MLOps: From Model-centric to Data-centric AI

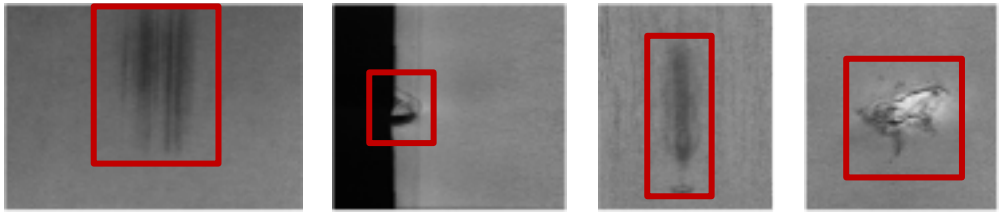
Andrew Ng

AI system = Code + Data

(model/algorithm)

Inspecting steel sheets for defects

Examples of defects

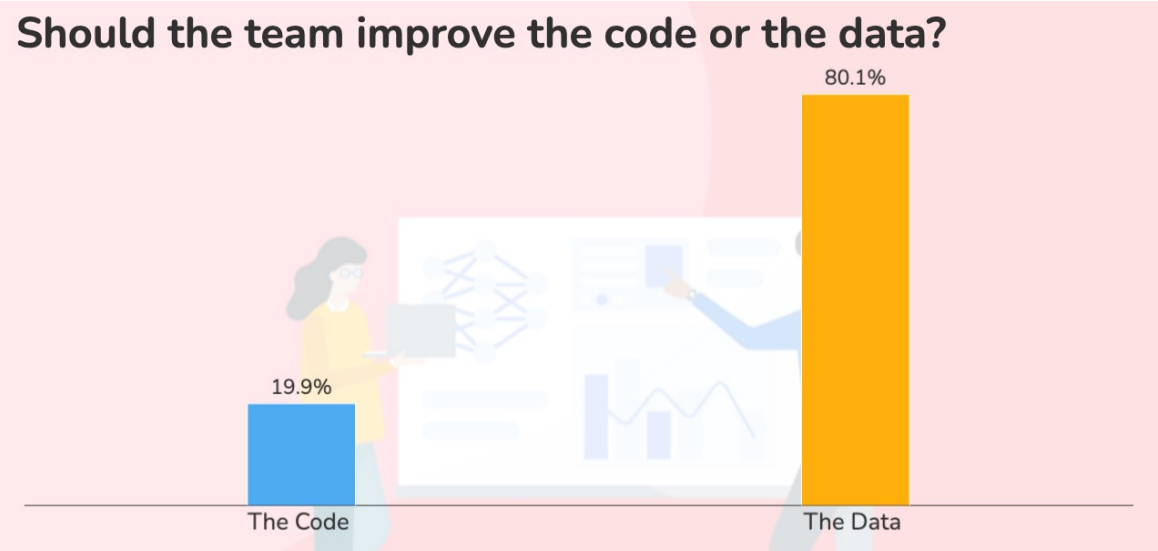


Baseline system: 76.2% accuracy

Target: 90.0% accuracy

Audience poll: Should the team improve the code or the data?

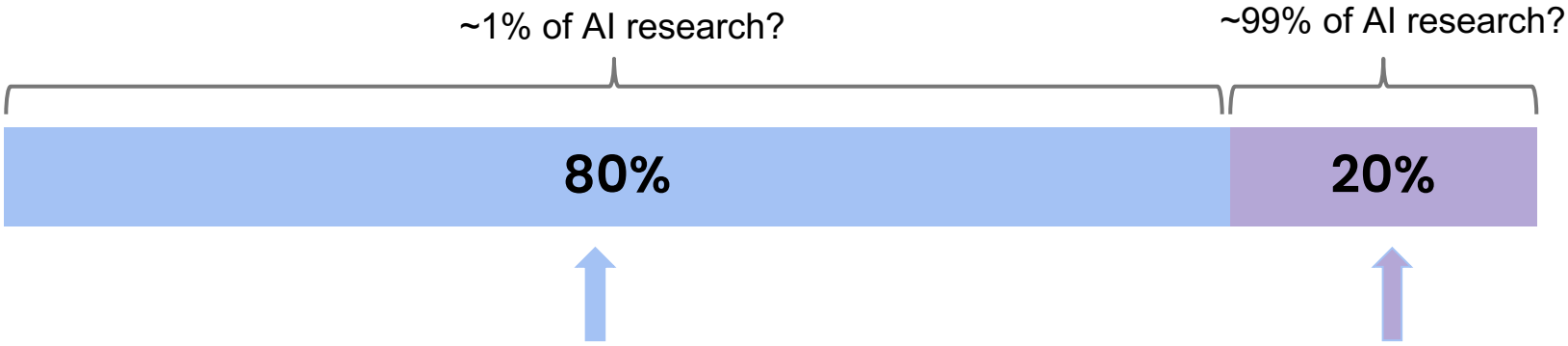
Poll results:



