Sensorless Synchronous Motors Classification Using Random Forest and Linear Support Vector Classifier

F Rahman, R R Julviar and I N Yulita

1Universitas Padjadjaran, Raya Bandung Sumedang KM.21 Street, Hegarmanah, Jatinangor, Kabupaten Sumedang, West Java45363

fazlur97@gmail.com
julviar17@gmail.com
intan.nurma@unpad.ac.id

Abstract. Automation system become increasingly complex and it needed to be monitored actively. Due to wide variance tasks to be done not suprising that the complexity problem occurs. The failure of the system can impact to economic loss for a company. For this purpose, this paper compare the methods to classify sensorless synchronous motors task to investigate the accuracy and precision of random forest classification and linear Support Vector Classifier (SVC). With cross-validation, the result show that the average accuracy is 99.816% with random forest and 66.04% with linear SVC. As the conclution, random forest classification is more precise than SCV method in this field.

Keywords: sensorless; random forest; svc; classification

1. Introductions

Smart city is a great concept which is cannot be separated from automation as it is today. Either from social point of view or Industrial ones, automation are essential for its resident. Smart city is affecting the resident starting from a small thing like sliding the door for you to the big things like opening a bridge so the ship may pass at the river.

A lot of system evolve to become an automation system starting from a simple task like open a door to the high complexity of industrial activity. Smart city cannot be separated from automation and also cannot be separated from motor-driven system. Motor-driven system is become a huge economical benefit for industry as such it is also a high loss if there were a failure in the system. The main problem in motor driven system is the complexity and it needed to be monitored actively. Due to wide variance tasks to be done not surprisingly that the complexity problem could occurs. The failure of the system can impact to the condition of the motor such as overheating, economic loss for a company. Thus, presently, there is a growing need and interest in developing system that can support the motor driven automation system so the diagnosis can be immediately handled by the system. This automation could also a big support for a city to evolve and become a smart city.
This paper presents a classification for sensorless synchronous motors. The process classifies data where it has a label or target class. The method used is based on machine learning. It studies data, recognizes patterns, and makes models based on historical data. The resulting model is then used to predict test data. Another research has been conducted to classify sensorless drive motor using a fuzzy pattern classification method. But on related paper, no study that focusing on random forest classification for sensorless drive motor. Thus, this paper propose to classify sensorless drive with random forest classification method and compare the accuracy with Linear Support Vector Classification. Both methods have been proven to provide good performance in classification.

2. Related Work

2.1 Random forest
The random forests algorithm is as follows:
1. Draw \( n_{\text{tree}} \) bootstrap samples from the original data.
2. For each of the bootstrap samples, grow an unpruned classification or regression tree, with the modification following: at each node, randomly sample \( m_{\text{try}} \) of the predictors and choose variables that have the best split. (Bagging can be thought of as the special case of random forests obtained when \( m_{\text{try}} = p \), the number of predictors.)
3. Aggregating the predictions of the \( n_{\text{tree}} \) trees to predict new data

2.2 Support Vector Machine
SVM is a machine learning algorithm, and it has developed by different formulations, and this method applicable as an effective classification too. At this following text, we use one of type of SVM, C-SVC that can works with different basic kernel. \( x_i \in \mathbb{R}^n, i = 1, ..., l \), it was training vectors in the two-class case and \( y_i \in \{1, -1\} \) was the corresponding class labels decision, the formula of C-SVC optimization for classification:

\[
\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\
\text{with constraints:}
\]

\[
y_i (w^T \Phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, ..., l.
\]

The dual problem definition is:

\[
\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha, 0 \leq \alpha \leq C, i = 1, ..., l,
\]

with constraints \( y^T \alpha = 0 \), where the vector of all ones is \( e \), \( C > 0 \) as the upper bound, \( Q \) is a \( l \) by \( l \) positive semidefinite matrix, \( Q_{ij} = y_i y_j K(x_i, x_j) \), and \( K(x_i, x_j) \equiv \Phi(x_i)^T \Phi(x_j) \) is the kernel. Function \( \Phi \) transforms training vector \( x_i \) into a higher (or it can infinite) dimensional space.

The decision function:

\[
\text{sgn} \left( \sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \right)
\]
2.3 Motor Drive Diagnosis
Motor Drive Diagnosis was analysed in some earlier publication. In\textsuperscript{9} publication show that ComRef method could increase the classification success for Motor Drive Dataset. In\textsuperscript{10} publication, fuzzy-pattern classification was conducted with Motor Drive Data.

