Session 4
Homework 3 Introduction

Fall 2025 ECE 157/272A
Seoyeon Kim, Heekyung Lee

Overview

- **Goal: Leverage LLM to complete a long piece of code and to understand the process and results**
- **WM-811K dataset & Feature Engineering**
- **Decision Tree vs Support Vector Machine Classifier**

Dataset

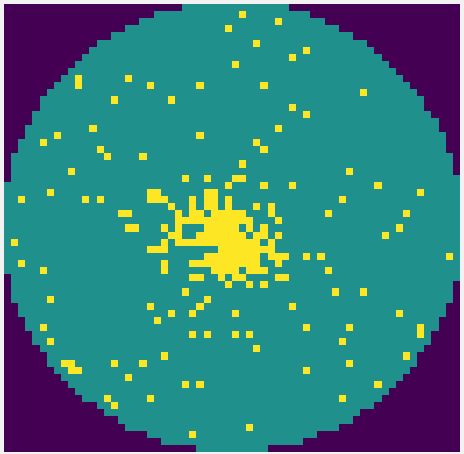
WM-811K wafer dataset

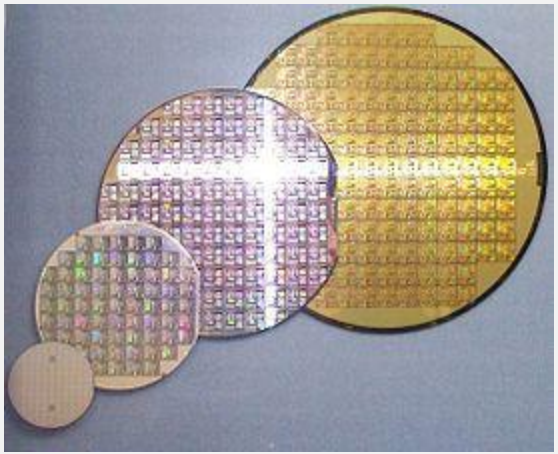
- Real-world data produced at a fab/foundry
- Wafer contains batches of functional circuits (dies)
- Every die is tested
- Detect failure patterns and perform root cause analysis to improve yield (% of passing dies)

No die (0)

Pass die (1)

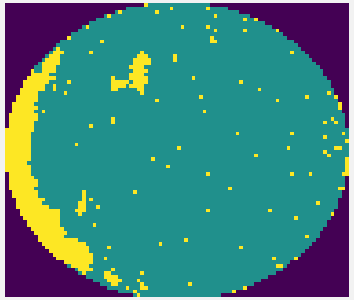
Fail die (2)



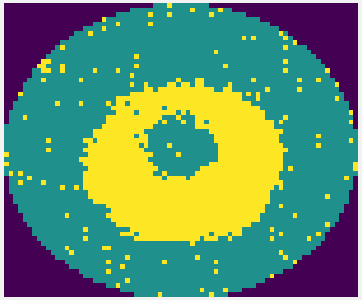


https://en.wikipedia.org/wiki/Wafer_(electronics)

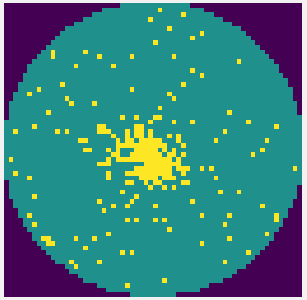
Dataset – Failure Patterns



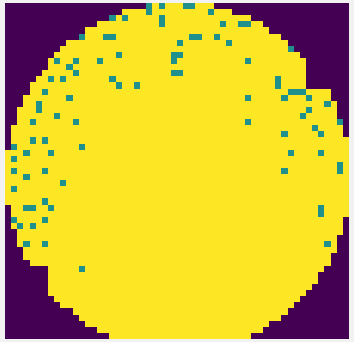
Edge-Loc



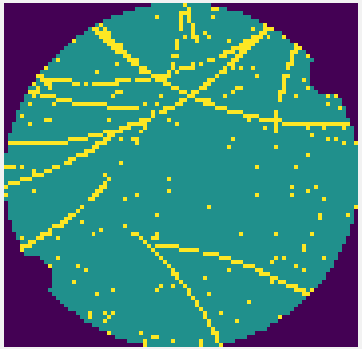
Donut



Center



Near-full



Scratch

Dataset – Structure

Column	Count	Type	Description
dieSize	1693	float	the number of dice on a wafer
failureType	1693	string	the type of failure pattern from 5 categories: "Center", "Edge-Loc", "Scratch", "Donut", "Near-full"
lotName	1693	string	the manufacturing lot name of a wafer
trainTestLabel	1693	string	denotes whether the sample is for training or testing in the original dataset
waferIndex	1693	integer	denotes the wafer index within a lot
waferMap	1693	numpy array	contains wafer maps *of varying size*. 0s denote no dice, 1s denote passing dice, 2s denote failing dice

Table 1: Real-world wafers [1]

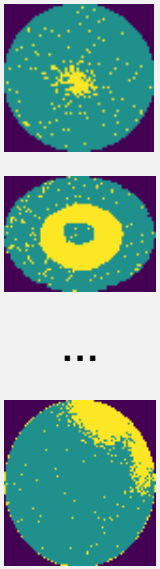
Overview

$$\begin{matrix} & f_1 & f_2 & \cdots & f_n \end{matrix}$$

$$\mathbf{X} = \begin{vmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \cdots \\ \vec{x}_m \end{vmatrix} = \begin{vmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{vmatrix} \quad \vec{y} = \begin{vmatrix} y_1 \\ y_2 \\ \cdots \\ y_m \end{vmatrix}$$

