Feature Selection for troubleshooting in complex assembly lines

Tobias Pfingsten, Daniel J.L. Herrmann, Thomas Schnitzler, Andreas Feustel, and Bernhard Schölkopf

***Abstract*— The final properties of sophisticated products can be affected by many unapparent dependencies within the manufacturing process, and the products' integrity can often only be checked in a final measurement. Troubleshooting can therefore be very tedious if not impossible in large assembly lines.**
In this paper we show that Feature Selection is an efficient tool for serial-grouped lines to reveal causes for irregularities in product attributes. We compare the performance of several methods for Feature Selection on real-world problems in mass-production of semiconductor devices.

***Note to Practitioners*— We present a data based procedure to localize flaws in large production lines: using the results of final quality inspections and information about which machines processed which batches, we are able to identify machines which cause low yield.**

***Index Terms*— Feature Selection, production chain, SVM.**

I. Introduction

On the following pages we describe how Feature Selection can be used as a tool to detect conspicuous processes in complex manufacturing lines. Our work was inspired by the situation given in a plant for semiconductor products, but applies as well to other computer–integrated assembly lines. In the manufacturing process of semiconductor products one deals with a great number of production steps that involve many different machines. Malfunctions can usually not be ruled out or identified in each processing step, and the actual quality of the product can only be evaluated in a final measurement.

A well known technique for fault detection is the so-called Taguchi method that uses orthogonal designs to effectively test complex systems [1]. In contrast to this experimental approach, which makes it necessary to launch test runs, we propose to use existing data from serial production by applying Feature Selection which has proven to be highly effective in new scientific fields such as bioinformatics [2]. Many previous works report on the use of Data Mining methods to analyze industrial data. Refer to [3], [4], [5] for recent review articles on fault detection. In contrast to our approach, previous works model specific processes to detect errors. In [6], for example, decision trees and neural nets are used to improve a cleaning process, in [7] Feature Selection is used to relate high dimensional electrical measurements to the yield. Our approach is novel in that it concentrates on the assembly line as a whole rather than building models for single processes.

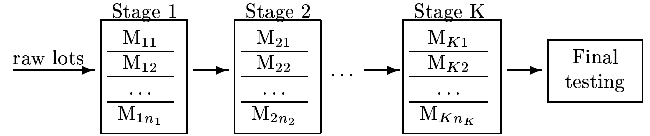

Fig. 1. Schematic view on how lots are handled during production. Each lot passes K stages with several parallel machines. After production lots are analyzed in a final test stage. Note that machines can be used in several stages.

In the examples which we shall describe later on, as many as 357 machines are used in 403 production stages. When final measurements show that some lots suffer from low yield due to previously unseen errors, one can suspect that one or more processes are subject to a disorder. Besides the results of the final measurements, we have access to the record of which machine handled which lot in each step. Our approach combines lot history and test results to locate dysfunctions which have not been captured by existing process control. While it can be hard to find causes for irregularities manually, located errors can be corrected systematically in little time.

We solve the task by treating the situation described above as a classification problem. High dimensional input vectors describe the history of each lot, while the targets are binary labels which divide the lots into regular and conspicuous ones. Feature Selection algorithms are designed to uncover combinations of variables which are related to the classification target. In our procedure we use these algorithms to find combinations of machines which are related to the irregularity.

We start with Section II by describing the problem, the available data and its preprocessing. The processed data is analyzed with different methods for Feature Selection, which we introduce in Section III. In Section IV we present results on data from our shop floor to demonstrate the effectiveness of the proposed approach and to give an instruction on how to use the methods in practice. We give a summary of the results in Section V.

II. Problem Setup and Preprocessing

When thinking of assembly lines, one usually has in mind a serial configuration of single machines that manipulate

Submitted March 2003. This is a preprint: accepted for publication in the IEEE Transactions on Automation Science and Engineering. Please respect the copyright: This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.

