# ML Project Checklist

**ESM3081 Programming for Data Science**

**Seokho Kang**

성균관대학교
SUNG KYUN KWAN UNIVERSITY(SKKU)

# Machine Learning Project Checklist

1. **Frame the problem and look at the big picture.**
2. **Get the data.**
3. **Explore the data to gain insights.**
4. **Prepare the data to better expose the underlying data patterns to ML algorithms.**
5. **Explore many different models and short-list the best ones.**
6. **Fine-tune your models and combine them into a great solution.**
7. **Present your solution.**
8. **Launch, monitor, and maintain your system.**

# Machine Learning Project Checklist

## 1. Frame the problem and look at the big picture.

a. Define the objective in business terms.

b. How will your solution be used?

c. What are the current solutions/workarounds (if any)?

d. How should you frame this problem (supervised/unsupervised, online/offline, etc.)?

e. How should performance be measured?

f. Is the performance measure aligned with the business objective?

g. What would be the minimum performance needed to reach the business objective?

h. What are comparable problems? Can you reuse experience or tools?

i. Is human expertise available?

j. How would you solve the problem manually?

k. List the assumptions you (or others) have made so far.

l. Verify assumptions if possible.

# Machine Learning Project Checklist

## 2. Get the data.

a. List the data you need and how much you need.

b. Find and document where you can get that data.

c. Check how much space it will take.

d. Check legal obligations, and get authorization if necessary.

e. Get access authorizations.

f. Create a workspace (with enough storage space).

g. Get the data.

h. Convert the data to a format you can easily manipulate (without changing the data itself).

i. Ensure sensitive information is deleted or protected (e.g., anonymized).

j. Check the size and type of data (time series, sample, geographical, etc.).

k. Sample a test set, put it aside, and never look at it (no data snooping!).

*** Note:** automate as much as possible so you can easily get fresh data.

# Machine Learning Project Checklist

## 3. Explore the data to gain insights.

a. Create a copy of the data for exploration (sampling it down to a manageable size if necessary).

b. Create a Jupyter notebook to keep a record of your data exploration.

c. Study each attribute and its characteristics:

- Name
- Type (categorical, int/float, bounded/unbounded, text, structured, etc.)
- % of missing values
- Noisiness and type of noise (stochastic, outliers, rounding errors, etc.)
- Possibly useful for the task?
- Type of distribution (Gaussian, uniform, logarithmic, etc.)

d. For supervised learning tasks, identify the target attribute(s).

e. Visualize the data.

f. Study the correlations between attributes.

g. Study how you would solve the problem manually.

h. Identify the promising transformations you may want to apply.

i. Identify extra data that would be useful (go back to “Get the Data”).

j. Document what you have learned.

*** Note:** try to get insights from a field expert for these steps.

# Machine Learning Project Checklist

## 4. Prepare the data to better expose the underlying data patterns to ML algorithms.

a. Data cleaning:
   - Fix or remove outliers (optional).
   - Fill in missing values (e.g., with zero, mean, median…) or drop their rows (or columns).

b. Feature selection (optional):
   - Drop the attributes that provide no useful information for the task.

c. Feature engineering, where appropriate:
   - Discretize continuous features.
   - Decompose features (e.g., categorical, date/time, etc.).
   - Add promising transformations of features (e.g., log(x), sqrt(x), x^2, etc.).
   - Aggregate features into promising new features.

d. Feature scaling: standardize or normalize features.

*** Notes:**

✓ Work on copies of the data (keep the original dataset intact).

✓ Write functions for all data transformations you apply, for five reasons:
   - So you can easily prepare the data the next time you get a fresh dataset
   - So you can apply these transformations in future projects
   - To clean and prepare the test set
   - To clean and prepare new data instances once your solution is live
   - To make it easy to treat your preparation choices as hyperparameters

# Machine Learning Project Checklist

## 5. Explore many different models and short-list the best ones.

a. Train many quick and dirty models from different categories (e.g., linear, naïve Bayes, SVM, Random Forests, neural net, etc.) using standard hyperparameters.

b. Measure and compare their performance.

