Robust, Non-Redundant Feature Selection for Yield Analysis in Semiconductor Manufacturing

Eric St. Pierre and Eugene Tuv

Intel Corporation, 4500 S Dobson Road,
Chandler, AZ 85286
{eric.r.st.pierre,eugene.tuv}@intel.com

Abstract. Thousands of variables are measured in line during the manufacture of central processing units (cpus). Once the manufacturing process is complete, each chip undergoes a series of tests for functionality that determine the yield of the manufacturing process. Traditional statistical methods such as ANOVA have been used for many years to find relationships between end of line yield and in line variables that can be used to sustain and improve process yield. However, a large increase in the number of variables being measured in line due to modern manufacturing trends has overwhelmed the capability of traditional methods. A filter is needed between the tens of thousands of variables in the database and the traditional methods. In this paper, we propose using true multivariate feature selection capable of dealing with complex, mixed typed data sets as an initial step in yield analysis to reduce the number of variables that receive additional investigation using traditional methods. We demonstrate this approach on a historical data set with over 30,000 variables and successfully isolate the cause of a specific yield problem.

Keywords: feature selection, yield analysis and improvement, random forest, gradient boosting.

1 Introduction

Modern semiconductor manufacturing trends have led to a large increase in the number of variables available with which to diagnose process yield. The increase has been primarily due to:

- System on a Chip(SoC): More components of a computer being built into a single chip. For example, rather than produce a cpu and a graphics chip separately, they are combined into a single chip. This usually adds to the total number of process steps required to build the product and the number of process control variables measured.
- Fault Detection and Classification (FDC): Almost all equipment used in producing cpus now has the capability to record information on how the tool is performing. For example, an etcher may record pressure, temperature, power consumption, and the states of various valves while it is processing. FDC alone has doubled the number of variables that can be potentially measured in line.

P. Perner (Ed.): ICDM 2011, LNAI 6870, pp. 204–217, 2011.

- Subentity Traceability: Most tools used in semiconductor factories (fabs) now record the path taken through them. For example, an etcher (entity) may have two chambers (subentities) being used in parallel, so it is no longer sufficient to simply record "Etcher1", now we must also record chamber as well, "ChamberA", in order to have a complete account of what entities processed specific material. Some tools have as many as 15 subentities.

The result is that the capability of the traditionally used ANOVA and graphical methods has been overwhelmed. The total number of variables measured on production material as it travels through the factory is approaching 50,000. It isn't plausible to try and mine such a large number of variables using only scatter plots and F-tests. However, the traditional methods have served well for many years and business processes and expertise have become well developed as to what statistical tests and supporting graphics are required to take a tool down from production or make a change to the manufacturing process. A filter is needed to reduce the number of variables that receive additional investigation with traditional methods.

Mining is typically being done to decrease the cost of manufacture by increasing the yield, performance, and reliability of the chips. Higher yields result in lower cost by reducing the amount of equipment needed to be purchased in order for the factory to meet a specific capacity and also lowers the amounts of consumables (chemicals and gases) needed to be used to meet the required output of good chips. Hundreds of chips are manufactured on 300mm diameter silicon wafers which are processed in batches of 25 called lots. Each lot travels through the line in a lot box, which has 25 slots. Once lots reach end of line, many tests of functionality, performance, and reliability are performed. Fabs are constantly looking for relationships between end of line yield and in line variables that can be used to maintain and improve the process yield.

1.1 History of Analysis Approach

Early in the 1990s, data storage and analysis software were Vax based. At this time, there were approximately 500 total variables measured in line. These variables were mostly SPC measurements (e.g. physical thickness of a deposited film), run card (entity that processed the lot and out date at each operation), and electrical test (e.g. electrical thickness of a film). It was possible to have nightly batch jobs run on the Vax which produced the output required for the analysis seen in Figure 1.

