Data Collection in the Wild

Vladimir Haltakov
BMW Group

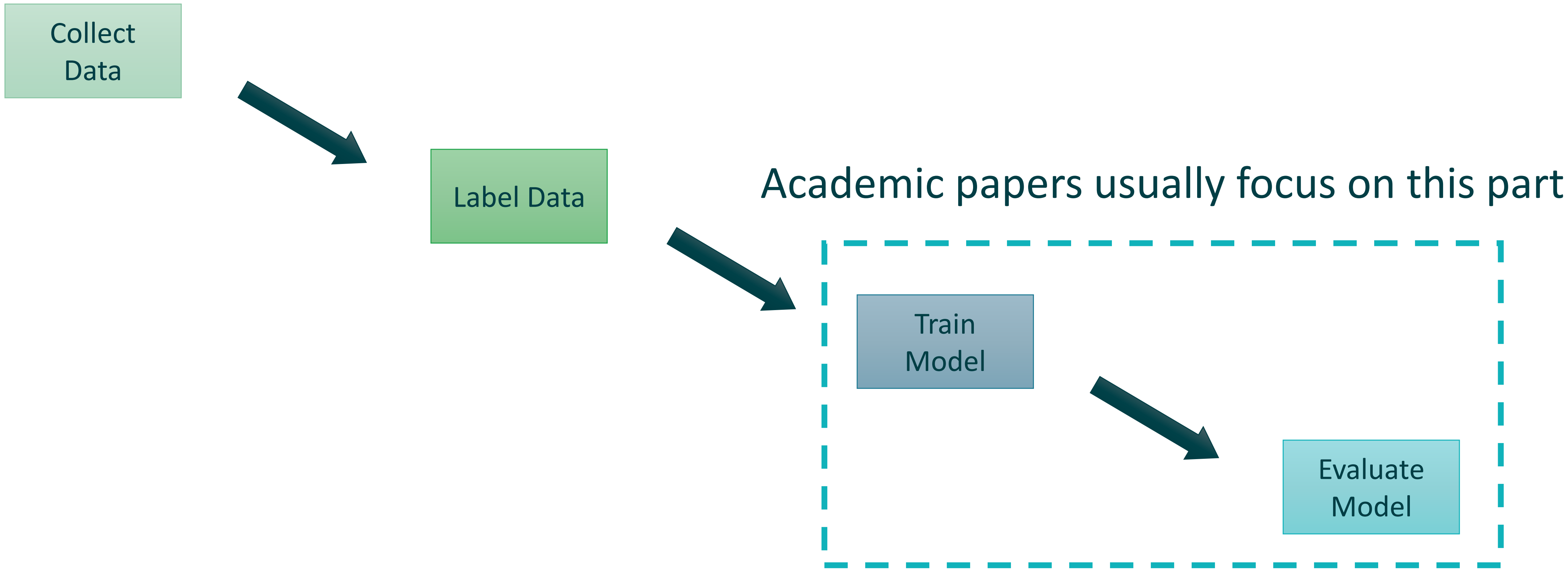
2021 embedded vision summit
VIRTUAL | MAY 25-27