- For each model, use *N*-fold cross-validation and compute the mean and standard deviation of the performance measure on the *N* folds.

c. Analyze the most significant variables for each algorithm.

d. Analyze the types of errors the models make.

- What data would a human have used to avoid these errors?

e. Have a quick round of feature selection and engineering.

f. Have one or two more quick iterations of the five previous steps.

g. Short-list the top three to five most promising models, preferring models that make different types of errors.

*** Notes:**

✓ If the data is huge, you may want to sample smaller training sets so you can train many different models in a reasonable time (be aware that this penalizes complex models such as large neural nets or Random Forests).

✓ Once again, try to automate these steps as much as possible.

# Machine Learning Project Checklist

## 6. Fine-tune your models and combine them into a great solution.

a. Fine-tune the hyperparameters using cross-validation.
   - Treat your data transformation choices as hyperparameters, especially when you are not sure about them (e.g., should I replace missing values with zero or with the median value? Or just drop the rows?).
   - Unless there are very few hyperparameter values to explore, prefer random search over grid search. If training is very long, you may prefer a Bayesian optimization approach.

b. Try Ensemble methods. Combining your best models will often perform better than running them individually.

c. Once you are confident about your final model, measure its performance on the test set to estimate the generalization error.

*** Notes:**

- ✓ You will want to use as much data as possible for this step, especially as you move toward the end of fine-tuning.
- ✓ As always automate what you can.

# Machine Learning Project Checklist

## 7. Present your solution.

a. Document what you have done.

b. Create a nice presentation.
   - Make sure you highlight the big picture first.

c. Explain why your solution achieves the business objective.
   - Don’t forget to present interesting points you noticed along the way.
   - Describe what worked and what did not.

d. List your assumptions and your system’s limitations.

e. Ensure your key findings are communicated through beautiful visualizations or easy-to-remember statements (e.g., “the median income is the number-one predictor of housing prices”).

# Machine Learning Project Checklist

## 8. Launch, monitor, and maintain your system.

a. Get your solution ready for production (plug into production data inputs, write unit tests, etc.).