Early 2000s saw the switch to Oracle databases from Vax and increased data storage capacity. Analysis was now done on desktops and laptops. Around 2005, automation capability was added that allowed subentity traceability (the complete path of a wafer through a process tool at each operation). This dramatically increased the amount of information available in the run card. No longer did we only know that the lot went through Tool1 and the date and time it did so, now we also knew each subentity that it visited inside the tool (e.g. what chambers, tracks, chucks, load locks, wafer transfer robots, and so forth) and the date

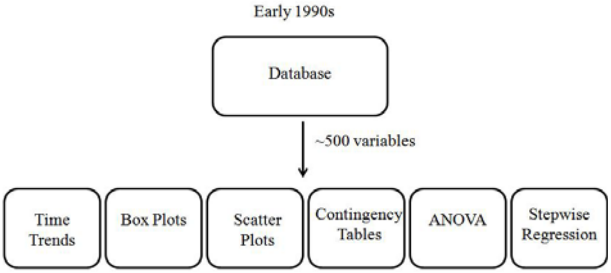


Fig. 1. Analysis in early 1990s

and times for each. Run cards went from a few hundred columns, to over two thousand. Also at this time, more operations were required to build the chip as graphics and other functionality was added to the cpu (called SoC). About 2 years later, FDC data was added to each tool set in the factory. The combination of SoC, FDC, and subentity traceability had increased the number of variables available into the tens of thousands. The analysis approach in Figure 1 was no longer practical.

The reaction to the large increase in variables led to the entity based approach shown in Figure 2. This approach emphasizes finding a suspect process tool, usually done by comparing with its peers at the same operation using contingency tables for categorical responses and ANOVA for continuous ones. If a suspect tool can be found, it acts as a filter, as now only variables associated with that tool and process operation need to be investigated. The approach has the advantage of reducing the initial search back to around 500 hundred variables (process operations), however, there are drawbacks. Historically, only 53% of yield issues can be traced to a single entity, so too often, a single entity at a single operation may not be able to be isolated. Several entities across several process operations may be suspect, so many variables may still need to be reviewed. Or, the yield problem may not be caused by a single tool. For example, it may be a sub optimal recipe being run across all tools at an operation. Or, there may be operations where only one tool is operating, so there are no peer tools to compare to. Also, while yield sustaining is typically about finding and fixing problem process tools, yield improvement is often a response surface problem where the goal is finding settings of inline parameters which provide higher yield. An entity filter approach is less useful for yield improvement. And finally, if a single tool at an operation can be found, there will still be 100 to 200 variables to search that are associated with that operation.

The approach we propose, shown in Figure 3, is to use a feature selection algorithm as the initial step. In this way, not only can process tools be searched during the initial step, but also every other variable in the data set. We have used this approach in production for the past 2 years, analyzing over 20 data sets ranging in size from 10,000 to 50,000 variables (in line measurements) and 1,000 to 10,000 rows (wafers). The data sets usually have more than one target variable

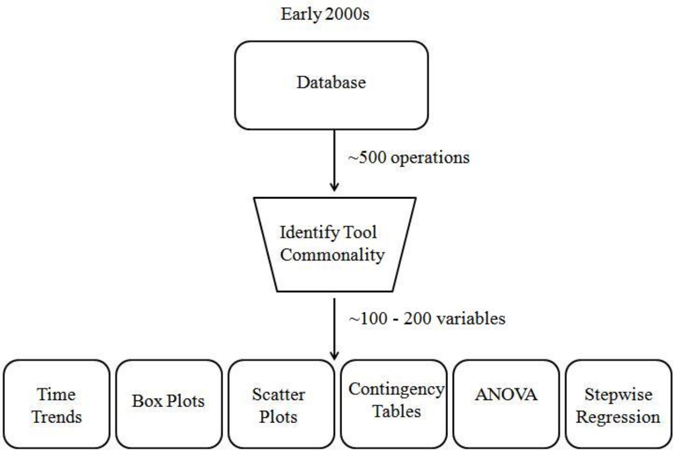


Fig. 2. Analysis in early 2000s

(unique yield problems), ranging between 2 to 12 target variables. The algorithm generally has reduced the number of variables requiring further investigation to 50 or less (per target), and greater reductions to between 5 and 10 (per target) are not uncommon. The advantage of this approach is that it searches more variables in less time and returns fewer and more likely to be relevant variables than does the entity based filter approach. It is also works equally well for yield sustaining (finding problems tools) as it does for yield improvement (find the best settings of relevant variables). There are other advantages as well, such as how it handles mixed variable types, missing data, and hierarchial variables that will be discussed further in the next section.

