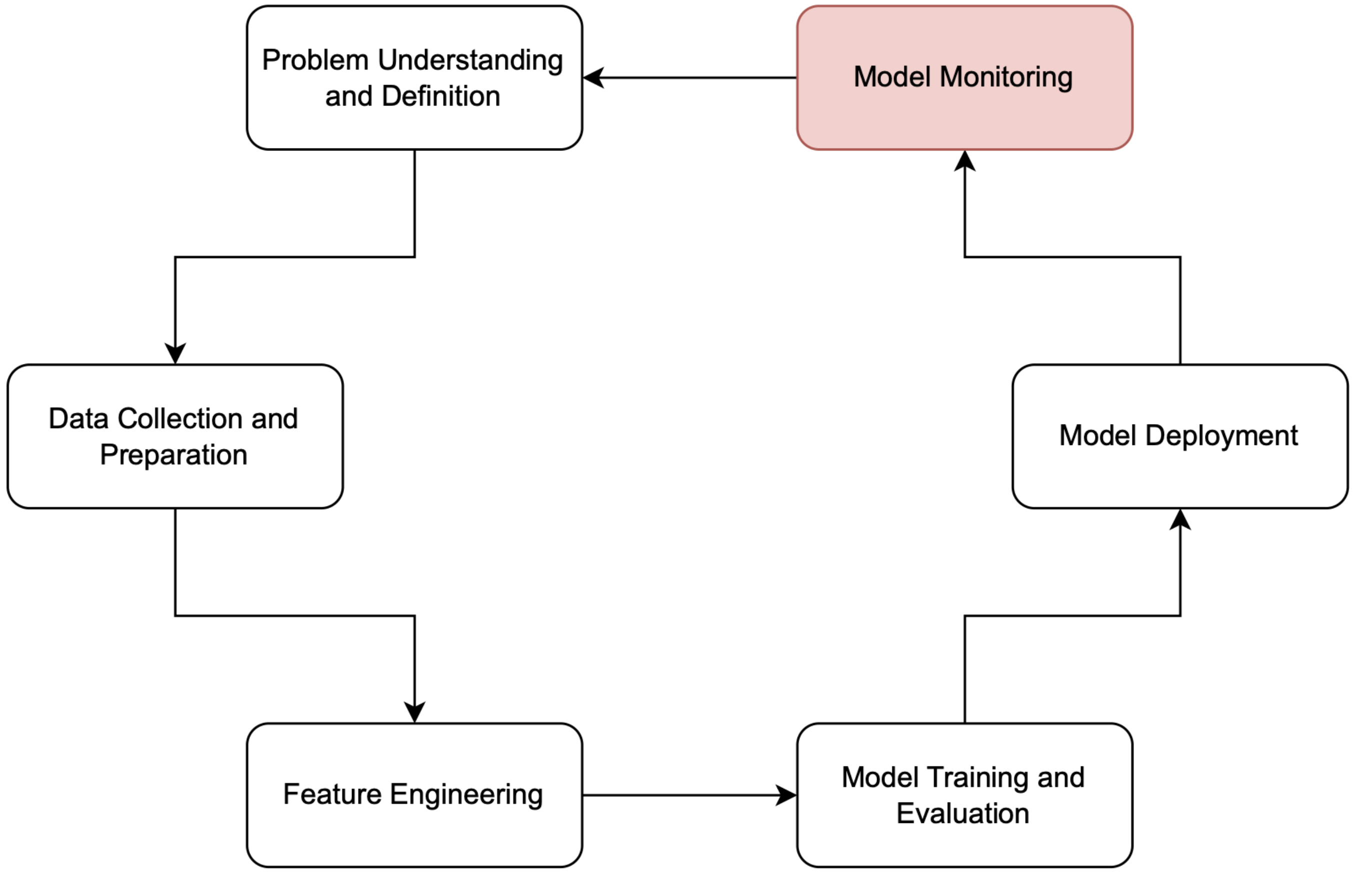
Monitoring and visualization

END-TO-END MACHINE LEARNING

Joshua Stapleton
Machine Learning Engineer

What's next?

- Trained, optimized, deployed, predicted... what next?
- Monitoring
  - Logging results
  - Visualizing performance



Logging with python

```
import logging
import matplotlib.pyplot as plt

# Setting up basic logging configuration
logging.basicConfig(filename='predictions.log', level=logging.INFO)

# Make predictions on the test set and log the results
for i in range(X_test.shape[0]):
    instance = X_test[i,:].reshape(1, -1)
    prediction = model.predict(instance)
    logging.info(f'Inst. {i} - PredClass: {prediction[0]}, RealClass: {y_test[i]}')
```

Logging with python (cont.)

```python
# Function to visualize the predictions from log
with open(logfile, 'r') as f:
    lines = f.readlines()
    predicted_classes = [int(line.split("Predicted Class: ")[1].split(",")[0]) \
        for line in lines]

    # Perform data analysis, visualization, etc.
    ...
```

