

**Naïve Bayes Classifier Optimization for Bearing Fault Detection**

Matthew J. Kenney

QM Analyst/Jr. Developer at Spring Point Solutions, LLC

mattjkenney@protonmail.com

*Matthew J. Kenney currently lives in Connecticut with his wife and two kids who are his greatest motivation to succeed in challenging circumstances. Matthew served six years in the US Navy – directly servicing motors and generators as an Electrician’s Mate for nuclear powered submarines. After the Navy, he immediately began working in Quality Assurance for nuclear energy critical motor repair at Schulz Electric in New Haven, CT. During his time at Schulz Electric, the era of Big Data was becoming popular, and he quickly realized the potential it had to serve the motor repair industry. He began study at Post University where he earned a B.S. Degree in Data Science. During his study, he developed a method for bearing fault detection with vibration analysis. This paper outlines his work.*

**ABSTRACT:**

Vibration analysis is an attractive option for bearing fault detection since vibration data gathering is non-destructive and can be achieved without equipment disassembly. Although, human interpretation requires much experience and knowledge to accurately interpret. Furthermore, detecting the fault location within the bearing itself is near impossible for slight imperfections in the bearing – complicating manufacturing improvements. Naïve Bayes machine learning classifiers have been proposed to accurately predict bearing health without the need for experienced personnel.

Naïve Bayes classifiers work well with small datasets – a symptom of which the needed training set suffers. However, since vibration and speed depend partly on previous magnitudes of the same parameter, the Naïve Bayes classifiers lack the feature independence assumption. Thereby lowering prediction accuracy.

This paper proposes a solution to improve a Naïve Bayes classifier for bearing fault location detection. Improvements are designed in three ways: 1) limiting predictors to Characteristic Bins, 2) determining the ideal feature-domain set for analysis, and 3) optimizing the number of bins and periods used for aggregating and calculating features. Finally, the optimized method is compared to the same method without predictor limitation and a Multinomial Naïve Bayes method from Scikit-Learn.

**KEY WORDS:** bearing fault analysis, vibration analysis, machine learning, Naive Bayes

**INTRODUCTION:**

As of October 2023, wind energy comprises 10.3% of all energy supply in the U.S. [1]. By 2030, the percentage is predicated to nearly double to 20% [1]. Meanwhile, onshore wind energy has the lowest Levelized Cost of Electricity (LCOE) among other sources (including Solar, Geothermal, Gas Peaking, Nuclear, Coal and Gas Combined Cycle) at $24 - 75/MWh [2]. Further, the LCOE of wind energy realized the second largest decline in LCOE between 2009 – 2023 at a 63% drop (the first being Solar PV-UtilityScale with a 83% drop) [2]. With the increasing demand for cheaper alternative energy, it’s easy to see quickly rising demand predicted in 2030.

Along with rising demand for supply, comes a rising demand for Wind Power Plant machine integrity. In 2015, the National Renewable Energy Laboratory (NREL – a national laboratory of the U.S. Department of Energy) established a database, now called the Gearbox Reliability Database (GRD), to track turbine gearbox faults from utility-scale Wind Power Plants. The Alliance for Sustainable Energy (the operator of the NREL) describes the purpose of the GRD:

1. Categorize top wind turbine gearbox failure modes,
2. Identify possible root causes, and
3. Direct future wind turbine gearbox reliability research and development (R&D) activities. [3]

From 750 records in the GRD between 2009 – 2015, 76% of the gearbox failures were caused by faulty bearings [3]. Thus, a strong interest has emerged in predictive bearing health analysis in the wind turbine energy industry. Vibration analysis is an attractive option since vibration data gathering is non-destructive and can be achieved without equipment disassembly. Although, such an analysis takes much experience and knowledge to accurately interpret. Therefore, machine learning solutions have been proposed to accurately interpret vibrational data without the need for technical experts.