3. Method

![Experiment Design](image)

The data is downloaded from UCI in format of txt. The field is space separated and the record is new line separated. We change the format to Comma Separated Value (CSV) so the data can be easily read than the txt format with simple text editor Notepad++. The CSV data then read using python pandas library for better and faster loading rather than parsing it manually with text manipulating code. Then we separate the label and training data before KFolding to make several training set and test set pair. After that we run the test for each classifier with the training set and test set pairs that have been created and populated the accuracy result of each test case. If the test had been conducted successfully the program will show the populated accuracy and its statistics.

This paper use Scikit-Learn library from python. Scikit-Learn\textsuperscript{11} is a machine learning library in Python that exposes wide variety of machine learning algorithm both supervised and unsupervised. It provide state-of-the-art implementation of many machine learning algorithms.
For the classification. We specified the parameter for each classification method as shown in Table I.

| Method         | Parameter                                      |
|----------------|-----------------------------------------------|
| **Random Forest** | min_samples_split=20  
The minimum number of samples required to split an internal node  
|                 | n_jobs =-2  
the number of processor that’s used for the processing 1 mean a single processor will be used 2 mean two processor while -1 means all and -2 means n-1 processor used.  
|                 | n_estimators =40  
The number of trees in the forest  
|                 | criterion='gini'  
The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain. Note: this parameter is tree-specific  
|                 | max_depth = none  
The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples  
|                 | min_samples_leaf = 1  
The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.  
|                 | min_weight_fraction_leaf = 0  
The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided  
|                 | max_features ='auto'  
The number of features to consider when looking for the best split  
|                 | max_leaf_nodes=None  
Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes  
|                 | min_impurity_decrease=0  
A node will be split if this split induces a decrease of the impurity greater than or equal to this value  
|                 | bootstrap=True  
Whether bootstrap samples are used when building trees  
|                 | oob_score=False  
Whether to use out-of-bag samples to estimate the generalization accuracy  
|                 | random_state=None  
If int, random_state is the seed used by the random number generator; If
RandomState instance, random_state is the random number generator; If None, the random number generator is the RandomState instance used by np.random.

- **verbose = 0**
  Controls the verbosity when fitting and predicting

- **warm_start=False**
  When set to True, reuse the solution of the previous call to fit and add more estimators to the ensemble, otherwise, just fit a whole new forest

| Linear SVC                                                                 |
|----------------------------------------------------------------------------|
| - **penalty =’l2’**                                                       |
| the norm used in the penalization. The ‘l2’ penalty is the standard used   |
| in SVC. The ‘l1’ leads to coef_vectors that are sparse.                   |
| - **loss =’squared_hinge’**                                               |
| the loss function. ‘hinge’ is the standard SVM loss (used e.g. by the SVC  |
| class) while ‘squared_hinge’ is the square of the hinge loss.             |
| - **dual =True**                                                         |
| Select the algorithm to either solve the dual or primal optimization      |
| problem. Prefer dual=False when n_samples > n_features.                   |
| - **tol =1e-4**                                                          |
| Tolerance for stopping criteria.                                          |
| - **C =1.0)**                                                            |
| Penalty parameter C of the error term.                                   |
| - **multi_class =’ovr’**                                                 |
| Determines the multi-class strategy if y contains more than two classes.  |
| Ovr stand for one vs rest                                                 |
| - **fit_intercept =True**                                                |
| Whether to calculate the intercept for this model. If set to false, no   |
| intercept will be used in calculations (i.e. data is expected to be      |
| already centered).                                                       |
| - **intercept_scaling default=1**                                        |
| When self.fit_intercept is True, instance vector x becomes [x, self     |
| .intercept_scaling]                                                      |
| - **verbose = 0**                                                       |
| Enable verbose output.                                                    |
| - **max_iter = 2000**                                                   |
| The maximum number of iterations to be run.                              |

4. **Experiment**

This paper Compare the accuracy of random forest classification and SVM classification. Sensorless Drive Diagnosis Data Set taken from UCI repository is used in this paper. This dataset contain 58510 data. The unit under investigation is composed of a synchronous motor and several attached components, e.g. bearings, axles, and a gear box. It, thus, represents a typical and crucial component within a plant or other machinery. The result of recording is consisting of 12 different operating condition with the 3 operational parameter rotational speed, load torque, and radial force. For the purpose of research, scikit-learn library is used for the SVM classification and random forest classification.
This paper uses 3 different parameters of KFolding: 4 folds, 10 folds, 100 folds to maintain objectivity of the test. The result of random forest presented in Table II are following: the classification average accuracies are roughly between 99.31% - 100%, minimal and maximal accuracy values show no big different with average accuracy. The result of SVM presented in Table III are following: the accuracies shows lower value than Random Forest classification and are roughly 64.01% - 65.12%. The minimal and maximal accuracy values shows higher difference with average accuracy. It also shown that linear SVC takes much longer time to classify. It is because Linear SVC on python didn’t have the capabilities to use multicore programming as it will only run on single thread if didn’t get combined with any ensemble method. The fastest rate of random forest is 17.2s per folds on 4 folds test, and the slowest is 61.22s per fold on 100 folds test.