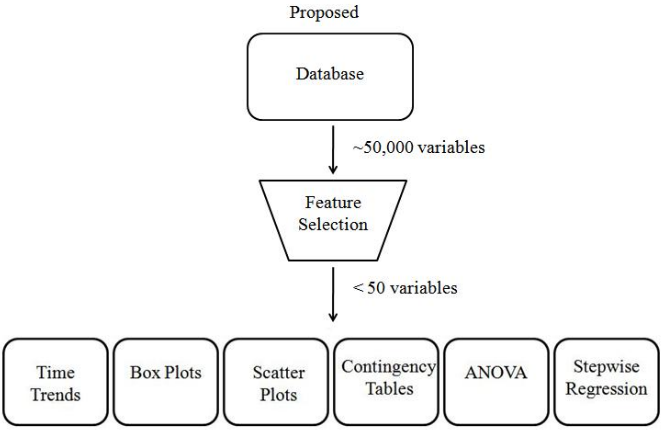


Fig. 3. Analysis with Feature Selection

2 Feature Selection

In theory, our goal in using a feature selection algorithm is to find a Markov Boundary (a minimum length list of only strongly relevant as well as non redundant, weakly relevant variables) for further investigation. But in practice, we are satisfied with a list of most of the relevant variables (strongly or weakly) with some redundancy. This is because we are using feature selection as an initial step, not to provide the final answer. We are looking for good leads to follow and expect that in the process of investigating these leads, we will learn more that will influence our search for the root cause of the yield problem. For more discussion of relevancy and redundancy please see [1],[2],[3].

The properties of our data sets make it necessary to have a feature selection algorithm that can successfully handle the following challenges:

- Mixed Data Types: Our data sets contain categorical variables, such as the recipe name that a lot received, as well as continuous variables such as the length of a transistor channel.
- Missing Data: Sometimes due to automation errors, but usually due to sampling. For example, only a few wafers in a lot might be sampled to measure the thickness of a deposited layer.
- Nested Variables: The most common way wafers are selected for analysis is to simply take all wafers that reached end of line in the past 30 days. The result of this method of selecting material for analysis is that we also now have roughly 30 days worth of material through all fab operations, thus process tools. This influences the way we want to search entities and subentities since time is nested within tool. By searching not only for a bad tool, but also the affected time period, we significantly increase our ability to find the correct signal over, say, a simple count of bad lots run on a tool over the entire 30 days.
- Multivariate: Interactions are not uncommon in our manufacturing process. For example, a lithography tool than tends to print metal lines wider than it peers combined with an etcher that tends to etch the line less than its peers is a combination that could push material out of spec and cause it to fail at end of line.
- Outliers: Not only caused by unusual material, but also because it is unfortunately common for metrology equipment to write nonsensical values to the database under certain conditions. For example, writing ”999” to the database when an upper limit for a variable is reached, or writing ”-100” for the wafer count through a tool when the actual wafer count can't be determined.
- Nonlinear Relationships: Very common. Due to the large number of variables, it isn't practical to try and determine a transformation that works for all of them unless doing something very basic like converting to ranks.
- Target Misclassification: Whether a specific wafer has the yield issue of interest or not is a judgement call and there will be inaccuracy in the label assigned to a wafer (e.g. GFA versus No GFA).

- Large Fraction Irrelevant: Our data sets may contain 50,000 variables, of which perhaps 5 are related to a specific yield problem.
- Multicolinearity: A large percentage of the variables in our data sets are highly correlated. For example, the time through the previous process operation is highly correlated with the time through the next process operation.