BMW GROUP

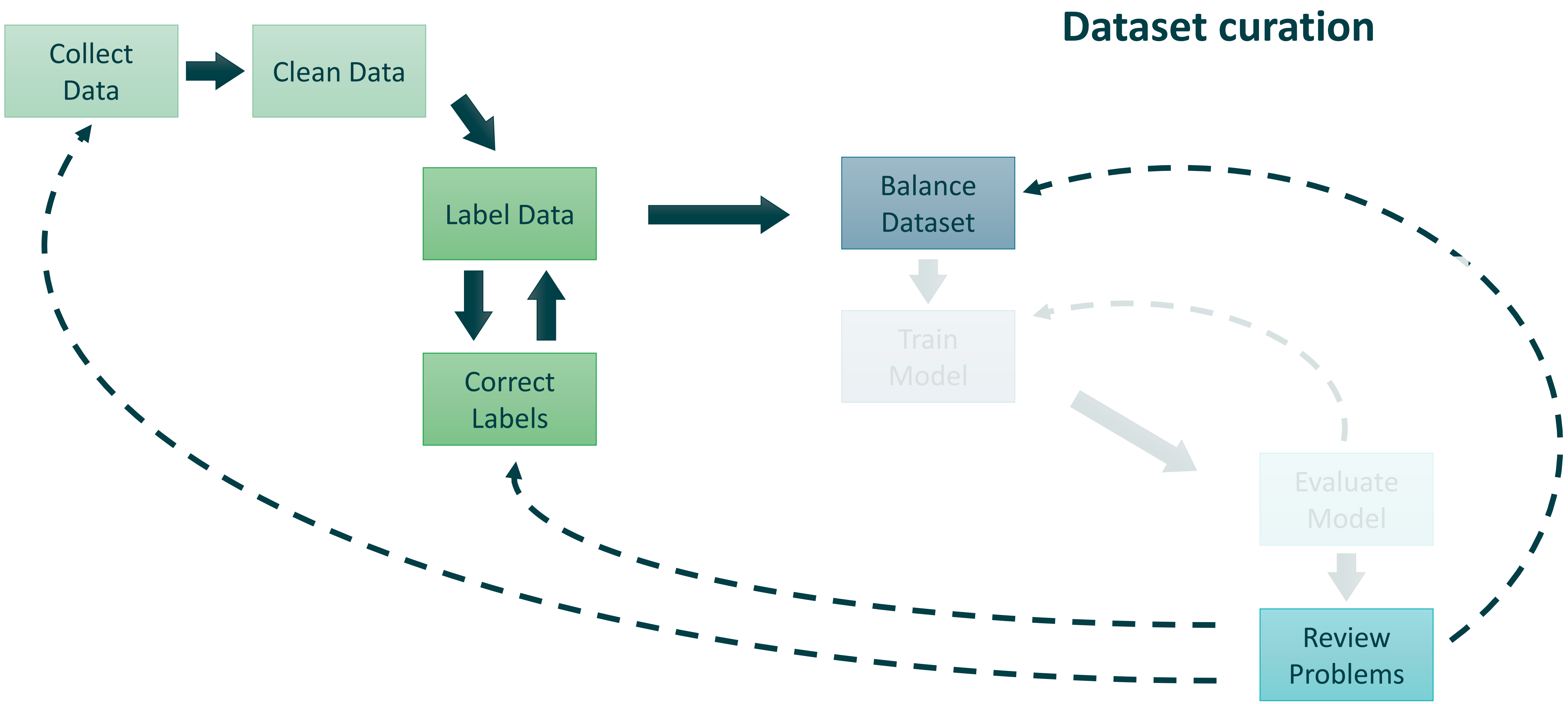
Data Collection for Real World Applications

- **Traffic sign recognition**
  - Robust detection and classification of traffic signs in **challenging conditions**
  - Support of all **country specific** traffic sign variants
  - **Organized** a **worldwide** data collection campaign
- **Traffic light recognition**
  - Robust detection performance in **rare situations**
  - Support for all **traffic light variants**
  - Created a **research dataset**

In a Perfect World...



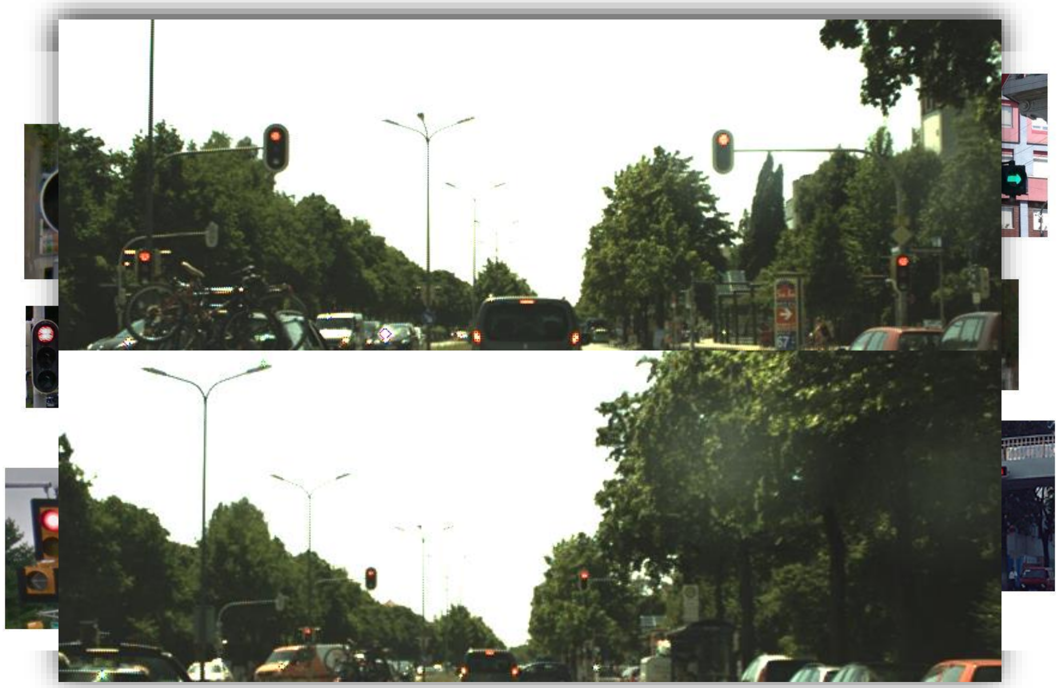
… in Reality



Dataset curation
Collect Data
Clean Data
Label Data
Correct Labels
Balance Dataset
Train Model
Evaluate Model
Review Problems

