Data Validation for Data Science

EuroPython July 2022 Tutorial

https://github.com/NatanMish/data_validation

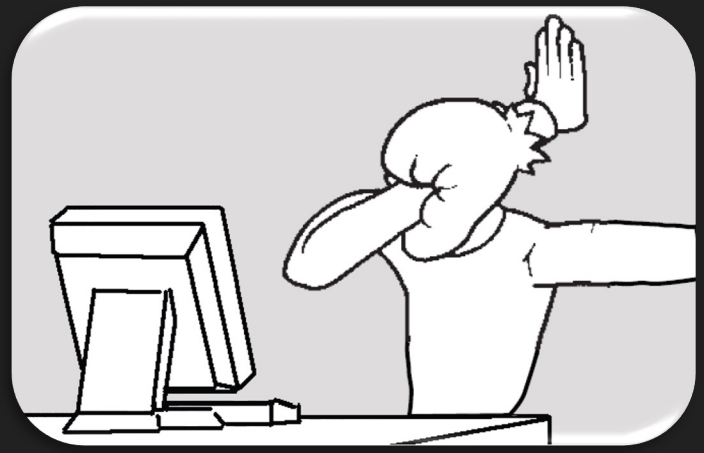
Natan Mish

Machine Learning Engineer

```
    114                 raise ValueError(
    115                     msg_err.format(
--> 116                         type_err, msg_dtype if msg_dtype is not None else X.dtype
    117                     )
    118                 )
```

```
ValueError: Input contains NaN, infinity or a value too large for dtype('float32').
```

Looks familiar?



```
home_data.LotFrontage.isna().sum()
```

```
259
```

```
feature_names = ['LotArea','YearBuilt','1stFlrSF','2ndFlrSF','FullBath','BedroomAbvGr','TotRmsAbvGrd', 'LotFrontage']
X = home_data[feature_names]
y = home_data.SalePrice
regr = RandomForestRegressor(max_depth=2, random_state=0)
regr.fit(X, y)
```

```
    114                 raise ValueError(
    115                     msg_err.format(
--> 116                         type_err, msg_dtype if msg_dtype is not None else X.dtype
    117                     )
    118                 )

ValueError: Input contains NaN, infinity or a value too large for dtype('float32').
```

About me

- Machine Learning Engineer @ Zimmer Biomet
- MSc in Social Data Science from the London School of Economics
- Previously worked in the finance and fintech industries

Also…

- I like finding bugs(especially if they're my own making)

Agenda

1. Who needs data validation anyway?
   - Data Integrity
   - Reliability
   - Readability
   - Familiarity
2. Validation in databases using Great Expectations
3. Validation in training pipelines using Pandera
4. Validation in model serving using Pydantic

Tutorial Instructions

- Click the "Open in Colab" link straight from the notebook on GitHub.

- Alternatively, use your own environment.
- Environment requirements: Python 3.8 and up with Jupyter installed, and Git for cloning the tutorial repo.
- Clone the repository from GitHub using the CLI:

```
git clone https://github.com/NatanMish/data_validation.git
```

Dataset – Ames house prices prediction

- As featured in Kaggle house prices prediction competition
- Simple to use and understand, used extensively in tutorials and courses.
- Variety of types of features – numerical, strings, categorical (80 different fields).
- Target variable is the house price.