Algorithm 1. Ensemble-Based Feature Selection, Classification

```
1.  set $\Phi \leftarrow \{\}$; $G_k(F) = 0, W_k = 0$
2.  for $k = 1, \ldots, K$ do
3.    set $V = 0$.
4.    for $r = 1, \ldots, R$ do
        $\{Z_1, \ldots, Z_M\} \leftarrow$ permute$\{X_1, \ldots, X_M\}$
        set $F \leftarrow X \cup \{Z_1, \ldots, Z_M\}$
        Compute class proportion $p_k(x) = exp(G_k(x)) / \sum_{l=1}^{K} exp(G_l(x))$
        Compute pseudo-residuals $Y_i^k = I(Y_i = k) - p_k(x_i)$
        $\mathbf{V}_{r.} = \mathbf{V}_{r.} + g_I(F, Y^k)$;
      endfor
5.    Element wise $\mathbf{v} = Percentile_{1-\alpha}(\mathbf{V}[\cdot, M+1, \ldots, 2M])$
6.    Set $\hat{\Phi}_k$ to those $\{X_k\}$ for which $\mathbf{V}_{.k} > \mathbf{v}$
      with specified paired t-test significance (0.05)
7.    Set $\hat{\Phi}_k = RemoveMasked(\hat{\Phi}_k, W_k + g_I(F, Y^k))$
8.    $\Phi \leftarrow \Phi \cup \hat{\Phi}_k$;
      for $k = 1, ..., K$ do
9.      $G_k(F) = G_k(F) + g_Y(\hat{\Phi}_k, Y^k)$
10.     $W_k(\hat{\Phi}_k) = W_k(\hat{\Phi}_k) + g_I(\hat{\Phi}_k, Y^k)$
      endfor
    endfor
11. If $\hat{\Phi}_k$ for all $k = 1, \ldots, K$ is empty, then quit.
12. Go to 2.
```

We will review the general concepts of the algorithm that we use, but for a full discussion of it, please see [4]. In general, the algorithm eliminates variables whose variable importance is determined to not be significantly greater than the importance of a random variable of the same distribution. This removes irrelevant variables. To eliminate redundant variables, the algorithm determines if masking is significant between two variables by comparing to that of a known random variable of the same distribution. If masking is significant, the variable with the highest importance is kept and the variables with significant masking scores to it are eliminated.

The most important concept of the algorithm is that it uses random contrasts with which to create a threshold to determine if a variable is important enough to keep. For each original variable, X_i, in the data set, a random contrast variable Z_i is created by permuting X_i. That way, there is a set of contrast variables that we know are from the same distribution as the original variables and should have no relationship with our target variable Y (since Z_i is a 'shuffled' X_i). A random forest [5] is then built using original and random contrast variables and the variable importance is calculated for all variables. The 90^{th} percentile (say) of the random contrasts variable importances is calculated. This process is replicated R times. For each replicate, we keep the variable importance of each original variable, and the 90^{th} percentile (say) of the importance across the contrasts. Now we have a distribution with which to calculate the mean for each original variables importance and a distribution with which to calculate the mean for the 90^{th} percentile of the random contrasts importance. A t-test is then conducted for each original variable with the null hypothesis that the mean variable importance for X_i is less than or equal to the mean of the 90^{th} percentile (say) of the variable importances of the contrast variables. Only original variables for which the null hypothesis is rejected are kept. This removes irrelevant variables. Please see Algorithm 1 for reference, with notation defined in Table 1.

To remove redundant variables, a very similar process is used. A modified surrogate score (called a masking score) is calculated between all pairs of variables. Masking scores between original variables and contrast variables are used for the same purpose as random contrast variable importances were used previously. That is, an upper percentile (say 75^{th} is calculated and kept for each replicate and after several replications a mean is calculated and this is the threshold against which all masking scores between original variables is compared to using a t-test. If the null hypothesis that the masking score between two original variables is less than or equal to the mean of the 75^{th} (say) is rejected, then that masking score is kept. One notable difference is that a gradient boosted tree (GBT) [6] is used to calculate surrogate scores instead of a random forest since a GBT tests each variable in each node, so richer, more effective masking information is obtained. To remove redundancy, the X_i with the highest importance score is kept and every other X_i with a significant masking score to it is eliminated. Please see Algorithm 2 for reference, with notation defined in Table 1. At this point, a GBT model is built using original variables that were kept, residuals are calculated to remove the affect of the variables found in this iteration, and the entire process begins over on the residuals with all original variables included for possible selection again. In this way, the algorithm can find weaker variables. The process stops when an iteration doesn't produce any important variables (or a predetermined upper bound).

