.
   v. Then, the BHI values are calculated using the MQE values of the remaining feature vectors.
   vi. The obtained BHI value of the last observation is then input to the bearing 1's SVR and the corresponding output obtained is the Life Percentage.
   vii. The obtained life percentage is converted to minutes at 100% life percentage using the 'minutes' features of the last observation.
   viii. The 'minutes' feature of the last observation is subtracted from the obtained 'minutes' feature at the 100% life percentage which gives the RUL of the test bearing.
   ix. This RUL obtained is added to the RUL array corresponding to the bearing 1 model.
   x. The steps i to ix are performed similarly with bearing 2's model.
   xi. Both the RUL arrays corresponding to the bearing 1's and bearing 2's model are returned.

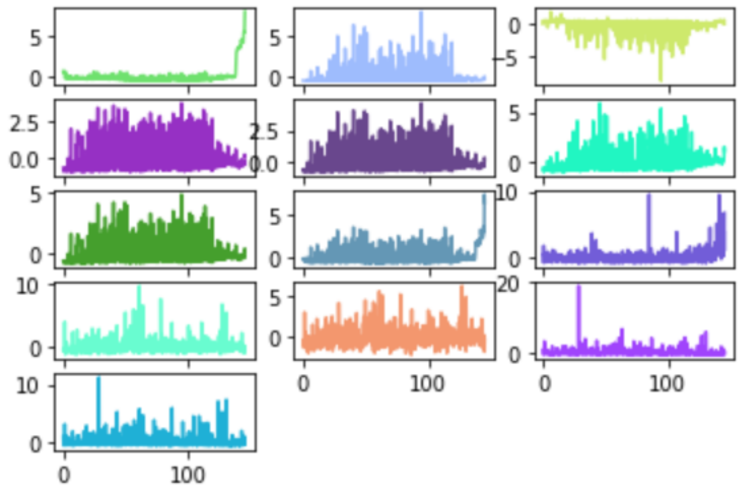
3. main():
   This function is the main function which calls the runModel() function and gets the RUL's to calculate the error of the overall models.

   Steps performed in this function:
   1) The runModel() function is called for each bearing conditions.

2) The obtained RUL arrays are concatenated into one array each for both the models.
3) The error is calculated between the predicted RULs and the actual RUL for both the model and the corresponding score is printed.

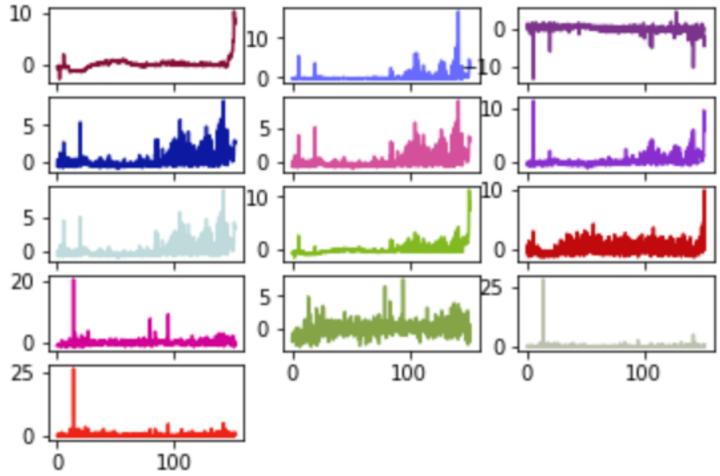
OUTPUT:

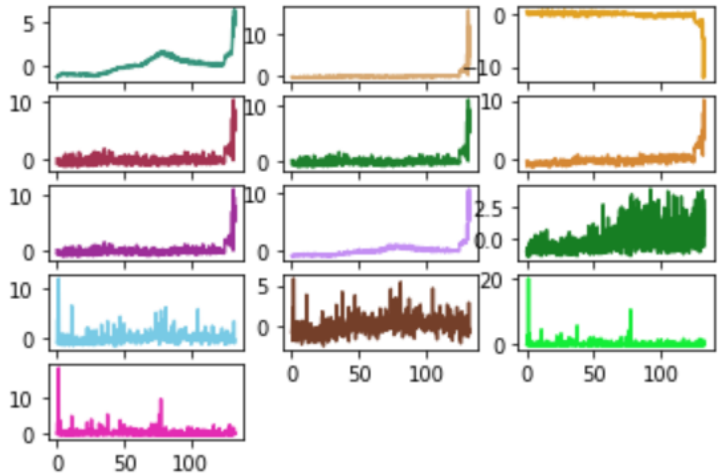




```
RUL for bearing1_3 predicted by 1st model: 2086.374718696843
RUL for bearing1_3 predicted 2nd model: 2536.566891466219
RUL for bearing1_4 predicted by 1st model: 1630.82043951897
RUL for bearing1_4 predicted 2nd model: 1300.2817178019513
RUL for bearing1_5 predicted by 1st model: 4952.012109060233
RUL for bearing1_5 predicted 2nd model: 2571.0956434908485
RUL for bearing1_6 predicted by 1st model: 2852.394716462576
RUL for bearing1_6 predicted 2nd model: 3575.494898116075
RUL for bearing1_7 predicted by 1st model: 4102.628395371307
RUL for bearing1_7 predicted 2nd model: 1957.3170227110438
```

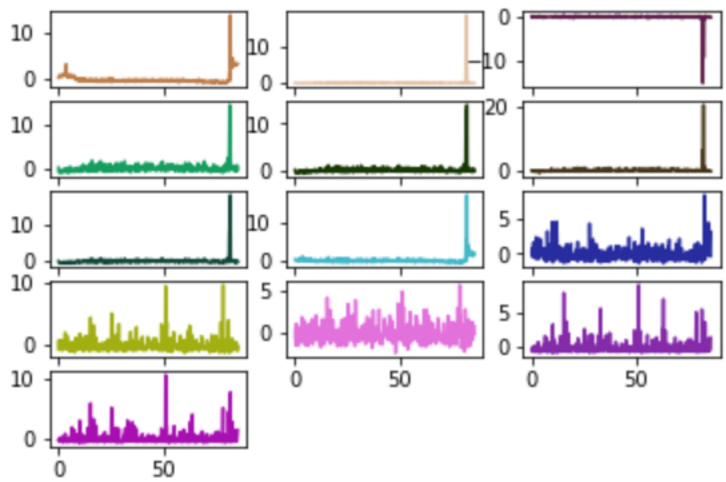
RUL for bearing1_7 predicted 2nd model: 1937.5176227116438