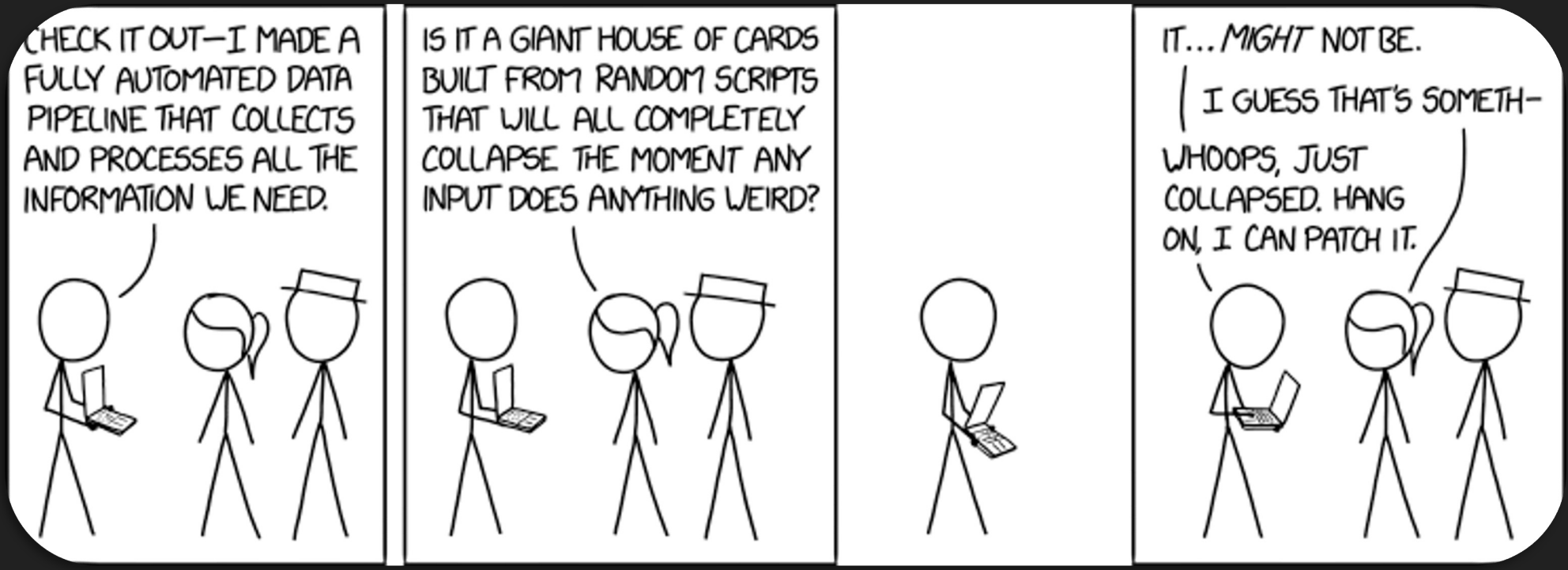
What does Data Validation mean?

- Data validation is the practice of checking the integrity, accuracy and structure of data before it is used.
- Useful in any kind of work that uses data – software development, research, accounting, military intelligence.
- We can validate types, ranges, consistency and incorporate business logic to make our project more robust.

How can data become invalid?

- Human error – for example when tagging labels for training models.
- Errors in the data generating code.
- Outdated/expired data sets.
- Bugs or changes in upstream data pipelines.

Why validate your data?



1. Data integrity

- Helps align the integrity throughout the entire system.
- This only relates to the logical integrity of the data, and not the physical one.

2. Reliability

- Prevent and detect bugs from the very first interaction with the data.
- Prevent and detect bugs throughout the data science product life cycle.
- Minimize surprises – know what input to expect and where to expect it.
- Can help with an early detection of a data drift.

