Drifting Away: Testing ML Models in Production

Chengyin Eng

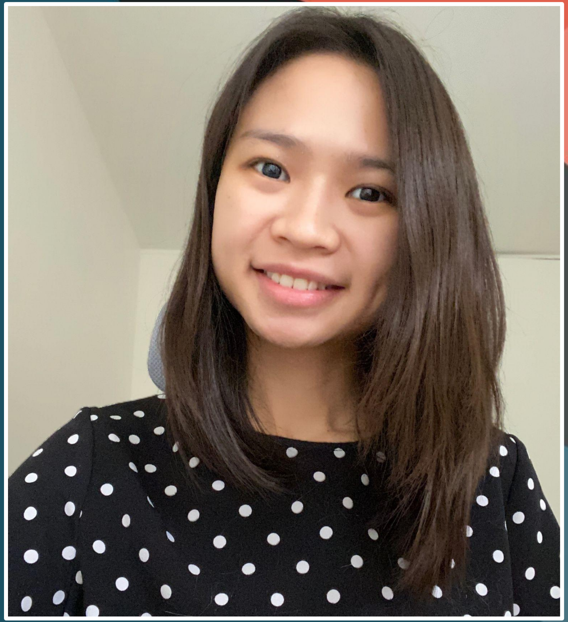
Niall Turbitt

About

Chengyin Eng

Data Scientist @ Databricks

- Machine Learning Practice Team
- Experience
  - Life Insurance
  - Teaching ML in Production, Deep Learning, NLP, etc.
- MS in Computer Science at University of Massachusetts, Amherst
- BA in Statistics & Environmental Studies at Mount Holyoke College, Massachusetts



databricks

About

Niall Turbitt

Senior Data Scientist @ Databricks

- EMEA ML Practice Team
- Experience
  - Energy & Industrial Applications
  - e-Commerce
  - Recommender Systems & Personalisation
- MS Statistics University College Dublin
- BA Mathematics & Economics Trinity College Dublin

databricks

Outline

- Motivation
- Machine Learning System Life Cycle
- Why Monitor?
  - Types of drift
- What to Monitor?
- How to Monitor?
- Demo

