Machine Learning in Manufacturing

Tejas Sanap

List of Figures

List of Tables

Contents

Chapter 1

Basics of Machine Learning

1 Introduction

Modern manufacturing systems are dynamic and complex, needing use of technologies that have the ability to learn and adapt to varying environment. Modern manufacturing setups generate large amounts of data which can be leveraged for process optimization, quality improvement and cost estimation. Artificial Intelligence-based technologies are available at cheaper rates through low-cost storage and computation and the availability of cloud services.

1.1 Trends in manufacturing

Newer technologies constantly appear, promising better performance at lower operating costs. However, certain trends can be observed, globally. More and more, companies are trying to integrate their business operations and engineering operations. Practices where sales, supply chain, inventory and production floor are tightly integrated within the ERP-paradigm, are strong examples. Companies benefit from such exchange of information between various organs of the company. An average modern customer demands higher levels of customization, at lower costs within the same time-frame. Conventional make-to-stock production models are unable to keep up with this demand-driven market. Thus, manufacturers are adopting newer production models such as make-to-order, configure-to-order and engineer-to-order. These models move away from the conventional mass production approach. These models enable custom configuration of each product according to individual customer's demands. As modern manufacturing technologies such as additive manufacturing become more mainstream, its process optimization will become an important task. Also, shifting the general focus of additive manufacturing from prototyping to mass production will become important. Manufacturing products like MEMS[1] devices need extremely tight process parameter monitoring and control. All of these tasks will need to be automated not only to increase the rate of production but also to ensure quality. Future factories will have to seek more sustainable processes to create sustainable products that have minimum adverse effect on the environment and better integrate with the rest of the product ecosystem.

1.2 Artificial Intelligence-based trends in manufacturing

Use of Artificial Intelligence (AI) in automation has been tremendous. However, adoption of AI-based technologies in the domain of manufacturing has been slower than anticipated[2]. Inhibiting factors to its adoption are things like the low human-resource costs in emerging countries, which didn't justify spending on AI and related automation. Inclusion of AI in the manufacturing process opens roads to new business models and diversification of markets. AI, in terms of manufacturing, requires cheap sensing technologies. Perception related hardware plays an important role in any automation project. Cheap sensing technologies enable machine vision, quality inspection and process monitoring[3]. Examples of AI include, precise quality control of integrated circuits to training a group of robots and humans to work in collaborative environments. AI finds application in areas out of automation and robotics. Numerous demonstrations of perceiving patterns of demand through various sales data merged with other socioeconomic and environmental data by various authors are available. Predictive maintenance of machines with Internet-of-things and adaptive process optimization with streaming data, are few of the examples.

2 Concepts of Machine Learning

Machine Learning (ML) is a sub-field of AI. While, AI in general tries to simulate or imitate human intelligence, ML focuses on the task of learning from various sources of data such as sensor data from production line and environmental data. ML algorithms use various statistical and probabilistic techniques to progressively improve performance on a specific task.

The formulation of any ML problem involves three elements: tasks, models and features. Models are constructed from selective features and trained using data[4].

The task of learning can be divided into two parts: *induction* and *deduction*. These terms in ML terminology are called as *training* and *prediction*. A suitable algorithm is trained using a given data-set during which it extracts various underlying patterns and behavior of various variables defined in the data-set. The algorithm will generally analyze the relationship between various variables temporally and spatially. ML algorithm will derive a high-level structure from low-level data.

ML is diverse field with many algorithms and theories. The relationship between various algorithms and theories and their conceptual structure, varies from author to author and is not commonly agreed upon. However, the general consensus divides machine learning into three types:

- *Reinforcement learning*, training data is incomplete and thus, the focus is on overall performance at the cost of making sub-optimal decisions. A feedback system is created in the form of rewards.

- *Supervised learning*, the training data is labeled into correct and incorrect responses. This type of learning includes provision of maximum feedback.

- *Unsupervised learning*, no distinction is made between the correct and incorrect response. Emphasis is on pattern discovery, clustering, and seeking out association rules. This type of learning includes no provision of feedback.

3 Strengths of Machine Learning

The NP-complete nature of manufacturing problems such as process scheduling and optimization, makes solving them a challenging task. Most manufacturing problems share certain characteristics which are offset by the strengths of ML. The key strength of ML is its ability to adapt to high dimensional and complex data.