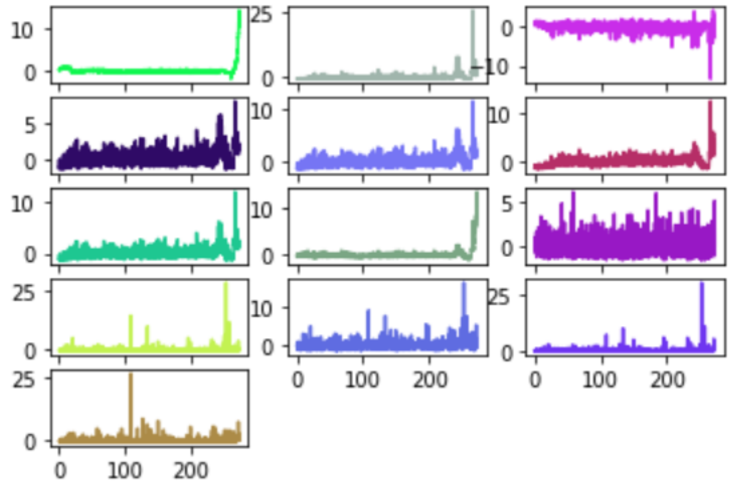




RUL for bearing2_3 predicted by 1st model: 6892.748543260713

```
RUL for bearing2_3 predicted by 1st model: 6892.748543260713
RUL for bearing2_3 predicted 2nd model: 2173.643872059183
RUL for bearing2_4 predicted by 1st model: 909.889755608069
RUL for bearing2_4 predicted 2nd model: 1085.7657694043994
RUL for bearing2_5 predicted by 1st model: 10136.951258021774
RUL for bearing2_5 predicted 2nd model: 2968.0315028949835
RUL for bearing2_6 predicted by 1st model: 1048.0168480977181
RUL for bearing2_6 predicted 2nd model: 3305.9938929838463
RUL for bearing2_7 predicted by 1st model: 1112.9566098521996
RUL for bearing2_7 predicted 2nd model: 1575.9553553908308
```





```
RUL for bearing3_3 predicted by 1st model: 839.1746094431966
RUL for bearing3_3 predicted 2nd model: 2272.3324351091796
Score of model built using training set 1: 0.23707479139445045
Score of model built using training set 2: 0.14974695345393452
```

DISCUSSION:

The Fourier transforms is an interesting way to get the frequency components of a time series signal. The frequency domain features itself are quite an intuitive way to represent the signals and the features it holds. The frequency domain features capture various things a time domain feature is unable to capture as we can see from the feature plots itself.

The Self Organizing Map itself is an interesting model which can be used to convert a high-dimensional feature space to a 2-dimesional space. Various 2-D topologies like rectangle, square, hexagon can be mapped upon using a Self Organizing Map which can ultimately lead us to learn new lower dimensional features.
The concept of training a SOM is itself new and interesting thing to know.

The Support Vector Regression hyperparameters when tuned showed different results for different combinations of hyperparameters. When the hyperparameters are tuned such that the model gives the highest R2 score on the training set, the same model gives much much worse predictions as I think it must be the case of overfitting. For a model tuned such that the R2 score is around 85% for the bearings corresponding to the conditions 1 gives pretty good results on the test set corresponding to same conditions but fail to give nice results on the bearings under conditions 2 and 3 when trained using the same hyperparameters.

The use of Life Percentage to calculate the Remaining Useful Life of the test bearings is very different and much reliable as it does not involve the need of setting a threshold.

Instead, if we have used the RUL directly as the final observations time minus the time at the current observation and then trained the model, then we must have to set a threshold for indicating failure. Once the predicted values cross the predetermined failure threshold, the bearing is set to be failed. The time duration between the current measurement point and the failure point gives the RUL of the bearing. The failure threshold that we need to set requires expertise in bearing conditions and their health.
So, the use of Life Percentage is much better and does not require setting any threshold.

The method used for calculation of Bearing Health Index at a particular point is itself quite meaningful. The formula for the same is

$$\mathrm{BHI}(m) = \exp\left(\frac{\sum_{l=1}^{m} \mathrm{MQE}(l)}{c \times m}\right), \quad m = 1, 2, \ldots\ldots\ldots. N$$

Here, the BHI at a particular instance is actually dependent on all the MQE values till that instance and this pretty clearly tells that the Bearing Health Index at a particular instance reflects all the MQE’s which has impact on the bearing’s health.

ACKNOWLEDGEMENT

We take this opportunity to thank the department of Computer Engineering and IT, VJTI for giving us an opportunity to work on this project. We would also like to thank our mentor Dr. S. S. Udmale Sir for his invaluable guidance and support throughout the project work. We would also like to congratulate our team members for their utmost efforts in making this project a success and implementing it to a deliverable level.

REFERENCES

1. https://journals.sagepub.com/doi/10.1177/0954406217700180
2. https://github.com/wkzs111/phm-ieee-2012-data-challenge-dataset
3. https://github.com/sevamoo/SOMPY
4. https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.fft.html
5. https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.hilbert.html