Tobias Pfingsten, Daniel J.L. Herrmann, Thomas Schnitzler and Andreas Feustel are with Robert Bosch GmbH, Stuttgart, Bernhard Schölkopf and Tobias Pfingsten are with Max Planck Institute for Biological Cybernetics, Tübingen

products in a given order. In complex manufacturing environments one often uses serial-group configurations instead—as shown in Figure 1—to optimize the throughput [8]. This design provides a group of machines for every stage, and lots can take different paths while being processed. Note that in semiconductor manufacturing machines can typically be used in several stages. The propagation of variation in serial-group lines is analyzed in [9].

Assume each lot passes K production stages and in each stage it can be handled by one of n_j machines. Computer-integrated production provides records of the so-called *lot history*—the information which machines has handled which batch—and the results of final measurements. We link these different types of data to locate flawed machines, and approach the root cause detection by analyzing the assembly line as a whole. In contrast, previous methods model specific processes to detect errors, for example by relating the yield to process parameters [7], [6]. Process models can be used to monitor unmeasurable parameters [4] and thus for fast error detection. However, when process flaws are rare, it is hard to use such models for troubleshooting [5]. Our approach directly relates observed errors to possible root causes without attempting to model the underlying processes. The lot history can be coded as a vector

$$\mathbf{x} = (x_1, x_2, \dots x_D) \tag{1}$$

in many different ways. A straightforward coding is to use one dimension for each stage and to assign some number to each machine. This coding, however, introduces an order within the sets of machines which does not reflect the structure of the underlying process. Instead, we construct a vector with one dimension for each possible combination of stage and machine by setting the corresponding value to one if the lot has passed the combination and to zero otherwise. Accordingly, the dimension D of this input vector is

$$D = \sum_{j=1}^{K} n_j \ , \tag{2}$$

and each dimension of the input vectors $\mathbf{x}$ directly corresponds to a certain machine being used in one particular stage. If we expect that a combination of some machines—no matter in what stages—causes the observed failures, we can add such features to the input vectors. We would have one additional dimension per machine, which is set to one when the machines was used in any process and to zero otherwise.

The term *feature* or variable refers to single entries $x_1, x_2, \dots x_D$ which make up the lot history. Note that the dimension of the constructed input vector is quite large—in our examples from mass production we dealt with approximately $2\,000$ features. However, a great advantage of this way of coding the lot history is that each feature directly corresponds to a localization in the process flow as shown in Figure 1. Features that are chosen by some extractor have a direct interpretation in terms of what happens during production.

After passing the production line, all units are tested in a final inspection and it is only at that point that one knows for sure whether they meet the quality requirements. A small fraction of units always fails in processes as complex as those used for semiconductor devices. Dysfunctions in the production line may be detected through an exceptionally low yield or the appearance of certain errors. We use those characteristics to identify "regular" ($y_\ell = 0$) and "conspicuous" ($y_\ell = 1$) lots and collect the grouping of all N lots in a vector

$$\mathbf{y} = (y_1, y_2, \dots y_N) \in \{0, 1\}^N \ . \tag{3}$$

How regularity is to be defined strongly depends on the type of product. For semiconductor devices the occurrence of patterns on wafers is an excellent criterion (see Section IV-A).

III. Feature Selection

In recent years Feature Selection has received a lot of interest, since many real-world problems involve a large number of variables. In prominent research areas such as text categorization [10] or bioinformatics [2] one is faced with up to a hundred thousand features. The aim of selecting informative variables—or combinations of those—can be to construct powerful predictors or to improve the understanding of the data itself. For reviews on Feature Selection please refer to [11] and [12]. All approaches for Feature Selection can be divided into the following two parts:

- Rank subsets of features by some measure for the information about the target.
- Choose a subset of features, which obtains a good ranking score, and use it to train a predictor.