Improving the code vs. the data

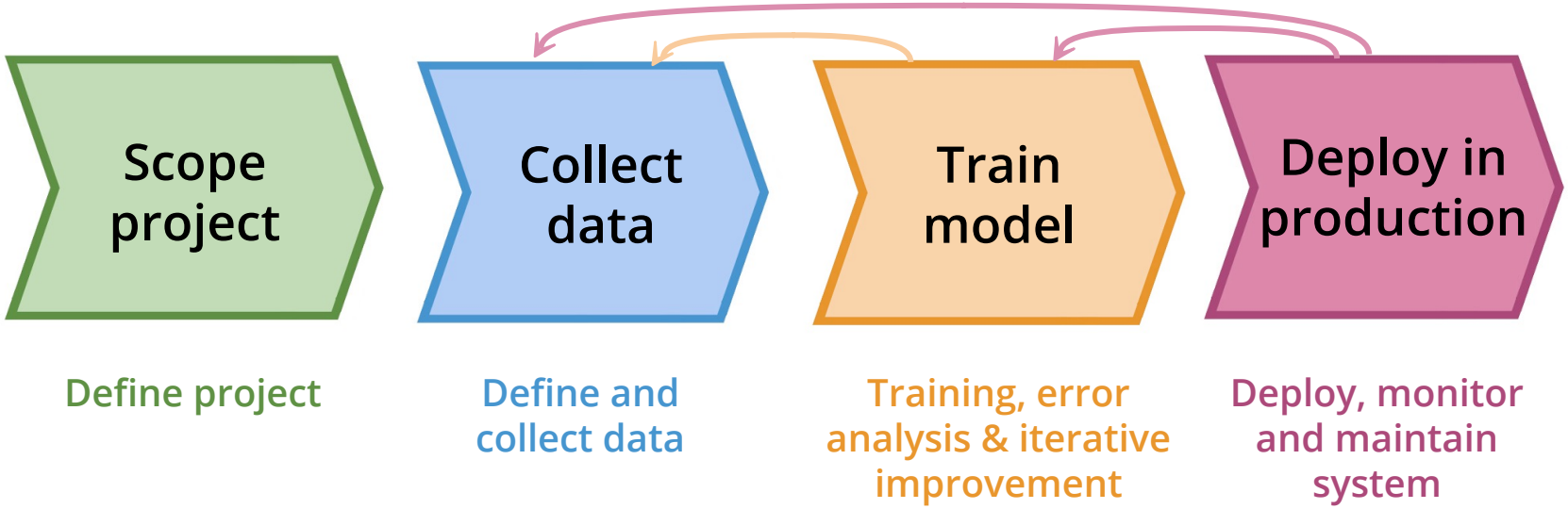
	Steel defect detection	Solar panel	Surface inspection
Baseline	76.2%	75.68%	85.05%
Model-centric	+0% (76.2%)	+0.04% (75.72%)	+0.00% (85.05%)
Data-centric	+16.9% (93.1%)	+3.06% (78.74%)	+0.4% (85.45%)

Data is Food for AI

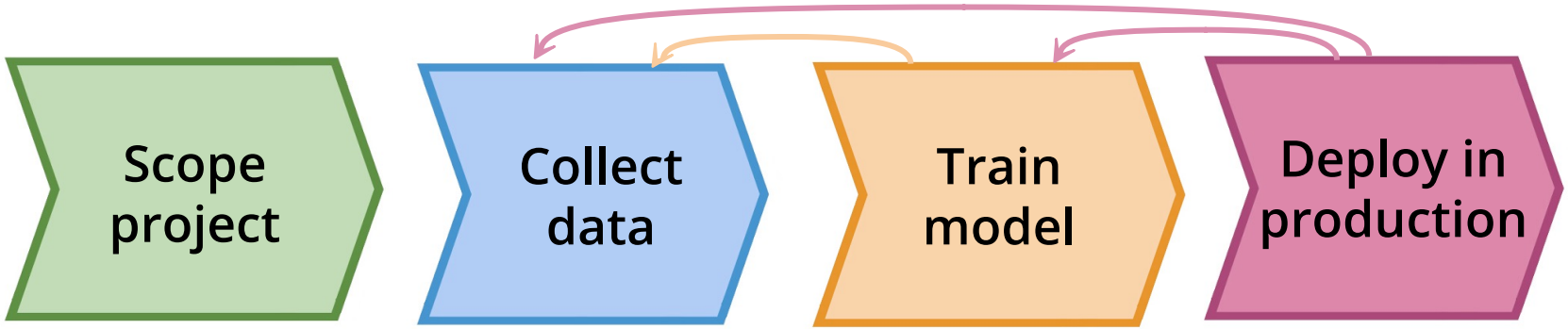


PREP	ACTION
Source and prepare high quality ingredients	Cook a meal
Source and prepare high quality data	Train a model