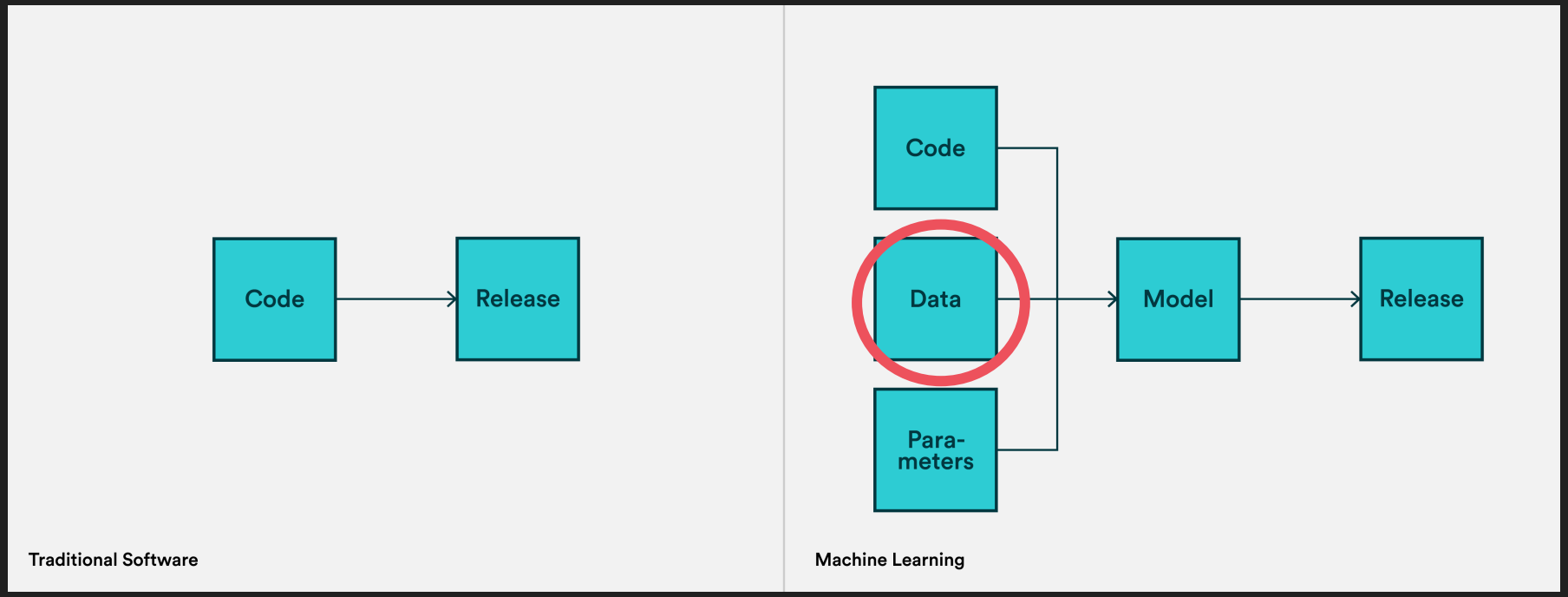
3. Readability

- Harness Python's type hints feature
- Some IDEs will highlight if something is wrong before we actually run anything.

4. Familiarity

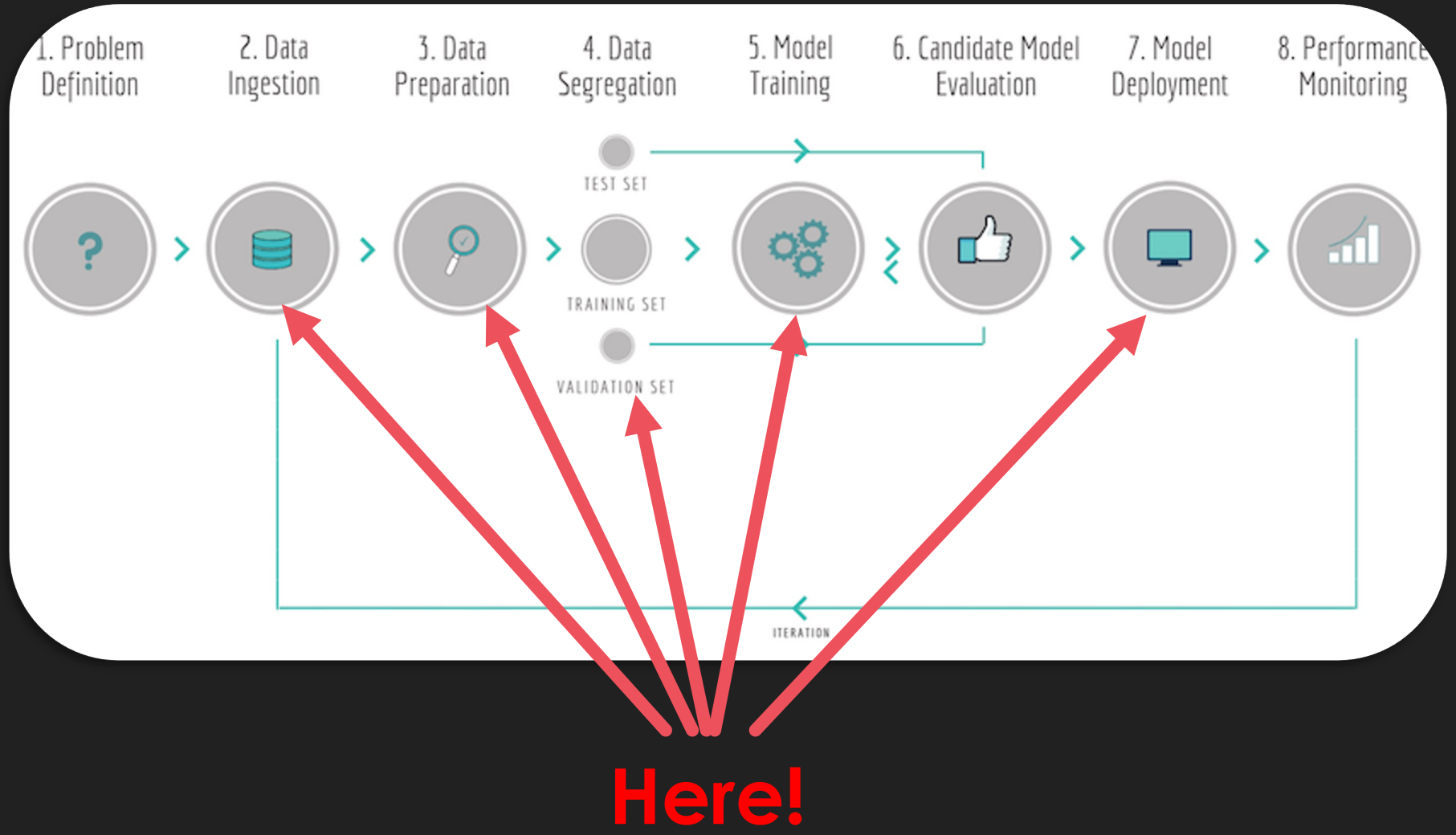
- Implementing validation steps will force us to get to know our data better.
- This helps us as data professionals gain domain knowledge.
- Find out what are the limits, gaps, types and ranges our database consists of.
- Some of the tools allow for creating automated data documentation.

