Larry Marshall
ACCESS ID: lmarshall

Meeg 591V
December 8th, 2022

Final Report

Adaptive Health Management for Rotary Systems

Abstract

The objective of this project is to evaluate a variety of supervised learning algorithms for classifying multiple fault scenarios for manufacturing systems and develop a proof-of-concept prototype for in-situ fault detection and classification. The manufacturing industry is moving from scheduled maintenance to condition-based maintenance due to the cost savings and flexibility offered by the latter. To achieve condition-based maintenance operators need fast and accurate methods of component fault detection and classification. In industry today there are numerous ways in which companies classify machinery faults, but usually, it is done as a binary classification problem i.e., is it healthy or not? Doing this binary classification is a great technique but it leaves lots of information on the table such as knowing which part has degraded, how it is degraded, and at which location, not allowing companies to make more informed decisions. With more information, they could be fault-adaptive, changing processes or sequences to maintain product throughput. This will require fault classification at the system level rather than only specific component monitoring. In literature researchers mainly focus on the classification of single ball-bearing faults, but they do not investigate multiple faulted bearings, warped shafts, or combinations of these scenarios all of which are major and very plausible fault scenarios. Hence, this project intends to fill these research gaps. This is done by creating datasets of single and multiple fault scenarios, creating machine-learning models that can accuratel y classify fault scenarios, and developing a prototype package for in-situ detection and classification using the best model.

Introduction

In the case of a bearing, it will be rated for a certain number of cycles, but this number is usually a computational or experimental mean value. Thus, the part may last significantly less or significantly more than this reported value. Using the technique of scheduled maintenance this part will be replaced at a pre-determined time interval regardless of the actual condition of the bearing. Since this technique does not consider the condition of the bearing two major issues can arise. Firstly, the part may catastrophically fail before its rated lifespan resulting in immense downtime, a great loss of revenue or this failure may damage other adjacent parts. The second scenario is a part that outlasts the rated life of the part. With scheduled maintenance, this part will be replaced and discarded though it is perfectly functional which is not an optimal use of resources. To address these issues condition-based maintenance can be implemented where we continuously monitor the health of the system and all its parts. Thus, only conducting maintenance when the part is going to fail. This method of maintenance also allows investors ample time to prepare for the maintenance of equipment resulting in dramatically reduced downtimes. Condition-based maintenance (CBM) is a maintenance program that recommends maintenance decisions based on the information collected through condition monitoring [1]. Though data can be readily collected this data must be pre-processed and used to make decisions. This is where machine learning techniques can be implemented. Since we are generating these datasets, therefore, knowing their labels hence, supervised learning can be used. Supervised learning is a subset of machine learning that involves predetermined output attributes besides the use of known input attributes. These algorithms attempt to predict and classify the predetermined attribute [2].

Many studies have been conducted on the use of vibration data for fault classification in mechanical rotary systems. For example, a study by Shuang and Meng (2007) where used Statistical theory and Principal Component Analysis for feature extraction on vibrating signals from rolling bearings. Then using these features to classify rolling bearing faults using Support Vector Machines achieving an accuracy of 97.1%. However, this study only investigates 3 types of faulted bearings [3]. In another study by Gunerkar and Jalan (2019), a K-nearest neighbors model was trained using vibrational and acoustic data. To extract features, they also used statistical theory and Principal Component Analysis. In their study, they achieved 100% accuracy but only classified 4 faulted bearing scenarios [4]. Lastly, a study by Chattopadhyay and Konar (2014) in their study they developed a scheme for multi-class fault detection of an induction motor. This was done using wavelet transform as the feature extraction technique and classified using the Support Vector Machine classifier. In this study they classified Broken rotor bar, bowed rotor, rotor unbalance, faulty bearing, voltage unbalance and stator fault achieving an accuracy of 97.14%. However, in this study, they did not consider multiple faults occurring simultaneously [5].

Final Report

Methodology

Experimental Setup

The first step of this project was to create datasets and test the classifiability of those datasets. Experiments were conducted using the Spectra Quest Machinery Fault simulator (fig.1) this device includes 4 faulted bearings: Inner raceway fault, Outer raceway fault, Combination fault, and Ball faulted bearings and two warped shafts: Middle bent and coupling end bent shafts. Accounting for single and dual fault scenarios this provides 38 faults and a no-fault condition. To collect data to train models National Instruments Compact DAQ system, piezoelectric accelerometers with an accuracy of 102mg/V, and DAQmx software were used. Experiments were conducted at three-frequency conditions 25 Hz, 50 Hz, and 75 Hz. Data were collected at a sampling rate of 6400 Hz for 10s. 25 data sets were collected for each test condition.