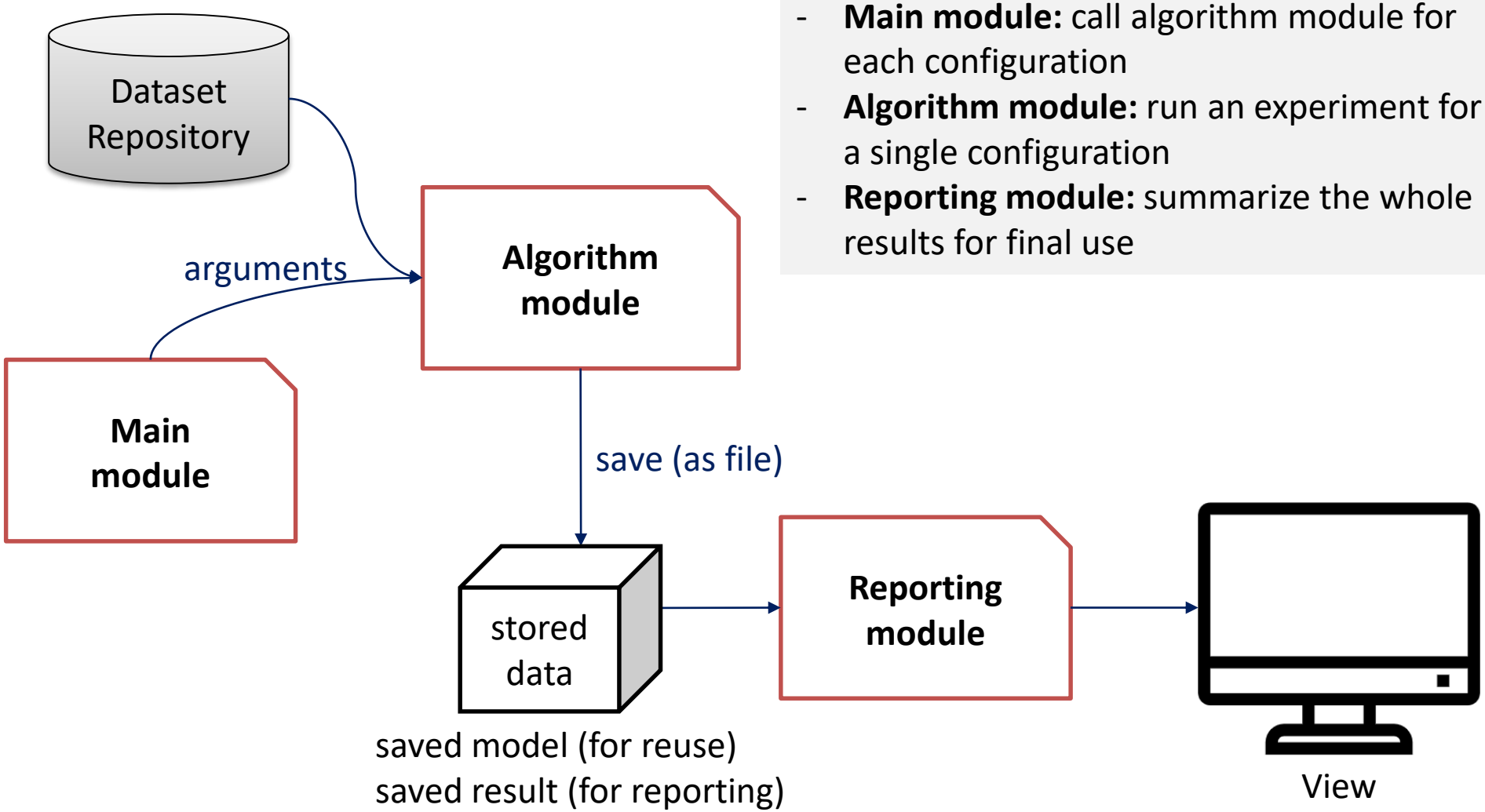
b. Write monitoring code to check your system's live performance at regular intervals and trigger alerts when it drops.

- Beware of slow degradation too: models tend to "rot" as data evolves.
- Measuring performance may require a human pipeline (e.g., via a crowdsourcing service).
- Also monitor your inputs' quality (e.g., a malfunctioning sensor sending random values, or another team's output becoming stale). This is particularly important for online learning systems.

c. Retrain your models on a regular basis on fresh data (automate as much as possible).

# Programming for ML Project

- **Main module:** call algorithm module for each configuration
- **Algorithm module:** run an experiment for a single configuration
- **Reporting module:** summarize the whole results for final use



