良率论文总结 V1

前景提要：

整个芯片生产过程中，所追求的是产品的良率，即一批生产批次的产品中符合最终要求的产品所占的比例。而影响产品最终良率的工艺流程有很多，如果采用穷举法将会耗费大量的时间和成本，因此如何能够利用已有的工艺数据来加速良率提升甚至预测良率显得尤为重要。

对于良率的提升有很多种方式和途径，例如**设计改变工艺参数选取的方法**在减少调参次数的同时达到较好的结果、例如**拟合已有的工艺参数和结果**预测出可能达到局部最优乃至全局最优的工艺参数和条件。

而在上述设计方法、拟合参数的过程中又涉及：**数据清洗**、**数据分类**、**数据降维**、**参数拟合**等步骤，各个步骤下又涉及**不同的算法**适用不同的数据。

本文旨在总结近年来有关“良率提升”、“实验改进”等方面的文章所用到的实验设计方法和数据处理模型。 （还在更新中…）

1. Design Of Experiment

试验设计是以**概率论**和**数理统计**为理论基础，经济地、科学地**安排试验**的一项技术。其发展大致经历了三个阶段：早期的单因素和多因素方差分析、传统的正交试验法和近代的调优设计法。

1.1. RSM（Response Surface Methodology）

响应面分析指：在多因素数量处理试验的分析中，可以分析试验指标（因变量）与多个试验因素（自变量）间的回归关系，这种回归可能是曲线或曲面的关系。

Response Surface Methodology 是一种应用于实验的方法，它先将所有的实验数据用 Box-Behnken Matrix 的形式表示（只有-1 0 1 三个取值），然后再根据矩阵的参数，列出响应面方程，最后再利用最小二乘法求出响应面方程的所有未知数前面的系数。例如有一个最终结果 Y，它会受到P_1、P_2、P_3 三个因素的影响，那么可以列出**三个自变量的二阶响应面方程来表示它**，如下所示:

$$Y = \beta_0 + \beta_1 P_1 + \beta_2 P_2 + \beta_3 P_3 + \beta_{11} P_1^2 + \beta_{22} P_2^2 + \beta_{33} P_3^2 + \beta_{12} P_1 P_2 + \beta_{13} P_1 P_3 + \beta_{23} P_2 P_3$$

其中，P_1、P_2、P_3为工艺参数，Y 为因变量，β 为回归系数。P_1、P_2、P_3由更敏感的工艺参数‘-’‘0’‘+’代替自变量实际的参数，其中‘-’‘0’‘+’**分别代表实际情况下**的小参数值、中等参数值、大参数值，分别用-1、0、+1 代表。

	P_1	P_2	P_3	Y
1	-	-	0	Y_1
2	-	+	0	Y_2
3	+	-	0	Y_3
4	+	+	0	Y_4
5	-	0	-	Y_5
6	-	0	+	Y_6
7	+	0	-	Y_7
8	+	0	+	Y_8
9	0	-	-	Y_9
10	0	-	+	Y_{10}
11	0	+	-	Y_{11}
12	0	+	+	Y_{12}
13	0	0	0	Y_{13}
14	0	0	0	Y_{14}
15	0	0	0	Y_{15}

1.1.1. Least Square Method

通过最小二乘方法拟合求得方程的回归系数。步骤如下：

① 确定目标函数：

$$Loss = \sum_{i=1}^{n} (Y_i - Y)^2$$

其中 n 为实验次数，Y_i为第 i 次实验的因变量值($i = 1, \ldots, 15$)，Y 为回归方程。（P_1、P_2、P_3为第 i 次实验的自变量取值，取值范围为-1、0、+1）

② 优化目标函数

为了使目标函数最优，对每个未知系数求其偏导数为 0，因为对于凸函数问题，使得函数对各未知数的偏导为 0 得到的值就是目标函数最小值，我们要的就是 Loss 越小越好，即：

$$\frac{\partial Loss}{\partial \beta_i} = 0(i = 0,1,2,3)$$
$$\frac{\partial Loss}{\partial \beta_{ij}} = 0(i = 1,2,3, j <= i)$$

③ 根据等式（3）求解所有的未知系数，采用梯度下降法计算非线性方程的梯度

梯度下降法步骤：

a. 对所有未知系数β赋初值(可以使用随机函数赋值)
b. 计算未知系数的梯度，代入 a 中的值，得到具体的梯度值

$$grad\_\beta_i = \frac{\partial Loss}{\partial \beta_i}$$

c. 代入 b 中的梯度值，更新未知系数（学习率可设置一个较小的数）：

$$\beta_i = \beta_i + learing\_rate * grad\_\beta_i$$

d. 使用 c 中更新后的值替换 a 的值
e. 当 loss 值小于我们设置的阈值后，可停止更新，否则回到步骤 b 计算下一个未知系数的梯度；亦可设置一个循环次数阈值，当循环次数大于设定的次数时，退出循环。