Since the algorithm is based on ensembles of trees (random forests and gradient boosted trees), it easily handles mixed data types, missing data, interactions, outliers, and nonlinear relationships [7]. Nested variables are handled by an interval search algorithm. To deal with the large fraction of irrelevant variables, we sample between 5 to 10 times more variables per split in the random forest than the usual $\sqrt{p}$ rule of thumb given in [5], although a system of learned weights like that proposed in [9] could also be used. Multicolinearity of the output is reduced by using Algorithm 2. We have found this feature selection algorithm to be very effective and will demonstrate our approach to using it in the next section.

Algorithm 2. RemoveMasked(F,W)

```
1   Let $m = |F|$.
2.  for $r = 1, \ldots, R$ do
3.      $\{Z_1, \ldots, Z_m\} \leftarrow$ permute$\{X_1, \ldots, X_m\}$
4.      set $F_P \leftarrow F \cup \{Z_1, \ldots, Z_m\}$
5.      Build GBT model $G_r = GBT(F_P)$.
6.      Calculate masking matrix $M^r = M(G_r)$ ($2m \times 2m$ matrix).
    endfor
7.  Set $M^r_{i,\alpha_m} = Percentile_{1-\alpha_m}(M^r[i, m+1, \ldots, 2m]),\ r = 1, \ldots, R$
8.  Set $M^*_{ij} = 1$ for those $i, j = 1 \ldots m$ for which $M^r_{ij} > M^r_{i,\alpha_m},\ r = 1, \ldots, R$
    with specified paired t-test significance (0.05), otherwise set $M^*_{ij} = 0$
9.  Set $L = F, L^* = \{\}$.
10. Move $X_i \in L$ with $i = \text{argmax}_i W_i$ to $L^*$.
11. Remove all $X_j \in L$ from $L$, for which $M^*_{ij} = 1$.
12. Return to step 10 if $L \neq \{\}$.
```

3 Analysis

Our data set has 31,600 variables and 9,300 rows (wafers). The variables are approximately one third subentity, one third FDC, and one third SPC, electrical test, and PM counters (wafer count since last preventative maintenance was performed). There are several target variables in the data set, but our analysis will focus on one specific yield issue, shown in Figure 4. Failing die are shown in black and the yield problem, known as a gross failure area (GFA), has a dark band of failing die in a single row near the top of the wafer. Wafers which have the GFA are classified as "1" and wafers with out the GFA are classified as "0". Our goal is to sift through the 31,600 variables and find a relevant, non redundant set of predictors for whether a wafer will have the GFA or not. Of the 9,300 wafers in the data set, 2,200 are classified as "1". For more on how we search large numbers of wafer maps for common spatial patterns and classify them, please see [8].

The entity based filter approach, shown previously in Figure 2, does not work for this GFA. No individual process tool can be isolated as the cause of this problem. To demonstrate this, we will look at the operation and tool set which was identified as most likely to be at fault (using the methods in [8]). Figure 5 is a time trend for all tools running at this operation. The green dots are lots with out the GFA (classified as "0"), the red dots are lots with the GFA (classified as "1"). The graph shows the 5 tools at a specific operation across a time span of about 1 month. A lot can only have a y axis value of "1" or "0", but to make the graph friendly to the eye, jitter has been added. Figure 6 shows box plots of similarity (a measure of how much a wafer looks like the GFA wafers) for each tool. Tool4 ran 63% of the total number of bad lots and 31% of the material it ran is affected. Tool4 ran 50% of the lots, so having 63% of the bad lots is not that unusual, also, several other tools at the operation ran a significant amount of the