Lifecycle of an ML Project



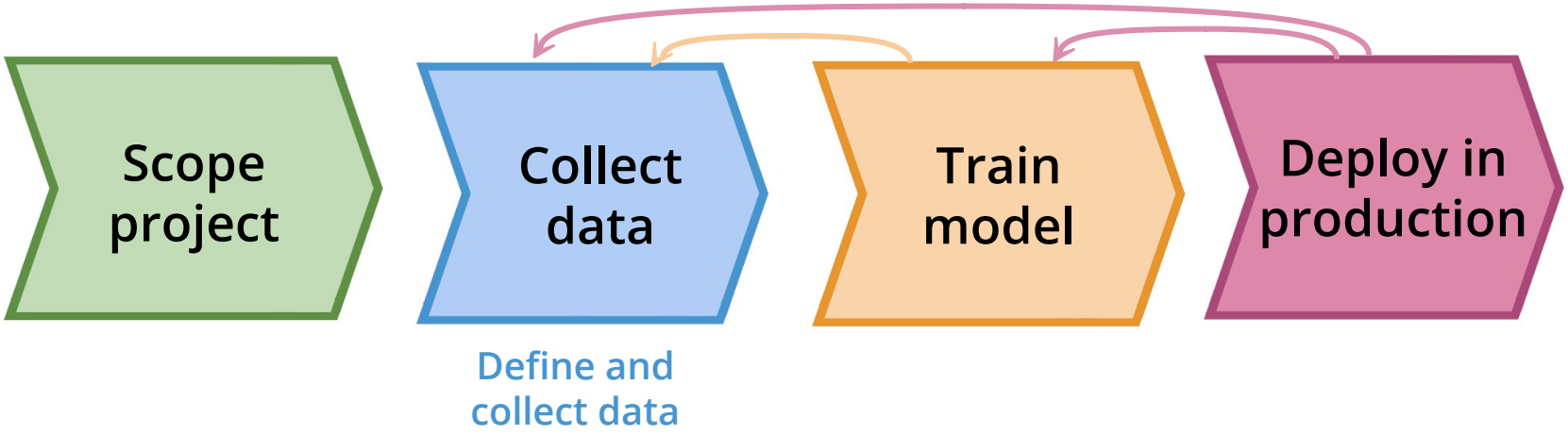
Scoping: Speech Recognition



Define project

Decide to work on speech recognition for voice search

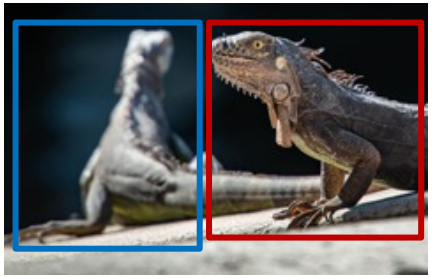
Collect Data: Speech Recognition

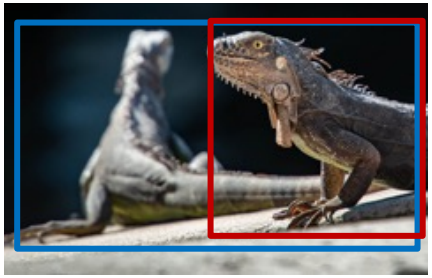


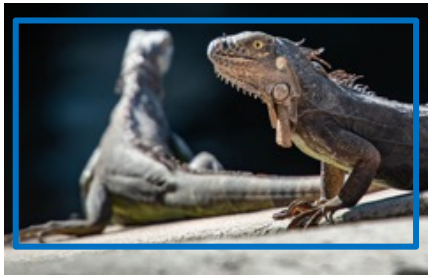
Is the data labeled consistently?

"Um, today's weather"
"Um... today's weather"
"Today's weather"

Iguana Detection Example







Labeling instruction:

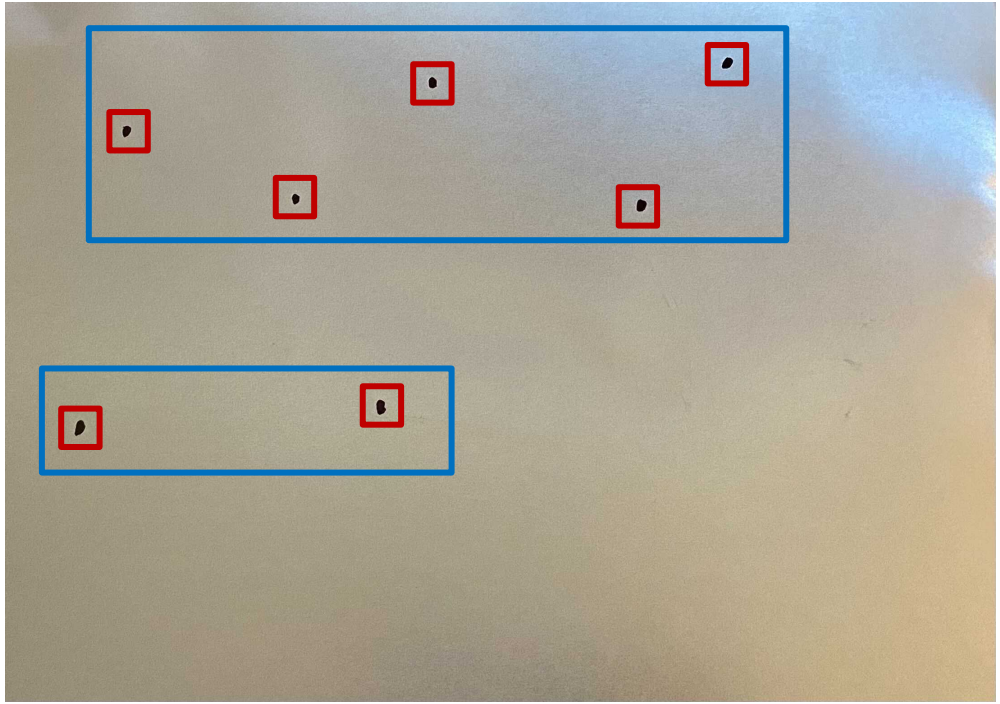
Use bounding boxes to indicate the position of iguanas

Making data quality systematic: MLOps

- Ask two independent labelers to label a sample of images.
- Measure consistency between labelers to discover where they disagree.
- For classes where the labelers disagree, revise the labeling instructions until they become consistent.

Labeler consistency example

Steel defect detection (39 classes). Class 23: Foreign particle defect.