**THE PROBLEMS FACING THE MACHINE LEARNING SOLUTION:**

The challenges machine learning engineers face attempting to design bearing fault detection with vibrational data are summed up in three categories: 1) large number of data entries per bearing, 2) low number of bearing samples, and 3) a lack of independent sample data. Each bearing dataset can contain hundreds of thousands of entries. That consumes significant computational resources – therefore, data reduction per bearing sample is much needed. At the same time, every sample dataset is from a single bearing instance in a lot; and every vibration in-part is affected by a previous vibration. That results in low prediction accuracy due to a lack of independent samples and sample data. Expounding on the issue, bearing vibration data is difficult to obtain – it requires specialized test set-ups, trained human resources for operation, and high energy expenses. In consequence, there exists a low number of unique bearing sample data (at least those that are publicly available); and within each sample, the data itself lacks sample independence.

For data reduction efforts, signal features are extracted from datasets. Feature values are statistical calculations that represent a characteristic of a subset of the underlying data. Using features in place of raw vibration signals, new challenges surface: 1) selecting the best feature calculation, and 2) optimizing the subset data size. Additionally, since features utilize the underlying data, they too suffer from sample dependence.

To handle the low bearing sample count, Naïve Bayes (NB) classifiers are an attractive option since they typically perform well with a low number of samples [4]. However, NB classifiers are problematic due to lacking the feature independence assumption.

Despite problem (3), there has been much success with a NB solution in past efforts for bearing fault detection. Zhang et al. [5] overcame the issue utilizing a Decision Tree (DT) to select low correlated features. Yi et al. [6] showed NB can achieve higher accuracy than other models with no feature engineering. Furthermore, Zhang et al. [7] found success with NB and highly correlated features for bearing remaining useful life (RUL) prediction; choosing only features >90% correlation.

This paper proposes another solution to improve a Naïve Bayes classifier for bearing fault location detection. Improvements are designed in three ways: 1) limiting predictors to Characteristic Bins, 2) determining the ideal feature-domain set for analysis, and 3) optimizing the number of bins and periods for aggregating and calculating features. Finally, the optimized method is compared the same method without predictor limitation and a Multinomial Naïve Bayes method from Scikit-Learn.

**NAÏVE BAYES INTRODUCTRION:**

Naïve Bayes equation calculates the *posterior probability*, , of the class according to the predictor where :

Eq. (1)

The term is the *prior probability* of the class. It describes the probability the class will occur based on the class size and total population. The term , is the *likelihood probability* all the predictors in will occur given the classification is true. The *evidence probability* term , is the probability all the predictors in will occur.

The posterior probability is calculated for each possible class. The object is then classified by the class with highest probability:

Eq. (2)

where, class index and the total number of classes.

**ALGORITHM DESIGN:**

A subsection of the domain containing an equal number of data points defines a *period*. A feature value is calculated with the aggregate of data within the period. Grouping the domain in period increments allows control over the data quantity. Lowering the number of periods will reduce the number of calculations for the Bayesian model, thereby saving computational cost and time. Raising the number of periods should raise the accuracy of the model. Feature – Domain engineering in this way is executed by Pseudocode 1.

**Pseudocode 1: Feature – Domain Engineering**

Inputs:

1. Vibration set files, such that each file contains a dataset of vibration velocities at a shaft rotational speed.
2. Number of Periods

Steps:

1. Initialize array
2. For each file in :
3. Extract vibration velocity , speed , and class label data.
4. If acceleration is preferred, convert to acceleration.
5. Sort by .
6. Aggregate equal quantities of data points by calculating a feature across each period for .
7. Build feature array | is the feature value in period in number of Periods.
8. Append to .
9. Return

The range of each period is broken up into vertical *bins*. Bin frequencies for each class are counted for each period. The bins on the extremities of the range for any period include the infinity value, and the bins in the middle are finite. For instance, if a period included 4 bins to cover a range of 20 points, the 3rd highest bin will hold all values between 10 – 15; and the 4th highest bin will hold all values from > 15. Therefore, for each period, all feature values will fall into exactly one bin. Likewise with period quantity, lowering the number of bins will save computational cost and time while raising bin quantity will raise accuracy.