- Use Python logging to trace model performance

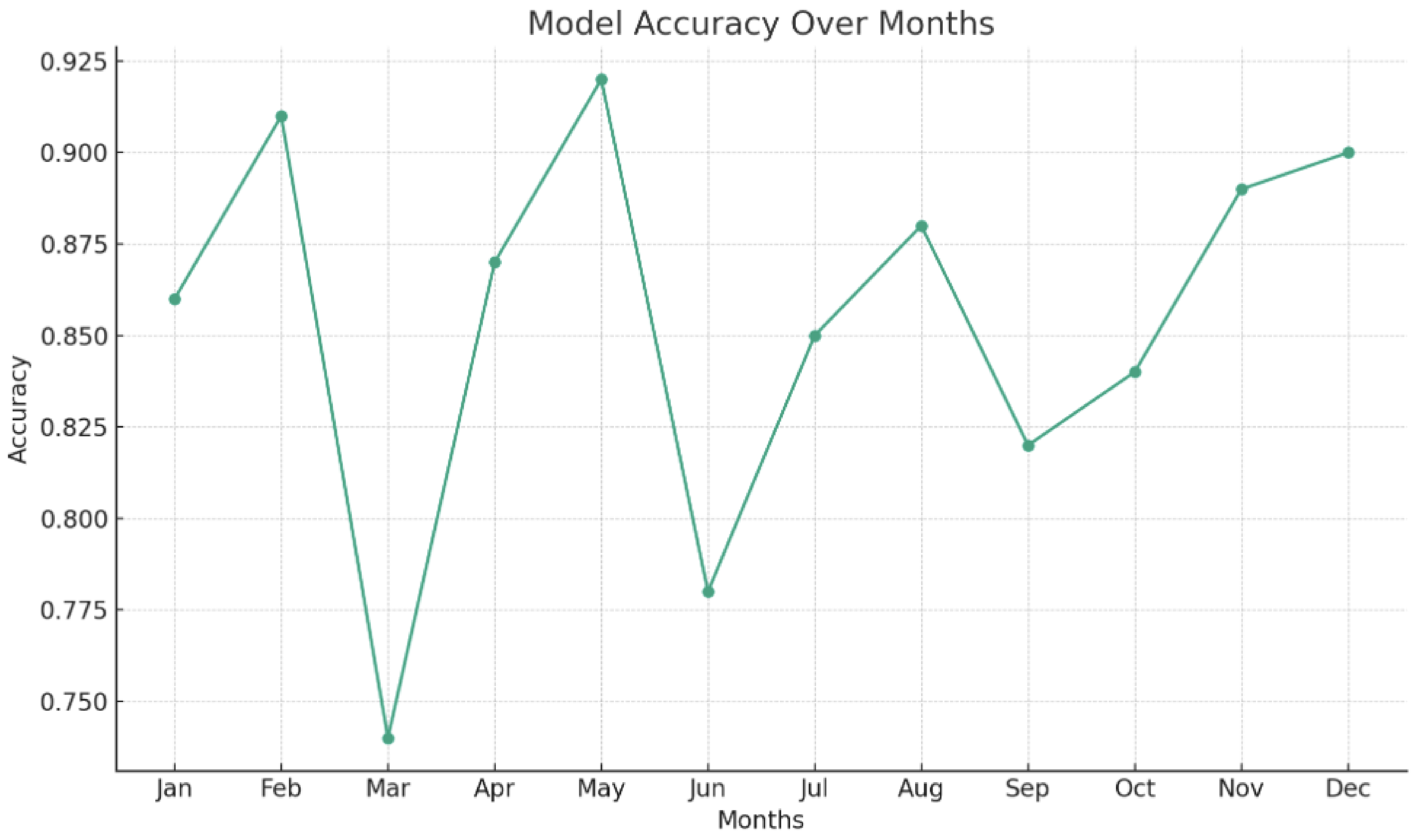
Visualization

- Inspect performance over time
- Transform raw data of inputs / predictions into insights

```python
import matplotlib.pyplot as plt


# Sample data: Random accuracy values for 12 months
months = ["Jan", "Feb", "Mar", ...]
accuracies = [0.86, 0.91, 0.74, ...]
plt.plot(months, accuracies, '-o')
plt.title("Model Accuracy Over Months")
plt.xlabel("Months")
plt.ylabel("Accuracy")
plt.show()
```

Visualization example



Logging

- Recording of events
  - Tracking variable values, Function calls
  - Information that informs execution + performance
- Monitoring helps track:
  - Usage, Performance, Errors/anomalies

