Data engineering and analytics for photolithography manufacturing process at DuPont

A practical approach from lab to fab

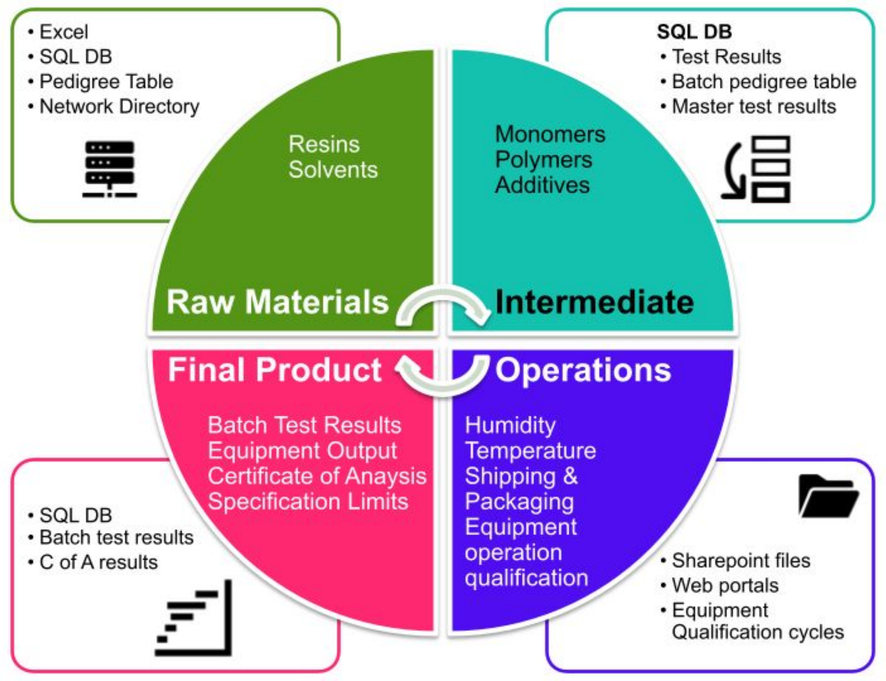
BACKGROUND:

- **Stringent requirements** on chemical suppliers to control material parameters used in semiconductor manufacturing
- Pre-emptively identify failures and **minimize manufacturing defects** using data science and statistics
- Success of statistics/ML requires **good data engineering practices** in a **challenging enterprise IT environment** with multiple systems

GOALS

Initial efforts focused on two targets

- **Data Availability**: Extraction/ Automation relieve domain-experts manual data organization
- **Data Accuracy**: Improve data quality



METHODS

3 phases: Assessment -> MVP -> Production

Understand the flow of data through systems by collaborating with domain experts

Start with a simple data stack - iterative development of data pipelines to get analysis-ready data

- Data ingestion: `pandas, openpyxl, pyodbc, requests`
- Data Exploration: `pandas, pandas-profiling`
- Data Quality: `thefuzz, assert, df.value_counts()`
- Code: Gitlab, Jupyter notebooks

Engage enterprise IT team: Confidentiality, security, access control (eg. creating views for specific products)