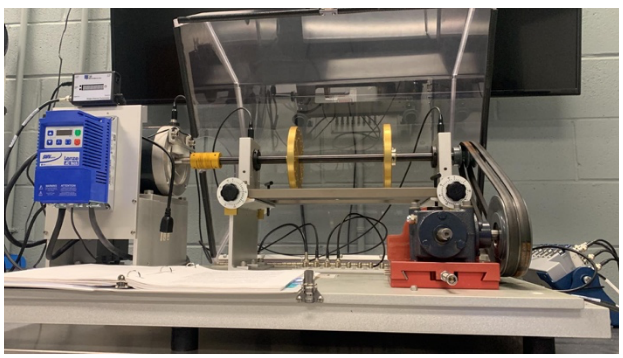


Figure 1 Experimental setup

Feature extraction

The current test setup of 8 accelerometers at a sampling rate of 6400 Hz for ten seconds provides a matrix of 512,000 elements. This large dataset should not be directly inserted into a machine-learning model because this runs the risk of overfitting, very slow training times, and low generalizability of the model, therefore, some characteristic features of the data must be extricated to greatly reduce those issues. Beginning in the time domain, common feature extraction methods usually involve transforming data into the frequency domain using the Fast Fourier Transform, Wavelet analysis, or a spectrogram. Though these methods are powerful tools R.S. Gunerkar and A.K. Jalan [3] displayed the accuracy of using statistical analysis such as median, minimum, maximum, skewness, kurtosis, RMS, and standard deviation for rolling element faults and this will be the feature extraction method explored in this report.

Algorithms

Due to all data being collected in the lab with its corresponding fault scenarios known, supervised machine learning algorithms will be most useful for this application. Supervised learning is a subset of machine learning that is defined by its use of labeled datasets to train algorithms to classify data or predict outcomes accurately [6]. Five supervised learning algorithms will be explored in this report K-nearest neighbors' algorithm (KNN), the Gaussian Naïve Bayes classifier algorithm (GNB), the Support Vector Machine algorithm (SVM), the Decision tree algorithm (DT), and the Random Forest algorithm (RF). The best algorithm will then be used for deployment. The first algorithm explored was the KNN algorithm. KNN algorithm is a non-parametric supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point [5]. GNB is a probabilistic classifier dependent on applying the Bayes theorem. GNB considers each attribute variable as an independent variable. The major advantage of GNB is that it requires little measure of training data which is vital for characterization and necessary for classification [7]. Support Vector Machines work by separating the different classes of data by a hyperplane corresponding to a decision function and the optimal hyperplane is the one with the maximal margin of separation between the classes [8]. Decision trees are a non-parametric supervised learning method. They make no assumptions about the underlying data distribution and are trained on labeled data to correctly classify previously unseen data [9]. Lastly, the Random Forest classifier algorithm. A random forest (RF) classifier is an ensemble classifier that produces multiple decision trees, classifying using a randomly selected subset of training samples and variables [10].

Deployment

In the wake of companies beginning to make the transition from scheduled-based maintenance to condition-based maintenance companies will now require inexpensive methods and devices that can be integrated into existing internet of things systems that can make informed decisions. In the laboratory setup (fig. 1 and 2) many expensive apparatuses and proprietary software were used to successfully conduct this experiment. Initially using this setup was useful in validating the method used ensuring that the only component for a failed experiment would be the machine learning algorithm selected and not the fidelity and accuracy of accelerometers and the data acquisition device. Once models were trained using collected data the best model was selected using accuracy as the metric, computational cost was not considered due to all models having low training times. After the best model was selected, it was then optimized to determine the least number of accelerometers needed to accurately classify fault scenarios.