Key part in the MLOps process



https://valohai.com/blog/difference-between-devops-and-mlops/

At what point in the model lifecycle should we implement data validation?



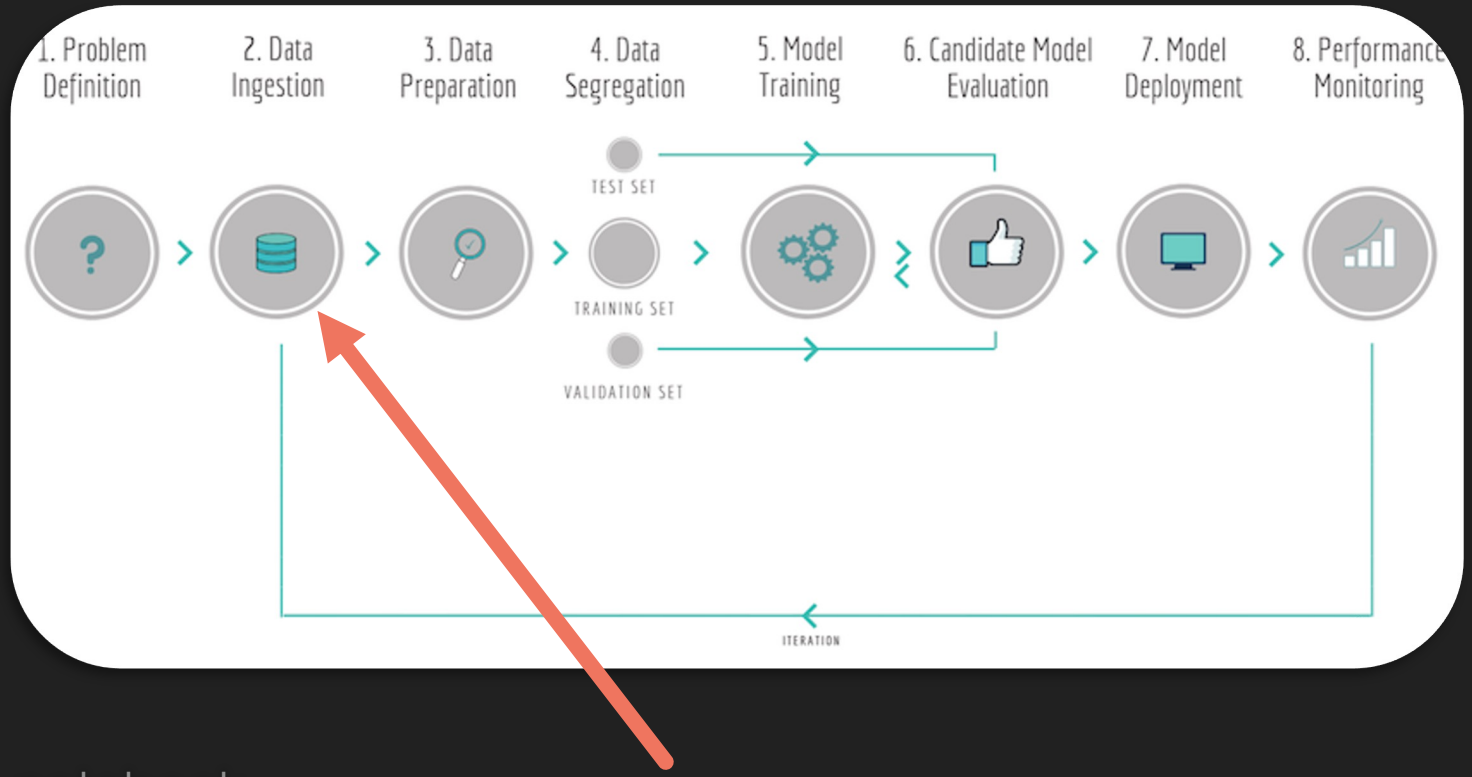
https://towardsdatascience.com/not-yet-another-article-on-machine-learning-e67f8812ba86