Manufacturing problems are characterized by a large number of features which often have a lot of data[1]. Problems with more than 10 dimensions are considered to be high-dimensional problems. 'Since most engineering and manufacturing problems are data-rich but are knowledge-sparse'[5]. The large dimensionality of a ML problem is not considered a problem in most cases. Even if, a problem was to reach the dimensionality threshold of a algorithm, various dimensionality reduction procedures are available. This allows use of seemingly irrelevant or useless data in our analysis that may turn out be relevant in a few cases. Due to the lack of transparency about what kind of data is being made available, feature selection becomes difficult. Thus, it is advisable to use as many features as possible. Especially in academia, data provided by industry is often anonymized. It is advisable to the beginner to not use geometric intuition when dealing with problems having high dimensionality. In ML litreature the problem of high dimensionality is often called the *curse of dimensionality.*

The complexity of manufacturing related problems is due to the numerous parameters that need to be tightly controlled in order to obtain the final quality that has been promised to the customer. Parameters are often interrelated, which makes monitoring them and adjusting the process like-wise, imperative. Normally, the relationship between various parameters are described using a multi-variate model. The modeling is generally done on the basis of first-principles related to the physics of the process. Such models assume some sort of functional dependency among the various parameters. Parameters usually exhibit non-linear and/or chaotic behaviors. However, first-principle based models might not capture the process in its entirety, leading to inaccurate predictions and sub-optimal results. The ML approach, which is data-driven, does not make any assumptions about the underlying physical reality of the process being observed. Instead, ML models only make elementary assumptions about the structure of data they will process. This proves to be advantageous as relationships between parameters is often implicit. In the particular case of complex manufacturing lines, identification of the root cause of an observed error becomes a tedious task. The ML approach is to identify the root cause by measuring its impact on the observed error, statistically[6].

The ability of ML models to tune themselves and adapt to changes according live streaming data has been widely demonstrated. Demonstrations include training of a collaborative working environment comprising of humans and robots. As most manufacturing environments are characterized by their dynamic and uncertain nature, the ability to learn and adapt to changing environment gives ML a very unique advantage. Special classes of ML algorithms exist which are designed and optimized to minimize training time and amount of data needed to reach a certain degree of accuracy. Examples of such ML model classes include Extreme Machine Learning (ELM)[7]. ELM algorithms have a training time thousands of times less than typical ML algorithms. ELM has been showed to very useful in data-driven modeling, in this particular case, to model the near chaotic behaviour of HCCI engines[8].

4 Selection of ML algorithm

Most problems in ML can be divided into three different categories, classification, regression and clustering. Classification deals with discrete sets and regression with continuous sets. Clustering is used for pattern searching or knowledge discovery. Classification and regression come under supervised learning and clustering comes under unsupervised learning. Classification involves assigning a label to an instance from a finite set of multiple labels. Thus, the set of all possible solutions contains a fixed number of discrete elements. In some use cases, it is preferable to abandon the concept of discrete classes altogether and instead, predict a real number from a set of continuous values. This is the case in regression. Clustering problems involves looking for pattern in a dataset, finding how the dataset can be divided in numerous clusters or finding the rules of association between various variables. Selection of ML algorithms depends upon the category under which the specific use-case will fall. After which, the algorithms under that category must be compared. Some algorithms are common to multiple categories.

4.1 Differences between some common algorithms[9]

Generally, Support Vector Machines (SVM) and neural networks tend to perform much better when dealing with multi-dimensions and continuous features. On the other hand, logic-based systems tend to perform better when dealing with discrete/categorical features. For neural network models and SVMs, a large sample size is required in order to achieve its maximum prediction accuracy whereas Naive Bayes (NB) may need a relatively small dataset.

SVMs are binary algorithms. In the training stage, multiple independent binary classifiers are constructed, each of which is based on a different partition of the set of the labels into two disjointed sets. In the second stage, the classification part, the predictions of the binary classifiers are combined to extend a prediction on the original label of a test instance.

There is general agreement that k-Nearest Neighbors (k-NN) is very sensitive to irrelevant features: this characteristic can be explained by the way the algorithm works. Moreover, the presence of irrelevant features can make neural network training very inefficient, even impractical.

Bias measures the contribution to error of the central tendency of the classifier when trained on different data[10]. Variance is a measure of the contribution to error of deviations from the central tendency. Learning algorithms with a high-bias profile usually generate simple, highly constrained models which are quite insensitive to data fluctuations, so that variance is low. Naive Bayes is considered to have high bias, because it assumes that the dataset under consideration can be summarized by a single probability distribution and that this model is sufficient to discriminate between classes. On the contrary, algorithms with a high-variance profile can generate arbitrarily complex models which fit data variations more readily. Examples of high-variance algorithms are decision trees, neural networks and SVMs. The obvious pitfall of high-variance model classes is overfitting.