For deployment, a basic store-bought three-axis accelerometer, and a raspberry pi 3 were used. Since the accelerometer used for initial data collection was different from the new accelerometer this would cause slightly different vibration patterns resulting in misclassified faults, moreover, a proportional constant would be needed to convert the raw data which would be difficult to do accurately. To circumvent this issue a much smaller dataset was collected using the new accelerometer and a new model was then trained. Figures 4 and 5 depict the accelerometer and fixture created to ensure data collected is consistent. Once the model was trained it was then integrated with a created graphical user interface (fig. 6) that can read, store, classify and display the fault scenario.

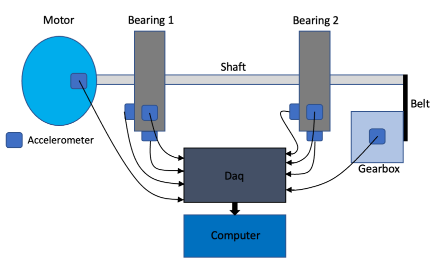


Figure 2 Laboratory experimental setup

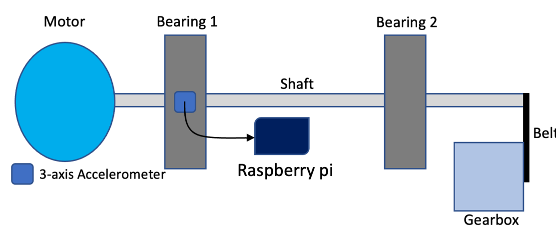


Figure 3 Deployed model setup

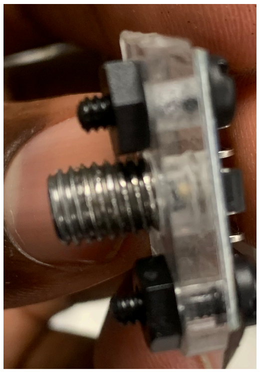


Figure 4 Accelerometer and created fixture

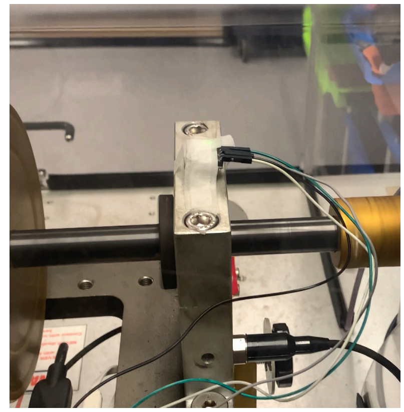


Figure 5 Accelerometer attached to bearing housing

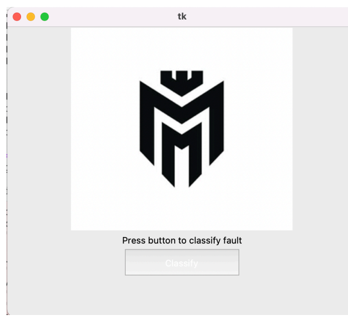


Figure 6 GUI for fault classification

Results and Discussion

For this project, five supervised machine-learning models were investigated and evaluated using the metric of accuracy for this initial algorithm selection process, all data used was collected at 25 Hz. Initially following the method established by Gunerkar, R. S., & Jalan, A. K. [3] the KNN algorithm was first used. This produced an accuracy of 63.077% (fig.7). It is believed that this low accuracy is due to the high dimensionality of the data. The KNN classifier assumes that similar data points share similar labels. Unfortunately, in high dimensional spaces, points that are drawn from a probability distribution, tend to never be close together [13]. To improve this a dimensionality reduction algorithm such as Principal component analysis can be used. The second algorithm used was the Support Vector Machine which produced an accuracy of 76.410% (fig 8). It is believed that the SVM algorithm suffers from the same issue as the KNN algorithm due to SVM separating classes using hyperplanes thus as the dimensions and classes increase the accuracy can decrease. The next algorithm tested was the Gaussian Naïve Bayes algorithm which result ed in an accuracy of 94.359% (fig 9). It is believed that since the GNB algorithm is a probabilistic algorithm using statistics and the normal distribution to classify since all features are created using statistical techniques it improves the accuracy of the algorithm. The fourth algorithm investigated was the Decision Tree algorithm resulting in an accuracy of 96.923% (fig 10). The last algorithm tested was the Random Forest algorithm which resulted in an accuracy of 98.462% (fig. 11). From these tests, it was found that the Random Forest algorithm was the most accurate classifying algorithm.