ML is everywhere, but often fails to reach production

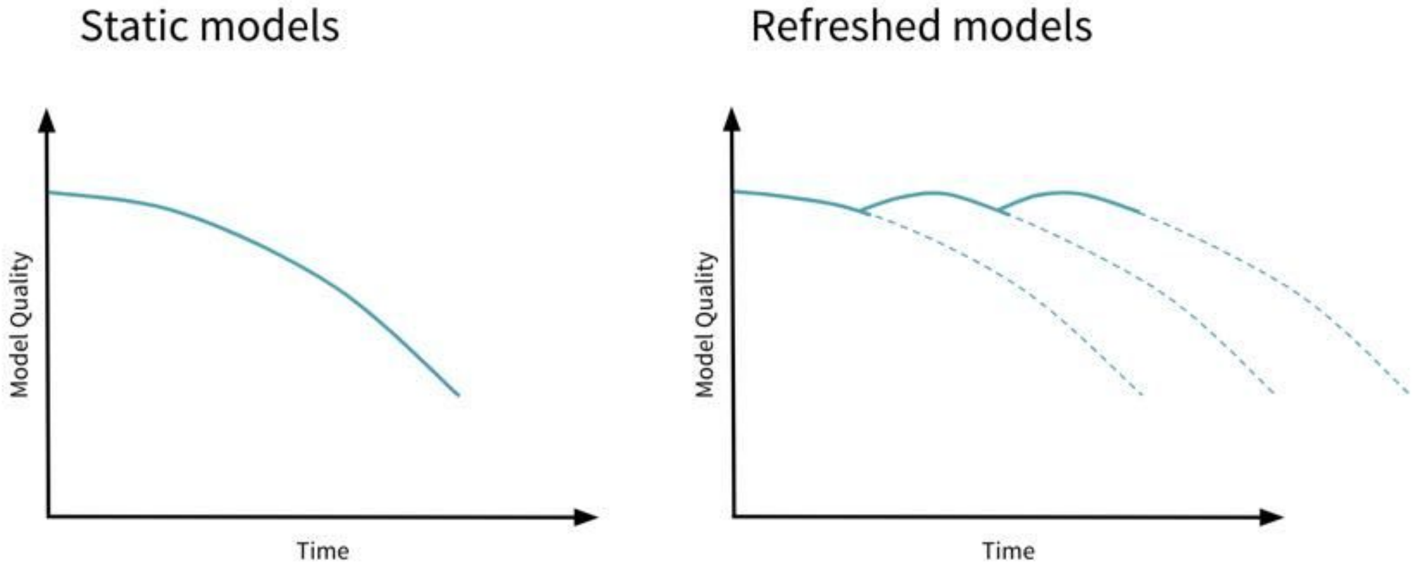
85% of DS projects fail

4% of companies succeed in deploying ML models to production

Source:
https://www.datanami.com/2020/10/01/most-data-science-projects-fail-but-yours-doesnt-have-to/

Why do ML projects fail in production?

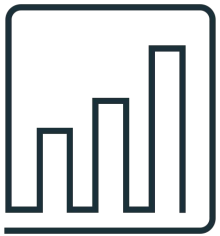
Neglect maintenance: Lack of re-training and testing



Source:
https://databricks.com/blog/2019/09/18/productionizing-machine-learning-from-deployment-to-drift-detection.html

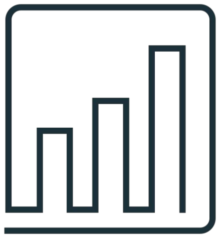
This talk focuses on two questions:

This talk focuses on two questions:



What are the statistical tests to use when monitoring models in production?

This talk focuses on two questions:



What are the statistical tests to use when monitoring models in production?

What tools can I use to coordinate the monitoring of data and models?

What this talk is *not*

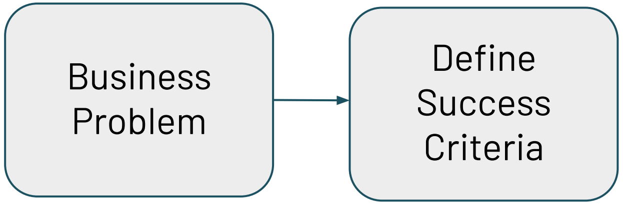
- A tutorial on model deployment strategies
- An exhaustive walk through of how to robustly test your production ML code
- A prescriptive list of *when* to update a model in production