Most decision tree algorithms cannot perform well with problems that require diagonal partitioning. The division of the instance space is orthogonal to the axis of one variable and parallel to all other axes. Therefore, the resulting regions after partitioning are all hyperrectangles. The Artificial Neural Networks(ANN) and the SVMs perform well when multi-collinearity is present and a nonlinear relationship exists between the input and output features. Lazy learning methods require zero training time because the training instance is simply stored. Naive Bayes methods also train very quickly since they require only a single pass on the data either to count frequencies (for discrete variables) or to compute the normal probability density function (for continuous variables under normality assumptions). Univariate decision trees are also reputed to be quite fastat any rate, several orders of magnitude faster than neural networks and SVMs.

Naive Bayes requires little storage space during both the training and classification stages: the strict minimum is the memory needed to store the prior and conditional probabilities. The basic k-NN algorithm uses a great deal of storage space for the training phase, and its execution space is at least as big as its training space. On the contrary, for all non-lazy learners, execution space is usually much smaller than training space, since the resulting classifier is usually a highly condensed summary of the data.

Moreover, Naive Bayes and the k-NN can be easily used as incremental learners whereas rule algorithms cannot. Naive Bayes is naturally robust to missing values since these are simply ignored in computing probabilities and hence have no impact on the final decision. On the contrary, k-NN and neural networks require complete records to do their work. Moreover, k-NN is generally considered intolerant of noise; its similarity measures can be easily distorted by errors in attribute values, thus leading it to misclassify a new instance on the basis of the wrong nearest neighbors. Contrary to kNN, rule learners and most decision trees are considered resistant to noise because their pruning strategies avoid overfitting the data in general and noisy data in particular.

What is more, the number of model or runtime parameters to be tuned by the user is an indicator of an algorithms ease of use. As expected, neural networks and SVMs have more parameters than the remaining techniques. The basic k-NN has usually only a single parameter(k) which is relatively easy to tune.

Logic-based algorithms are all considered very easy to interpret, whereas neural networks and SVMs have notoriously poor interpretability. k-NN is also considered to have very poor interpretability because an unstructured collection of training instances is far from readable, especially if there are many of them. While interpretability concerns the typical classifier generated by a learning algorithm, transparency refers to whether the principle of the method is easily understood. A particularly eloquent case is that of k-NN; while the resulting classifier is not quite interpretable, the method itself is quite transparent because it appeals to the intuition of human users, who spontaneously reason in a similar manner. Similarly, Naive Bayes' is very transparent, as it is easily grasped by users like physicians who find that probabilistic explanations replicate their way of diagnosing. Similarly, Naive Bayes' explanations in terms of the sum of information gains is very transparent, as it is easily grasped by users like physicians who find that explanations replicate their way of diagnosing.

Finally, decision trees and NB generally have different operational profiles, when one is very accurate the other is not and vice versa. On the contrary, decision trees and rule classifiers have a similar operational profile. SVM and ANN have also a similar operational profile. No single learning algorithm can uniformly outperform other algorithms over all datasets.

4.2 Ensembling of multiple algorithms

The concept of combining classifiers is proposed as a new direction for the improvement of the performance of individual classifiers. The goal of classification result integration algorithms is to generate more certain, precise and accurate system results. Although or perhaps because many methods of ensemble creation have been proposed, there is as yet no clear picture of which method is best. Mechanisms that are used to build ensemble of classifiers include:

1. Using different subsets of training data with a single learning method.
2. Using different training parameters with a single training method.
3. Using different learning methods

The possibility of integrating two or more algorithms together to solve a problem should be investigated. The objective is to utilize the strengths of one method to complement the weaknesses of another. If we are only interested in the best possible classification accuracy, it might be difficult or impossible to find a single classifier that performs as well as a good ensemble of classifiers. Despite the obvious advantages, ensemble methods have at least three weaknesses. The first weakness is increased storage as a direct consequence of the requirement that all component classifiers, instead of a single classifier, need to be stored after training. The total storage depends on the size of each component classifier itself and the size of the ensemble (number of classifiers in the ensemble). The second weakness is increased computation because in order to classify an input query, all component classifiers (instead of a single classifier) must be processed. The last weakness is decreased comprehensibility. With involvement of multiple classifiers in decision-making, it is more difficult for non-expert users to perceive the underlying reasoning process leading to a decision.