Where is the number of instances in class and is the total population including all classes, prior probabilities are calculated:

Eq. (3)

Where is the frequency count, the likelihood probability of bin in period for class becomes:

Eq. (4)

Likewise, the equation for the evidence probability of bin in period becomes:

Eq. (5)

Pseudocode 2 shows the progression for training the model.

**Pseudocode 2: Training**

Inputs:

1. Pseudocode 1 where files array for training set
2. Number of Bins per Period.

Steps:

1. With :
2. Count for all instances.
3. Count for each unique class label.
4. Create dictionary | class label = Eq. (3), for each unique class label.
5. Calculate bin edges for each period and per period.
6. Create array | bin edges for bin in period .
7. With , count for all unique class labels.
8. Create dictionary | class label = Eq. (4), for all bins in .
9. Create dictionary | Eq. (5), for all bins in .
10. Create dictionary | class label = array containing bin identifiers for all bins occupied by that class.
11. Return

Some bins are not used at all in a particular class, resulting in a 0 probability for Eq. (4). Likewise, some bins are not used at all in any class, resulting in a 0 probability for Eq. (5). Both circumstances show erroneous probabilities. Suppose a vibration set fits in 99 of 100 bins for one known class. Then Eq. (4) becomes 0, where it is actually very likely the bearing belongs to the class since it matches the bin makeup by 99%. Therefore, the model only considers the bins from the intersection of those in the class from the training set and the test samples, called *characteristic bins,* for determining Equations (4) and (5). For instance, let’s say bins correspond to the bin numbers for periods . Now, suppose a test sample uses bins and class A uses bins . Then, with *Characteristic Bins*, only the bins in periods are used for Equations (4) and (5). Without the *Characteristic Bin* utilization, all the bins from the test set are used, regardless of the class characteristic bins. Pseudocode 2

**Pseudocode 3: Naïve Bayes Algorithm with Characteristic Bin Utilization**

Inputs:

1. Pseudocode 1 where files array for test set
2. Pseudocode 2

Steps:

1. With :
2. With , create array containing the bin identifiers of all bins occupied by .
3. Initialize array .
4. For each unique class label in :
5. Calculate Eq. (1) | .
6. Append to .
7. Class label Eq. (2) with probability array .
8. Return class label.

**DATA SOURCE:**

The data contains vibration amplitude and speed recordings captured by [8]. Five bearing classes were evaluated: healthy, inner fault, outer fault, ball fault, and combination fault. Each class was tested in 4 rotational speed conditions with 3 trials: increasing speed, decreasing speed, increasing then decreasing, decreasing then increasing. The total lot consists of 60 datasets. Vibration readings were sampled at 200,000 Hz for 10 seconds, totaling 2 million datapoints per dataset. Speed was measured by an encoder: EPC model 775, 1024 CPR (Cycles Per Revolution). The encoder scale is not reported but assumed to be x2 based on the reported frequency range per test and the resulting velocity calculations. Vibration amplitudes were measured with an accelerometer: ICP accelerometer, Model 623C01. The experimental bearings were ER16K ball bearings. The test rig is composed of an AC Drive, motor and two bearings for support: one healthy, the other experimental (see Figure 1).

Diagram

Description automatically generated

**Figure 1: Test Rig** [8]

**EXPERIMENT 1 – FEATURE SELECTION:**

Five features and two domains are compared totaling 10 tests. The domains include shaft rotational Velocity and Acceleration. Where is an array containing encoder pulse count, is the encoder cycles per revolution, the sampling rate is and assuming a scale of 2, Velocity is calculated:

Eq. (6)

Acceleration is calculated by finding the change of Velocity and dividing it by time. Five features are compared: Skewness, Kurtosis, Crest, Shape, Impulse, and Margin (see Table 1).