When combining some kind of Feature Selection with a predictor (such as SVMs, Neural Nets or Decision Trees) one can think of three possible constructions (see [13] and [14]). *Wrapper* approaches use the predictions of the induction algorithm itself to do the selection, while *filter* approaches do Feature Selection separately from induction. *Embedded methods* make use of the complete specification of a certain predictor rather than using it as an exchangeable "black box".

Wrappers and embedded methods optimize the predictive performance of an induction algorithm, but they can be computationally very demanding. As we use Feature Selection to identify irregularities in the production line, we are not really interested in predicting the products' quality, but a sensible feature ranking. Therefore we choose the computationally simpler filter methods and use the classifier only after the ranking to validate the results of the ranking procedure.

When adapting Feature Selection methods for the presented setup, we have to take into account the following peculiarity of our data: While most methods assume a balanced ratio of target labels, a serial production is usually not error-prone and we expect only a small fraction of examples to show flaws. Therefore the definition of the classifier has to reflect that target values of zero dominate.

A. Feature ranking

As pointed out, we choose a filter approach to rank and extract informative features. Filters estimate the information content of features independently of a classifier, and are therefore computationally attractive and comparably easy to implement.

A standard criterion to rank single features is the *Fisher Score* (FS)

$$\mathrm{FS}(j) = \frac{\left(\mu_{j(1)} - \mu_{j(0)}\right)^2}{\sigma^2_{j(1)} + \sigma^2_{j(0)}} \tag{4}$$

with $\mu_{j(1/0)}$ denoting the mean and $\sigma^2_{j(1/0)}$ the variance of feature j for all examples with target 1/0. The Fisher Score can be calculated for both, binary and continuous features.

Another univariate ranking criterion is the *Mutual Information* (MI) between single features x and the target y

$$\mathrm{I}(x;y) = \mathrm{H}(x) + \mathrm{H}(y) - \mathrm{H}(x,y) \tag{5}$$

—H denoting the entropy—which we estimate by its empirical counterpart. We can normalize the MI by dividing it by the entropy $\mathrm{H}(y)$, making values range between zero (completely uninformative) and one (complete dependence). As the target and input vectors are binary, we can compute all quantities without further approximations.

The univariate measures, FS and MI, have two major drawbacks. First, correlations between features cannot be captured and thus several equivalent features might be selected. Second, effects which are caused by a combination of features can only be found via their independent contributions. A greedy search as proposed in [15], called Conditional Mutual Information (CMI) criterion, solves both problems. Based on the MI criterion, the CMI constructs a set of features by adding features which are informative in combination with the previous selection:

1) Choose the first feature to be

$$\nu(1) = \underset{n\in\{1\ldots K\}}{\operatorname{argmax}} \left\{ \mathrm{I}(y;x_n) \right\} . \tag{6}$$

The first selected feature is the one that carries maximal information about the target.

2) Choose the $l+1^{\text{st}}$ feature to be

$$\nu(l+1) = \underset{n\in\{1\ldots K\}}{\operatorname{argmax}} \left\{ I(y;x_n|x_{\nu(1)}\ldots x_{\nu(l)}) \right\} . \tag{7}$$

The feature $x_{\nu(l+1)}$, which we add to the subset $x_{\nu(1)}\ldots x_{\nu(l)}$, is the one with maximal MI with the target, given the set which is already selected. For reasonably sized data sets this quantity cannot be evaluated for computational reasons and we resort to a greedy approximation, which replaces the argument by the minimal MI given any feature from the subset:

$$\nu(l+1) \approx \underset{n\in 1\ldots K}{\operatorname{argmax}} \{ \min_{m\le l} [I(y;x_n|x_{\nu(m)}] \} . \tag{8}$$

B. Validation

In a setup which is comparable to ours, [2] compares various classifiers in combination with two Feature Selection criteria. It is shown that different classifiers vary greatly in their predictive performance. In the end we are interested in how to find subsets of features which are related to the target, rather than in the absolute predictive performance of certain classifiers. Hence we restrict ourselves to the widely used C-SVM classifier and compare different ranking criteria.