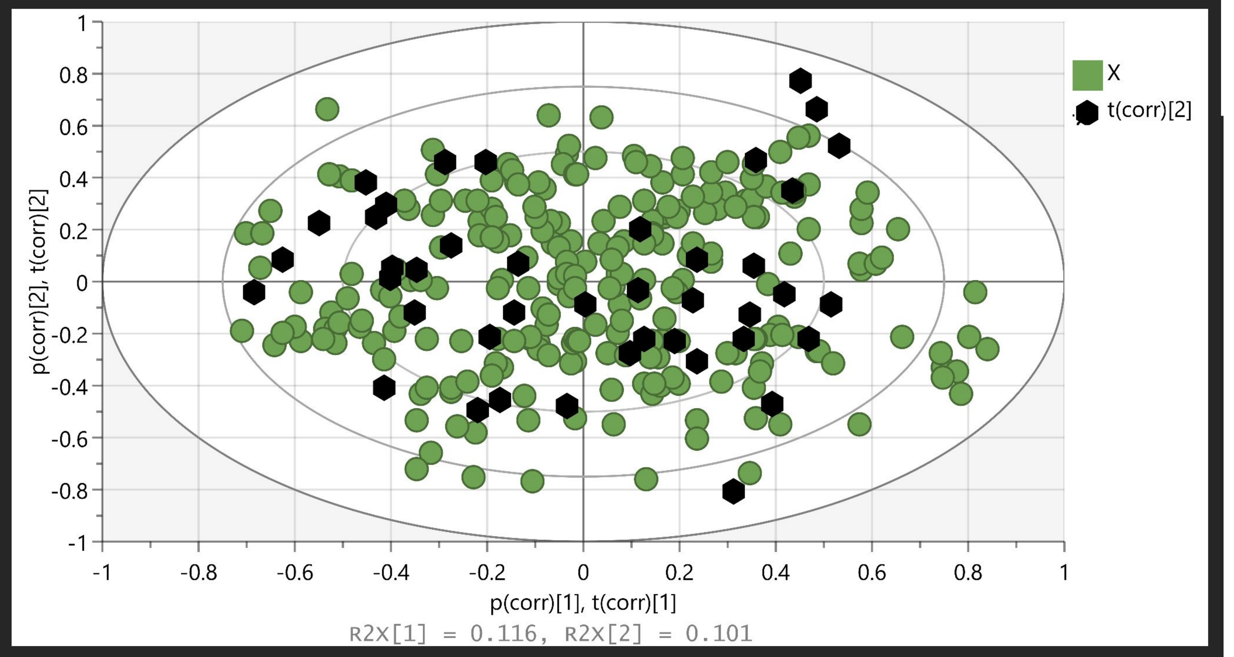
DUPONT™ logarithm labs

Focus on **data engineering** and **data quality**, collaboration with **domain experts**, engagement with **enterprise IT** and use of **scientific python libraries** helped **improve time to analysis by >80%** for photolithography manufacturing data

Take a picture to download code samples

https://github.com/logarithmlabs/scipy2023_sample_code

Biplot of final batches

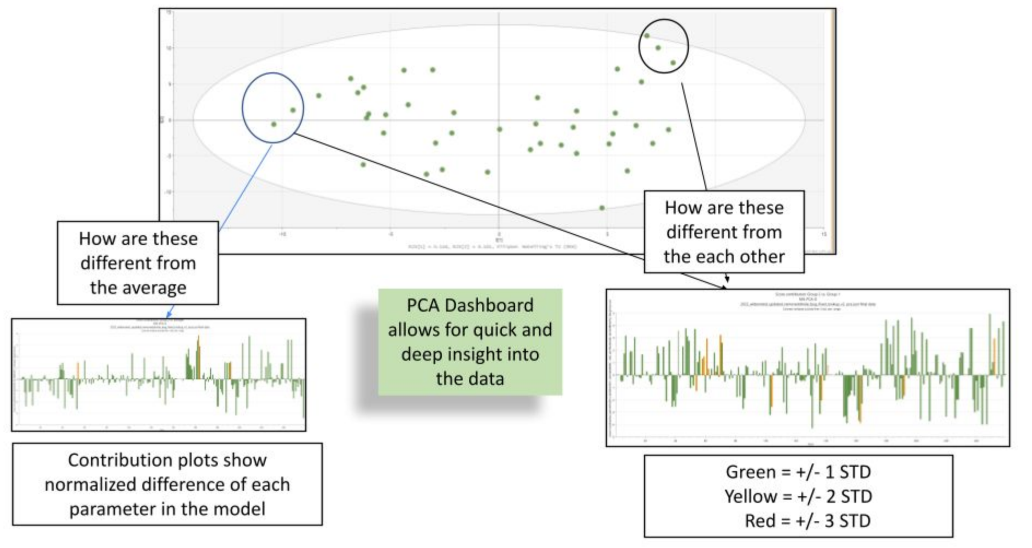


Black hexagons are the batches
Green dots are the batch parameters
Position relative to differences from each other / batch

RESULTS

- Building an **understanding of data systems** and data generation "workflows" is essential in an enterprise setting
- A **simple python-based data stack** helped jumpstart our efforts and get initial wins
- **Excel** is widely used in the enterprise. `pd.read_excel` is a start. **Data transformations are key**
- **Correcting data errors is an iterative process**, best done collaboratively with data + domain experts
- **Using `assert` statements generously catches incorrect data assumptions** in data pipeline code early and helps debug data issues quicker
- Different product groups make **slightly different assumptions about data**
- **Templated Jupyter notebooks** with product-specific data transformations **enable domain experts to utilize the python data stack**
- **`join` on data generated from varied systems requires domain expertise**
- **Enhancing raw data with missing metadata can surface additional insights** (eg. mapping product name to product kind)
- Getting **data into the right "shape"** for analysis often exposes issues with "unclean" data

The turnaround time for downstream, domain-specific analysis improved by >80% when domain experts got clean data in the right shape using automation



Avishek Panigrahi
avishek@logarithmlabs.com
Stefan J Caporale
stefan.caporale@dupont.com
Abhishek Shrivastava
abhishek.shrivastava@dupont.com
Sumanth Sekar
sekar.s.dhanasekaran@dupont.com