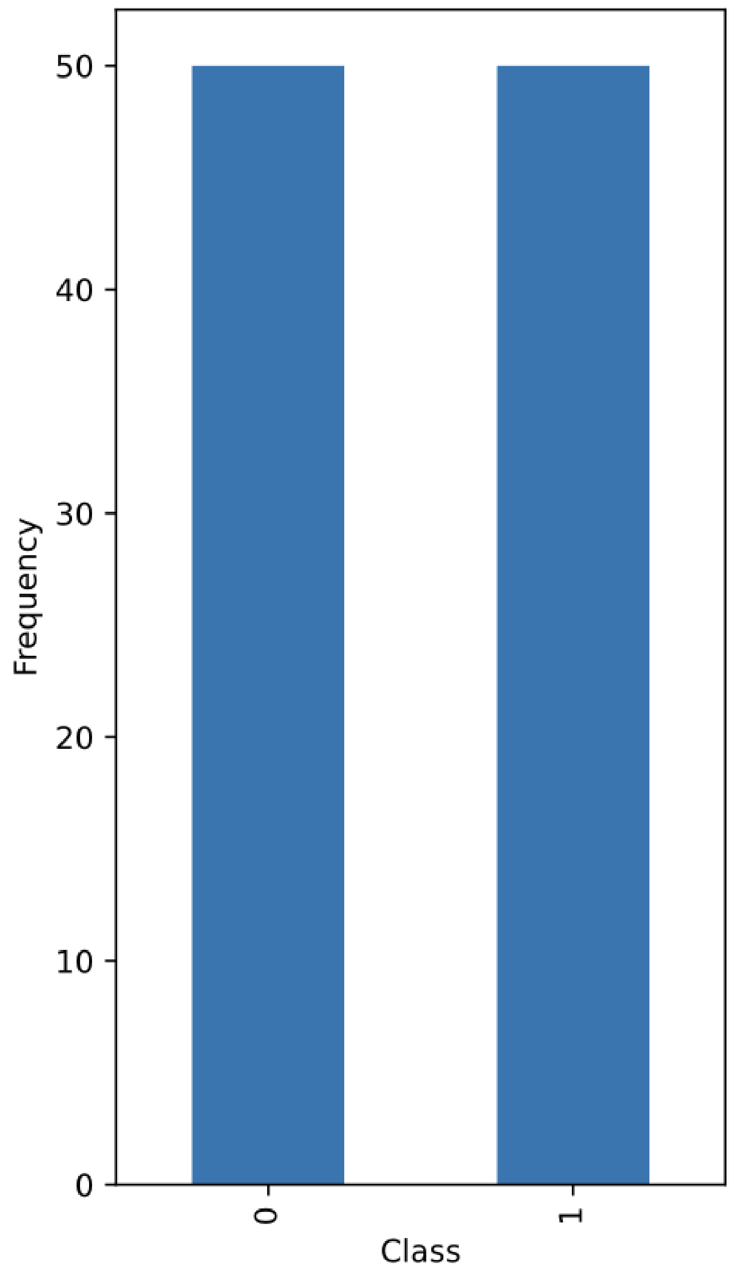
```
2023-08-04 09:15:20 [INFO] Model version 1.2.7 started
2023-08-04 09:15:45 [INFO] Preprocessing input data for prediction
2023-08-04 09:15:47 [DEBUG] Input data shape: (1, 12)
2023-08-04 09:15:48 [INFO] Making prediction
2023-08-04 09:15:50 [DEBUG] Output prediction: [0.78]
...
```

Visualization examples

- Helpful metric for our model: balanced accuracy over time
- Spot trends, see if performance degrades
- See if retraining is necessary
- Choose helpful metrics for our use-case

Example:

- Balanced accuracy changes relative to expected, real-world rate
- Potentially indicative of problem
- Choose and evaluate



Let's practice!