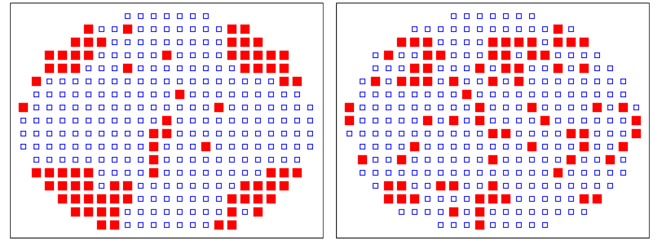

Fig. 2. Artificial examples for pass/fail wafermaps showing areas of high failure rates that form a suspicious cloverleaf-like pattern (left) and randomly distributed errors for comparison (right).

The weighted C-SVM uses the loss function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\left(c_1 \sum_{i;y_i=1} \xi_i + c_0 \sum_{i;y_i=0} \xi_i\right), \tag{9}$$

where we choose the weighting parameters

$$c_1 = \frac{\#[y=0]}{\#[y]} \quad \text{and} \quad c_0 = \frac{\#[y=1]}{\#[y]} . \tag{10}$$

The loss function directly corresponds to unbalanced classes, by making the loss for incorrect classification of the two groups independent of their respective sizes.

We choose a Gaussian kernel function $k(\mathbf{u},\mathbf{v}) = \exp\{-\gamma|\mathbf{u}-\mathbf{v}|^2\}$ and do an extra 10 fold Cross Validation (CV) on each training set to determine the parameters γ and C using *LIBSVM* [16]. There is a vast literature on SVMs and we refer the reader to e.g. [17] or [18] for details.

Corresponding to the loss function in (10) we use the "balanced score" by [2] to rate predictions:

$$\mathrm{Score}(y,\hat{y}) = \frac{1}{2}\left[\frac{\#\{y=\hat{y}=1\}}{\#\{y=1\}} + \frac{\#\{y=\hat{y}=0\}}{\#\{y=0\}}\right] . \tag{11}$$

The score function uses a weighting that assigns equal mass to correct classification of both groups, as we divide the number of correct predictions by the number of test cases which actually belong to the corresponding class. If, say, one class is perfectly predicted while nothing is known about the other, we obtain a score of roughly $\frac{3}{4}$. Our test scores are obtained using a 10-fold CV scheme. Note that we exclude the test set both during the ranking and the training of the classifier to avoid over-fitting.

IV. Evaluation and Practical Use

In the following we describe in detail how the above methods can be used to locate flaws, and we assess their performance on real-world data sets. We present two examples,

Name	# stages	# machines	# features	# lots tot.	# class 1
PC1	403	357	1896	112	41
PC2	355	331	1758	98	28
YC1	157	142	779	870	11
YC2	339	391	2104	261	37

TABLE I

Used data sets from mass production of different products.

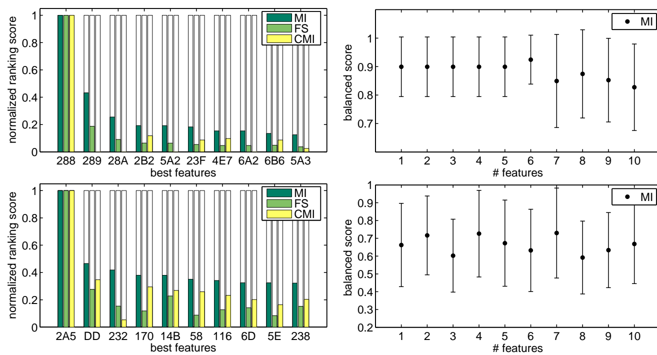

Fig. 3. Scores for data sets PC2 (top) and YC1 (bottom). The feature ranking (left) measures how well combinations of stages and machines correlate with the grouping into good and flawed lots. For these data sets all measures extract very clearly that one combination is responsible for the flaws, #288 for PC2 and #2A5 for YC1. The validation scores (right) show that the grouping could be understood to great extend with the top ranked feature only.

where we identify conspicuous lots using occurrences of patterns on wafers, and two where we use unusually low yield. Table I summarizes details of the data sets. All examples represent data which have been recorded during mass production in our plant.