Data Collection

- Sample the **real world distribution** as accurately as possible.
  - Lighting conditions
  - Distance and viewpoints
  - Object variations
  - Problem specific variations
- **Define the boundaries** of your dataset
- **Plan the collection** of the images carefully

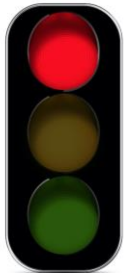
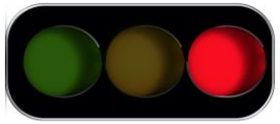


Beware of Sampling Bias

The dataset **does not** accurately **represent** the real **distribution** → **biased dataset**

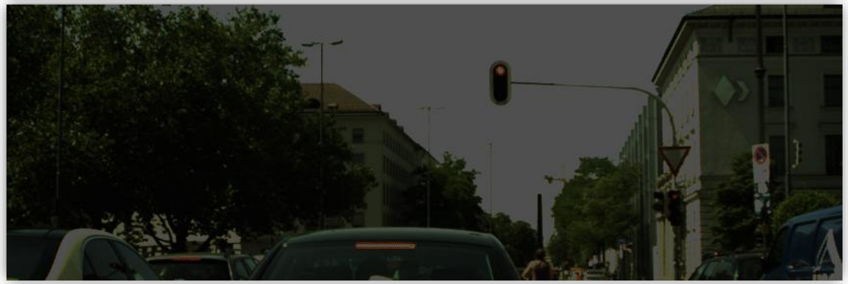
⚠ We cannot detect the bias during development ⚠

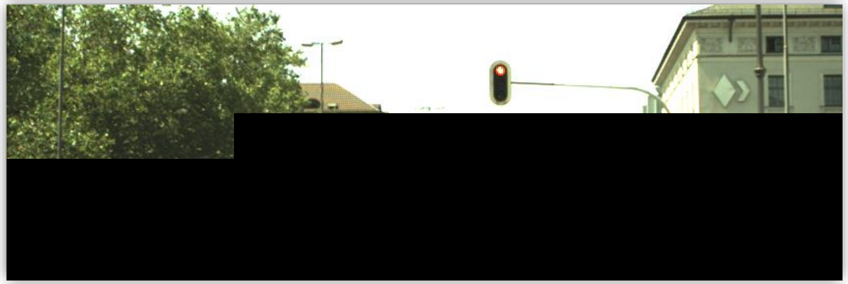
- Both **training** and **test** data will contain the **same bias**
- The model will achieve **high score** on the **test data**
- The model will **perform poorly** when **deployed** in the **real world**

If we train only with  we will not be able to detect 

Data Cleaning

- **Remove bad samples** from the dataset
  - **Overexposed** or **underexposed** images
  - Images in **irrelevant** situations
  - **Faulty** images
- Bad images **reduce** the **performance** of our model





Label the Data

- Different **types of labeling**: manual, semi-automatic, fully-automatic, self-supervised
- **Which data to label?**
  - Can we label **all data**?
  - Can we perform **per-labeling** during data collection?
  - How to label according to the **real distribution**? (avoid Sampling Bias)
- **Iterative process**: label → train → evaluate → choose difficult samples → label

Label Correction

- You will get **wrong labels!** Humans make mistakes…
- Wrong labels can **hurt the model performance** and lead to **wrong conclusions**
- Plan a **process** to **correct labels**
  - Label samples **multiple times**
  - **Spot checks** before training to find systematic problems
  - **Improve** labeling **guidelines** and **tools**
  - **Review** test results and **fix** labels

Label Errors in Research Datasets

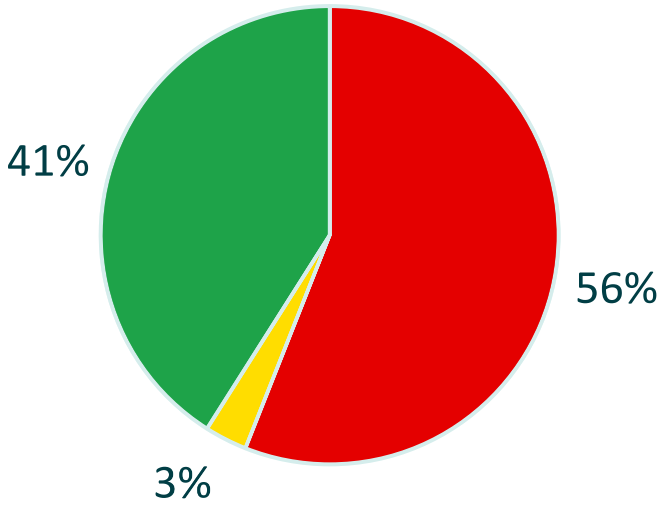
- **New study** from MIT on **label errors** in popular research datasets (e.g. **ImageNet**)
  - *Northcutt et al. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks, 2021*
- On **average 3.4% label errors** in the test dataset (5.8% in ImageNet)
- Models performing **worse on the wrong labels**, perform **better on the corrected labels**!
  - The better models are often much **smaller**!