2. FDC (Fault Detection and Classification)

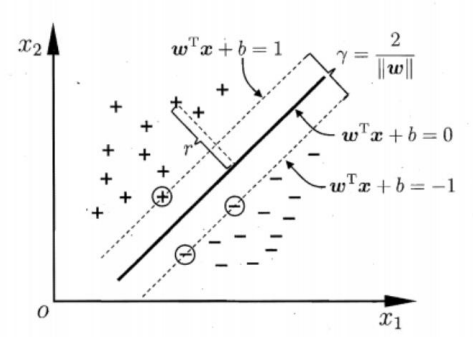
2.1. Classification

分类就是将数据区分分类，其特点是分类的类别已经预定义了，分类算法的基本功能是做**预测**。我们已知某个实体的具体特征，然后想判断这个实体具体属于哪一类，或者根据一些已知条件来估计感兴趣的参数。

2.1.1. SVM（Support Vector Machine）

支持向量机（Support Vector Machines，SVM）是一种**二分类模型**，它将实例的特征向量映射为空间中的一些点，SVM 的**目的就是想要画出一条线，以“最好地”区分这两类点**，以至如果以后有了新的点，这条线也能做出很好的分类。SVM 适合中小型数据样本、非线性、高维的分类问题。

虚线上的点到划分超平面的距离都是一样的，实际上只有这几个点共同确定了分类的线（也称作超平面）的位置，因此被称作 “支持向量（support vectors）”，支持向量机的最终目标，就是能找到一个这样的平面使得两条虚线之间的距离$\gamma = \frac{2}{\|\omega\|}$ 最短。如果看做一个线性规划问题，那么使得$\gamma = \frac{2}{\|\omega\|}$ 最短就是优化目标。



划分超平面可以定义为一个线性方程：$\omega^T X + b = 0$，其中：

※ $\omega = \{\omega_1; \omega_2; ...; \omega_d\}$是一个法向量，决定了超平面的方向，d 是特征值的个数；

※ X为训练样本；

※ b为位移项，决定了超平面与原点之间的距离。

只要确定了法向量ω和位移 b，就可以唯一地确定一个划分超平面。划分超平面和它两侧的边际超平面上任意一点的距离为$\frac{1}{\|\omega\|}$，利用一些数学推导可以将$\gamma = \frac{2}{\|\omega\|}$最大的优化问题变为有限制的凸优化问题（convex quadratic optimization），进而可以利用 Karush-Kuhn-Tucker (KKT)条件和拉格朗日公式，推出最大边际的超平面可以被表示为以下“决定边界 (decision boundary)”：

$$d(X^T) = \sum_{i=1}^{n} y_i \alpha_i X_i X^T + b_0$$

※ n 是支持向量点的个数，因为大部分的点并不是支持向量点，只有个别在边际超平面上的点才是支持向量点。那么我们就只对属于支持向量点的进行求和；

※ X_i为支持向量点的特征值；

※ y_i是支持向量点X_i的类别标记（class label），比如+1 还是-1；

※ X^T是要测试的实例，想知道它应该属于哪一类，把它带入该方程；

※ α_i和b_0都是单一数值型参数，由以上提到的最优算法得出，α_i是拉格朗日乘数。

每当有新的测试样本X，将它带入方程，根据方程结果的正负来进行二分类。

2.1.2. KNN（K Near Neighbor）

2.2. Clustering

聚类的目的也是把数据分类，但是事先我是不知道如何去分的，完全是算法自己来判断各条数据之间的相似性，相似的就放在一起。聚类算法的功能是**降维**。假如待分析的对象很多，我们需要归归类，划划简，从而**提高数据分析的效率**。

在 Intelligent Manufacturing 中可以应用于缺陷类型数据的聚类，将多种影响或者导致同种 defect 的影响参数聚类在一起，以达到数据降维作用。

"**聚类算法**"试图将数据集中的样本划分为若干个通常是不相交的子集，每个子集称为一个**“簇”(cluster)**，通过这样的划分，每个簇就可能对应于一些潜在的概念或**类别**。

2.2.1. K-Means

K-Means 是聚类算法中的最常用的一种，算法最大的特点是简单，好理解，运算速度快，但是**只能应用于连续型**的数据，并且一定要在聚类前需要**手工指定**要**分成几类**。

kmeans 算法又名 k 均值算法,K-means 算法中的 k 表示的是聚类为 k 个簇，means 代表取每一个聚类中数据值的均值作为该簇的中心，或者称为质心，即用每一个的类的质心对该簇进行描述。

其算法思想大致为：先从样本集中**随机选取 k 个样本作为簇中心**，并计算所有样本与这 k 个“簇中心”的距离，对于每一个样本，将其**划分到与其距离最近**的“簇中心”所在的簇中，对于新的簇计算各个簇的新的“簇中心”。