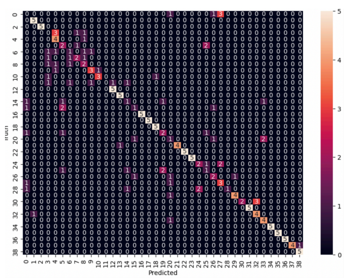


Figure 7 KNN model confusion matrix

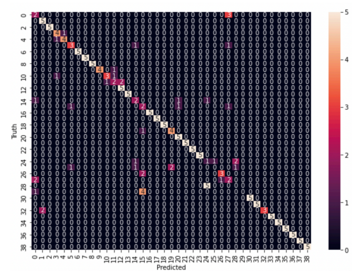


Figure 8 SVC model confusion matrix

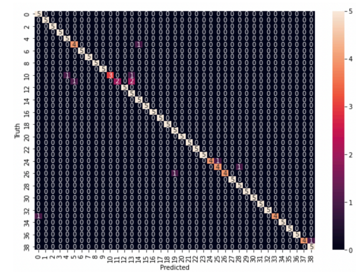


Figure 9 GNB model confusion matrix

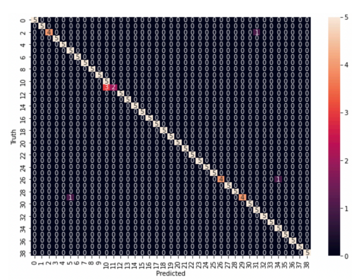


Figure 10 DT model confusion matrix

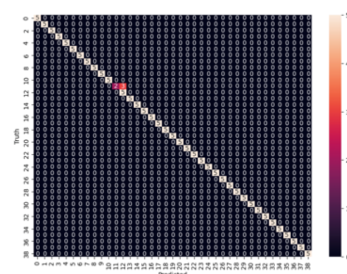


Figure 11 RF model confusion matrix at 25 Hz

Now that the most accurate model has been selected it was then trained and tested at various frequencies to test the generalizability of the model. Training and testing at 50 Hz resulted in an accuracy of 100% (fig. 11). Training and testing at 75 Hz also resulted in an accuracy of 100% (fig. 12). This proves the generalizability of the model.

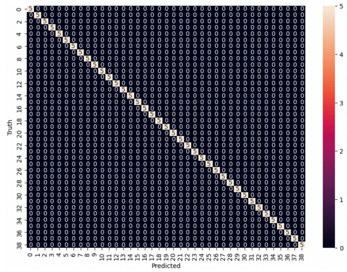


Figure 12 RF model confusion matrix at 50 Hz

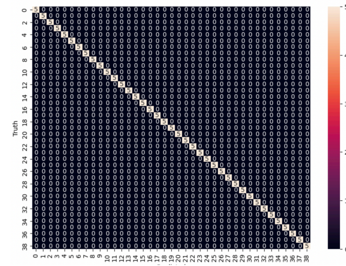


Figure 13 RF model confusion matrix at 75 Hz

To further test the adaptability of the model it was trained on data at 25Hz and tested o n data at 50Hz (fig. 12). This resulted in an accuracy of 2.564%. This result raises issues in the a pplication of this method since most machinery operates at various rpm.

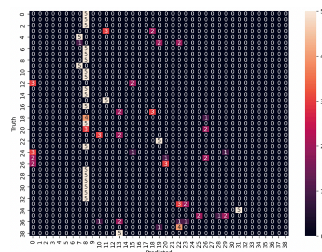
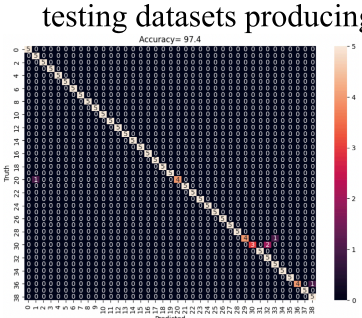
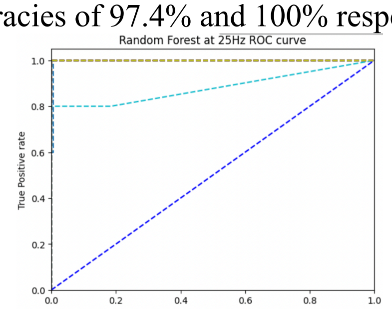


Figure 14 RF model trained on 25Hz and tested on 50 Hz

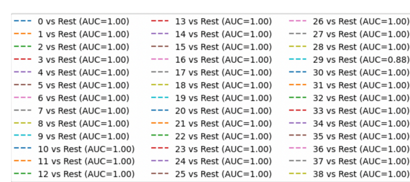
To combat this issue two additional models were created and tested the first model was a random Forest Classifier trained on 50% 25Hz and 50% 50 Hz. It was then tested on 25Hz and 50Hz testing datasets producing accuracies of 97.4% and 100% respectively.