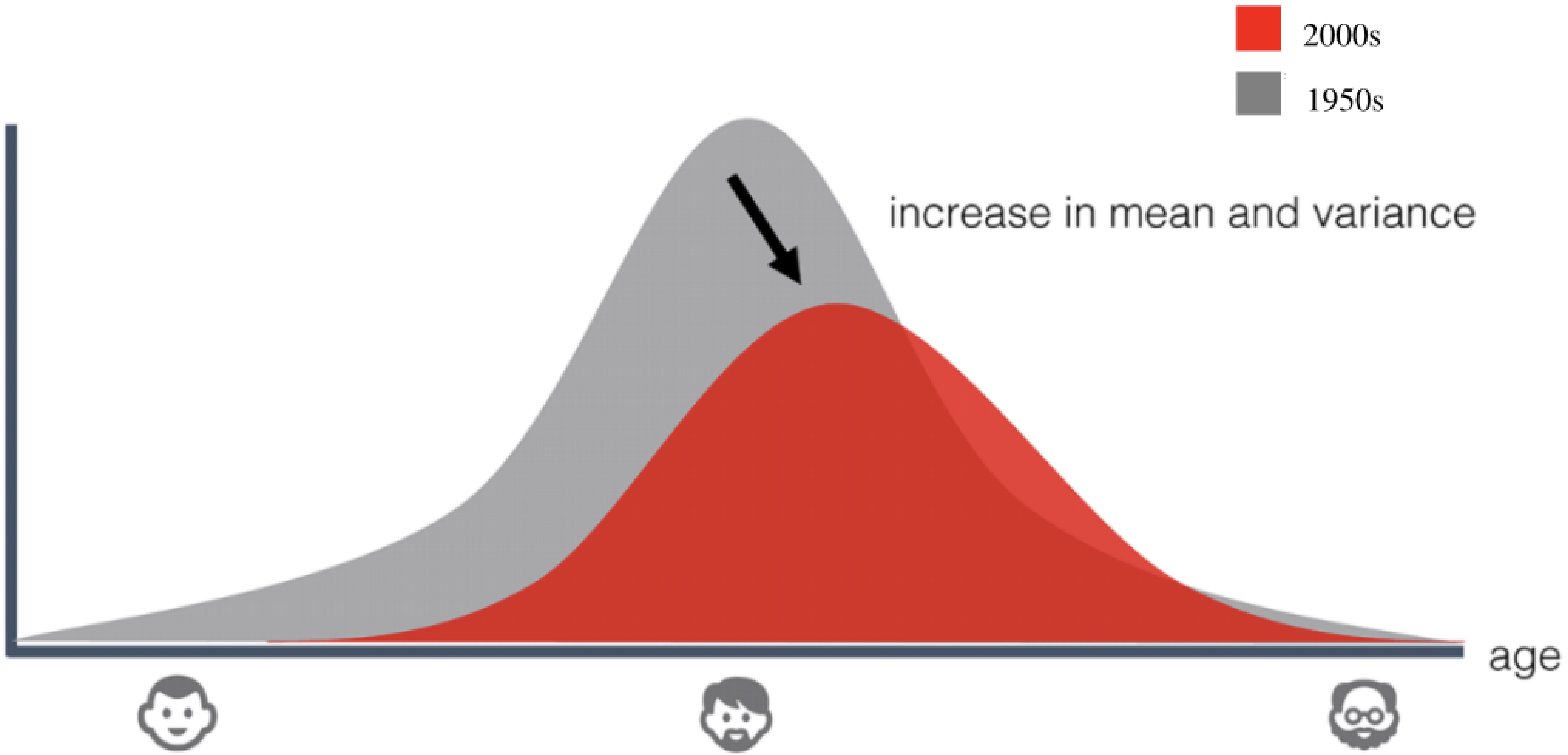
END-TO-END MACHINE LEARNING

Data drift

END-TO-END MACHINE LEARNING

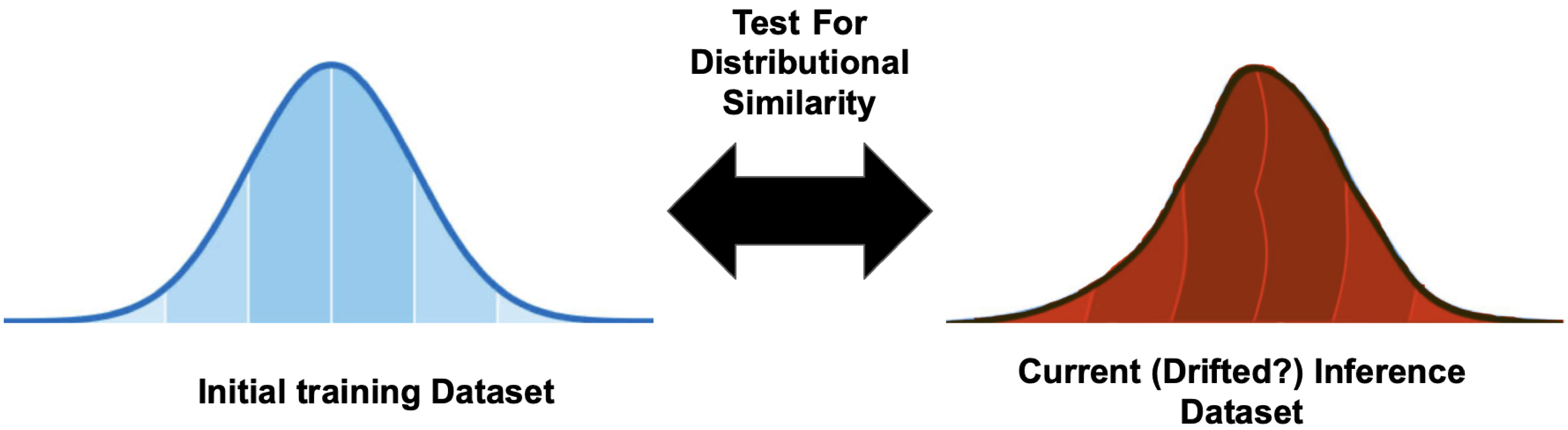
Joshua Stapleton
Machine Learning Engineer

The need for data drift detection



The Kolmogorov-Smirnov test

- Commonly used for detecting data drift
- Compares differences between dataset samples to determine distributional similarity



Using the ks_2samp() function

- `ks_2samp()` function returns two values: test statistic, p-value.
- Use p-value to accept/reject the null hypothesis of distributional similarity.

```
from scipy.stats import ks_2samp
```

```
# load the 1D data distribution samples for comparison
sample_1, sample_2 = training_dataset_sample, current_inference_sample
```

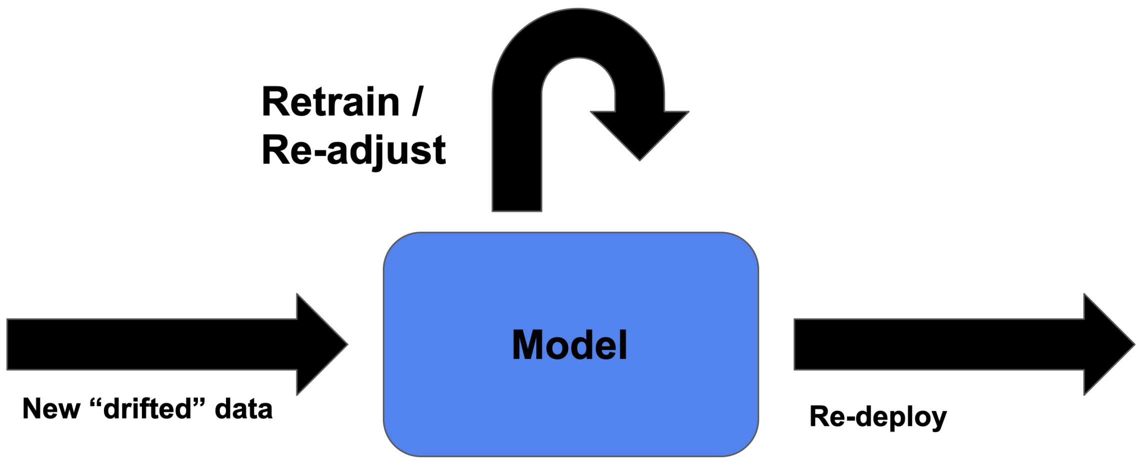
```
# perform the KS-test - ensure input samples are numpy arrays
test_statistic, p_value = ks_2samp(sample_1, sample_2)
```

```
if p_value < 0.05:
    print("Reject null hypothesis - data drift might be occuring")
else:
    print("Samples are likely to be from the same dataset")
```

Correcting data drift

Update model to account for new data

- Retrain model
- Re-adjust / update model parameters



Not enough new/inference data?