根据以上描述，我们大致可以猜测到实现 kmeans 算法的主要四点：

（1）**簇个数 k 的选择**；（人工确定）

（2）各个样本点到“簇中心”的距离；

（3）根据新划分的簇，更新“簇中心”；

（4）重复上述 2、3 过程，直至"簇中心"没有移动。

距离计算方式：

$$d(x,y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+...+(x_n-y_n)^2}=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

对于分类后的产生的 k 个簇，分别计算到簇内其他点距离均值最小的点作为质心（对于拥有坐标的簇可以**计算每个簇坐标的均值作为质心**）

聚类优化截止条件：

① 当每个簇的质心不再改变时就可以停止 K-means；

② 设置一个循环次数 loop_max，当循环次数到达 loop_max 时，停止 K-means。

2.2.2. PCA（principal component analysis）

主成分分析算法（PCA）是最常用的**线性降维方法**，它的目标是通过某种线性投影，将**高维的数据**映射到**低维的空间**中，并期望在所投影的维度上数据的**信息量最大**（方差最大），以此使用较少的数据维度，同时保留住较多的原数据点的特性。

算法步骤：

① 去除平均值
② 计算协方差矩阵
③ 计算协方差矩阵的特征值和特征向量
④ 将特征值排序、保留前 N 个最大的特征值对应的特征向量
⑤ 将原始特征转换到上面得到的 N 个特征向量构建的新空间中

具体推导：

假设有 M 个样本$\{X^1, X^2, ..., X^M\}$，每个样本有 N 维特征$X^i = (x_1^i, x_2^i, ..., x_N^i)^T$，每一个特征$x_j^i$都有各自的特征值。

第一步：对所有特征进行去中心化：即对所有样本，每一个特征都减去对应的平均值；

第二步：求协方差矩阵 C

$$C = \begin{pmatrix} cov(x_1, x_1) & \cdots & cov(x_1, x_n) \\ \vdots & \ddots & \vdots \\ cov(x_n, x_1) & \cdots & cov(x_n, x_n) \end{pmatrix}$$

在上述 C 矩阵中，对角线上是$x_1 \sim x_n$的方差，非对角线上是协方差。协方差大于 0 则代表x_i, x_j若有一个增，则另一个也增；协方差小于 0 则代表x_i, x_j若有一个增，则另一个减；协方差为 0 时，则二者独立。协方差绝对值越大，则代表两者对彼此的影响越大，反之越小。

其中:

$$cov(x_1, x_1) = \frac{\sum_{i=1}^{M} (x_1^i - \overline{x_1})(x_1^i - \overline{x_1})}{M - 1}$$

根据协方差计算公式我们就得到了这 M 个样本在这 N 维特征下的协方差矩阵 C。

求协方差矩阵 C 的特征值λ和相对应的特征向量 u（每一个特征值对应一个特征向量）：

$$Cu = \lambda u$$

特征值λ会有 N 个，每一个λ_i对应一个特征向量u_i，将特征值λ按照从大到小的顺序排序，选择最大的前 k 个，并将其对应的 k 个特征向量取出来，得到一组

$$\{(\lambda_1, u_1), (\lambda_2, u_2), ..., (\lambda_k, u_k)\}$$

第三步：将原始特征投影到选取的特征向量上，得到降维后的新 k 维特征。这个**选取最大的前 k 个特征值**和**相对应的特征向量**，并进行投影的过程，就是**降维的过程**。

对于每一个样本X^i，原来的特征是$\left(x_1^i, x_2^i, ..., x_n^i\right)^T$，投影之后的新特征是$\left(y_1^i, y_2^i, ..., y_k^i\right)^T$，新特征的计算公式如下：

$$\begin{bmatrix} y_1^i \\ \vdots \\ y_k^i \end{bmatrix} = \begin{bmatrix} u_1^T \cdot (x_1^i, x_2^i, ..., x_n^i)^T \\ \vdots \\ u_k^T \cdot (x_1^i, x_2^i, ..., x_n^i)^T \end{bmatrix}$$

最大方差理论:**方差越大，信息量就越大**。协方差矩阵的每一个特征向量就是一个投影面，每一个特征向量所对应的特征值就是原始特征投影到这个投影面之后的方差。由于投影过去之后，我们要尽可能保证信息不丢失，所以要选择具有较大方差的投影面对原始特征进行投影，也就是选择具有较大特征值的特征向量。然后将原始特征投影在这些特征向量上，投影后的值就是新的特征值。

每一个投影面生成一个新的特征，k 个投影面就生成 k 个新特征。

PCA 降维的目的，就是为了在尽量保证“信息量不丢失”的情况下，对原始特征进行降维，也就是**尽可能将原始特征往具有最大信息量的维度上进行投影**。将原特征投影到这些维度上，使降维后信息量损失最小。

3. Testing