**Table 1: Features**

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Description | Equation [9] |  |
| Skewness | A measure of symmetry across the sample mean [10]. |  | Eq. (7) |
| Kurtosis | A measure of tail heaviness of the normal distribution. A heavier tail has more outliers [10]. |  | Eq. (8) |
| Crest | The peak to RMS ratio of a waveform [11]. |  | Eq. (9) |
| Shape | Parameter that affects the general shape of a distribution [12]. |  | Eq. (10) |
| Impulse | Height of peak to the mean signal level [13]. |  | Eq. (11) |
| Margin | Peak amplitude to squared mean of squared roots of absolute amplitudes – also called “clearance factor” [13]. |  | Eq. (12) |

Figure 2 shows the mean Feature of all datasets per period for all Feature – Domain sets, in 20 period increments. It’s clear from Figure 2 that a combination of faults among all Feature – Domain sets are distinct. Likewise, inner-race faults are distinct except with the Skewness feature. The goal now is to determine which Feature – Domain set best distinguishes between the remaining conditions: ball fault, outer race fault, and healthy.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Figure 2: Feature Means in 20 Period Increments, Grouped by Sorted Domains**

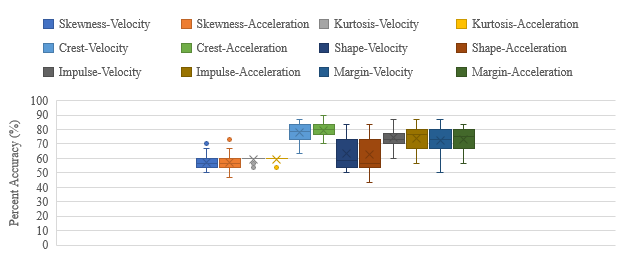
Each feature was tested for model Accuracy along Velocity, and then Acceleration. Model parameters were set at , , and 50% random sampling. New training and test samples were selected for each test. Table 2 shows statistical data for model Accuracy after running the classifier 50 times. Figure 3 shows the results of Table 2 in a box and whisker plot.

**Table 2: Classification Accuracy Statistics by Feature-Domain Sets, n = 50**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Accuracy (%) a | | | | | | |
| Feature | | Mean | Std. Dev. | Min | Max | Median | 25% | 75% |
| Skewness | | | | | | | | |
|  | Velocity | 56.933 | 5.032 | 50.000 | 70.000 | 56.667 | 53.333 | 60.000 |
|  | Acceleration | 57.200 | 5.527 | 46.667 | 73.333 | 56.667 | 53.333 | 60.000 |
| Kurtosis | | | | | | | | |
|  | Velocity | 58.933 | 2.375 | 53.333 | 60.000 | 60.000 | 60.000 | 60.000 |
|  | Acceleration | 59.067 | 2.337 | 53.333 | 60.000 | 60.000 | 60.000 | 60.000 |
| Crest | | | | | | | | |
|  | Velocity | 77.867 | 5.938 | 63.333 | 86.667 | 78.333 | 73.333 | 83.333 |
|  | Acceleration | 79.600 | 4.886 | 70.000 | 90.000 | 80.000 | 76.667 | 83.333 |
| Shape | | | | | | | | |
|  | Velocity | 63.987 | 11.963 | 50.000 | 90.000 | 60.000 | 53.333 | 73.333 |
|  | Acceleration | 63.333 | 12.293 | 43.333 | 90.000 | 56.667 | 53.333 | 73.333 |
| Impulse | | | | | | | | |
|  | Velocity | 74.267 | 5.596 | 60.000 | 86.667 | 73.333 | 70.000 | 76.667 |
|  | Acceleration | 73.933 | 7.429 | 56.667 | 86.667 | 76.667 | 67.500 | 80.000 |
| Margin | | | | | | | | |
|  | Velocity | 72.200 | 8.237 | 50.000 | 86.667 | 73.333 | 66.667 | 79.167 |
|  | Acceleration | 73.333 | 7.284 | 56.667 | 83.333 | 75.000 | 66.667 | 80.000 |

a Highlighted fields show where the metrics for mean or standard deviation are the best performing.