Balance the Dataset

Problem: some classes appear more often than others.

⚠ Classifiers **ignore underrepresented** classes ⚠

Traffic light colors distribution



97% accuracy if yellow is ignored completely

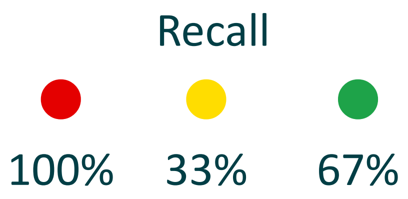
Balance the Dataset

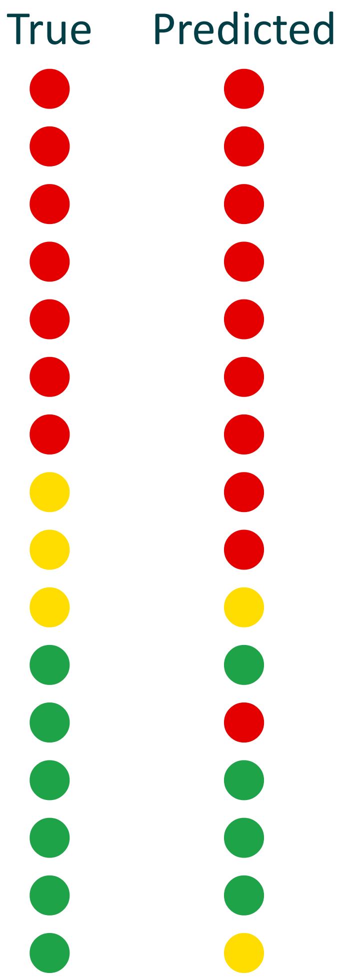
Problem: some classes appear more often than others.

⚠ Classifiers **ignore underrepresented** classes ⚠

Use the right **evaluation metrics:**

- Recall





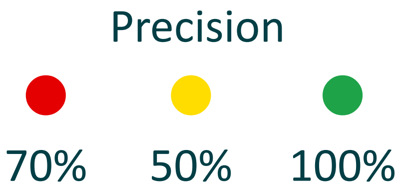
Balance the Dataset

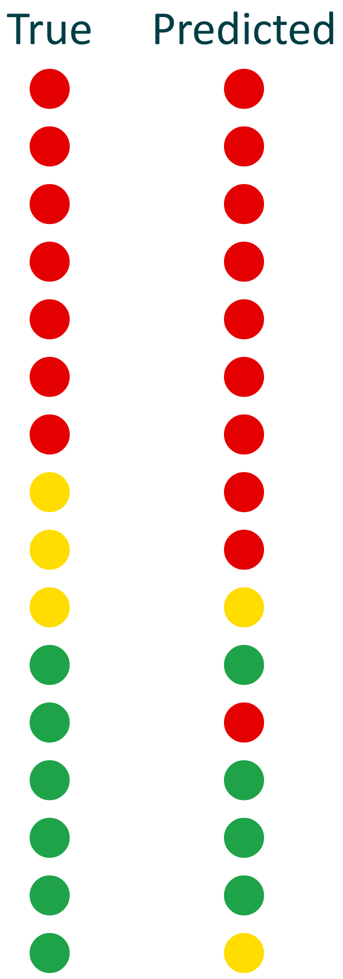
Problem: some classes appear more often than others.

⚠ Classifiers **ignore underrepresented** classes ⚠

Use the right **evaluation metrics:**

- Recall
- Precision





Balance the Dataset

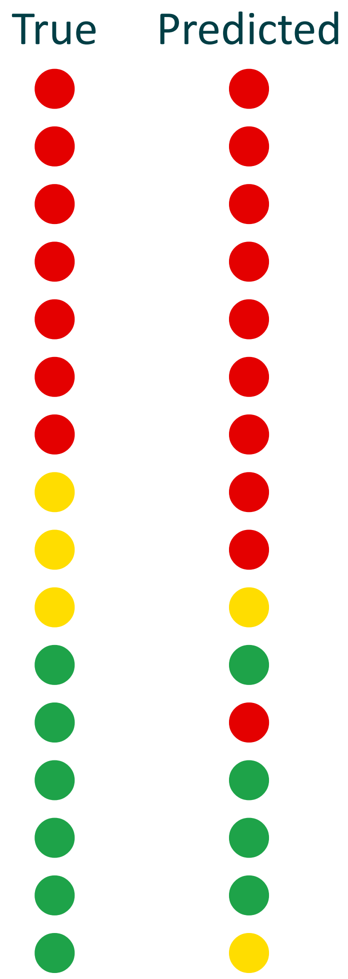
Problem: some classes appear more often than others.