Hands on tutorial

1. Dataset validation with  great_expectations
2. Training pipelines validation with pandera
3. Model serving validation with Pydantic

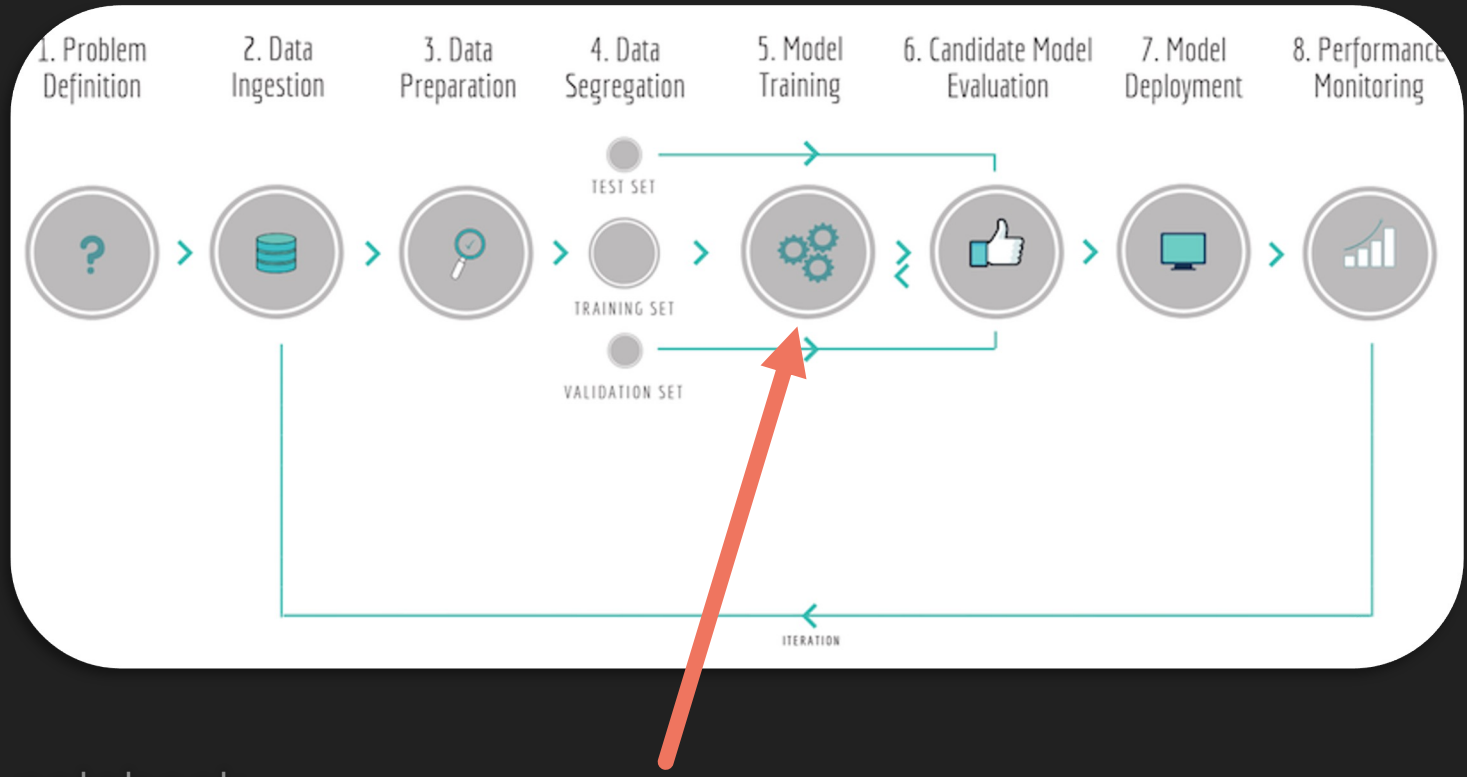
*Each of the tools could potentially be used for any of the components