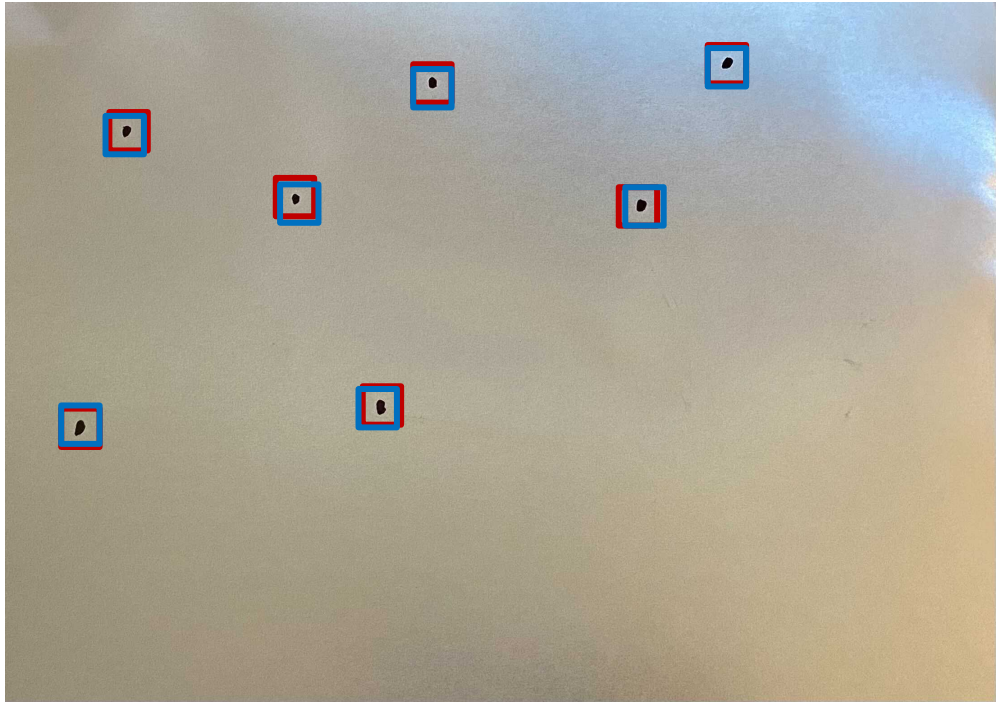


Labeler 1

Labeler 2

Labeler consistency example

Steel defect detection (39 classes). Class 23: Foreign particle defect.



Labeler 1

Labeler 2

Making it systematic: MLOps

Model-centric view

Collect what data you can, and develop a model good enough to deal with the noise in the data.

Hold the data fixed and iteratively improve the code/model.

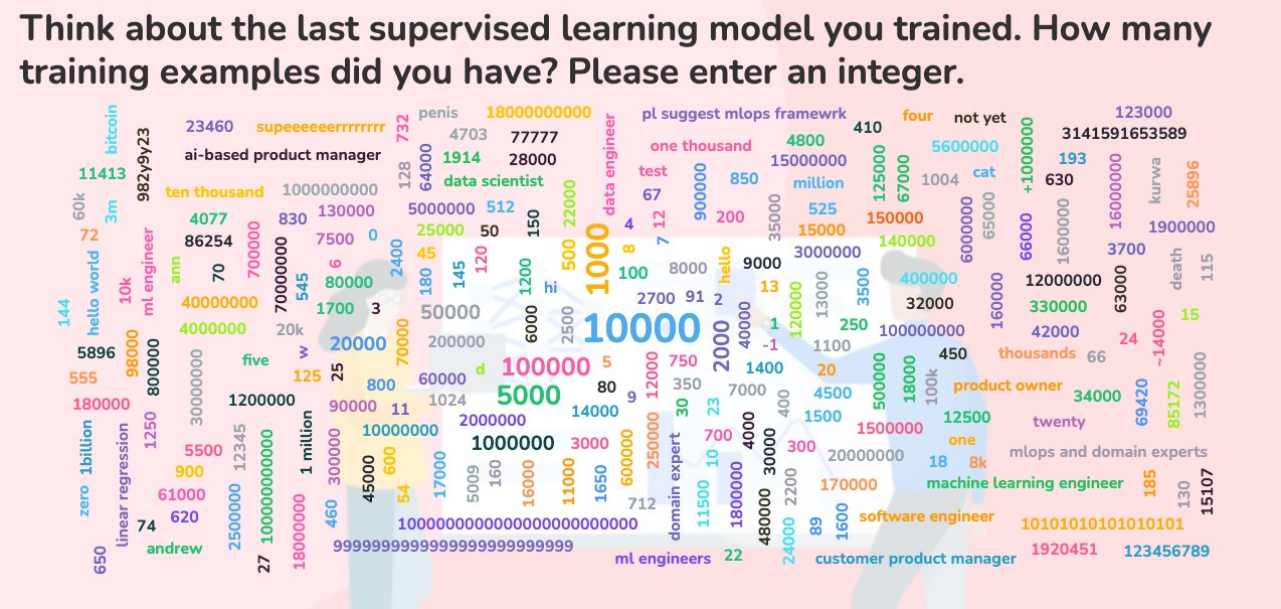
Data-centric view

The consistency of the data is paramount. Use tools to improve the data quality; this will allow multiple models to do well.