Machine Learning System Life Cycle

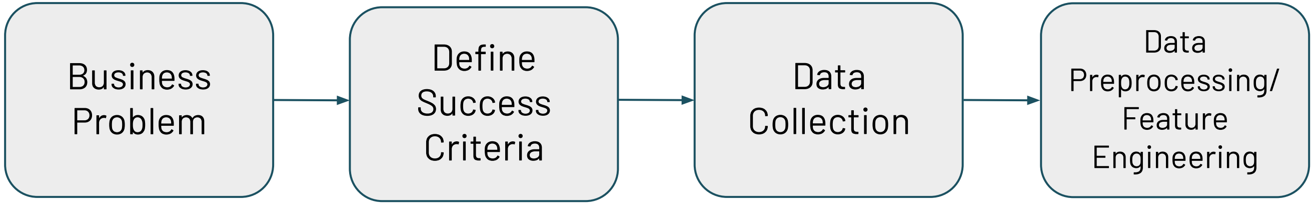
ML system life cycle

Business Problem

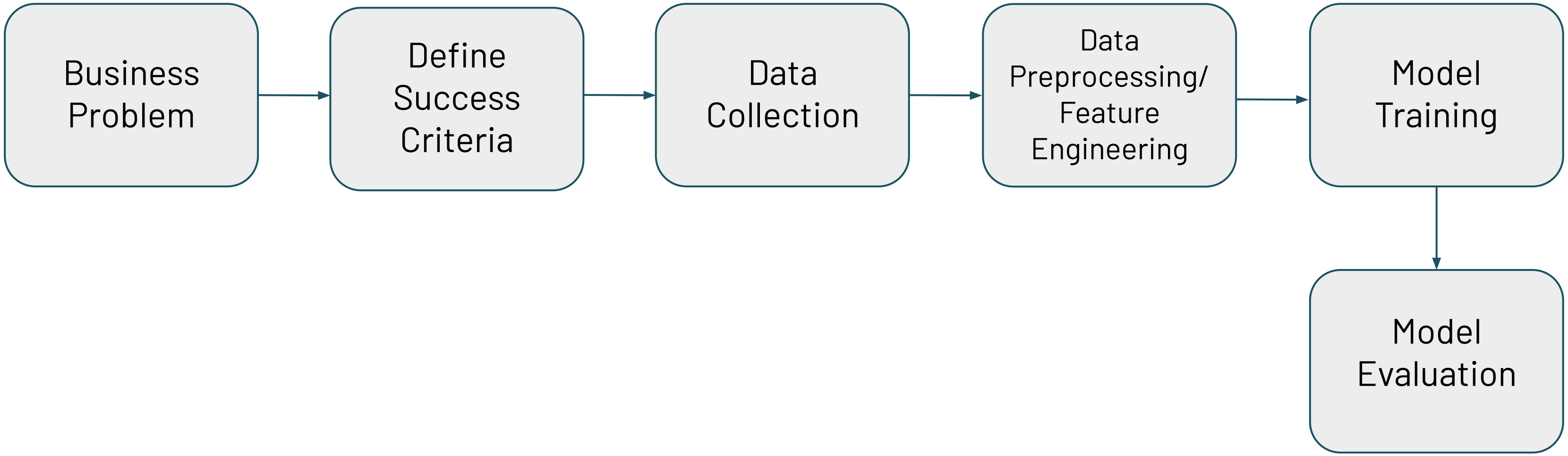
ML system life cycle



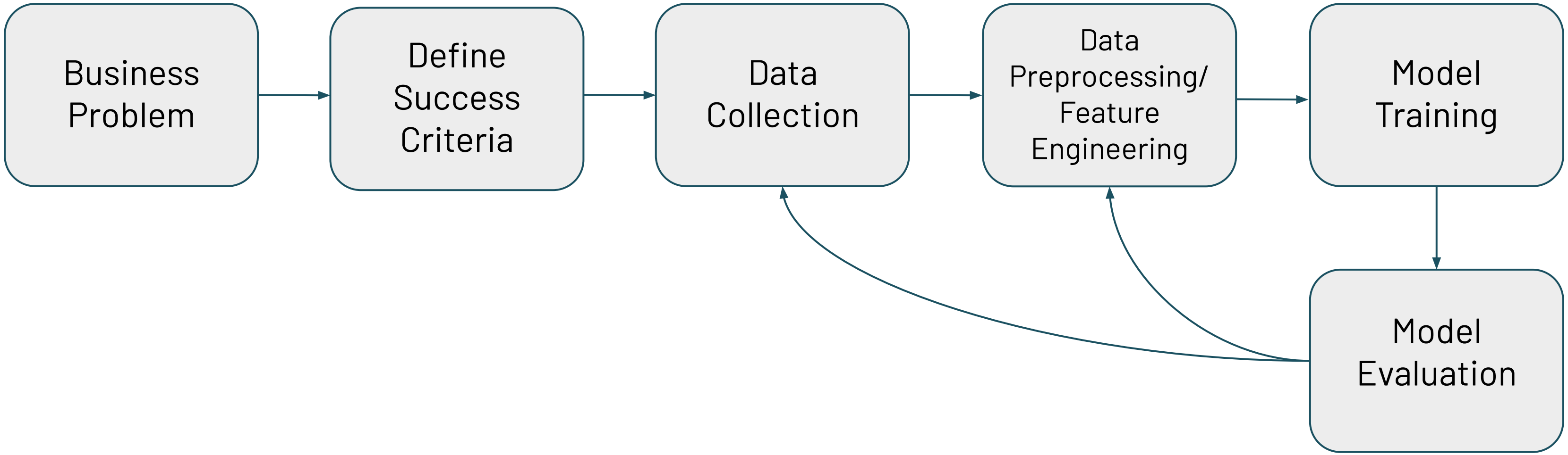
ML system life cycle



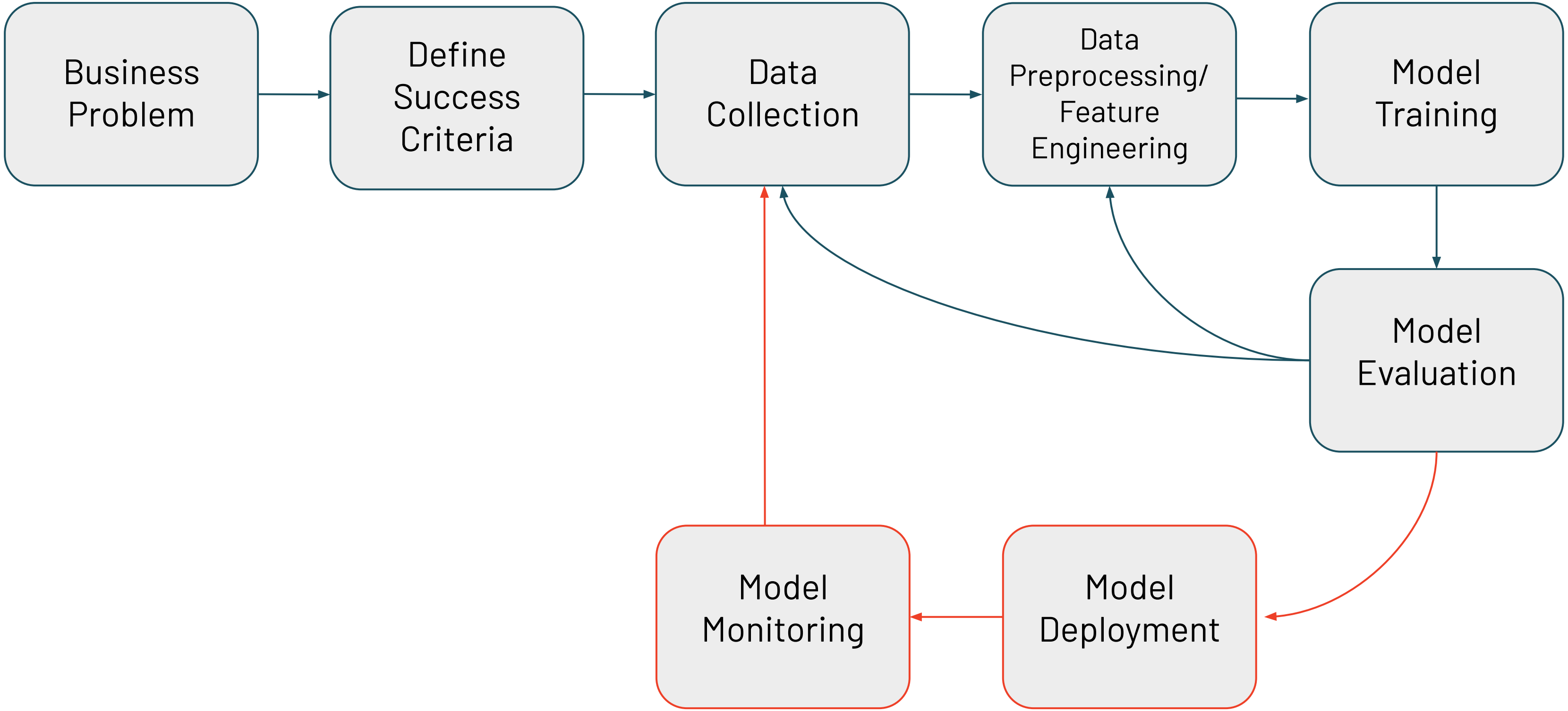
ML system life cycle



ML system life cycle



ML system life cycle