⚠ Classifiers **ignore underrepresented** classes ⚠

Use the right **evaluation metrics:**

- Recall
- Precision
- Confusion Matrix

True \ Predicted			
	100%	0%	0%
	67%	33%	0%
	17%	17%	66%



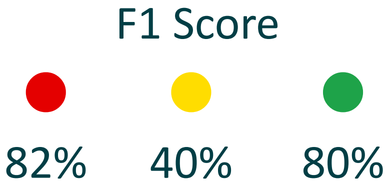
Balance the Dataset

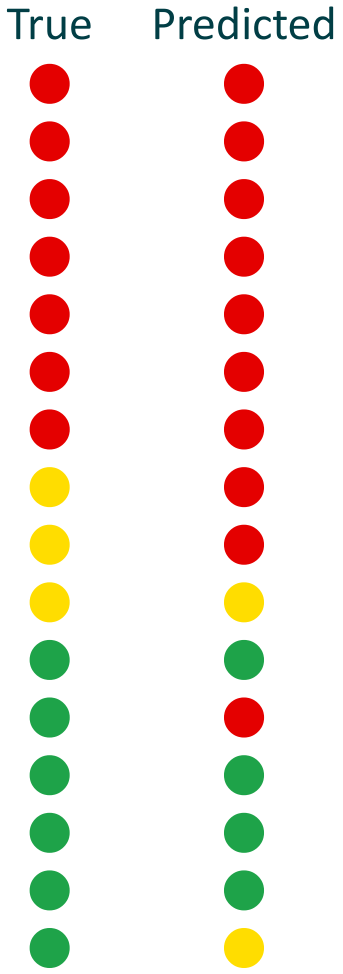
Problem: some classes appear more often than others.

⚠ Classifiers **ignore underrepresented** classes ⚠

Use the right **evaluation metrics:**

- Recall
- Precision
- Confusion Matrix
- **F1 score**



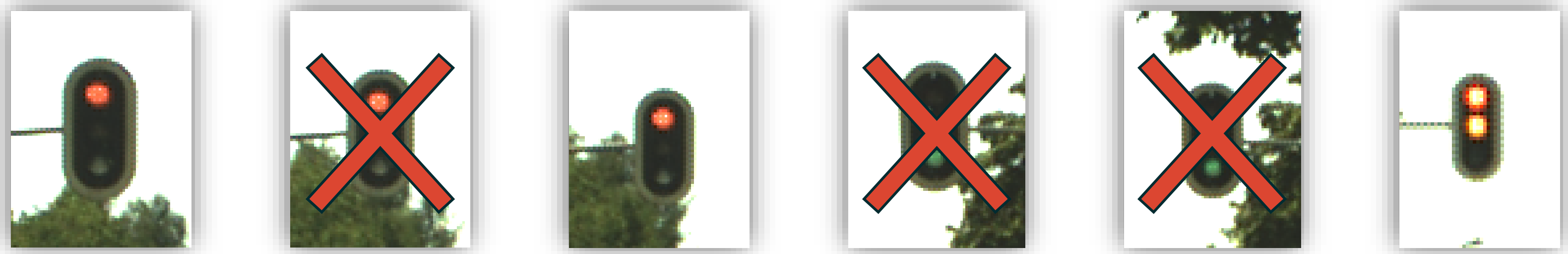


Balance the Dataset - Undersampling

Remove examples of the **dominant** classes

- **Randomly** throw away samples

We lost all green samples!



Balance the Dataset - Undersampling

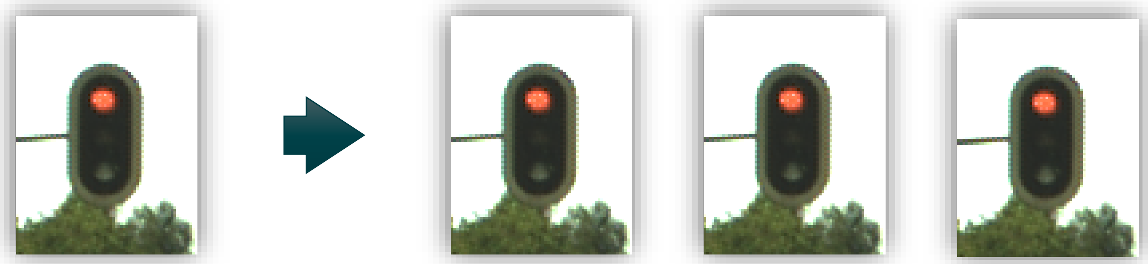
Remove examples of the **dominant** classes

- **Randomly** throw away samples
- Throw away **similar images**
  - Compute image **features** (e.g. using a pretrained CNN)
  - **Cluster** images by visual appearance (e.g. k-means, DBSCAN)
  - **Remove similar** samples (e.g. Near-Miss, Tomek Links)