(a) Confusion Matrix

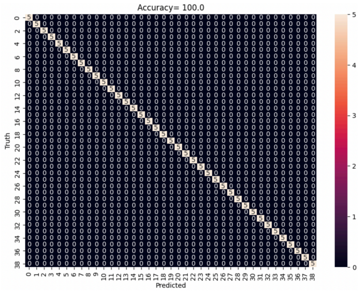


(b) ROC Curve

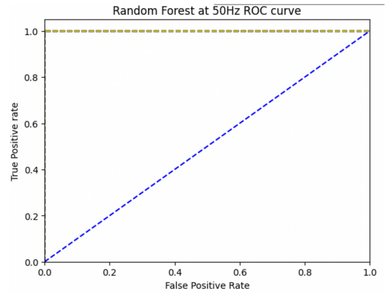


(c) Area Under Curve

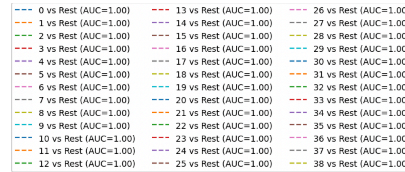
Figure 15 Results from model tested on 25 Hz



(a) Confusion Matrix



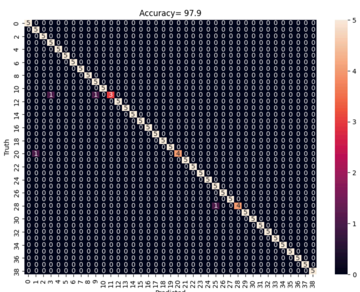
(b) ROC Curve



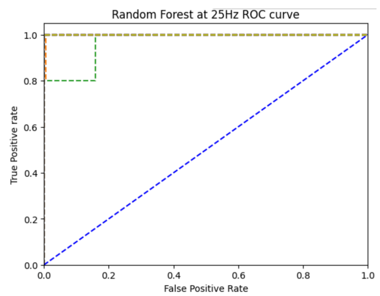
(c) Area Under Curve

Figure 16 Results from model tested on 50 Hz

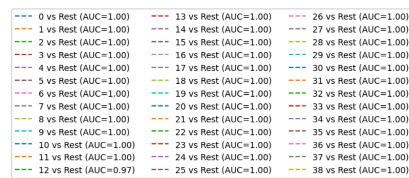
The second model was trained at 33.33% 25Hz, 33.33% 50Hz, and 33.33% 75Hz the model was then tested at those frequencies. This resulted in an accuracy of 97.9% 99.5%, and 99.5% respectively.



(a) Confusion Matrix

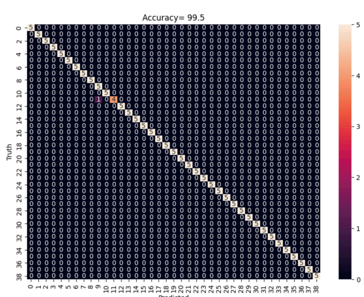


(b) ROC Curve

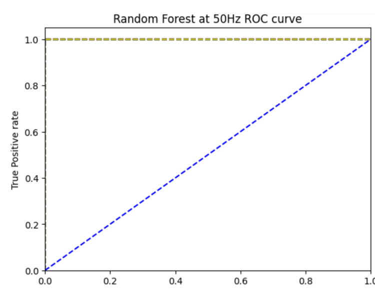


(c) Area Under Curve

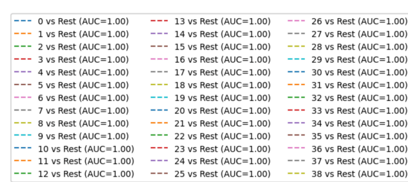
Figure 17 Results from model tested on 25 Hz