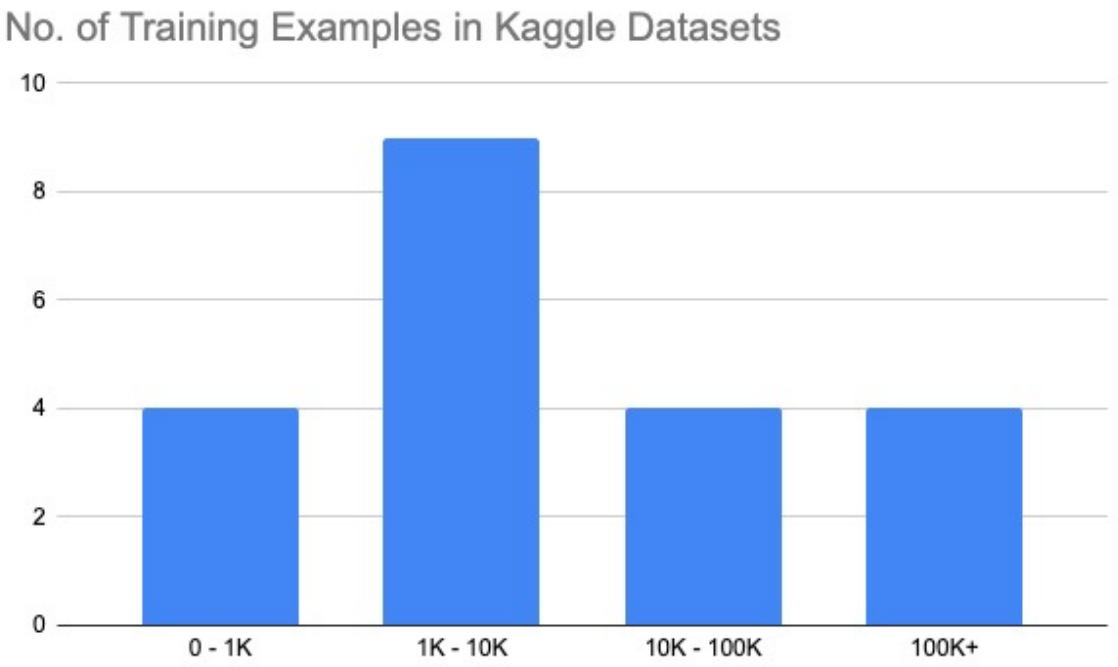
Hold the code fixed and iteratively improve the data.

Audience poll: Think about the last supervised learning model you trained. How many training examples did you have? Please enter an integer.

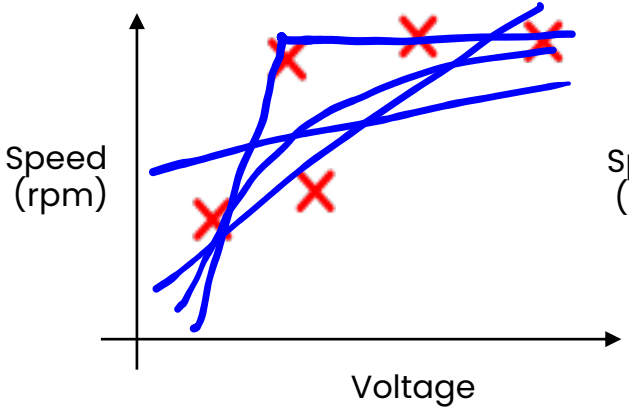
Poll results:



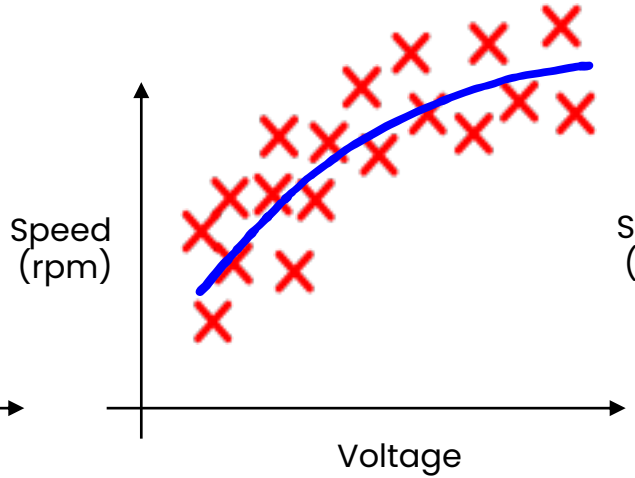
Kaggle Dataset Size



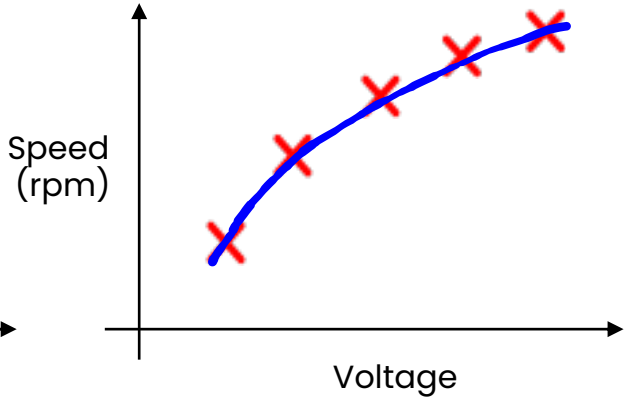
Small Data and Label Consistency



- Small data
- Noisy labels



- Big data
- Noisy labels



- Small data
- Clean (consistent) labels

Theory: Clean vs. noisy data

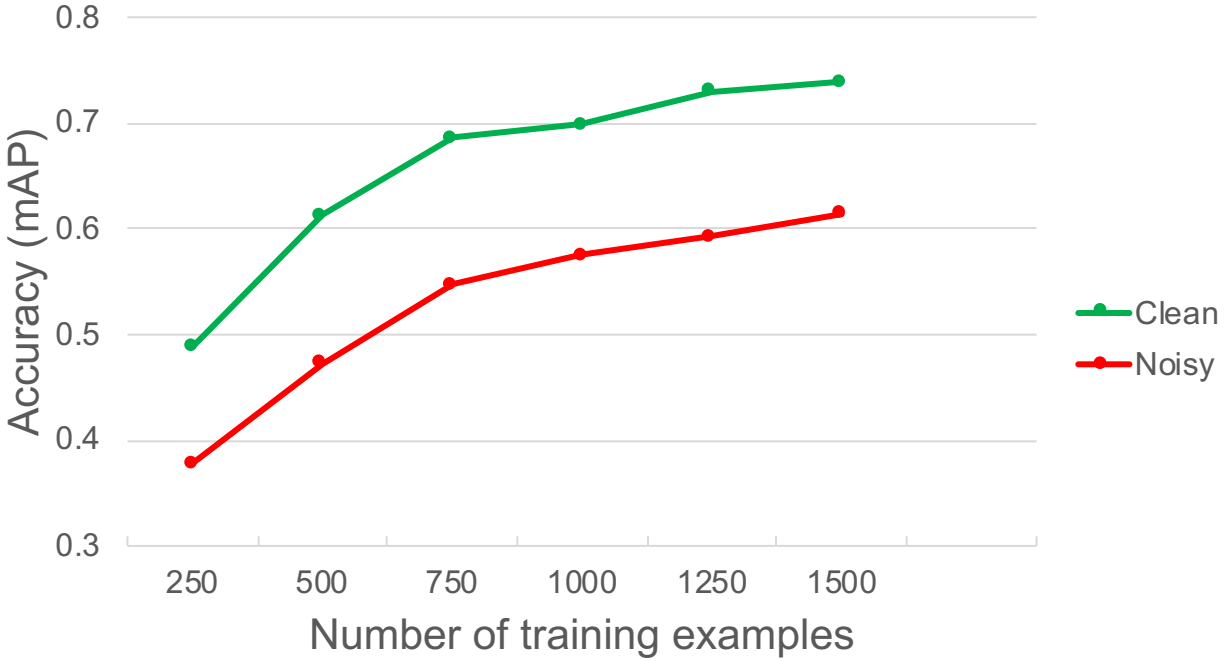
You have 500 examples, and 12% of the examples are noisy (incorrectly or inconsistently labeled).

The following are about equally effective:

- Clean up the noise
- Collect another 500 new examples (double the training set)

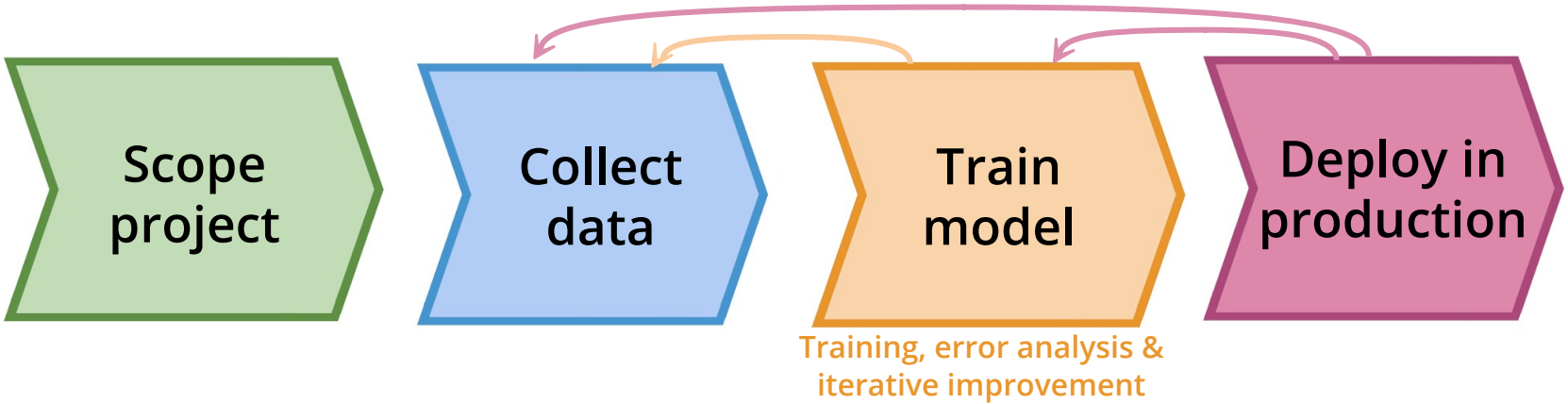
With a data centric view, there is significant of room for improvement in problems with <10,000 examples!

Example: Clean vs. noisy data



Note: Big data problems where there's a long tail of rare events in the input (web search, self-driving cars, recommender systems) are also small data problems.