# Debugging

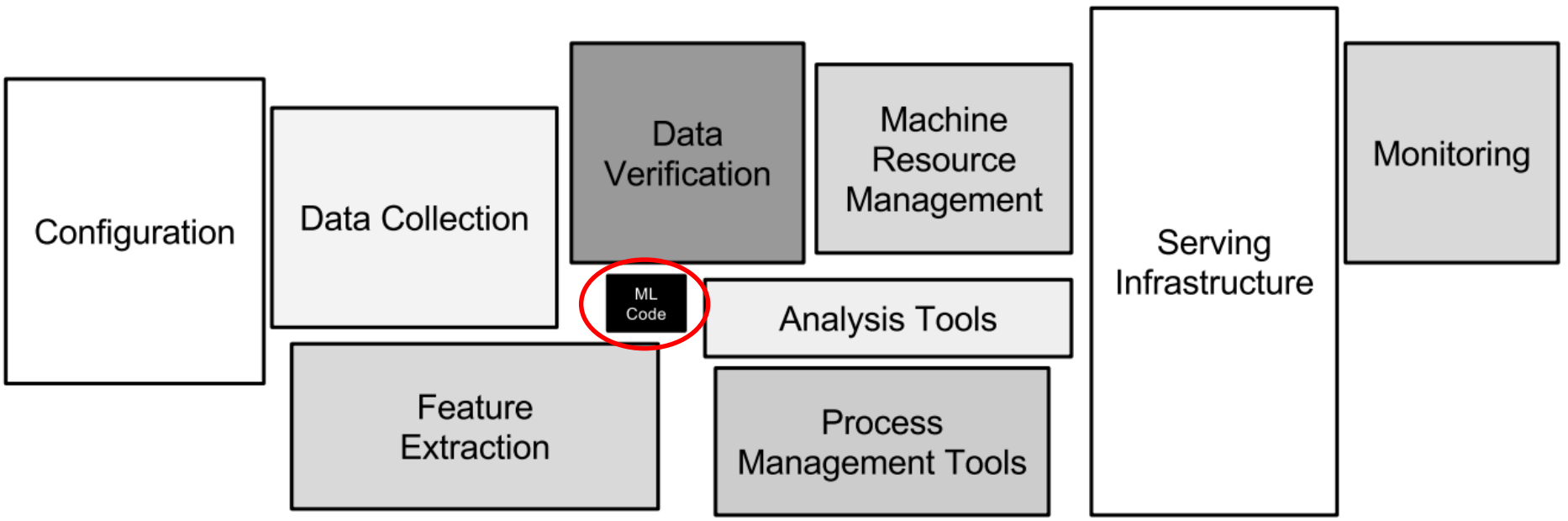
- **It's usually difficult to find the reason why a machine learning system doesn't work or performs poorly.**
  - In most cases, we don't know the expected behavior or suboptimal behavior of the system.
  - Multiple components in the system can adapt each other. If one component is broken, the other components can adapt.
  - Sometimes, there's a software/hardware defect.

- **If things are problematic, you should…**
  - Inspect data, software, and hardware
  - Use a small dataset
  - Visualize the model in action
  - Visualize the worst mistakes
  - …

# Debugging

# ML Practice

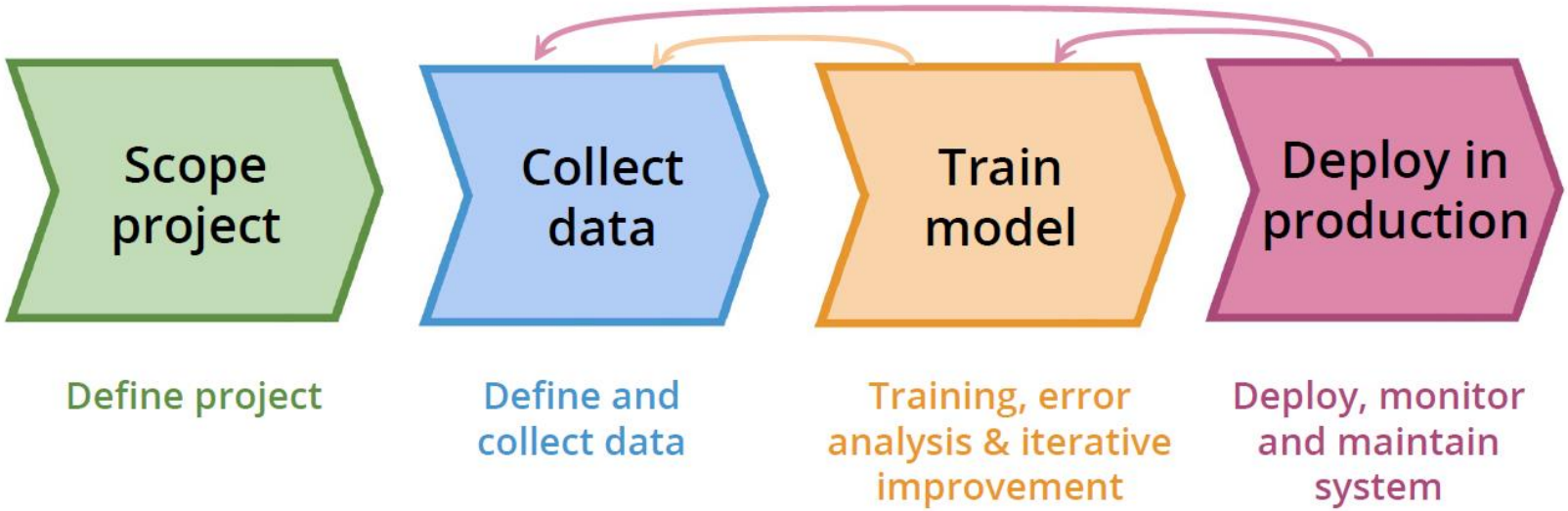
- **Real-World ML System**
  - Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.




