第一屆半導體大數據分析競賽
Big Data Analytics for Semiconductor Manufacturing
初賽 report

指導老師: 林軒田 副教授

台灣大學 資訊工程學系四年級 李廣和
台灣大學 資訊工程學系四年級 鄒侑霖
台灣大學 資訊工程學系四年級 宋昊恩

Abstract

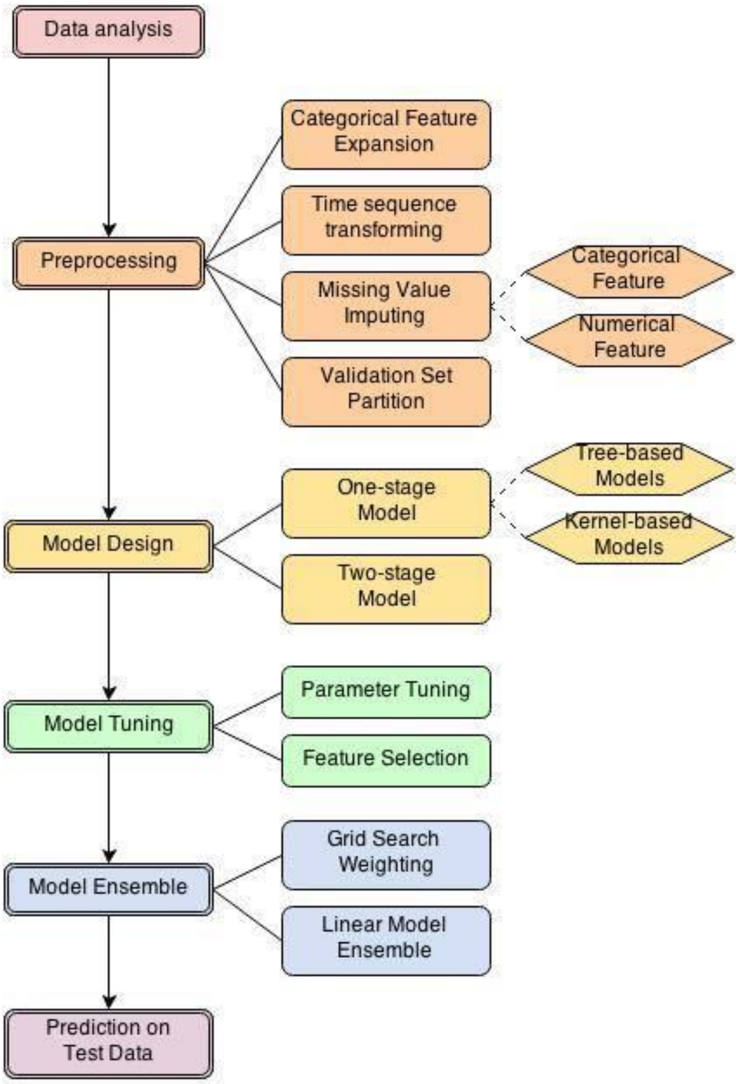
The semiconductor is a mainstream material using as the foundation of LEDs, solar cells, transistors…, etc. The development of semiconductor manufacture in Taiwan is well-known across the world. So, it is our mission to do research on the analysis and prediction on this task to provide cutting-edge technology in order to improve the quality of producing semiconductors.

Data Preprocessing

At first, we simply concatenate data from different files and create an initial training set. However, there are a lot of missing data and noise.

There are lots of categories in each correlated feature in a string format indicating the name of the category in that feature (In the left hand side in the ***Figure 1***, the feature “Recipe0” has 3 categories: Recipe0-0, Recipe0-1, ”NA”). In order to make use of the statistic models to analyze and predict the desired work, we have to convert the string format into numeric values. Here we introduce a binary expansion method as depicted in ***Figure 2***.