Train model: Speech Recognition

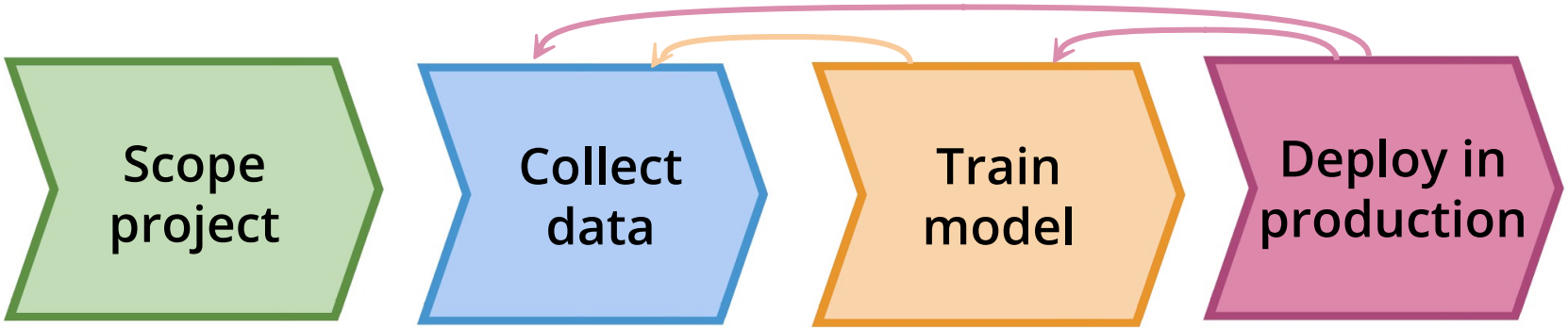


Error analysis shows your algorithm does poorly in speech with car noise in the background. What do you do?

Model-centric view
How can I tune the model architecture to improve performance?

Data-centric view
How can I modify my data (new examples, data augmentation, labeling, etc.) to improve performance?

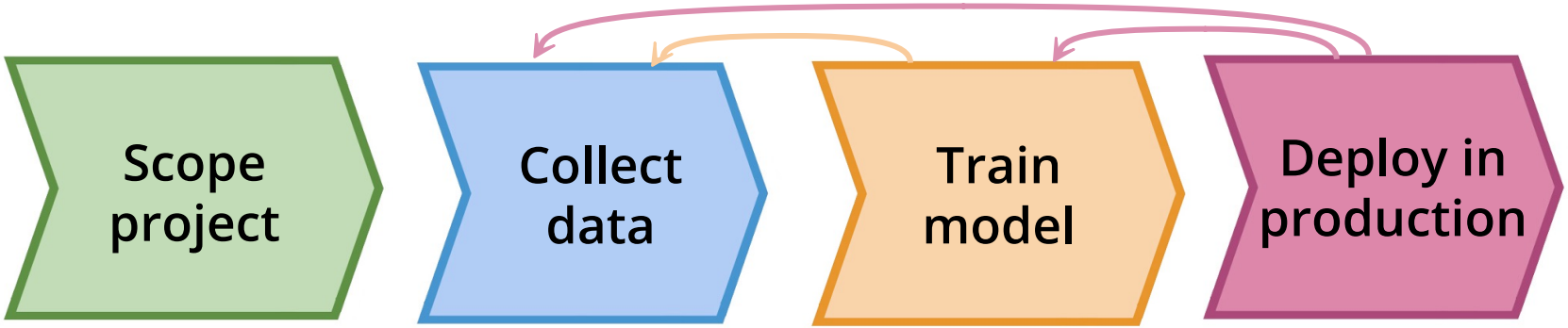
Train model: Speech Recognition



Making it systematic – iteratively improving the data:

- Train a model
- Error analysis to identify the types of data the algorithm does poorly on (e.g., speech with car noise)
- Either get more of that data via data augmentation, data generation or data collection (change inputs x) or give more consistent definition for labels if they were found to be ambiguous (change labels y)

Deploy: Speech Recognition



Monitor performance in deployment, and flow new data back for continuous refinement of model.

- Systematically check for concept drift/data drift (performance degradation)
- Flow data back to retrain/update model regularly

Deploy, monitor and maintain system

Making it systematic: The rise of MLOps

	Code
Creation	Software engineers
Quality/ Infrastructure	DevOps

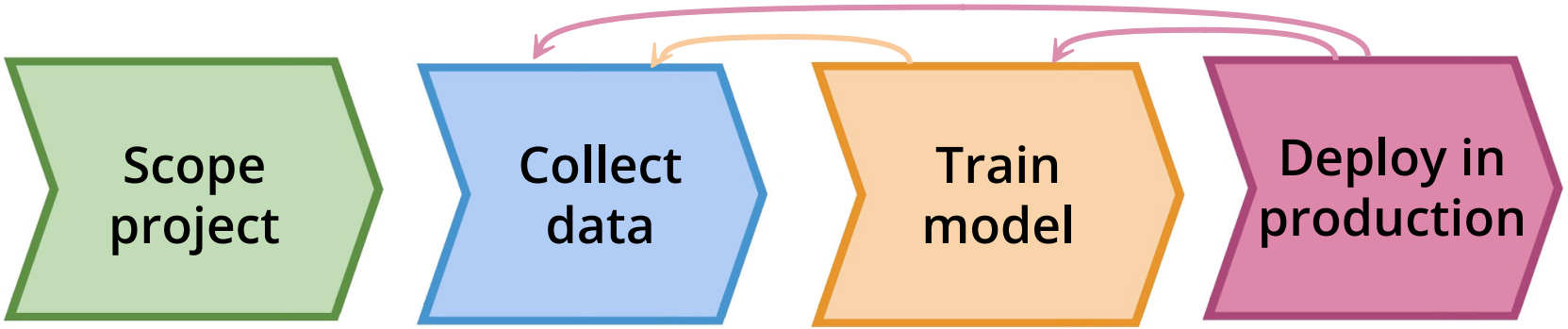
Making it systematic: The rise of MLOps

AI systems	=	**Code**	+	**Data**
Creation		Software engineers		ML engineers
Quality/ Infrastructure		DevOps		MLOps

