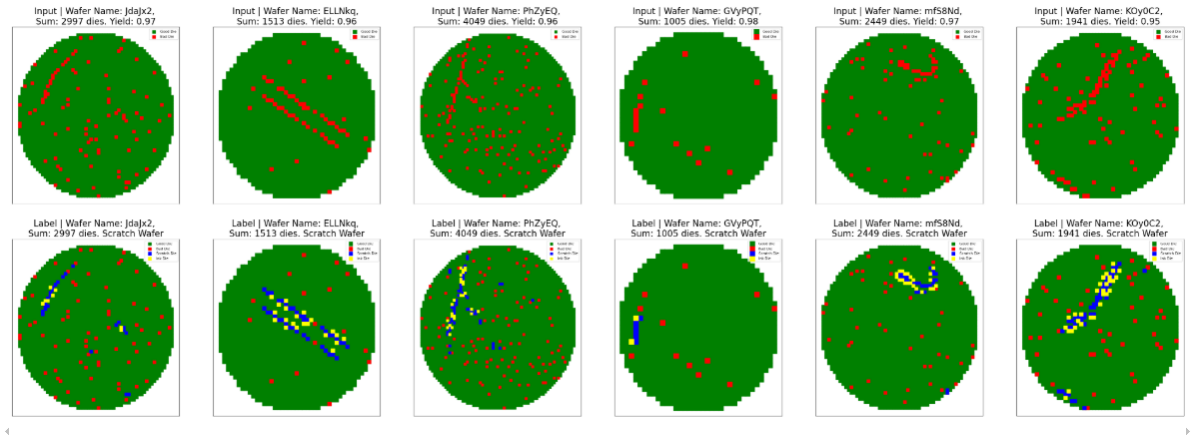
On the first page, I'll show the results, and then the process.

Results:

This is the result of my model on the test set:



As a business consideration, I added an output indicating whether the wafer is scratched and the size of the scratch:

	WaferName	HasScratch	ScratchPercent
0	ELLNkq	True	3.899537
1	GVyPQT	True	0.796020
2	JdaJx2	True	1.067734
3	KOy0C2	True	3.967027
4	PhZyEQ	True	1.951099
5	mfS8Nd	True	2.000817

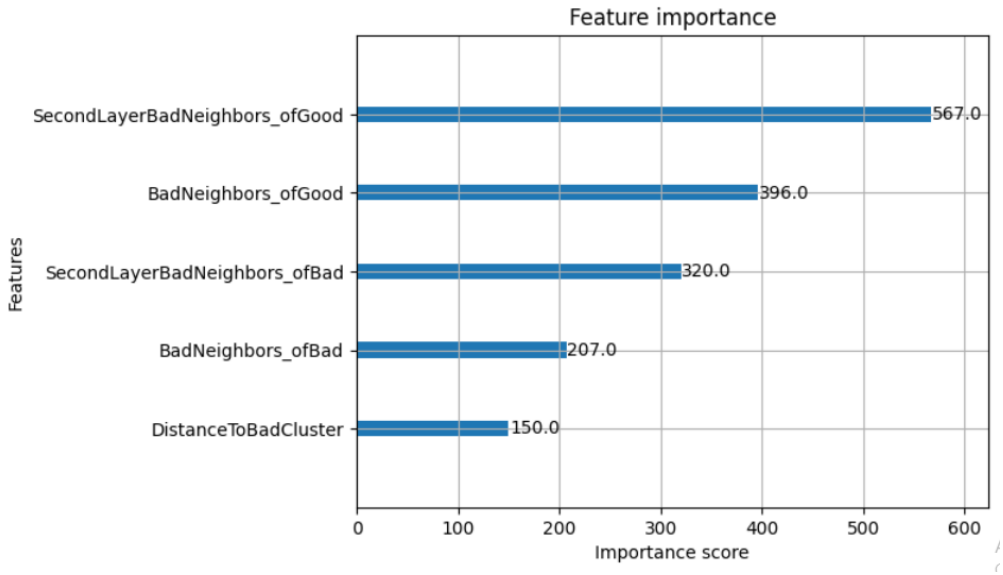
The process I went through:

Found different wafer sizes. Checked if size affects defect rate. No strong link, but smallest wafers seem slightly more prone to defects.

```
Average defect rate by wafer size:
   WaferSize  DefectRate
0     1005.0    0.041189
1     1513.0    0.037805
2     1941.0    0.038410
3     2449.0    0.036768
4     2997.0    0.035885
5     3405.0    0.035086
6     4049.0    0.034905
```