Chapter 2

Algorithms

1 Support vector machine

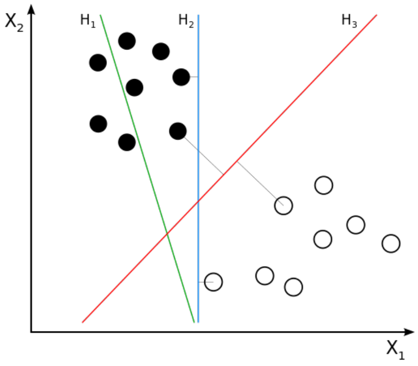

Figure 2.1: Linear SVM seperating vector using hyperplanes

SVM's are a set of ML methods that are used for classification, regression and outlier detection. Outliers are points that lie far away from the distribution mean. Outliers tend to affect model performance, adversely. classifying data is a common task in machine learning. suppose all the given data points in a data set each belong to one of two classes and, the task is to find out the class of a new data point. In the case of SVMs each data point a n-dimensional vector in a n-dimensional vector space. The vector space can be divided using a $(n-1)$ dimensional hyperplane. We want to know if such a hyperplane exists. A hyperplane for which the distance from it to the nearest data point on each side is maximized, is called *maximum margin hyperplane* and the classifier it defines, is called the *maximum margin classifier*.

In the figure 2.1, we can observe that SVM would choose a separating hyperplane for two classes of points in 2D. H1 does not separate the classes. H2 does, but only with a small margin. H3 separates them with the maximum margin.

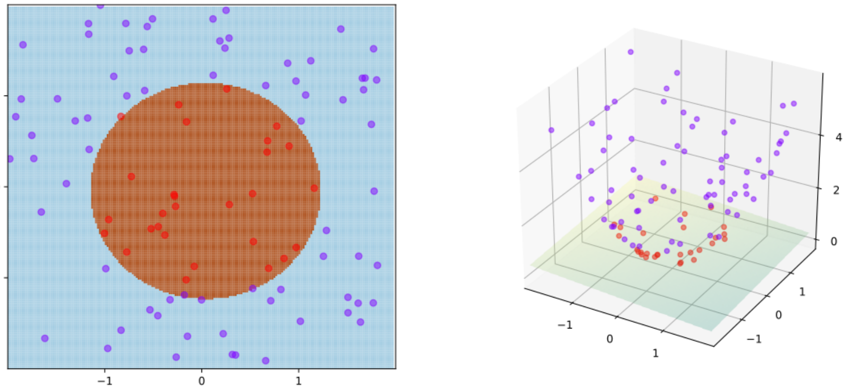

Figure 2.2: Non-linear SVM

Given a training dataset of n points of the form,

$$(\vec{x_1}, y_1), \cdots, (\vec{x_n}, y_n) \tag{2.1}$$

Where each x_i is p-dimensional vector and each y_i has a value of $+1$ or -1 indicating the class to which it belongs. The SVM algorithm tries to find out the "maximum margin hyperplane" which divides the dataset into two groups, one where the value y_i corresponding to x_i has the value $+1$ or-1. Any hyperplane can be defined as the set of point $\vec{x}$ satisfying

$$\vec{w} \cdot \vec{x} - b = 0 \tag{2.2}$$

Where $\vec{w}$ is the normal vector to the hyperplane. One of the few drawbacks of SVM is that the generalization error increases with the number of dimensions.

SVM algorithms are used to classification tasks and regression tasks, they mainly find application in predicting performance parameters or yields from processes. SVM's can be used as non-linear classifiers and regressor[11].

2 Naive-Bayes Classifier

The Bayes theorem is stated mathematically as the following equation:

$$P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B)} \Rightarrow \text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \tag{2.3}$$

The Naive Bayes classifier is a family of probabilistic classifiers based on Bayes theorem with assumptions of strong independence between the features. Bayes theorem is a theorem in Probability concerning conditional probability. Basically, Abstractly, Naive Bayes is a *conditional probability* model: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \ldots, x_n)$ representing some n features (independent variables), it assigns to this instance probabilities,

$$p(C_k \mid x_1, \ldots, x_n) \tag{2.4}$$

for each of K possible outcomes or classes C_k.