Balance the Dataset - Oversampling

Add **new examples** from the **underrepresented** classes

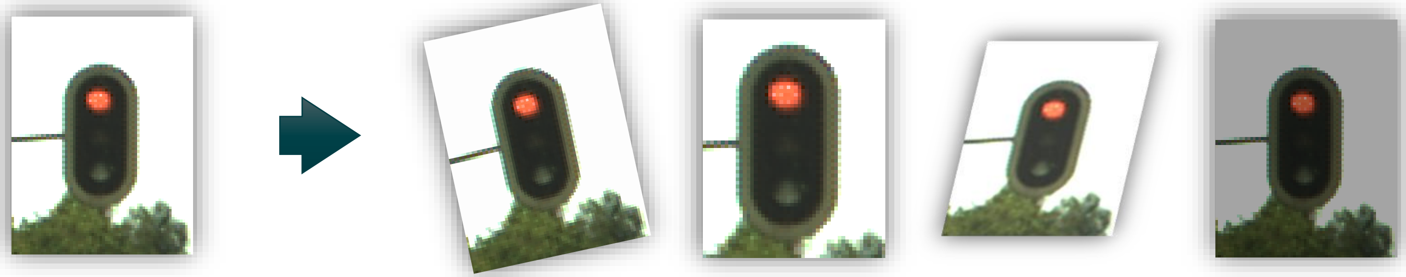
- **Repeat** samples (prone to overfitting)



Balance the Dataset - Oversampling

Add **new examples** from the **underrepresented** classes

- **Repeat** samples (prone to overfitting)
- Data **augmentation** (e.g. rotate, flip, zoom, skew, change color)



Balance the Dataset - Oversampling

Add **new examples** from the **underrepresented** classes

- **Repeat** samples (prone to overfitting)
- Data **augmentation** (e.g. rotate, flip, zoom, skew, change color)
- **SMOTE** (Synthetic Minority Oversampling Technique)
  - Create new samples by combining samples in feature space

Balance the Dataset - Oversampling

Add **new examples** from the **underrepresented** classes

- **Repeat** samples (prone to overfitting)
- Data **augmentation** (e.g. rotate, flip, zoom, skew, change color)
- **SMOTE** (Synthetic Minority Oversampling Technique)
  - Create new samples by combining samples in feature space
- **Synthetic** images (GAN, simulation) – render completely new images

Richter et al. Playing for Data: Ground Truth from Computer Games. ECCV 2016

Balance the Dataset - Adapt the Loss Function

Set higher **penalties** for underrepresented classes in the loss function

- No changes to the data needed
- Similar effect as removing or duplicating samples
- **Finer control** on the weights

Examples

- PyTorch: `torch.nn.CrossEntropyLoss(weight=None, ...)`
- TensorFlow: `tf.keras.Model.fit(class_weight=None, ...)`

Evaluate Model and Review Problems

Bad performance on the **test** dataset? **Problem** with the **model**?

A lot of the times the **problem** is in the **dataset** and not in the model**:**

- **Bad data** samples (remove)
- **Wrong** labels (correct)
- **Bugs** in the **evaluation** metrics (fix code)
- **Lack** of **training data** (collect and label more)

Dataset **curation** is an **iterative process**!

Beware of Simpson's Paradox

Evaluate model on current data

- 1000 training and 1000 test samples → **90%** accuracy

Label **additional 200 samples**, retrain and evaluate again

- 1100 training and 1100 test samples → **85%** accuracy

The additional data breaks the model? Should we remove it?