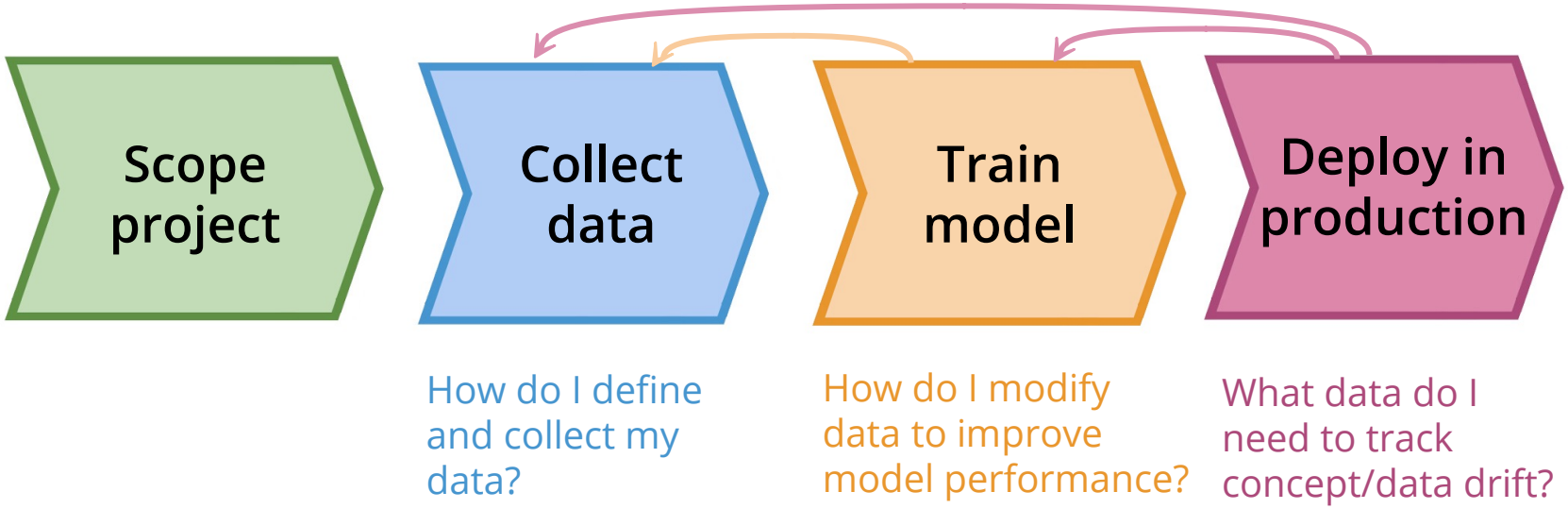
Traditional software vs AI software

Traditional software

AI software

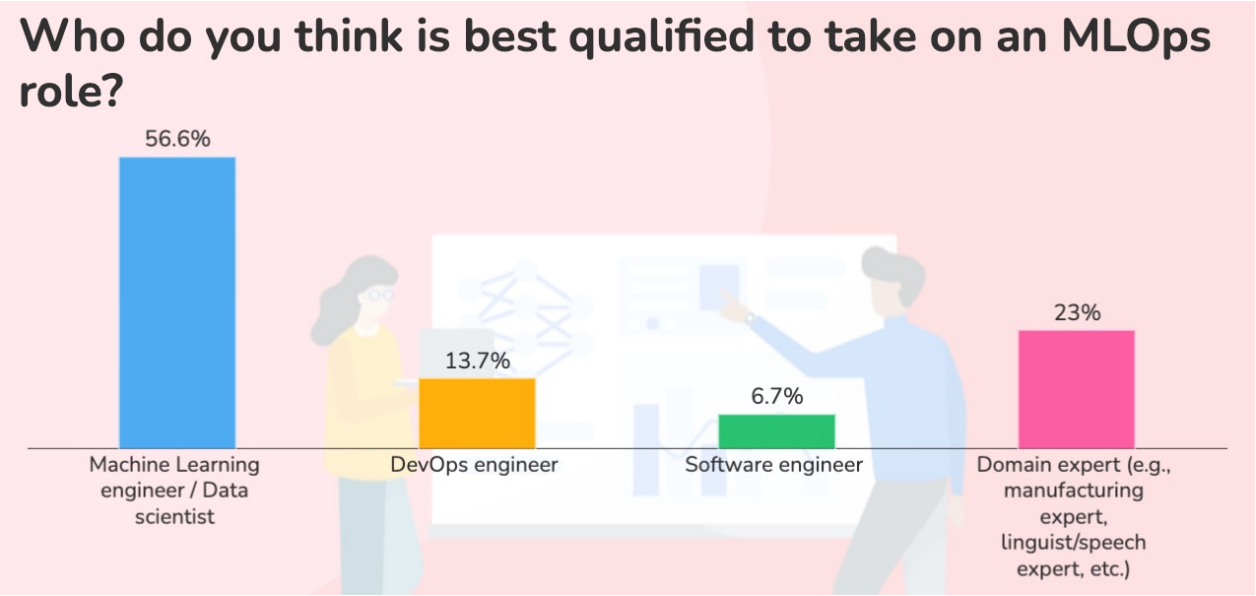


MLOps: Ensuring consistently high-quality data



Audience poll: Who do you think is best qualified to take on an MLOps role?

Poll results:



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

Takeaways: Data-centric AI

MLOps' most important task is to make high quality data available through all stages of the ML project lifecycle.

AI system = Code + Data

Model-centric AI

How can you change the model (code) to improve performance?

Data-centric AI

How can you systematically change your data (inputs x or labels y) to improve performance?

Important frontier: MLOps tools to make data-centric AI an efficient and systematic process.