(a) Confusion Matrix

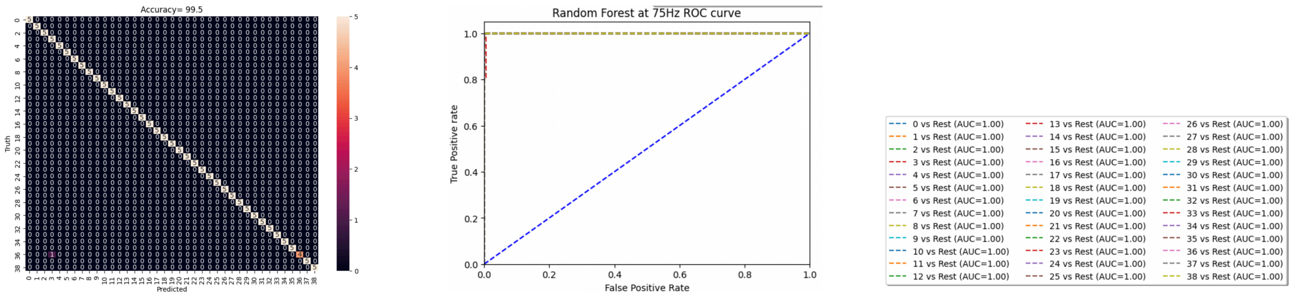


(b) ROC Curve



(c) Area Under Curve

Figure 18 Results from model tested on 50 Hz



(a) Confusion Matrix *(b) ROC Curve* *(c) Area Under Curve*

Figure 19 Results from model tested on 75 Hz

The next task would be to optimize the model. This is done by reducing the number of accelerometers used from eight to a minimum of three testing along the way and checking for any reduction in accuracy. A minimum of three accelerometers was selected due to store-bought accelerometers already measuring vibrations in 3 axes thus in this setup three accelerometers are the minimum and can be increased by a multiple of three. From figures 11, 20, and 21 it can be seen that reducing the number of accelerometers from 8 to 6 to 3 does not negatively impact the accuracy of the model all resulting in an accuracy of 98.462%.

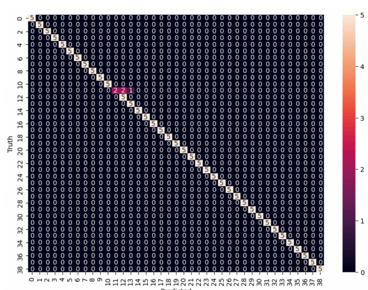


Figure 20 RF model confusion matrix using 6 accelerometers

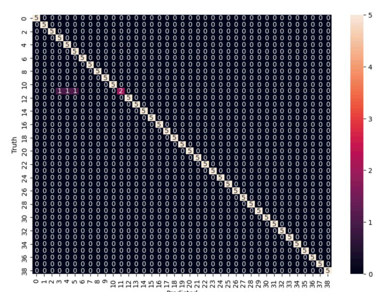


Figure 21 RF model confusion matrix using 3 accelerometers

The last step in this project was to reconduct this experiment using the deployed model setup (fig 3). As a proof-of-concept data was collected for 4 fault scenarios and a nominal scenario using the raspberry pi 3 and 3-axis accelerometer. A Random Forest model was then trained to utilize this data which resulted in 100% accuracy (fig. 22). This model was then saved and integrated into the created graphical user interface and used to classify fault scenarios in real-time (fig. 23).

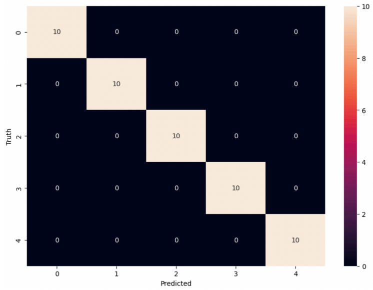


Figure 22 RF model confusion matrix using data collected from raspberry pi

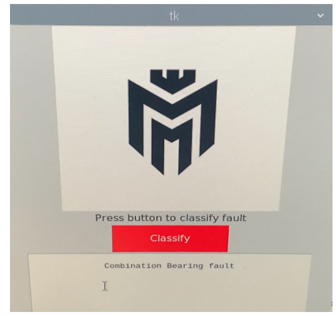


Figure 23 Model correctly classifying fault scenario

In this project, the metric used was accuracy which meant an incorrect classification carries the same weight regardless of what the incorrect classification was which would not be the case in reality. For example, if there was a bearing 1 ball bearing fault and the model predicted that it was a bearing 2 inner raceway fault or a shaft fault, these incorrect predictions carry much greater severity due to the incorrect location prediction versus if the prediction was bearing 1 outer bearing fault. Hence another metric must be found or developed to incorporate these factors.