Database validation with Great Expectations



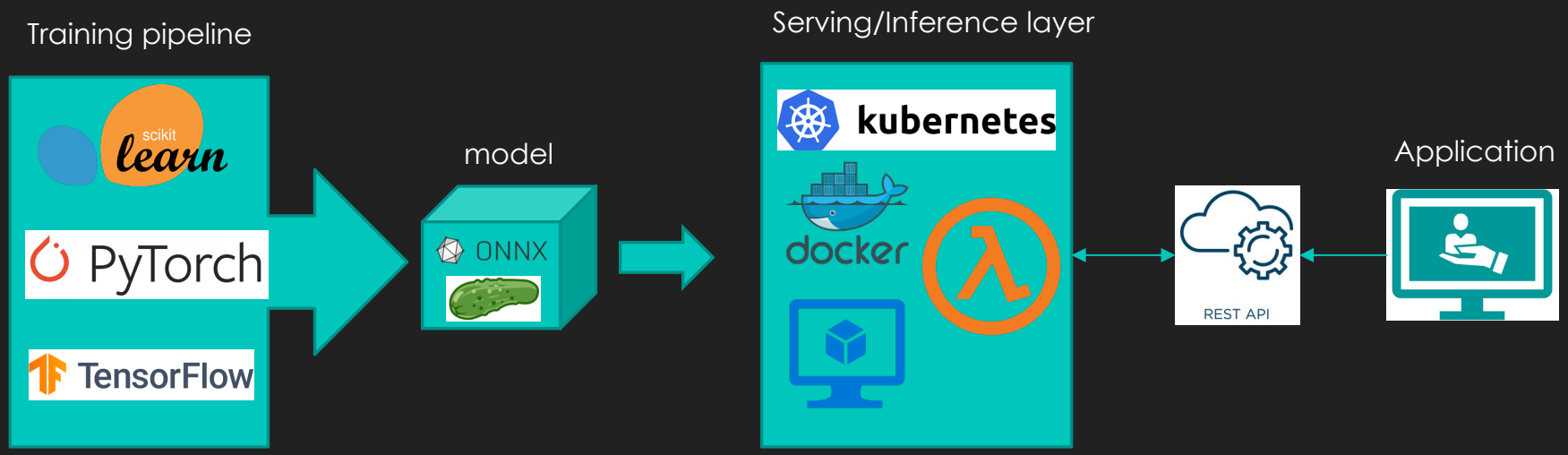
- Follow the instructions in the notebook.

Training pipeline validation with Pandera

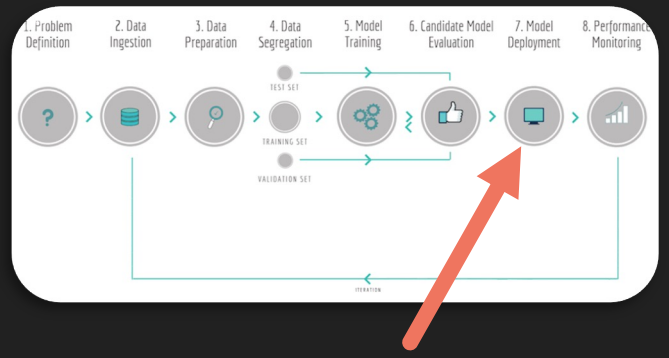


- Follow the instructions in the notebook.