Between Velocity and Acceleration, for all features, there appears no significant difference in mean or median. Kurtosis clearly showed the least variation with a near flat-line appearance in a box plot among all others (see Figure 3). Additionally, Table 2 shows Kurtosis-Acceleration held the lowest standard deviation at 2.337% and 60% accuracy across the board spanning the 25 percentiles to the maximum in both domains. In terms of Accuracy, Crest-Acceleration achieved the highest at 79.6%, 95% CI [78.2, 81.0].



**Figure 3: Classifier Accuracy by Feature – Domain**

**EXPERIMENT 2 – PERIOD AND BIN QUANTITY OPTIMIZATION:**

By varying bin and period sizes, we can find the optimal values for each parameter. Since in Experiment 1, Crest-Acceleration had the highest Accuracy and Kurtosis-Acceleration had the least variation, both were selected for Experiment 2. The model was trained and tested, on three occasions, each with the same and , but with new training and test samples. Where both and , for each set of 3 tests: method parameters were adjusted such that every combination of and was tested. For each set of three tests, mean accuracy was calculated. Figure 4 and Table 3 show the resulting data from Crest-Acceleration. Figure 5 and Table 4 show the results for Kurtosis-Acceleration.

**Figure 4: Mean Classifier Accuracy with Crest-Acceleration, Varying Period and Bin Quantity, n=3**

**Figure 5: Mean Classifier Accuracy with Kurtosis-Acceleration, Varying Period and Bin Quantity, n=3**

Figure 5 shows raising the number of bins per period raises classifier accuracy, plateauing where , for the Kurtosis-Acceleration feature set. There is no clear improvement with either feature set while varying period quantity, and the Crest-Acceleration feature set shows no improvement with varying either parameter. The maximum accuracy achieved is 97% with the Kurtosis-Acceleration feature set where and ; and where and (see Table 4). The CA feature set achieved at most 83% accuracy where and ; and where and (see Table 3).

**Table 3: Mean Classifier Accuracy with Crest-Acceleration, Varying Period and Bin Quantity, n=3 a, b**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Number of Bins per Period | | | | | | | | | |
| Number of Periods | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| 10 | 69 | 81 | 77 | 70 | 66 | 71 | 71 | 64 | 72 | 70 |
| 20 | 78 | 82 | 78 | 81 | 76 | 77 | 76 | 66 | 70 | 69 |
| 30 | 79 | 77 | 73 | 72 | 77 | 72 | 68 | 70 | 72 | 69 |
| 40 | 79 | 73 | 80 | 79 | 80 | 79 | 76 | 80 | 67 | 71 |
| 50 | 78 | 80 | 77 | 71 | 73 | 80 | 73 | 82 | 76 | 80 |
| 60 | 80 | 73 | 74 | 76 | 80 | 71 | 77 | 77 | 71 | 74 |
| 70 | 82 | 76 | 77 | 80 | 69 | 78 | 69 | 72 | 70 | 68 |
| 80 | 79 | 78 | 83 | 77 | 79 | 77 | 76 | 78 | 72 | 77 |
| 90 | 77 | 80 | 78 | 83 | 82 | 78 | 78 | 72 | 76 | 73 |
| 100 | 73 | 73 | 80 | 80 | 79 | 76 | 77 | 71 | 76 | 71 |

a Highlighted fields show where the maximum accuracy rate is achieved. b Data units= %