Table 1. Notation in Algorithms 1-3

K	Number of classes (if classification problem)
X	set of original variables
Y	target variable
M	Number of variables
R	Number of replicates for t-test
α	quantile used for variable importance estimation
α_m	quantile used for variable masking estimation
Z	permuted versions of X
W	cumulative variable importance vector.
W_k	cumulative variable importance vector for k-th class in classification.
Φ	set of important variables
F	current working set of variables
$\mathbf{V}$	variable importance matrix $(R \times 2M)$
$\mathbf{V}_{r.}$	rth row of variable importance matrix $\mathbf{V}$, $r = 1 \dots R$
$\mathbf{V}_{.j}$	jth column of matrix $\mathbf{V}$
$g_I(F,Y)$	function that trains an ensemble of L trees based on variables F and target Y, and returns a row vector of importance for each variable in F
$g_Y(F,Y)$	function that trains an ensemble based on variables F and target Y, and returns a prediction of Y
$G_k(F)$	current predictions for log-odds of k-th class
$GBT(F)$	GBT model built on variable set F
$M(G)$	Masking measure matrix calculated from model G
M^k	Masking matrix for k-th GBT ensemble G_t.
M^*	Masking flags matrix

affected material. Combining this information, it isn't convincing that Tool4 is the cause of the GFA. Since even the best candidate to use as an entity filter does not look likely to be the cause, an analyst would then revert to the approach shown in Figure 1, which is tedious and time consuming given the large number of variables. Variables will need to be filtered for outliers, transformations may be needed, and different statistical models will be needed for continuous versus categorical explanatory variables. Also, no redundancy elimination is embedded into the traditional methods, so the output is likely to contain a large number of highly correlated variables, which clutter the output. For example, electrical test measurements are often reported on their natural, but also log scale and similar measurements are often taken on different test structures on the wafer. This can cause what is basically a single variable (say a leakage measurement) to show up many times in the output in slightly different forms and cause the analyst to errantly pass over other important variables in the list.

Using the approach in Figure 3, we will use the feature selection algorithm described earlier. No outlier removal, transformations, or pre filtering of the data is required. The output shown in Figure 7 is the cumulative variable importance reported by the algorithm. Seven variables have importance above 1%. From a

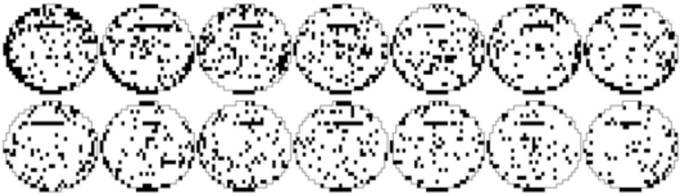


Fig. 4. Wafers demonstrating fail pattern

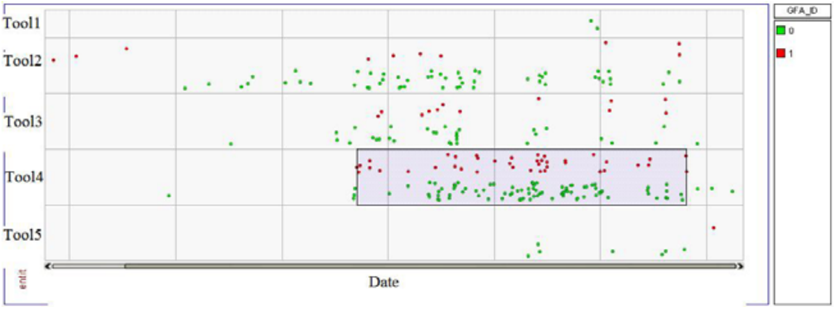


Fig. 5. Time trend for tools at a specific process operation. Red dots are affected lots classified as '1'. Jitter has been added.

large list of variables, we have filtered it to less than 10 to investigate further. The algorithm ran in under 10 minutes on a desktop computer running 32 bit Windows XP.

Queue time between two specific process operations has the largest variable importance. This is the time between when a lot finished processing at operation i and started processing at operation j. A bubble plot is shown in Figure 8 with the trend between the fraction of wafers with the GFA and the queue time. The size of the bubble indicates the number of wafers with the specific queue time. Clearly, once queue time decreases below 100, the fraction of wafers affected with the GFA rapidly rises.