Sculley, David, et al. "Hidden technical debt in machine learning systems." *Advances in neural information processing systems*. 2015.

# ML Practice

- **Typical Lifecycle of ML Project**



# Data-Centric AI/ML

*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***

**AI/ML System = Code(Model/Algorithm) + Data**

| How can you change the model (code) to improve performance? | How can you systematically change your data (inputs x or labels y) to improve performance? |
|---|---|
| **Model-centric view** | **Data-centric view** |
| Collect what data you can, and develop a model good enough to deal with the noise in the data. | The consistency of the data is paramount. Use tools to improve the data quality; this will allow multiple models to do well. |
| Hold the data fixed and iteratively improve the code/model. | *Hold the code fixed and iteratively improve the data.* |

# Data-Centric AI/ML

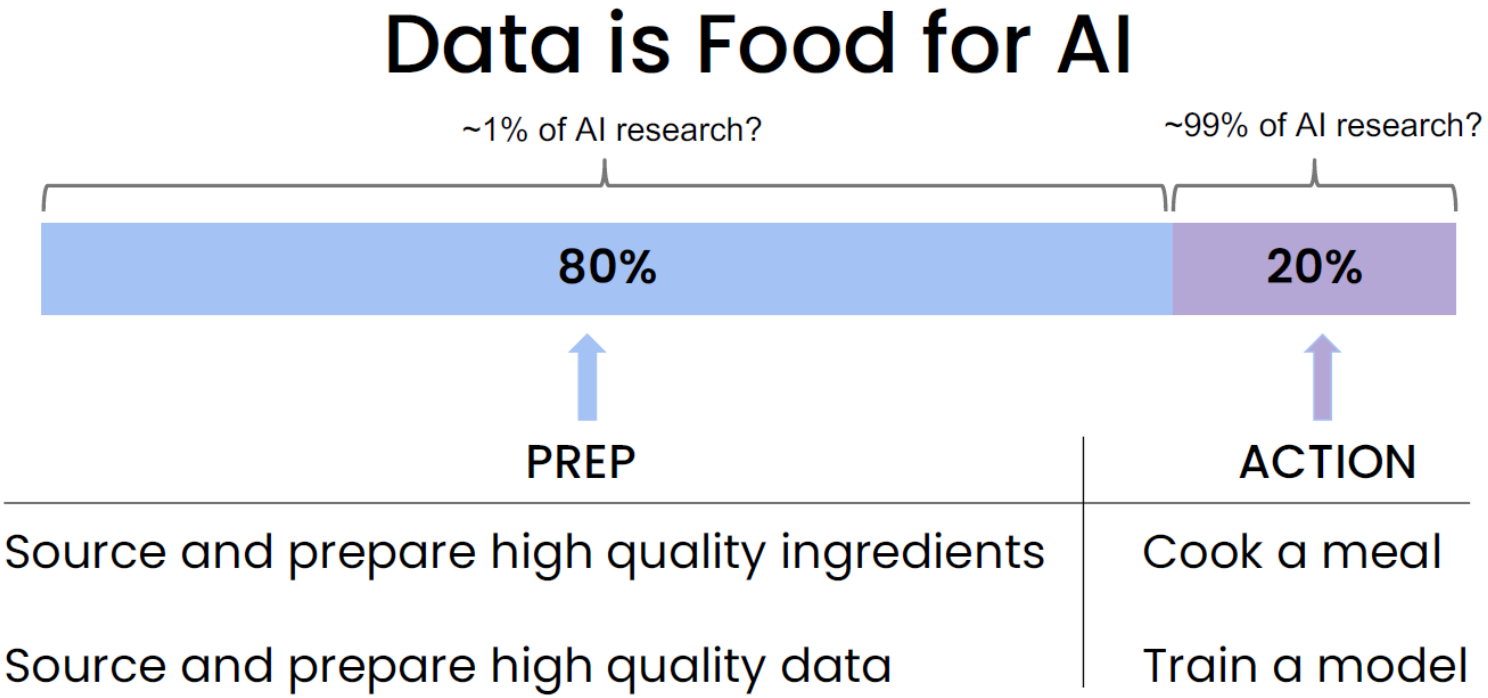
*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***

