

Remaining Useful Life Prediction of Cutting Tools Based on Support Vector Regression

*A Report
Submitted by*

TEJAS BHATAMBAREKAR (181080016)

KAPIL BHISE (181080017)

PRATHMESH CHANDWADE (181080018)

GAURAV CHAMBHARE (181080025)

TY BTECH IT

IN

INFORMATION TECHNOLOGY

At

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE

Department of CE & IT

MUMBAI

MARCH 2021

iii. Problem Statement

Remaining useful life (RUL) prediction of cutting tools is critical to effective condition based maintenance for reducing downtime, ensuring quality and avoiding accidents. In this paper, a RUL prognostic method based on support vector regression (SVR) is proposed for predicting cutting tool's life. The proposed method consists of two main phases: an off-line phase and an on-line phase. In the first phase, the signal features are extracted from raw data, and then the SVR models with considering different length of signals at past times are established to reflect the relationship between monitoring data and tool life. In the second phase, the constructed models are used to predict the cutting tool's RUL, and the best signal length for accurate prediction result is obtained. The proposed method is applied on experimental data taken from a computer numerical control (CNC) rotor slot machine in a factory. The result shows the validity and practicability of this method.

iv. Dataset

IEEE PHM 2012 Challenge Dataset

v. Motivation

For our project we wished to implement the knowledge that we have learned from class as well as learning various new techniques and approaches we can take in the process of creating a fine working model.

While searching appropriate topics of interest from various research papers, we came across our current topic. We became interested by reading the content of said research paper and decided to select it as one of our topics for recommendations

For Remaining Useful Life prediction, various methods have been proposed in literature. Support Vector Regression is one of these methods. Support Vector Regression is a regression algorithm that supports both linear and non-linear regressions.

In our project, we are estimating RUL of ball bearings based on IEEE PHM 2012. We have learnt Support Vector Regression during Machine Learning course, and this motivated us to further study about it. We referred a research paper in SVR applied for RUL prediction of cutter. This helped us in determining our approach of using SVR in RUL prediction of given dataset.

vi. Methodology

We have used a two-step approach for RUL estimation based on the research paper referred to.

The first step involves extraction of features from vibration data from the dataset. The features extracted from the dataset are as follows:

1. Root Mean Square (RMS)
2. Amplitude

We normalized the extracted features using the min-max method for better applicability in the model.

In the second step, we developed a SVR model based on the normalized values of features. We trained the model using a learning dataset. The developed model was then applied to a test dataset for RUL estimation.

We have used the SVR model with kernels. The common kernel functions used for SVR approaches are linear, polynomial and RBF. Literature suggests that RBF performs better than other kernels in most of the cases. Thus, we have used the RBF kernel for the SVR model.

We have used following values for parameters of SVR model using RBF kernel:

$$\varepsilon = 0.1, C = 4, \gamma = 0.5$$

vii. Contribution

We contributed in the project in the following manner:

1. Feature Extraction:

We decided on the features to be used for the model.

We researched for a way to extract feature from given dataset

We extracted the features from the dataset.

We defined functions for feature extraction and extracted features for learning and testing dataset.

2. Model Selection:

In the SVR model, various kernels can be used. Literature suggests using RBF kernel. We verified it by running models based on linear, polynomial and RBF kernels and concluded that RBF gives best fitting.

We ran the SVR model using the RBF kernel.

3. Calculating score:

We trained the model using a learning set and used it on the training set and calculated accuracy score for the outcome.