Features of samples were given in homework 2

Overview



$$f_1 \quad f_2 \quad \cdots \quad f_n$$

$$\vec{y} = \begin{vmatrix} y_1 \\ y_2 \\ \cdots \\ y_m \end{vmatrix}$$

Need to extract the features from the wafer maps in this homework

Tasks

Data Inspection

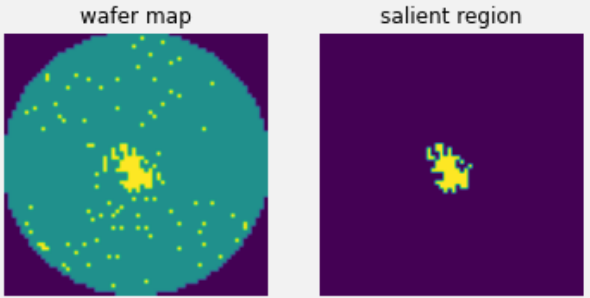
- Inspect the data frame information, visualize the failure patterns

Data Preparation (Part 1)

- Resize the wafer maps to (64, 64)
- Convert the failureType column to numerical values

Feature Engineering

- Extract features from the wafer maps
- Many features can be implemented using Scikit-Image, specifically regionprops

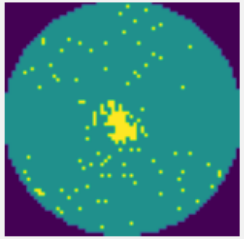


Feature Engineering Hints 1

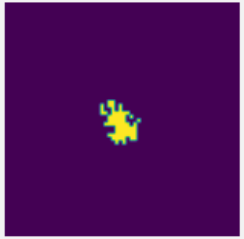
- **Scikit-Image label**
  - Label connected regions of an integer array
  - https://scikit-image.org/docs/stable/api/skimage.measure.html#skimage.measure.label

- **Scikit-Image regionprops**
  - Measure properties of labeled image regions
  - https://scikit-image.org/docs/stable/api/skimage.measure.html#skimage.measure.regionprops

wafer map



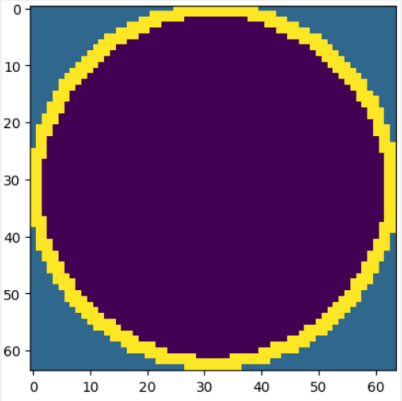
salient region



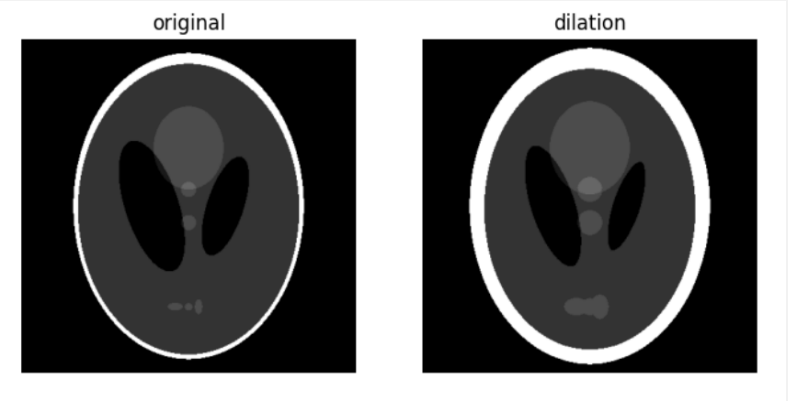
```
# label the salient region connected components and get their regions properties
labeled_image = label(salient_region, background=NO_DIE, connectivity=2)
region_props = regionprops(labeled_image)

# find perimeter and wafer map radius
perimeter = region_props[0].perimeter
```

Feature Engineering Hints 2

- **For get_edge_yield_loss and ring_label_from_outside**
  - **Scikit-Image dilation**
  - https://scikit-image.org/docs/stable/api/skimage.morphology.html#skimage.morphology.dilation
  - https://scikit-image.org/docs/stable/auto_examples/applications/plot_morphology.html#dilation



Outer Most Two Rings Mask



Dilation Example from Scikit-Image

Tasks

Data Preparation (Part 2)

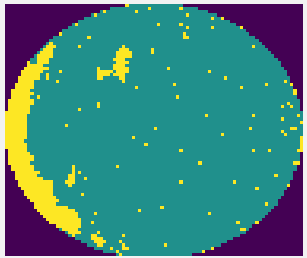
- Split the dataset into training and validation
- Do not normalize the data since we want to directly related the decision tree rules back to extract features

Model Training and Validation (Decision Tree)

- Train a decision tree model on the training set
- Evaluate the performance on the validation set
- Understand feature importance and decision rules based on the decision tree plot.

- Example:

- Intuitive description: **long thin** pattern **along the edge**
- Decision rule: **major axis ratio > 1** & **minor axis ratio < 0.097** & **minDistFromCenter > 28** & ***yield loss < 0.5***



Tasks

Model Training and Validation (Support Vector Classifier)

- Train a decision tree model on the training set
- Evaluate the performance on the validation set

Submission

- dt_scores.csv and svc_scores.csv: CSV files with one column, failureType, containing your models' predictions for the wafers in wafermap testing.npy.

- Submit your code in notebook format (.ipynb). Use LLM API in the file as much as possible, instead of using ChatGPT, Claude, or Gemini.