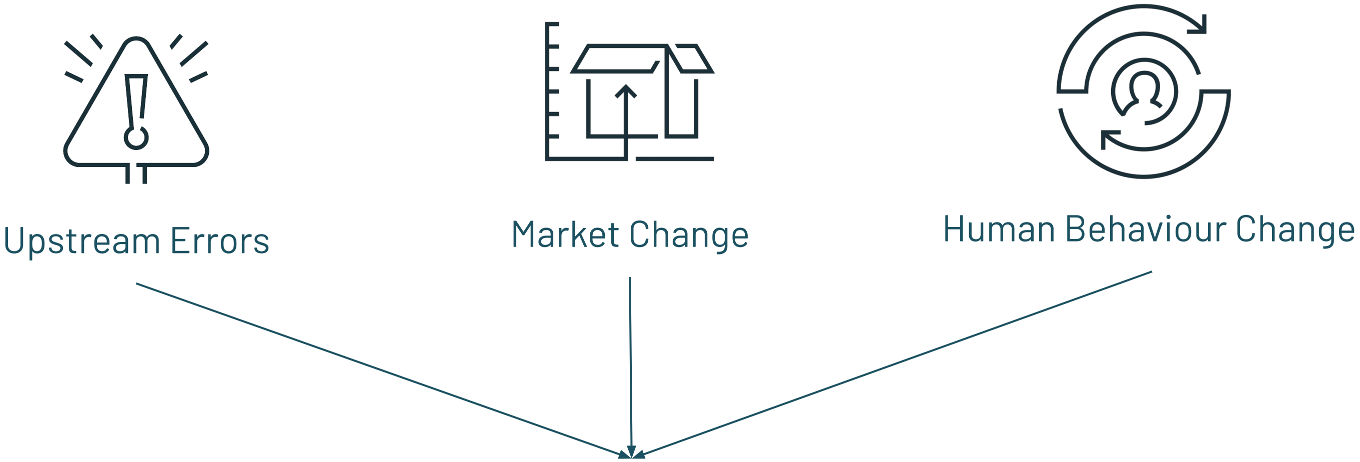


Why Monitor?

Model deployment is not the end

It is the beginning of model measurement and monitoring

- Data distributions and feature types can change over time due to:



Potential model performance degradation

Models *will* degrade over time

Challenge: catching this when it happens

Types of drift

Feature Drift