## Improving the code vs. the data

| | Steel defect detection | Solar panel | Surface inspection |
|---|---|---|---|
| Baseline | 76.2% | 75.68% | 85.05% |
| Model-centric | +0% (76.2%) | +0.04% (75.72%) | +0.00% (85.05%) |
| Data-centric | +16.9% (93.1%) | +3.06% (78.74%) | +0.4% (85.45%) |

# Data-Centric AI/ML

*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***



# Data-Centric AI/ML

*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***

## From Big Data to Good Data

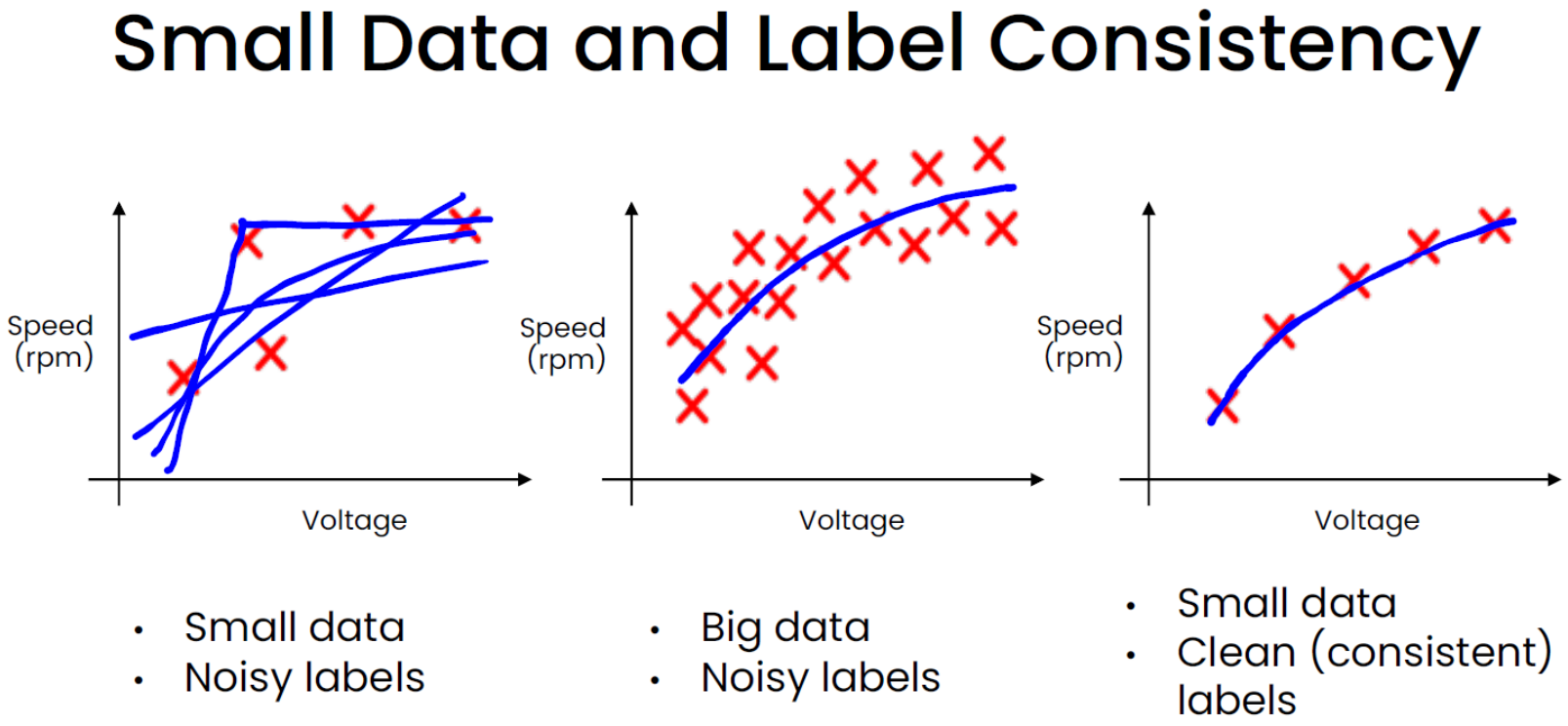
MLOps' most important task: **Ensure consistently high-quality data in all phases of the ML project lifecycle.**