Beware of Simpson's Paradox

Evaluate model on current data

- 1000 training and 1000 test samples → **90%** accuracy

Label **additional 200 samples**, retrain and evaluate again

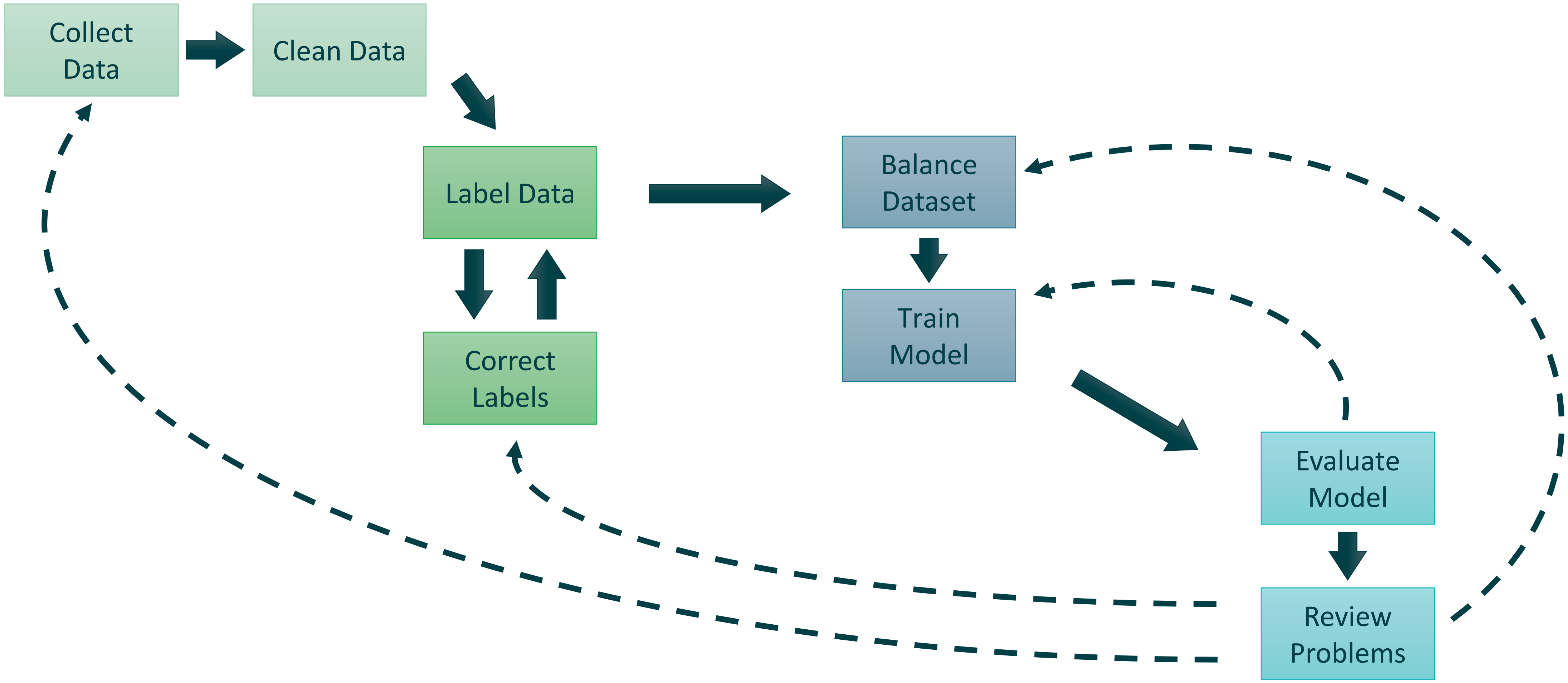
- 1100 training and 1100 test samples → **85%** accuracy

The model actually got better!

	Initial model (1000 samples training)	**Retrained model (1100 samples training)**
Accuracy on initial 1000 samples	90% (900/1000)	91% (910/1000)
Accuracy on new 100 samples	-	25% (25/100)
Overall accuracy	90% (900/1000)	85% (935/1100)

The new samples are much more difficult

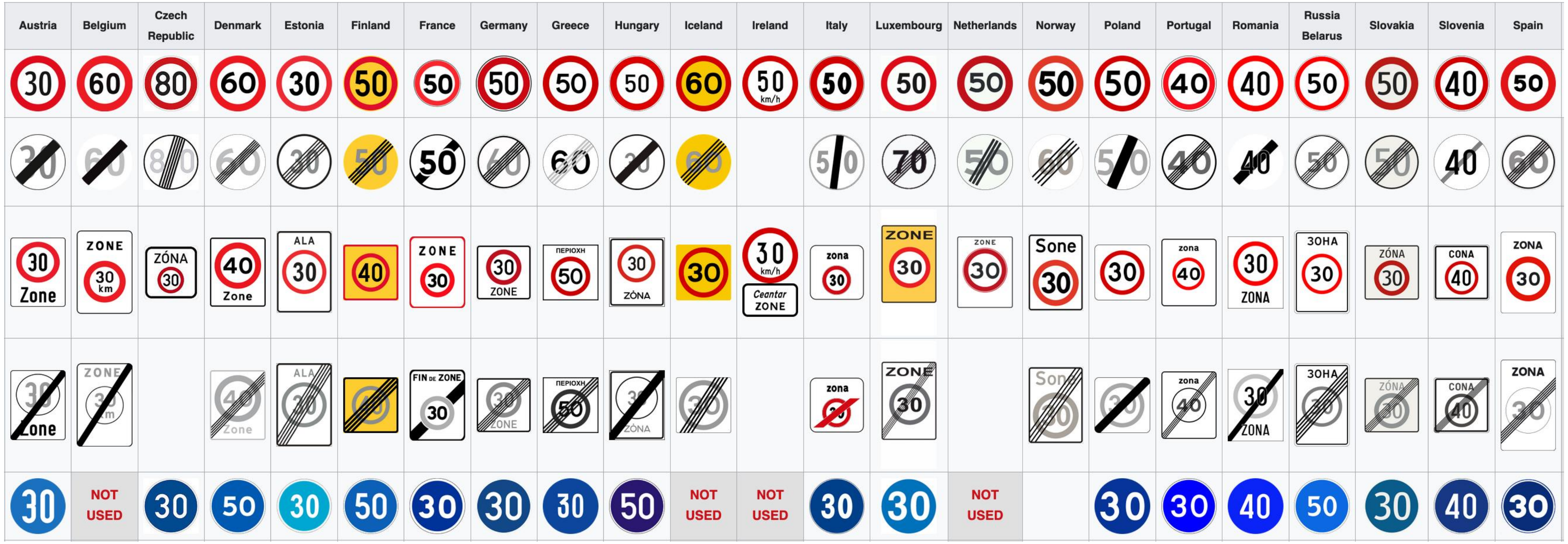
Dataset Curation



Concept Drift

Your model starts performing **worse with time** when **deployed**

The problem - the **real world changes!**



Source: https://en.wikipedia.org/wiki/Comparison_of_European_road_signs

Concept Drift

Your model starts performing **worse with time** when **deployed**

The problem - the **real world changes!**

Source: https://www.arabianbusiness.com/transport/402769-new-abu-dhabi-speed-signs-include-140kph-160kph-limits