In practice, we are only concerned with the numerator. The denominator is a constant as it doesn't depend on C. The numerator is equivalent to the *joint probability* model,

$$p(C_k, x_1, \cdots, x_n) \tag{2.5}$$

This can also be rewritten as,

$$\begin{aligned} p(C_k, x_1, \cdots, x_n) =& \, p(x_1, \cdots, x_n, C_k) \\ =& \, p(x_1 \mid x_2, \cdots, x_n, C_k) p(x_2, \cdots, x_n, C_k) \\ =& \, p(x_1 \mid x_2, \cdots, x_n, C_k) p(x_2 \mid x_3, \cdots, x_n, C_k) \\ & \, p(x_3, \cdots, x_n, C_k) \\ =& \, \cdots \\ =& \, p(x_1 \mid x_2, \cdots, x_n, C_k) p(x_2 \mid x_3, \cdots, x_n, C_k) \\ & \, p(x_3 \mid x_4, \cdots, x_n, C_k) \cdots p(x_{n-1} \mid x_n, \cdots, C_k) \\ & \, p(x_n|C_k) p(C_k) \end{aligned} \tag{2.6}$$

Now, the "naive" conditional independence assumptions made initially state that every feature x_i is conditionally independent of feature x_j provided $i \neq j$, given the category C_k. This means that

$$p(x_i \mid x_{i+1}, \cdots, x_n, C_k) = p(x_i \mid C_k) \tag{2.7}$$

The above model allows us to calculate the probability of individual features. Now, a classifier can be constructed from this model by turning it into an optimization problem by combining it with a decision rule. The NB algorithm mainly finds application in monitoring and control applications.

3 Comparison of supervised learning algorithms[9]

Parameter	Decision Trees	ANN	NB	kNN	SVM
Accuracy	▽▽	▽▽▽	▽	▽▽	▽▽▽▽
Speed of Classification	▽▽▽	▽	▽▽▽▽	▽	▽▽
Tolerance to missing values	▽▽▽▽	▽▽▽▽	▽▽▽▽	▽	▽▽▽▽
Tolerance to irrelevant attributes	▽▽▽	▽	▽▽	▽▽▽▽	▽▽
Tolerance to redundant attributes	▽▽	▽▽	▽	▽▽	▽▽▽
Tolerance to highly interdependent values	▽▽	▽▽▽	▽	▽	▽▽
Dealing with binary/ discrete/continuous attributes	▽▽▽▽	▽▽▽ (not discrete)	▽▽▽ (not continuous)	▽▽ (not directly discrete)	▽▽ (not discrete)
Tolerance to noise	▽▽	▽▽	▽▽▽	▽	▽▽
Tendency to overfit	▽▽▽	▽	▽▽▽	▽▽▽	▽▽
Attempts for incremental learning	▽▽	▽▽▽	▽▽▽▽	▽▽▽▽	▽▽
Model parameter handling	▽▽▽	▽	▽▽▽▽	▽▽▽	▽

Table 2.1: ▽▽▽▽ and ▽, are the best and worst performance, respectively.

Chapter 3

Conclusion

In conclusion, we can say that ML offers practical solutions to problems in Manufacturing. Having analyzed the trends in manufacturing and AI, we have come to realize the use of ML is inevitable. We hope, that the comparative study helps practitioners not choose the wrong algorithm. Detailed discussion of two common algorithms, SVM and NB, is done.

Bibliography

[1] C. I. Thorsten Wuest, Daniel Weimer and K.-D. Thoben, "Machine learning in manufacturing: advantages, challenges, and applications," *Production and Manufacturing Research*, vol. 4, no. 1, pp. 23–45, 2016.

[2] P. Dorfman, "3 advances changing the future of artificial intelligence in manufacturing."

[3] C. I. Thorsten Wuest and K.-D. Thoben, "An approach to monitoring quality in manufacturing using supervised machine learning on product state data," *Journal of Intelligent Manufacutring*, 2013.

[4] P. Flach, *Machine Learning.* Cambridge University Press, 2012.

[5] S. C.-Y. Lu, "Machine learning approaches to knowledge synthesis and integration tasks for advanced engineering automation," *Computers in Industry*, vol. 15, no. 1, pp. 105 – 120, 1990.

[6] J. T. Pfingsten, "Machine learning for mass production and industrial engineering," 2007.

[7] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 2, pp. 985–990 vol.2, July 2004.

[8] A. Vaughan and S. V. Bohac, "An extreme learning machine approach to predicting near chaotic HCCI combustion phasing in real-time," *CoRR*, vol. abs/1310.3567, 2013.

[9] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica*, vol. 31, pp. 249–268, 2007.

[10] E. Bauer and R. Kohavi, “An empirical comparison of voting classification algorithms: Bagging, boosting, and variants,” *Machine Learning*, vol. 36, pp. 105–139, Jul 1999.

[11] T. D. Po-Sen Huangy and M. Hasegawa-Johnson, “Multi-sensory features for personnel detection at border crossings,” 2013.