viii. Pseudo Code

```
#imported all the libraries required in the code
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR

#define a normalise function according to formula given in research paper
def norm(list):
    mini = min(list)
    maxi = max(list)
    for i in range(len(list)):
        list[i] = 0.1 + (0.8*(list[i]-mini))/(maxi - mini)
def normalise(list_1, list_2, list_3):
    norm(list_1)
    norm(list_2)
    norm(list_3)

def calculate(rul, rms, amplitude, sizes, learning_bearing, number, something):
    print(something)
    for i in range(len(learning_bearing)):
        if((something == "Test_set" and i>1) or something=="Learning_set"):

k=pd.read_csv("https://raw.githubusercontent.com/wkzs111/phm-ieee-2012-data-challenge-dataset/master
/" + something + "/Bearing" + "{:01d}".format(number) + "_" + "{:01d}".format(i+1) +
"/acc_0"+"{:04d}".format(learning_bearing[i]-1)+".csv", header=None)

        finalTime = (k.iloc[-1][0]*3600) + (k.iloc[-1][1]*60) + k.iloc[-1][2]

        for j in range(1, learning_bearing[i]):

f=pd.read_csv("https://raw.githubusercontent.com/wkzs111/phm-ieee-2012-data-challenge-dataset/master
/" + something + "/Bearing" + "{:01d}".format(number) + "_" + "{:01d}".format(i+1) +
"/acc_0"+"{:04d}".format(j)+".csv", names = ["hr", "min", "sec", "msec", "horacc", "veracc"])

        #rul
        currentTime = (f.iloc[-1][0]*3600) + (f.iloc[-1][1]*60) + f.iloc[-1][2]
        rul.append(finalTime - currentTime)

        # #RMS
        currAcceleration = f.iloc[:, [4]]
```

```

MSE = np.square(currAcceleration).mean()
RMSE = math.sqrt(MSE)
rms.append(RMSE)

#Amplitude
AMPL = math.sqrt(2)*(RMSE)
amplitude.append(AMPL)

#size
sizes.append(k.size)

new_rms = []
new_amplitude = []
new_sizes = []
new_rul = []

# learning_bearing_1 = [2804, 872]
learning_bearing_1 = [2804]
learning_bearing_2 = [912, 798]
learning_bearing_3 = [516, 1638]
# testing_bearing_1 = [0, 0, 1803, 1140, 2303, 2303, 1503]
# testing_bearing_2 = [0, 0, 572, 172, 1203, 613, 2003, 573] #from 6 to 7
testing_bearing_3 = [0, 0, 353]

calculate(new_rul, new_rms, new_amplitude, new_sizes, learning_bearing_1, 1, "Learning_set")

#call normalise on all the lists
normalise(new_rul, new_amplitude, new_rms)

#create dataframe using obtained normalised lists
df = pd.DataFrame(list(zip(new_rms, new_amplitude, new_rul)), columns=['RMS', 'Amplitude', 'RUL'])

testing_new_rms = []
testing_new_amplitude = []
testing_new_sizes = []
testing_new_rul = []
# print(len(testing_bearing_3))
# calculate(testing_new_rul, testing_new_rms, testing_new_amplitude, testing_new_sizes,
testing_bearing_3, 3, "Test_set")

#call normalise on all the lists
normalise(testing_new_rul, testing_new_amplitude, testing_new_rms)

#create dataframe using obtained normalised lists

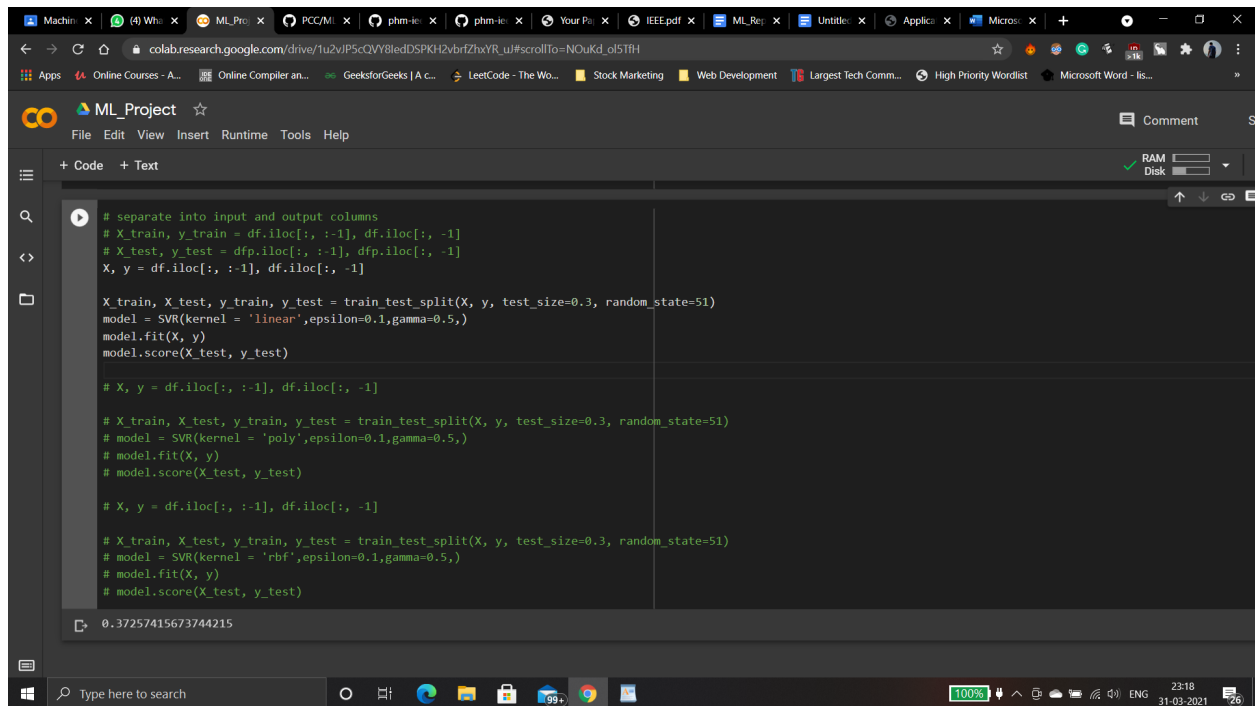
```

```
dfp = pd.DataFrame(list(zip(testing_new_rms, testing_new_amplitude, testing_new_rul)), columns
=['RMS', 'Amplitude', 'RUL'])
```

```
# separate into input and output columns
# X_train, y_train = df.iloc[:, :-1], df.iloc[:, -1]
# X_test, y_test = dfp.iloc[:, :-1], dfp.iloc[:, -1]
X, y = df.iloc[:, :-1], df.iloc[:, -1]
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
model = SVR(kernel = 'rbf',epsilon=0.1,gamma=0.5,)
model.fit(X, y)
model.score(X_test, y_test)
```