A. Finding conspicuous lots

Recall the notation introduced in Section II, where we collected the lot history in a vector $\mathbf{x}$ (1) and a grouping of N conspicuous and flawless lots in a vector $\mathbf{y}$ (3). While the lot history is automatically recorded, the grouping is done on the basis of final measurements.

For semiconductor products, patterns which appear on so called *wafermaps* are a good indicator for specific flaws in the production line. Semiconductor products are fabricated and tested as batches on discs, called wafers, and wafermaps are obtained by arranging test results of single units according to the position they have on those wafers—see Figure 2 for exemplary maps. The patterns may be identified by an operator or may be found by unsupervised data mining methods, which have proven to be very effective in automatically grouping wafermaps in the database [19]. Data sets PC1 and PC2 are examples for a pattern-based grouping. A classification based on patterns is an excellent guidance for our Feature Selection approach, as it is highly improbable for a pattern to show up by pure chance. In the data sets YC1 and YC2 we defined lots to be conspicuous when a threshhold for the total error rate was exceeded.

B. Feature ranking

The feature ranking, which we introduced in III-A, establishes a correlation between the grouping $\mathbf{y}$ and the process histories $\mathbf{x}$ of the lots. In a representation as in Figure 3 we plot the correlation scores between the occurrence of errors and the use of machines for different processes. Note that we only show the ten features which obtain the highest scores.

In both data sets, PC2 and YC1, all measures consistently rank one feature far higher than any other (#288 and #2A5 respectively), indicating a large correlation with the defect.

On the PC2 data it is interesting to compare different measures. The features #289 and #28A are ranked 2nd and 3rd by FS and MI, while CMI finds them to be highly correlated to the first feature #288. A closer look at the production chain shows that all top three features belong to the same production stage and represent the only machines to choose from in this stage. They are strongly correlated as in this stage only one of these machines can be used at a time. The same gap can be observed in the PC1 data (Figure 4), where combination #404 is found to be completely uninformative if one knows whether the lot has passed #675. CMI finds small sets of informative features, ignoring redundant information such as feature #404. As we use a combination of all ranking measures in our plots, we obtain a good indicator for dependencies between the features. Errors, which are caused by a coinciding use of several machines, are found reliably by CMI as features are considered jointly.

Data set PC2 represents a recent problem. Our maintenance team could be directed to the identified production stage, in which the machines #288, #289 and #28A were used (see figure 3), and verified machine #289 to have caused the error. YC1 is a historic data set where the cause had been found beforehand in a time-consuming manual check. The feature #2A5, which was found by our ranking on only 11 conspicuous lots, correctly represents the machine known to have caused the error.

C. Validation

Feature ranking alone can already give a prognosis on where to find flaws in the production line. However, as we are dealing with as many as 2 000 features and as few as 11 flawed lots, the results need to be validated to avoid over-fitting. The validation, which we have described in Section III-B, uses a cross validation scheme in which the SVM classifier is trained on the selected features. Based on the predictions for the independent test sets we calculate the score defined by (11).