- Re-train model on mixed dataset
- Increase amounts of new data

Further resources for detecting and rectifying data drift

- **Population Stability Index (PSI)**
  - Compares single categorical variables / columns
- **Evidently**
  - Open-source Python library
  - Robustly test and correct for data drift
- **NannyML**
  - Monitor deployed model performance

Let's practice!

END-TO-END MACHINE LEARNING

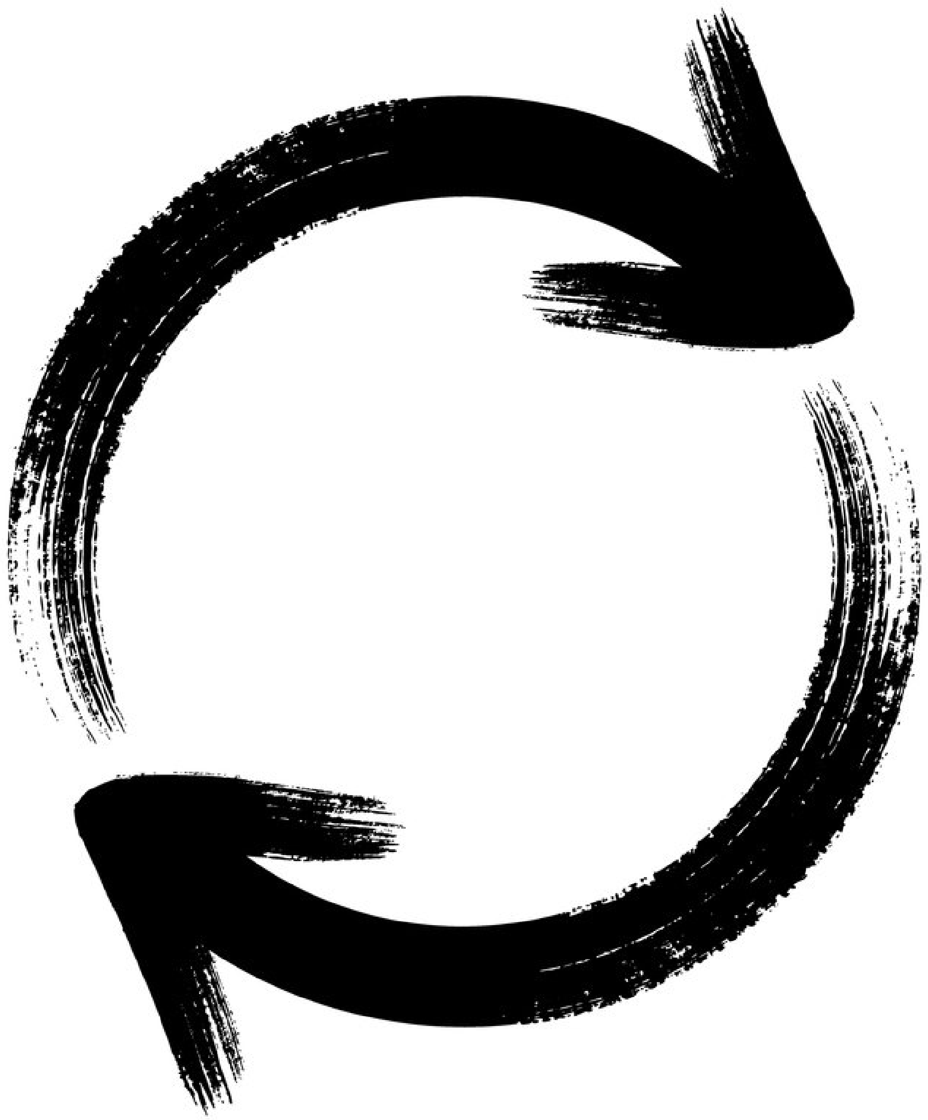
Feedback loop, re-training, and labeling