Output



```
# separate into input and output columns
# X_train, y_train = df.iloc[:, :-1], df.iloc[:, -1]
# X_test, y_test = dfp.iloc[:, :-1], dfp.iloc[:, -1]
X, y = df.iloc[:, :-1], df.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
model = SVR(kernel = 'linear',epsilon=0.1,gamma=0.5,)
model.fit(X, y)
model.score(X_test, y_test)

# X, y = df.iloc[:, :-1], df.iloc[:, -1]

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'poly',epsilon=0.1,gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)

# X, y = df.iloc[:, :-1], df.iloc[:, -1]

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'rbf',epsilon=0.1,gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)
```

0.37257415673744215

Machine x (4) Whi x ML_Pro x PCC/ML x phm-ic x phm-ic x Your Pa x IEEEpdf x ML_Re x Untitle x Applic x Micro x +

colab.research.google.com/drive/1u2vIP5cQVY8ledDSPKH2vbrfZhwYR_uI#scrollTo=NOuKd_o15TH

Apps Online Courses - A... Online Compiler an... GeeksforGeeks | A c... LeetCode - The Wo... Stock Marketing Web Development Largest Tech Comm... High Priority Wordlist Microsoft Word - Is...

ML_Project ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM 100% Disk

```
# separate into input and output columns
# X_train, y_train = df.iloc[:, :-1], df.iloc[:, -1]
# X_test, y_test = dfp.iloc[:, :-1], dfp.iloc[:, -1]
X, y = df.iloc[:, :-1], df.iloc[:, -1]

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'linear', epsilon=0.1, gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
model = SVR(kernel = 'poly', epsilon=0.1, gamma=0.5,)
model.fit(X, y)
model.score(X_test, y_test)

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'rbf', epsilon=0.1, gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)
```

0.08731048839387867

Type here to search

100% 23:19 31-03-2021

Machine x (4) Whi x ML_Pro x PCC/ML x phm-ic x phm-ic x Your Pa x IEEEpdf x ML_Re x Untitle x Applic x Micro x +

colab.research.google.com/drive/1u2vIP5cQVY8ledDSPKH2vbrfZhwYR_uI#scrollTo=NOuKd_o15TH

Apps Online Courses - A... Online Compiler an... GeeksforGeeks | A c... LeetCode - The Wo... Stock Marketing Web Development Largest Tech Comm... High Priority Wordlist Microsoft Word - Is...

ML_Project ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM 100% Disk

```
# separate into input and output columns
# X_train, y_train = df.iloc[:, :-1], df.iloc[:, -1]
# X_test, y_test = dfp.iloc[:, :-1], dfp.iloc[:, -1]
X, y = df.iloc[:, :-1], df.iloc[:, -1]

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'linear', epsilon=0.1, gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
# model = SVR(kernel = 'poly', epsilon=0.1, gamma=0.5,)
# model.fit(X, y)
# model.score(X_test, y_test)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=51)
model = SVR(kernel = 'rbf', epsilon=0.1, gamma=0.5,)
model.fit(X, y)
model.score(X_test, y_test)
```

0.7531883107908562

Type here to search

100% 23:19 31-03-2021

ix. Discussion

First of all, we researched for a way to extract given features from the given dataset. We checked all of the referred papers as well but couldn't find any legible formulae. Then we got the formula for RMS calculation from the vibrational energy which was one of the given parameters. We then extracted RMS and amplitude and created our dataset. Then using these features we created a svr model using linear, polynomial and RBF kernels. We observed that the score prediction of RBFkernel was at par from linear and polynomial. So we decided to continue with the RBF kernel.

Based on our assumption, we decided to pick a single training set and split it into testing and training sets. We calculated the accuracy of 75.31% for the same.

We faced a difficulty because our research paper had assumed that RUL of the training data was already provided which was not true in our case so we calculated the RUL of given bearing in a case as the final recording time - current case recording time. Here, we assume that after the final recording time, bearing is worn out.

From this project, we have gained thorough knowledge of how research papers are published and contents and format of the same. We have also gained insights in the research field and read various research papers from different authors.