**Good data is:**

- Defined consistently (definition of labels y is unambiguous)
- Cover of important cases (good coverage of inputs x)
- Has timely feedback from production data (distribution covers data drift and concept drift)
- Sized appropriately

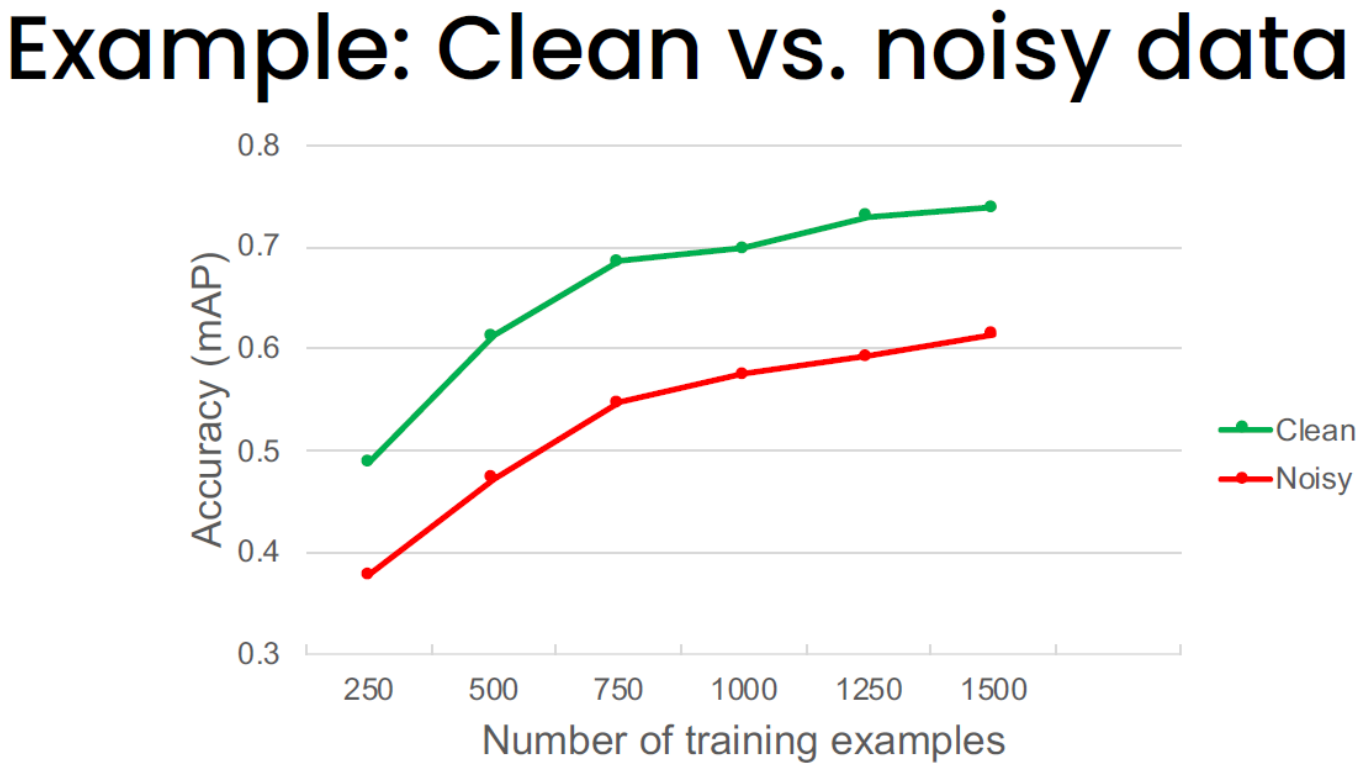