**Table 4: Mean Classifier Accuracy with Kurtosis-Acceleration, Varying Period and Bin Quantity, n=3 a, b**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Number of Bins per Period | | | | | | | | | |
| Number of Periods | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | |
| 10 | 60 | 63 | 73 | 73 | 78 | 89 | 89 | 91 | 87 | 82 | |
| 20 | 58 | 67 | 74 | 81 | 80 | 90 | 92 | 88 | 89 | 90 | |
| 30 | 58 | 68 | 71 | 74 | 78 | 89 | 92 | 91 | 91 | 89 | |
| 40 | 60 | 67 | 79 | 78 | 81 | 92 | 88 | 90 | 93 | 91 | |
| 50 | 60 | 68 | 76 | 80 | 88 | 89 | 89 | 93 | 90 | 91 | |
| 60 | 59 | 71 | 77 | 81 | 80 | 87 | 92 | 91 | 93 | 91 | |
| 70 | 60 | 71 | 86 | 84 | 89 | 87 | 92 | 92 | 96 | 94 | |
| 80 | 60 | 78 | 78 | 77 | 86 | 86 | 89 | 94 | 97 | 96 | |
| 90 | 60 | 69 | 77 | 74 | 81 | 90 | 93 | 92 | 96 | 92 | |
| 100 | 60 | 73 | 77 | 86 | 86 | 84 | 91 | 92 | 93 | 97 | |

a Highlighted fields show where the maximum accuracy rate is achieved. b Data units= %

**EXPERIMENT 3 – OPTIMIZED ALGORITHM COMPARISIONS:**

The optimization test was conducted with only three samples per parameter set to minimize the time required to run the test. Therefore, the test was repeated for the best performing classifier with the lowest parameter constraints and a larger sample size. Table 5 shows summary statistics for accuracy with Kurtosis-Acceleration where , , and . For comparison, the identical training and test samples were classified using 1) the presented algorithm in this paper using only characteristic bins, *Algorithm A*; 2) the same algorithm using all bins where sample weights= 0.01 (prevents division by 0), *Algorithm B*; and 3) the Multinomial Naïve Bayes method from Scikit-Learn [14], also where sample weights= 0.01, *Algorithm C*. Finally, the training time is measured and recorded for *Algorithms A & B* and *Algorithm C* in Table 7.

**Table 5: Classifier Accuracy Summary Statistics with KA, 80 Periods, 90 Bins per Period, and n = 50**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Accuracy (%) | | | | | | |
| Algorithm | Mean | Std. Dev. | Min | Max | Median | 25% | 75% |
| A | 93.267 | 3.531 | 86.667 | 100.000 | 93.333 | 93.333 | 96.667 |
| B | 59.000 | 1.684 | 56.667 | 63.333 | 60.000 | 56.667 | 60.000 |
| C | 46.200 | 5.871 | 33.333 | 63.333 | 46.667 | 43.333 | 50.000 |

**Table 7: Classifier Training Time**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Time (seconds) | | | | | | |
| Algorithm | Mean | Std. Dev. | Min | Max | Median | 25% | 75% |
| A & B | 221.121 | 21.464 | 206.639 | 285.871 | 219.027 | 208.281 | 220.406 |
| C | 0.001 | 0.001 | 0.000 | 0.003 | 0.001 | 0.001 | 0.002 |

**DISCUSSION:**

It logically follows that raising the bin quantity per period will raise accuracy with nearly all features:

1. More bins per period creates a greater chance for a bearing health class to exclusively occupy a bin; and therefore, creating more *characteristic bins* for the class.
2. With more characteristic bins, there is a greater opportunity for a test set to have more matching characteristic bins for any class.
3. If the test set has more matching characteristic bins than bins overall, the likelihood and evidence probabilities are calculated using only the characteristic bins which leads to a more accurate result.

One exception, as we saw, was Crest-Acceleration where improvements were insignificant while raising bin quantity, peaking at 40 bins. Kurtosis-Acceleration, on the other hand, outperformed Crest-Acceleration by raising bin quantity. Notice that Kurtosis-Acceleration shows a clearer distinction between flaw types (see Figure 2) which was detected by raising the bin quantity.