Concept Drift

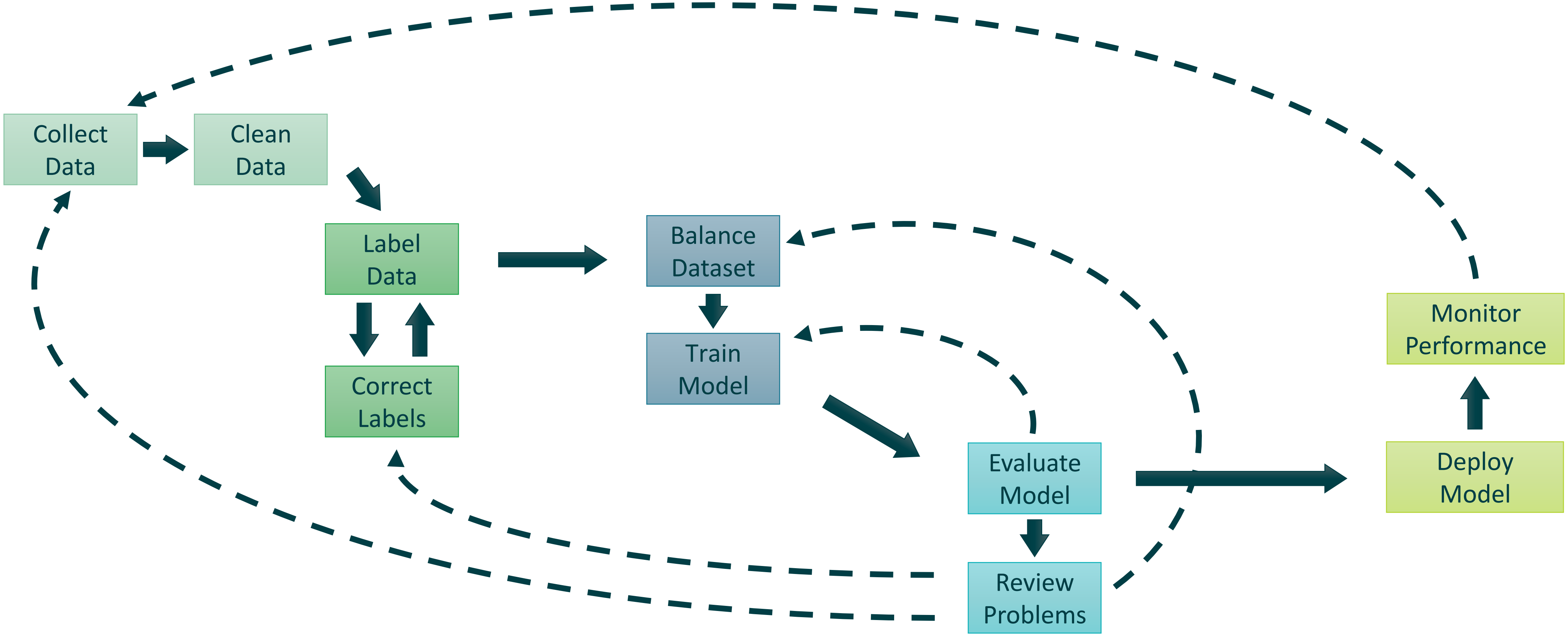
Your model starts performing **worse with time** when **deployed**

The problem - the **real world changes!**

Be prepared to **adapt your model** after it is deployed

- **Continuously evaluate** the performance of the deployed model
- Define a process to **collect data** from **production**
- Define a process to **retrain** your model **continuously** and **redeploy**

Concept Drift - Monitoring the Model in Production



Summary

- **Dataset curation** is crucial for a **good** model **performance**
- Dataset curation is an **iterative process**
- Beware of **common biases** that may lead to wrong conclusions
- Be prepared to handle **concept drift**

Resources

Imbalanced-learn Python library

https://github.com/scikit-learn-contrib/imbalanced-learn

Survey on deep learning with class imbalance

https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0192-5

Traffic lights recognition dataset

http://campar.in.tum.de/Chair/ProjectTrafficLightsDetection

Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks

https://arxiv.org/abs/2103.14749

2021 Embedded Vision Summit

Watch my other talk

"Is my Model Performing Well? It Depends…"