The position of the wafer in the lot box has the next largest variable importance. Wafers travel through the fab in batches of 25 in lot boxes with 25 slots. Slot indicates the position a wafer was in the lot box. A bubble plot is shown in Figure 9 with the trend of fraction of wafers with the GFA versus the slot position. The size of the bubble indicates the number of wafers from each slot. As slot position increases, so does the fraction of wafers affected with the GFA, peaking around slot 17 and then declining, but still high, toward slot 25.

Of the remaining variables with importances above 1%, none have interesting trends. We won't show plots of all the remaining variables, but will show of variable 3 to demonstrate that the remaining variables do not have as strong of trends as did the queue time and slot position variables. Figure 10 shows fraction

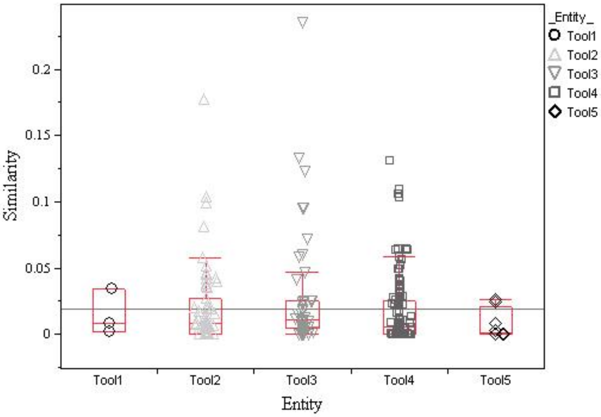
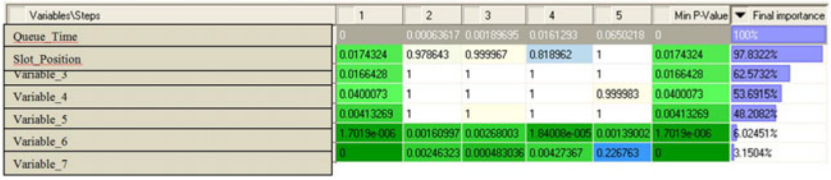


Fig. 6. Box plots of similarity to the affected wafers for each process tool at a specific operation



Variables\Steps	1	2	3	4	5	Min P-Value	Final importance
Queue_Time	0	0.00063617	0.00189695	0.0161293	0.0650218	0	100%
Slot_Position	0.0174324	0.978643	0.999967	0.818962	1	0.0174324	97.8322%
Variable_3	0.0166428	1	1	1	1	0.0166428	62.5732%
Variable_4	0.0400073	1	1	1	0.999983	0.0400073	53.6915%
Variable_5	0.00413269	1	1	1	1	0.00413269	48.2082%
Variable_6	1.7019e-006	0.00160997	0.00268003	1.84008e-005	0.00139002	1.7019e-006	6.02451%
Variable_7	0	0.00246323	0.000483036	0.00427367	0.226763	0	3.1504%

Fig. 7. Cumulative variable importance from feature selection algorithm

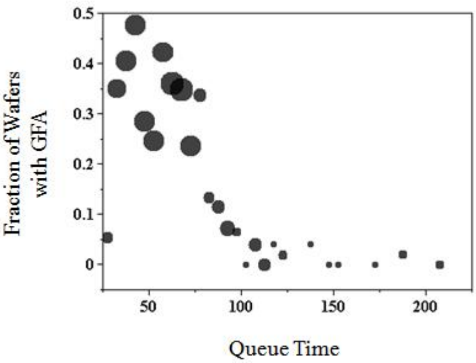


Fig. 8. Fraction of wafers affected by queue time. Bubble size varies with number of wafers.

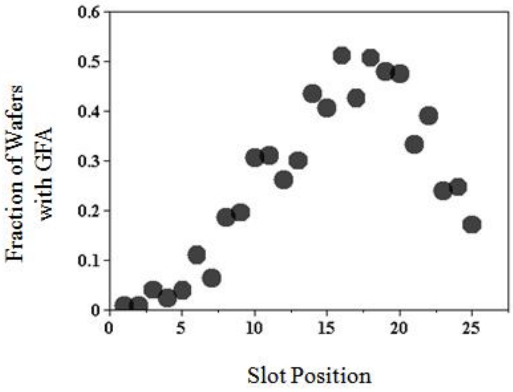


Fig. 9. Fraction of wafers affected by slot position. Bubble size varies with number of wafers.