Future development can be achieved with these suggestions:

1. Build two Bayesian Models: 1) for all conditions, and 2) for only ball fault, outer race fault, and healthy conditions. First, compare the unknown to the first model. If the label is not an inner race or combination fault, then compare it to the second model. Inner race and combination faults are easily distinguishable among most features. Since the second model removes inner race and combination fault data, bin sizes will be smaller and therefore will more easily capture characteristic bins for the conditions with tighter margins.
2. Determine bin quantities based on the smallest margin between conditions per period. Rather than using a fixed bin quantity per period, dynamically scaling bin quantities controls for variances between periods.
3. Determine methods to lower training time. For example, using best practices in code mechanics.

There is a significant trade-off between training time for accuracy while considering the methods in this paper over a traditional multinomial Naïve Bayes algorithm. Although training time amounts to roughly 3 ½ minutes, the outcome is upward from 90% accuracy (93.3%, 95% CI [92.3, 94.3]) – which is worth the wait for most applications of this nature.

In conclusion, the Naïve Bayes model and Characteristic Bin Utilization algorithm presented in this paper can be a steppingstone for unlocking a powerful tool for bearing fault detection. This tool will not only benefit the Wind Power industry, but all industries that rely on continuous rotational machinery operation.

**ACKNOWLEDGEMENTS:**

Post University – Here, I was pushed hard to quickly learn the foundational and advanced data science tools utilized in the work presented in this paper.

Schulz Electric (a Timken Company) – The many years spent with Schulz Electric helped me gain appreciation for motor shops and their customers, faced with the challenging feat of troubleshooting and preventing mechanical faults.

**REFERENCES:**

|  |  |
| --- | --- |
| [1] | U.S. Energy Information Administration, "What is U.S. electricity generation by energy source?," 2023. [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=427&t=3. |
| [2] | G. Bilicic and S. Scroggins, "LAZARD’S LEVELIZED COST OF ENERGY ANALYSIS—VERSION 16.0," LAZARD, 2023. |
| [3] | Alliance for Sustainable Energy, LLC, "Wind Turbine Gearbox Damage Distribution Statistics 2016," 2023. [Online]. Available: https://grd.nrel.gov/. |
| [4] | Z. Zhang, "Naive Bayes Explained," 14 April 2019. [Online]. Available: https://towardsdatascience.com/naive-bayes-explained-9d2b96f4a9c0. |
| [5] | N. Zhang, L. Wu, J. Yang and Y. Guan, "Naive Bayes Bearing Fault Diagnosis Based on Enhanced Independence of Data," *Sensors,* vol. 18, pp. 463-480, 2018. |
| [6] | X. J. Yi, Y. F. Chen and P. Hou, "Fault diagnosis of rolling element bearing using Naïve Bayes classifier," *Vibroengineering PROCEDIA,* vol. 14, pp. 64-69, 2017. |
| [7] | N. Zhang, L. Wu, Z. Wang and Y. Guan, "Bearing Remaining Useful Life Prediction Based on Naive Bayes and Weibull Distributions," *Entropy,* vol. 20, no. 12, p. 944, 2018. |
| [8] | H. Huang and N. Baddour, "Bearing Vibration Data under Time-varying Rotational Speed Conditions," *Mendeley Data, V2,* 2019. |
| [9] | K. H. Hui, C. S. Ooi, M. H. Lim, M. S. Leong and S. M. Al-Obaidi, "An improved wrapper-based feature selection method for machinery fault diagnosis," *PLoS ONE,* vol. 12, no. 12, pp. 1-10, 2017. |
| [10] | National Institute of Standards and Technology, "Measures of Skewness and Kurtosis," [Online]. Available: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm. |
| [11] | Mobius Institue, "Vibration Analysis Dictionary: Terms - Mobius Institute," 2022. [Online]. Available: https://www.mobiusinstitute.com/vibration-analysis-dictionary/. |
| [12] | S. Glen, "Shape Parameter: Definition and Examples," 2023. [Online]. Available: https://www.statisticshowto.com/shape-parameter/. |
| [13] | The MathWorks, Inc., "Signal Features - MATLAB & Simulink," 2023. [Online]. Available: https://www.mathworks.com/help/predmaint/ug/signal-features.html. |
| [14] | F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 12, pp. 2825-2830, 2011. |