Conclusion

In conclusion, this project successfully evaluated a variety of supervised learning algorithms for classifying multiple fault scenarios for manufacturing systems and develop a proof-of-concept prototype for in-situ fault detection and classification. This project investigates five algorithms namely K-nearest neighbor, Support Vector Machine, Gaussian Naïve Bayes, Decision Trees, and the Random Forest algorithms. The K-nearest neighbor model had an accuracy of 63.077%. The Support Vector Machine model resulted in an accuracy of 76.410%. The Gaussian Naïve Bayes model resulted in an accuracy of 94.359%. The Decision Trees model resulted in an accuracy of 96.923%. The Random Forest model was found to be the most accurate at 98.462%. A Random Forest model was then created that can accurately classify faults at various frequencies. Then deployed the models to a raspberry pi 3.

The major limitation for the future advancement of this work will be data collection. Hence, future work will work toward creating digital twins of systems that can simulate fault scenarios allowing the virtual creation of datasets.

Data

All data used in this project is available upon request.

References

[1] Jardine, A. K. S., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mechanical Systems and Signal Processing, 20(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

[2] Gupta, A. (2016). Machine learning with MATLAB.

[3] Lu Shuang, & Li Meng. (2007). Bearing Fault Diagnosis Based on PCA and SVM. 2007 International Conference on Mechatronics and Automation, 3503–3507. https://doi.org/10.1109/ICMA.2007.4304127

[4] Gunerkar, R. S., & Jalan, A. K. (2019). Classification of Ball Bearing Faults Using Vibro-Acoustic Sensor Data Fusion. Experimental Techniques (Westport, Conn.), 43(5), 635–643. https://doi.org/10.1007/s40799-019-00324-0

[5] Chattopadhyay, P., & Konar, P. (2014). Feature Extraction using Wavelet Transform for Multi-class Fault Detection of Induction Motor. Journal of the Institution of Engineers (India). Series B, Electrical Engineering, Electronics and Telecommunication Engineering, Computer Engineering, 95(1), 73–81. https://doi.org/10.1007/s40031-014-0076-1

[6] By: IBM Cloud Education. (n.d.). *What is supervised learning?* IBM. Retrieved November 23, 2022, from https://www.ibm.com/cloud/learn/supervised-learning

[7] *What is the K-nearest neighbors algorithm?* IBM. (n.d.). Retrieved November 23, 2022, from https://www.ibm.com/topics/knn

[8] Karatzoglou, A., Meyer, D., & Hornik, K. (2006). Support Vector Machines in R. *Journal of Statistical Software*, *15*(9), 1–28. https://doi.org/10.18637/jss.v015.i09

[9] H. Kamel, D. Abdulah and J. M. Al-Tuwaijari, "Cancer Classification Using Gaussian Naive Bayes Algorithm," *2019 International Engineering Conference (IEC)*, 2019, pp. 165-170, doi: 10.1109/IEC47844.2019.8950650.

[10] Fletcher, S., & Islam, M. (2019). Decision Tree Classification with Differential Privacy: A Survey. ACM Computing Surveys, 52(4), 1–33. https://doi.org/10.1145/3337064

[11] Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. ISPRS Journal of Photogrammetry and Remote Sensing, 114, 24–31. https://doi.org/10.1016/j.isprsjprs.2016.01.011

[12] Cornell University. (n.d.). *Lecture 2: k-nearest neighbors*. Lecture 2: K-nearest neighbors / curse of dimensionality. Retrieved November 25, 2022, from https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote02_kNN.html

[13] Guan, H., Li, J., Chapman, M., Deng, F., Ji, Z., & Yang, X. (2013). Integration of orthoimagery and lidar data for object-based urban thematic mapping using random forests. International Journal of Remote Sensing, 34(14), 5166–5186. https://doi.org/10.1080/01431161.2013.788261