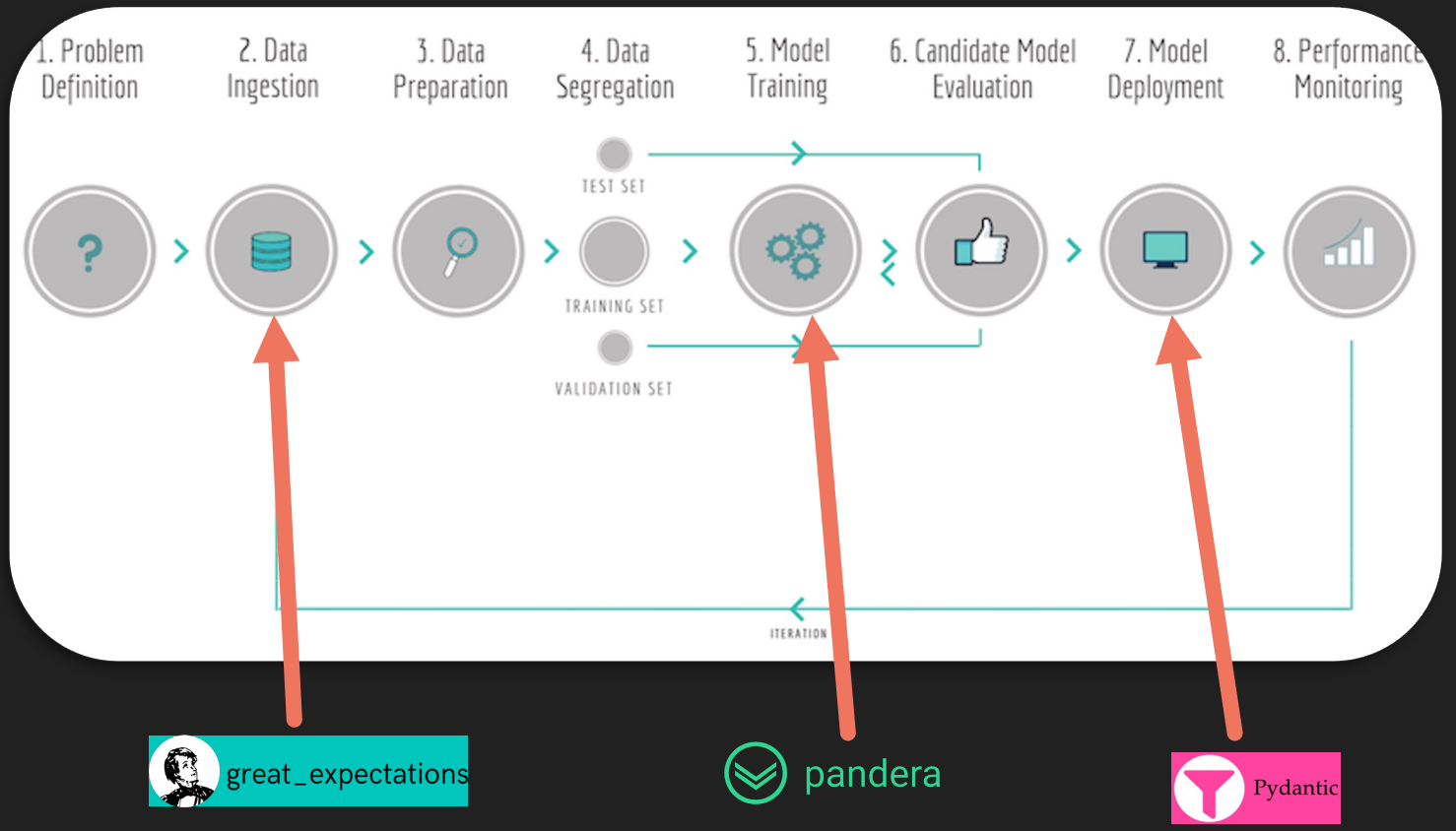
Serving data validation with Pydantic



- Follow the instructions in the notebook.



To wrap things up…



Thank you for listening!

Any questions?