END-TO-END MACHINE LEARNING

Joshua Stapleton
Machine Learning Engineer

Feedback loop

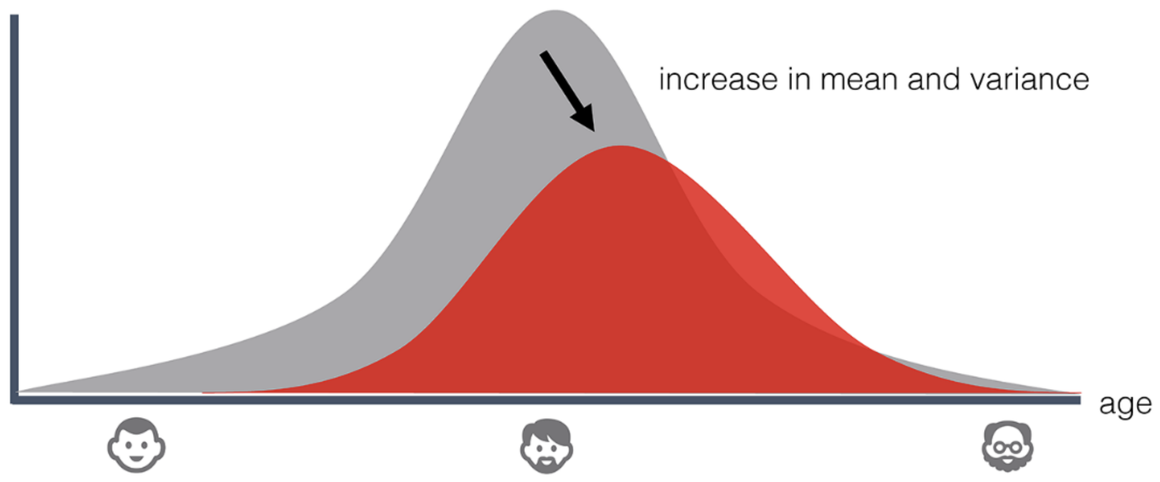
- Model output considered as system input:
  - Using metrics/predictions to inform system evolution
  - Can use model monitoring
- Integral part of ML:
  - Allows for rapid learning and adjustment
  - Better adapt to change



Feedback loop implementation

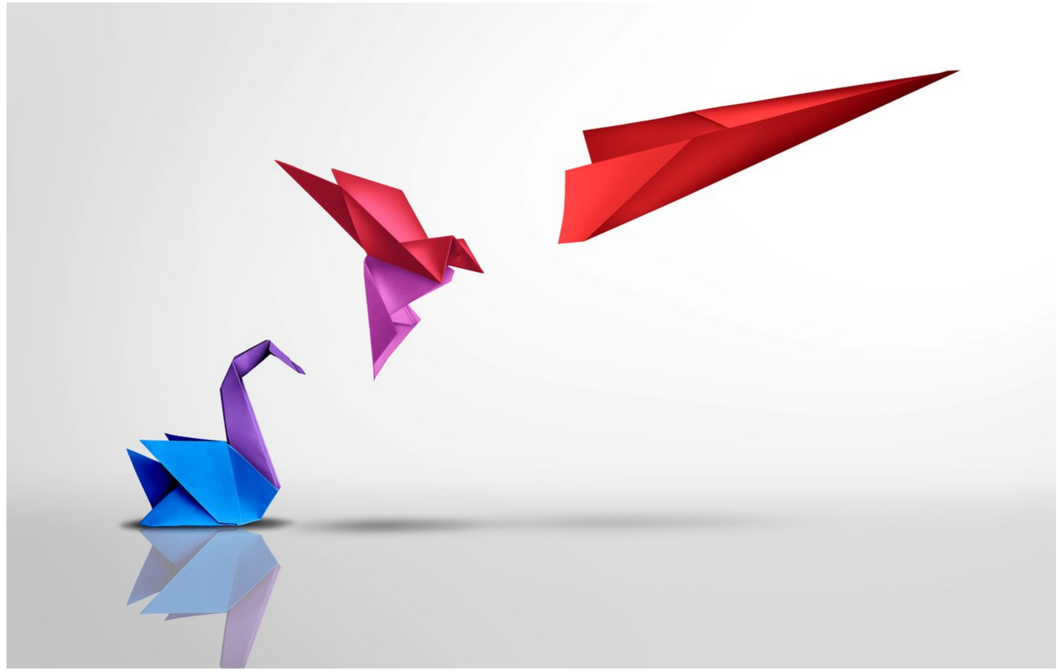
Data drift detection

- Input data distribution changes over time
- Feedback loop: retrain on newer data



Online learning

- Periodically retrain based on changing data
- Beyond data drift: adapts to changes in data structure



Dangers of feedback loops

Dangers...

- Model's outputs affect inputs
- Eg: social media recommendation:
  - Maximize user engagement
  - Learns to serve certain type of content
  - Causes user to view more of this content
  - etc.
- Develops undesired behavioral patterns
- More dangerous when automated

Better usage of feedback loop

- Reactive:
  - Human in the loop
  - Model's predictions don't change input data
- Caution and oversight are key!

Let's practice!