# Data-Centric AI/ML

*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***



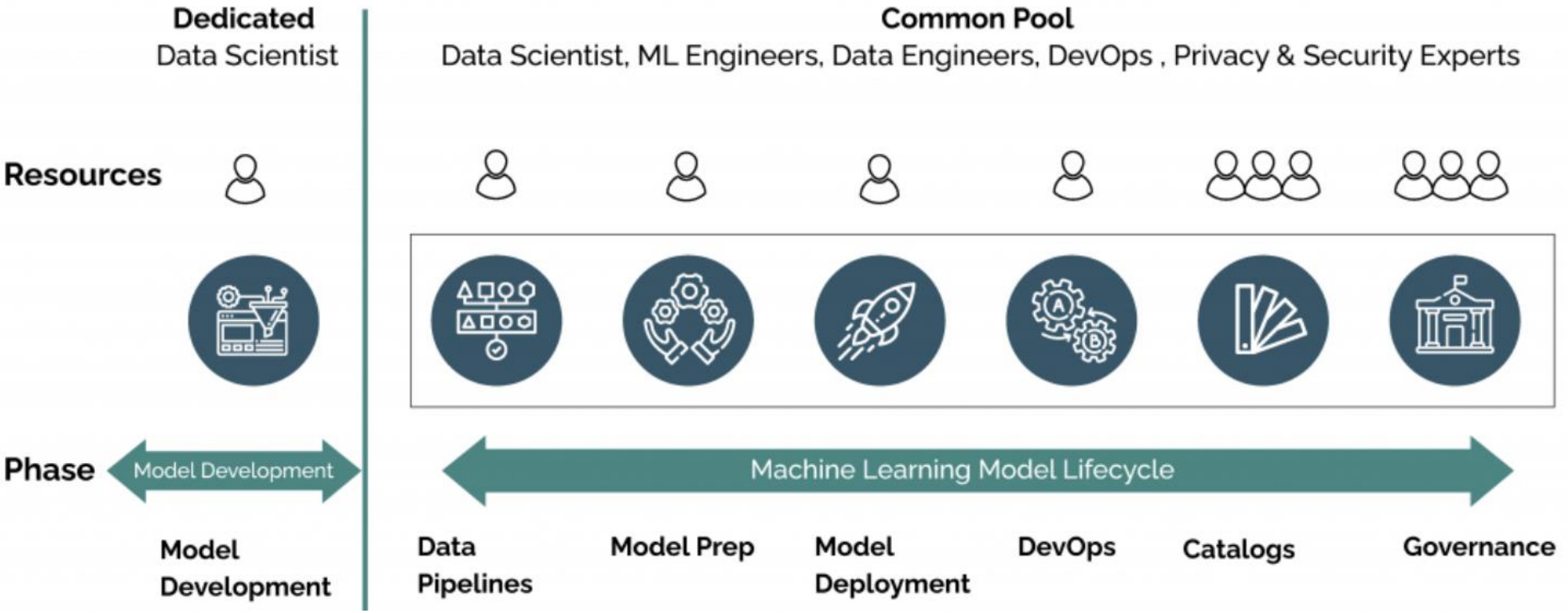
# Data-Centric AI/ML

*from* ***Andrew Ng's Talk "MLOps: From Model-Centric to Data-Centric AI"***



# MLOps

- **Machine Learning Operations (MLOps)**



# MLOps

- **Machine Learning Operations (MLOps)**