We realize that different folds might lead to different accuracy value. In this experiment, bigger folds consume more time to execute.

Table 2. Classification rates for 5x2 cross validation test with Random Forest Classification

| Folds | Average Accuracy | Min   | Max   | Elapsed Time |
|-------|------------------|-------|-------|--------------|
| 4     | 99.80 %          | 99.73%| 99.85%| 70.92 second |
| 10    | 99.81 %          | 99.64%| 99.81%| 382.04 second|
| 100   | 99.84 %          | 99.31%| 100.00%| 6122.84 second|

Table 3. Classification rates for 5x2 cross validation test with SVM Classification

| Folds | Average Accuracy | Min   | Max   | Elapsed Time |
|-------|------------------|-------|-------|--------------|
| 4     | 66.11 %          | 63.13%| 67.40%| 1270.69 second |
| 10    | 67.56%           | 65.58%| 70.16%| 4364.79 second|
| 100   | 64.45%           | 52.35%| 71.35%| 16.954.78 second|

5. Conclusion
Wrong decision on sensorless drive motor could lead to a trouble of overheating or break the motor. This paper address the problem on smart city that a simple system like motor failure could mean a big trouble. Thus in this paper we solve it with random forest classifier and SVC to classify the data quickly and accurate. With cross-validation 4, 10, 100 folds, the result show that the average accuracy is 99.816% with random forest and 66.04% with linear SVC. As the conclusion, random forest classification is more precise than SVC method in this field. It is also a faster method as shown in the table in forth chapter. In the light of this matter this paper could be a reference for quick classification of this data set without sacrificing much accuracy so the decision can be taken fast and accurate.

6. Reference
[1] Enge-rosenblatt, O., Bayer, C., & Brüning, A. (2016). Complex Analysis of Monitoring and Process Data for Effective Manufacturing, (December).
[2] Yulita, I. N., Fanany, M., & Arymurthy, A. (2017). Bi-directional Long Short-Term Memory using Quantized data of Deep Belief Networks for Sleep Stage Classification (pp. 530–538). Bali: Procedia Computer Science.
[3] Helene, D. (2014). Combinatorial Refinement of Feature Weighting for Linear Classification.
[4] Yulita, I. N., Fanany, M., & Arymurthy, A. (2018). Fast Convolutional Method for Automatic Sleep Stage Classification. Healthcare Informatics Research, 170–178.
[5] Liaw, A., & Wiener, M. (2014). Classification and Regression by randomForest, (November 2001).
[6] Novakovic, J., & Veljovic, A. (2011). C-support vector classification: Selection of kernel and
parameters in medical diagnosis. *SISY 2011 - 9th International Symposium on Intelligent Systems and Informatics, Proceedings*, 465–470. https://doi.org/10.1109/SISY.2011.6034373

[7] Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, 20(3), 273–297. https://doi.org/10.1023/A:1022627411411

[8] Vapnik, V. N. (1998). Statistical Learning Theory. *Adaptive and Learning Systems for Signal Processing, Communications and Control*, 2, 1–740. https://doi.org/10.2307/1271368

[9] Helene, D., & Uwe, M. (2014). Fast Classification in Industrial Big Data Environments.

[10] Bayer, C., Enge-rosenblatt, O., Dicks, A., & Lohweg, V. (2013). Sensorless drive diagnosis using automated feature extraction, significance ranking and reduction. *Sensorless Drive Diagnosis Using Automated Feature Extraction, Significance Ranking and Reduction*, (September). https://doi.org/10.1109/ETFA.2013.6648126

[11] Pedregosa, F., Weiss, R., & Brucher, M. (2011). Scikit-learn: Machine Learning in Python, 12, 2825–2830.

**Acknowledgments**

We would like to thank Martyna Bator from University of Applied Science for providing helpful support regarding Motor Drive Diagnosis dataset.

This article is presented at the International Conference on Smart City Innovation 2018 that supported by the United States Agency for International Development (US-AID) through the Sustainable Higher Education Research Alliance (SHERA) Program for Universitas Indonesia’s Scientific Modeling, Application, Research and Training for City-centered Innovation and Technology (SMART CITY) Project, Grant #AID-497-A-1600004, Sub Grant #IIE-00000078-UI-1.