END-TO-END MACHINE LEARNING

Serving the model

END-TO-END MACHINE LEARNING

Joshua Stapleton
Machine Learning Engineer

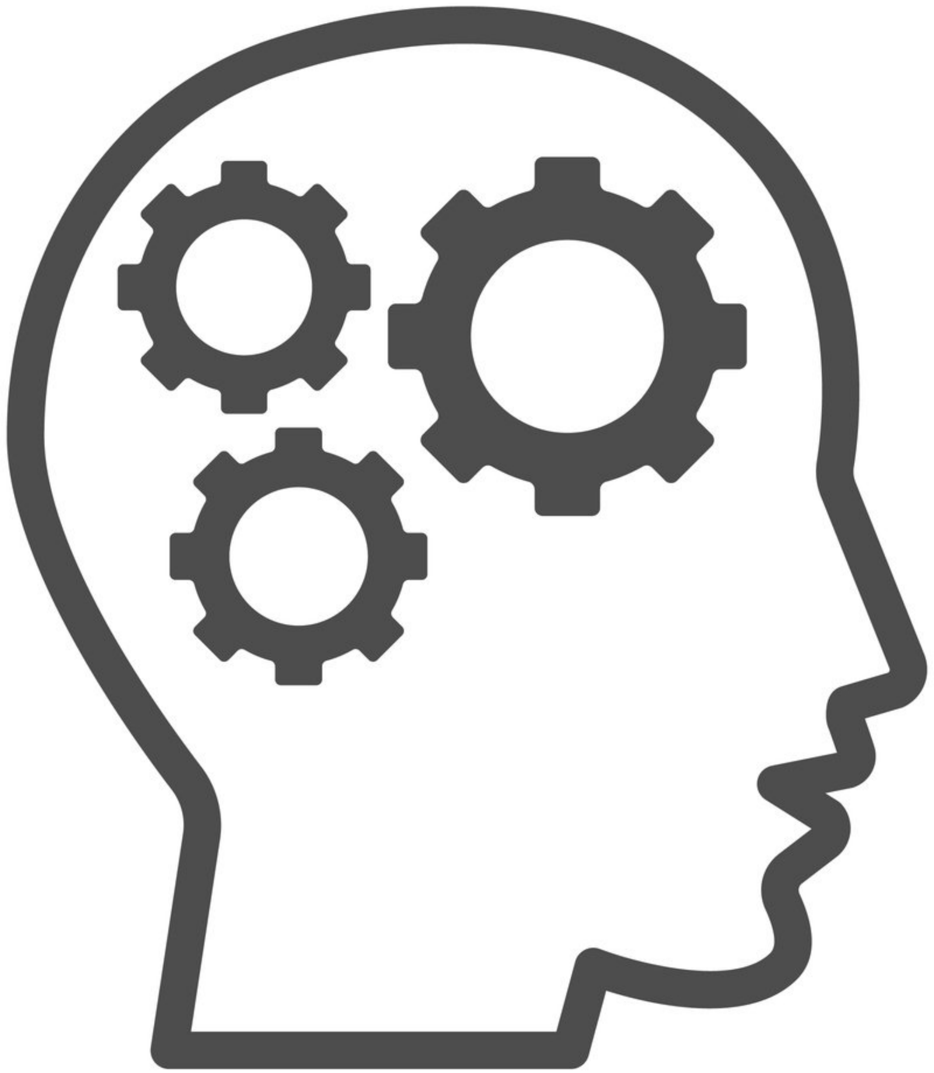
Model-as-a-service

- Stakeholders / users access model over internet
- Surface model through portal
  - Users post queries / data
  - Receive diagnosis / predictions
- Concerns:
  - Rural clinic / no internet access
  - Highly secure environment / sensitive data

On-device serving

Integrated serving architectures

- Edge-computing
- Helpful for unreliable internet cases



Pros and cons of on-device serving

Pros:

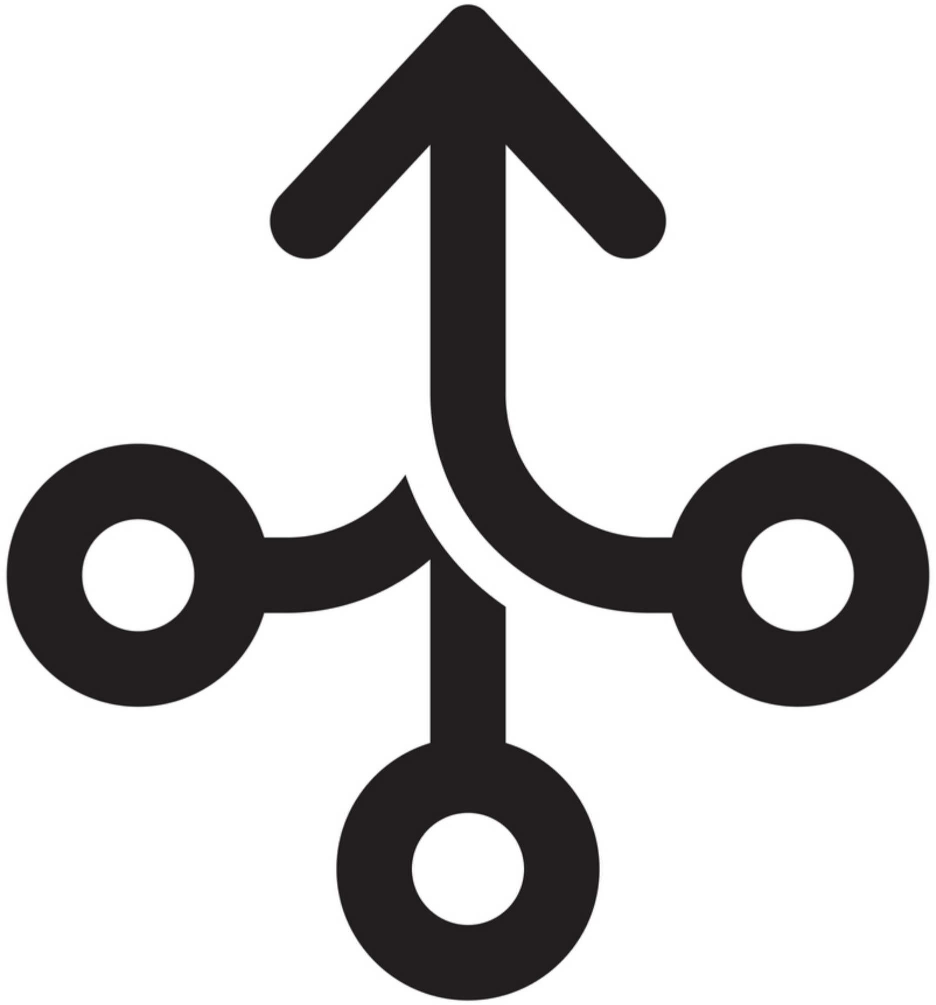
- Lower latency
- Security
- Applications for remote / disconnected areas