For the two data sets PC2 and YC1 the validation score strongly affirms the ranking. On PC2 we obtain a good score on the best feature #288 only. The score is as high as 90% and thus confirms that most flaws can be explained by the corresponding combination of machine and stage. In this example machine #289 in the same stage produces the error, and using feature #288 the SVM predicts flaws if the lot has *not* passed machine #288. For the YC1 data we observe lower

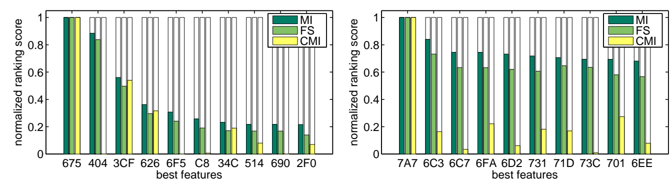

Fig. 4. Ranking scores for data sets PC1 (left) and YC2 (right). For PC1 three features obtain relatively high FS and MI scores in comparison to the rest, CMI finds the second feature to be completely dependent on the first. In the YC2 data the ranking gives poor separation between the features.

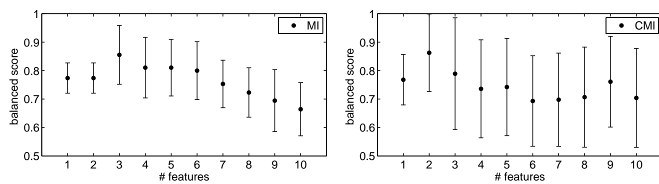

Fig. 5. Rating for PC1. The plot shows mean CV-rankings for the MI (left) and the CMI (right) filter. Using the best three features given by MI or the best two features given by CMI a mean score of 0.87 out of a possible 1 is obtained. We omit the FS as it gives results equivalent to the MI criterion.

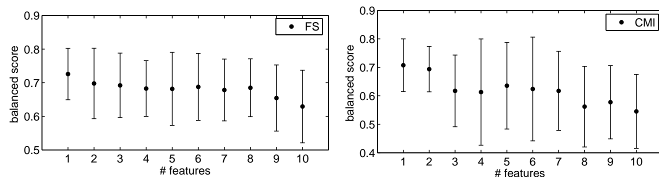

Fig. 6. Rating for data set YC2. Shown are the ratings using the FS and the CMI filter. The univariate MI leads to results very similar to that of the FS, the rating leads to a mean score of approximately 72% on the feature that is ranked highest.

scores with high variance, as we only have 11 affected lots in 870 examples.

While the ranking shows only one prominent combination in the above cases, the separation between the features is not as clear for the other two data sets (see Figure 4).
Three features can be identified in PC1 to be strongly correlated to the error. The CMI measure finds #675 and #404 to be mutually redundant, while #3CF is independent of the first two. The validation results in Figure 5 give more insight, showing that the combinations #675 and #3CF together lead to 87% of the maximal score. Therefore we are given high evidence that the error is caused by a combination of the machines behind the features #675 and #3CF, or #404 and #3CF. We were able to validate this result with a manual check in the shop floor.

On the YC2-data neither feature ranking nor validation gives a clear picture. The maximal rating shows a relatively low mean score of 72% (see Figure 6). The data alone do apparently not contain enough information to identify the cause of the irregularity clearly in this example.

While the ranking provides a fast estimate for the impact of the features, by training the SVM and testing its predictions in the validation step, we can capture all possible interactions and check how much of the grouping could really be understood. Especially when we have just a small number of lots in the database, we can only be sure of avoiding over-fitting if we test predictions in a CV scheme: the feature ranking alone might point to a wrong trace.

V. Conclusion

The aim of this work was to construct a procedure to reliably locate dysfunctions in complex production lines based on few observed irregular lots. The approach uses existing data and can be used where we lack extra measurements that are required by traditional statistical process control. In contrast to previous works, which focus on single processes, our method embraces the manufacturing chain as a whole and requires only the lot history and final test results. We use four data sets, which represent irregularities in the mass production of semiconductor devices, to explain how we have applied the method in practice and to assess its performance. Our data sets include two examples where errors are reflected by patterns on wafermaps. For such cases our method can be combined with unsupervised pattern recognition to build an automatic control mechanism on top of existing process control.