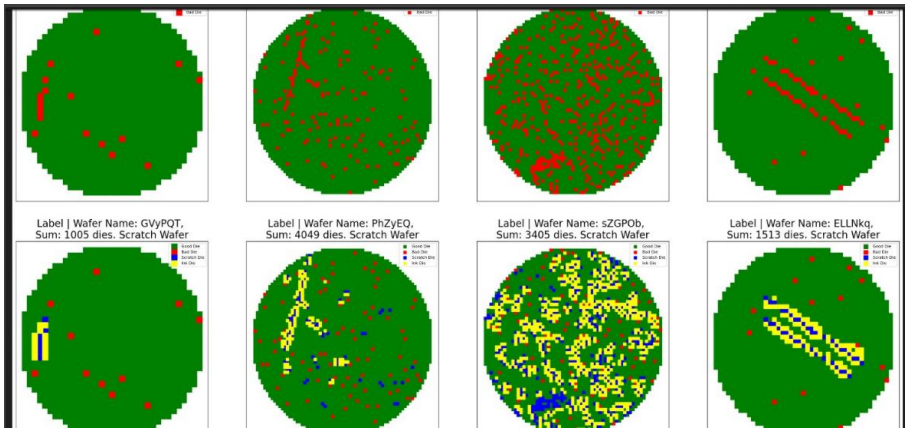
Features it creates:

BadNeighbors_ofGood: Number of "bad" neighbors (IsGoodDie == False) within a distance of 1.5 around good dies.
BadNeighbors_ofBad: Same as above, but around bad dies.
SecondLayerBadNeighbors_ofGood: Number of bad dies in the second layer (neighbors of neighbors) around good dies.
SecondLayerBadNeighbors_ofBad: Same as above, for bad dies.
DistanceToBadCluster: Distance from each die to the nearest bad die.



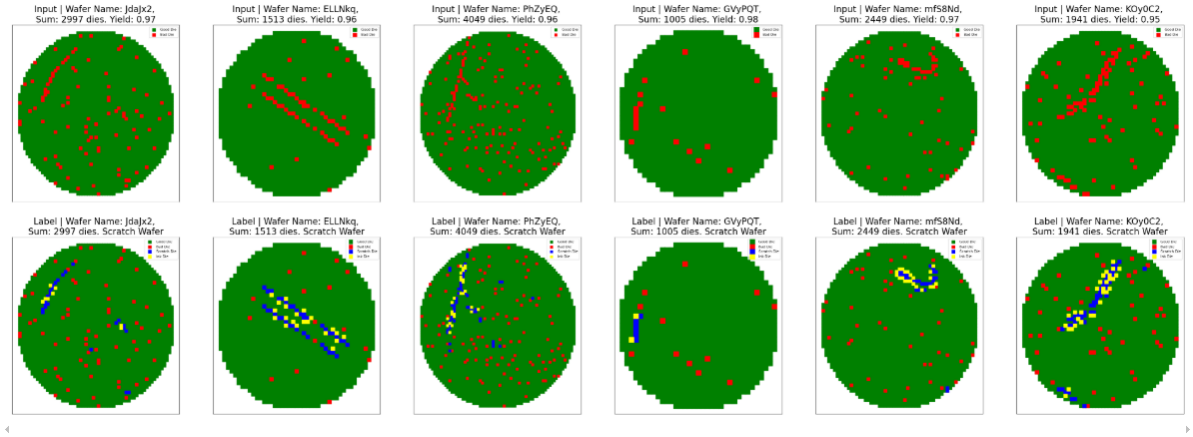
Model

Tested XGBoost and Random Forest—XGBoost performed better.
Chose simple, efficient models over neural networks for faster, low-resource deployment, aligning with business constraints.



At first, the model tended to detect scratches as much wider than they actually were.

I preferred using XGBoost because it allows fine-tuning the scale_pos_weight parameter, which is especially helpful in this highly imbalanced dataset—where good dies significantly outnumber bad ones.
After some tuning, I achieved:



Despite a few misses, the model successfully detects scratches.
With **85% recall**, it's effective in identifying most defective dies:

```
Accuracy: 0.9903
Precision (True class): 0.4439
Recall (True class): 0.8500
F1-score (True class): 0.5832
ROC AUC: 0.9938

Classification Report:
              precision    recall  f1-score   support

       False       1.00      0.99      1.00    361632
        True       0.44      0.85      0.58      2907

    accuracy                           0.99    364539
   macro avg       0.72      0.92      0.79    364539
weighted avg       0.99      0.99      0.99    364539
```