Figure1. Workflow

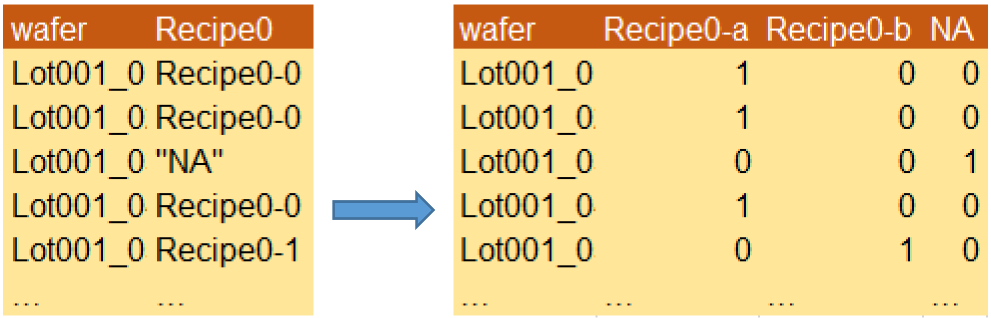


Notice that the above method only applies for categorical features but not for timestamp features and we treat “NA” as a category because we think a feature being NA indicates that it is different from all the other categories in the same feature column.

As for the timestamp feature, it’s easier to convert it into numerical value, so we subtract each time stamp by “1970/01/01” to get the time period value as the feature.

➩ E.g. $1981/02/03 \text{ —> } [(11 * 365) + 31 + 2 \text{ days}] * 86400 \text{ sec/day} = 349747200$

Figure 2: Binary Expansion



wafer	Recipe0
Lot001_0	Recipe0-0
Lot001_0	Recipe0-0
Lot001_0	"NA"
Lot001_0	Recipe0-0
Lot001_0	Recipe0-1
...	...

wafer	Recipe0-a	Recipe0-b	NA
Lot001_0	1	0	0
Lot001_0	1	0	0
Lot001_0	0	0	1
Lot001_0	1	0	0
Lot001_0	0	1	0
...	...	...	...

Validation setting

In order to produce convincing result of a regression model, we have to create a hold out dataset for validating our model performance and parameter tuning.

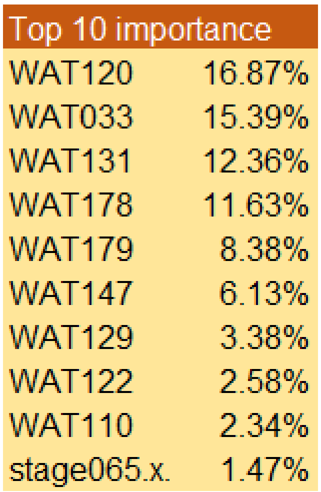
According to the work of *B Efron et all*[1], we introduce a boot technique for randomly sample 63.2% data as training data and the rest as validation data with stable performance control.

Key feature for predicting *CP*

Up to now, we have lots of features, but not sure for correlation of each feature and *CP,* therefore, we start from simple models, like linear regression, to other complicated models and tree based models, such as SVM, Random forest and Gradient Boosting Machine.

Furthermore, we conduct an experiment use all features including WAT, which is not given for the testing set. We use Gradient Boosting Machine with Gaussian distribution, which is a tree-based model, to do prediction on WAT features. Taking advantage of tree-based model, we can make use of the nature of model to get each feature's importance. So we have the feature importance distribution in the following ***Figure 3***:

Based on this analysis, we know that only the top-10 feature achieves 80% feature importance. So once we have *9 WAT values* and *stage065.x*, we are able to achieve high performance on the test set. But, unfortunately, the values of WAT are **not given in testing set**, so we propose a 2-stage framework for this task in ***Figure 4***.



Top 10 importance	
WAT120	16.87%
WAT033	15.39%
WAT131	12.36%
WAT178	11.63%
WAT179	8.38%
WAT147	6.13%
WAT129	3.38%
WAT122	2.58%
WAT110	2.34%
stage065.x.	1.47%