The proposed scheme uses different feature ranking methods in combination with a validation based on SVM classification. The univariate measures, FS and MI, rank the features independently, and enable us to find correlations between single entries in the lot history and the occurence of irregularities. However, these simple measures cannot capture dependencies between the features, and errors which are caused by the interplay of several processes might not be detected. CMI is constructed to take such interdependencies between features into account. By combining all measures we can thus identify dependencies between the features and locate errors which are caused by a coaction of several processes. The SVM classifier, which we use in a cross validation scheme to validate the ranking, guarantees reliable results. We avoid the phenomenon of over-fitting, often observed in setups with a small ratio of observations and features, and therefore our method requires only a small number of observations.

The results on our benchmark data sets show that the proposed scheme is a powerful tool to complete the existing process control in semiconductor manufacturing. We believe that the results also hold for other serial–group assembly lines, where the lot history is recorded in a database. The proposed ranking and validation methods are relatively easy to implement and give a fast overview on the impact of different machines or stages.

Acknowledgment

The authors thank T.N. Lal, M. Bensch, and M. Schröder for helpful discussions.

References

[1] G. Taguchi, R. Jugulum, and S. Taguchi, *Computer-Based Robust Engineering*. ASQ Quality Press, 2004.
[2] J. Weston, F. Pérez-Cruz, O. Bousquet, O. Chapelle, A. Elisseeff, and B. Schölkopf, "Feature selection and transduction for prediction of molecular bioactivity for drug design," *Bioinformatics*, vol. 19, no. 6, pp. 764–771, 2003.
[3] R. Isermann, "Model-based fault-detection and diagnosis—status and applications," *Annual Reviews in Control*, vol. 29, no. 1, pp. 71–85, 2005.
[4] ——, "Process fault detection based on modeling and estimation methods—a survey," *Automatica*, vol. 20, no. 4, pp. 387–404, 1984.
[5] J. M. Agosta and T. Gardos, "Bayes network "smart" diagnostics," *Intel Technology Journal*, vol. 8, no. 4, pp. 361–372, 2004.
[6] D. Braha and A. Shmilovici, "Data mining for improving a cleaning process in the semiconductor industry," *IEEE Transactions on Semiconductor Manufacturing*, vol. 15, no. 1, pp. 91–101, 2002.
[7] M. Bensch, M. Schröder, M. Bogdan, and W. Rosenstiel, "Feature selection for high-dimensional industrial data," in *ESANN*, 2005.
[8] R. Webbink and S. Hu, "Automated generation of assembly system-design solutions," *IEEE Transactions on Automation Science and Engineering*, vol. 2, no. 1, pp. 32–39, 2005.

[9] Q. Huang and J. Shi, "Stream of variation modeling and analysis of serial-parallel multistage manufacturing systems," *J. of Manufacturing Science and Engineering*, vol. 126, no. 3, pp. 611–618, 2004.
[10] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proc. of International Conference on Machine Learning*, 1997, pp. 412–420.
[11] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, Eds., *Feature extraction, foundations and Applications*. Springer, to appear.
[12] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.
[13] R. Kohavi and G. John, *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998, ch. The Wrapper Approach, pp. 33–50.
[14] T. Lal, O. Chapelle, J. Weston, and A. Elisseeff, "Embeded methods," in *Feature extraction, foundations and Applications*, I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, Eds. Springer, to appear.
[15] F. Fleuret, "Fast binary feature selection with conditional mutual information," *JMLR*, vol. 5, pp. 1531–1555, 11 2004.
[16] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
[17] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.
[18] B. Schölkopf and J. Smola, *Lerning with Kernels*. MIT Press, 2002.
[19] G. D. Nicolao, E. Pasquinetti, G. Miraglia, and F. Piccinini, "Unsupervised spatial pattern classification of electrical failures in semiconductor manufacturing," IAPR - TC3, Florence, 9 2003.