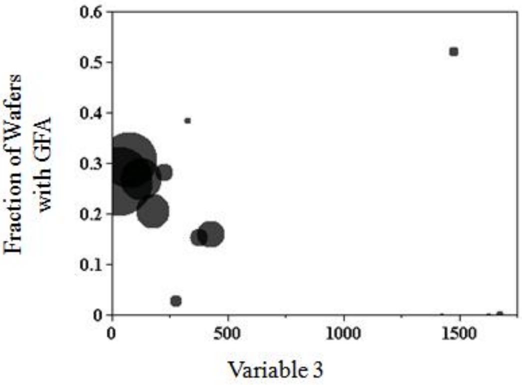


Fig. 10. Fraction of wafers affected versus variable 3. Bubble size varies with number of wafers.

of wafers affected with the GFA versus variable 3. The trend is uninteresting since it seems that the feature selection algorithm picked up on it due to a small cluster of wafers in the upper right hand corner of the graph.

4 Discussion of Results

Like many other industries, we struggle to make use of the almost overwhelming amount of information that we collect. The example shown previously is typical of the challenge we face. Prior to the use of feature selection as a filter, it would take multiple analysts using scatter plots, box plots, linear models, and ANOVA, weeks to find the 2 of over 30,000 variables that can be used to eliminate the yield issue. Now, including time to extract the data, it can be done in 2 or 3 days by one person.

As a result of this analysis, a delay was implemented in production control to ensure that wafers sat the required time before the next operation so that the impact of the GFA could be contained. Once the containment was implemented, the GFA was no longer seen at end of line. The information is also fed back to the process designers for a possible process change and a potential long term solution that would eliminate the need for the containment.

While our example focused on a single yield issue, there are usually multiple yield issues affecting a fab at any given time. Through the use of feature selection, we can now find important clues for 4 or 5 yield issues in the time that it used to take us to do just one. This has allowed us to meet the ever increasing yield goals set at our fabs and is key in being able to continue to meet these goals.

5 Conclusions

The significant increase in the number of variables measured in line during the production of cpus has made ANOVA based and graphical methods impractical to use as tools for the initial search of variables that contain information about yield issues. The entity based filter approach which came into use as a reaction to this increase addresses the need for a filter, but only for cases where yield issues can be isolated to a small number of process tools. Our proposed approach of using a feature selection algorithm to filter variables addresses the need for a filter and works for all cases. We demonstrated the effectiveness of this approach on a historical data set by finding the 2 variables in a data set of 30,000 variables that could be used to contain the yield issue.

References

1. Yu, L., Liu, H.: Efficient Feature Selection via Analysis of Relevance and Redundancy. J. of Mach. Learn. Res. 5, 1205–1224 (2004)
2. Koller, D., Sahami, M.: Toward Optimal Feature Selection. In: Proceedings of the Thirteenth International Conferrence on Machine Learning, pp. 284–292. Morgan Kaufmann Publishers, San Francisco (1996)
3. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant Features and the Subset Selection Problem. In: Machine Learning: Proceedings of the Eleventh International Conference, pp. 121–129. Morgan Kaufmann Publishers, San Francisco (1994)
4. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature Selection with Ensembles, Artificial Variables, and Redundancy Elimination. J. of Mach. Learn. Res. 10, 1341–1366 (2009)
5. Breiman, L.: Random Forests. Mach. Learn. 45(1), 5–32 (2001)
6. Friedman, J.H.: Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics 29, 1189–1232 (2001)

7. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman and Hall, New York (1998)
8. St. Pierre, E.R., Tuv, E., Borisov, A.: Spatial Patterns in Sort Wafer Maps and Identifying Fab Tool Commonalities. In: IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp. 268–272. IEEE, Los Alamitos (2008)
9. Borisov, A., Eruhimov, V., Tuv, E.: Dynamic soft feature selection for tree-based ensembles. In: Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L. (eds.) Feature Extraction, Foundations and Applications. Springer, New York (2005)