Figure 3: feature importance with validation MSE = 1.943

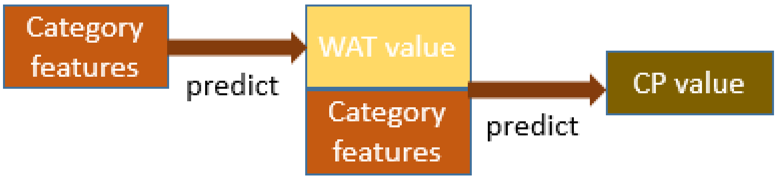
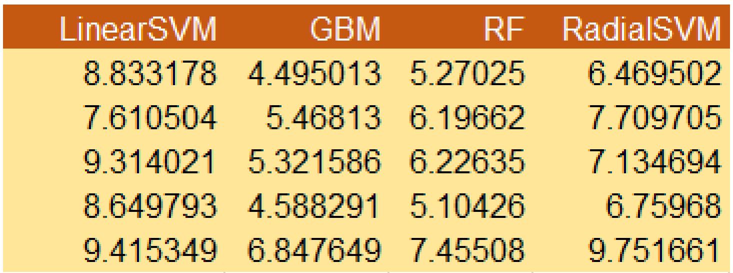


Figure 4: 2 stage model

[1] Efron, Bradley, and Robert Tibshirani. "Improvements on cross-validation: the 632+ bootstrap method." *Journal of the American Statistical Association* 92.438 (1997): 548-560.

Experiment Results

First, we run some models on the data after preprocessing, and tune their parameters with grid search. To make our results concrete, we repeat the experiment five times, and the results are listed as below:



LinearSVM	GBM	RF	RadialSVM
8.833178	4.495013	5.27025	6.469502
7.610504	5.46813	6.19662	7.709705
9.314021	5.321586	6.22635	7.134694
8.649793	4.588291	5.10426	6.75968
9.415349	6.847649	7.45508	9.751661

Figure 5: experiment results with 5 times bootstrap validation

We can find that the GBM model consistently outperform other models with different training/validation setting, which indicates that GBM is more suitable for the task in the dataset. Thus we choose GBM as our best single model.

Notice that the above result only derived from the feature set without **WAT** features. Thus, in order to achieve better performance, we try to set WAT as the 1st stage prediction target and combine them with original dataset in 2nd stage as **Figure 4**. However, our experiment reveals that although **WAT** features are key elements for getting a captivating results (validation MSE = 1.934 in our experiment), but our models perform poorly in those WAT values in respect to MSE measurement.

Afterwards, we try some basic ensemble schemes to utilize our various model settings. We choose simple linear model and grid search weighted blending as our ensemble method. The basic idea of ensemble is that every model has pros and cons, so if we leverage the results with different models, we can compensate each model’s weakness with other models’ strengths. But the final results of both our ensemble schemes show that they are poorer than the original single GBM model. So we adopt the GBM as our final prediction model.

Furthermore, we do some statistically analysis on the original TRAIN_CP distribution and our prediction. It shows that our prediction follows similar distribution with the TRAIN_CP. Therefore, we are more confident about our prediction. ☺

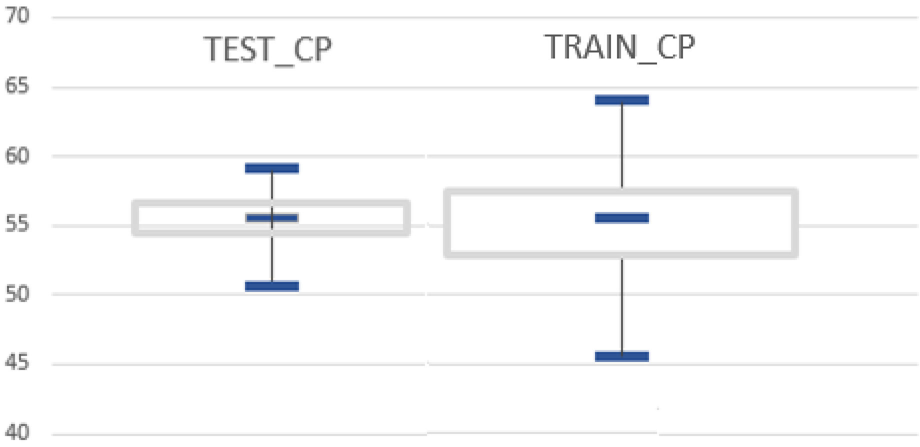


Figure 6: Box and whisker plot of our prediction/original CP