Cons:

- Resource constraints
- Model updates
- Monitoring

Implementation strategies

- Pruning
- Transfer Learning
- Use Dedicated Frameworks



Let's practice!

END-TO-END MACHINE LEARNING

Wrap-up

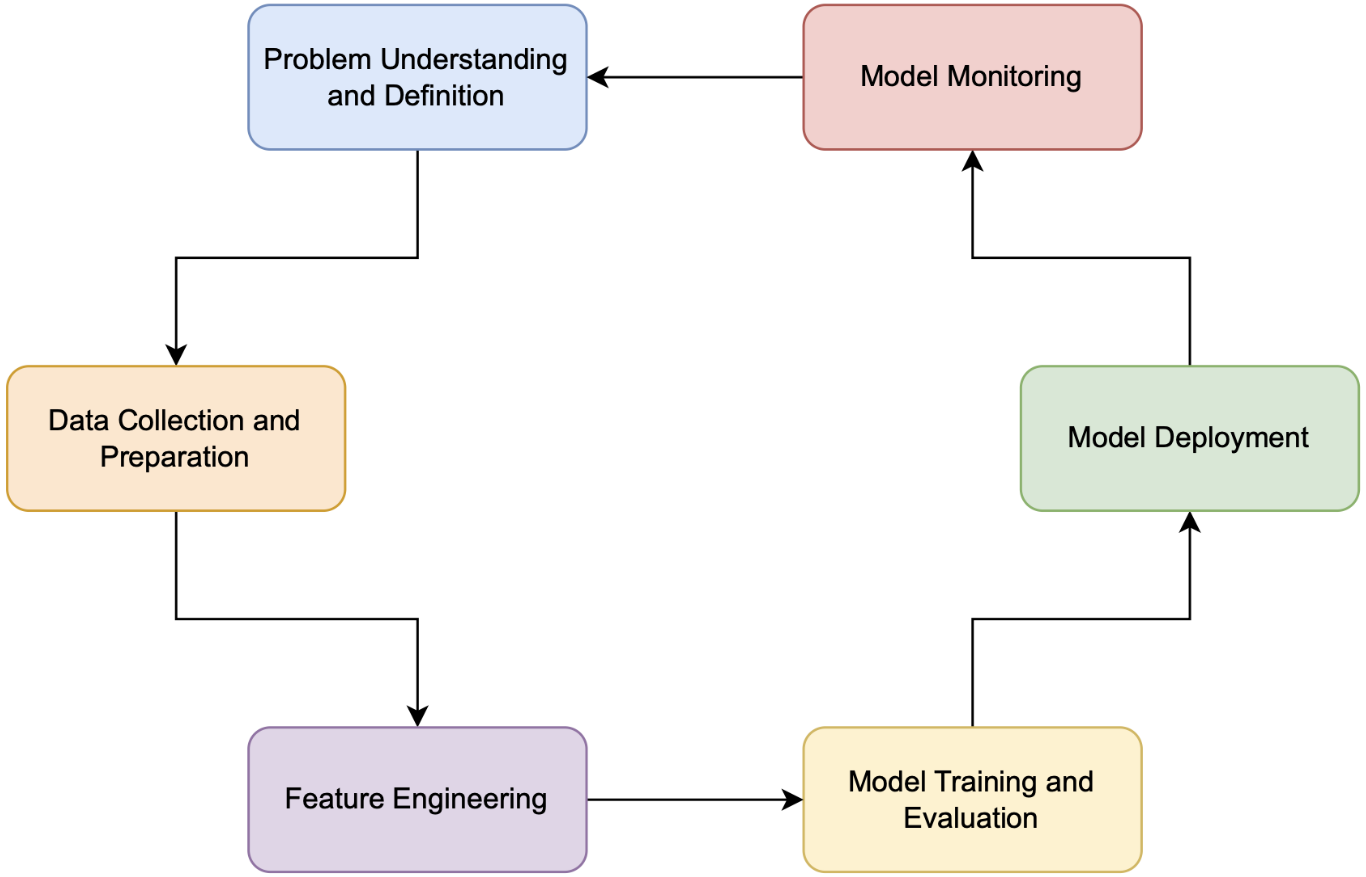
END-TO-END MACHINE LEARNING

Joshua Stapleton
Machine Learning Engineer

How far we've come

Built a full ML pipeline:

- Problem Definition
- Data Cleaning
- Feature engineering and selection
- Model Training and evaluation
- Model deployment using TDD and CI/CD
- Model monitoring
- Feedback loop



What's next?

?

So much more to do!

- ML lifecycle isn't a once-off solution.
  - It's iterative, organic.
  - Continue to improve!

Let's practice!

END-TO-END MACHINE LEARNING