Input feature(s) distributions deviate

Label Drift

Label distribution deviates

Prediction Drift

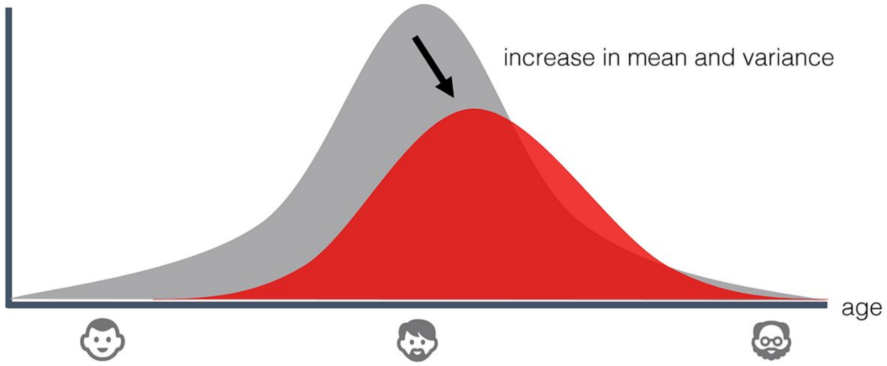
Model prediction distribution deviates

Concept Drift

External factors cause the label to evolve

Feature, Label, and Prediction Drift

Categories	Expected	Observed	Total
A	25	35	60
B	25	20	56
C	25	25	50
D	25	20	45
Total	100	100	100

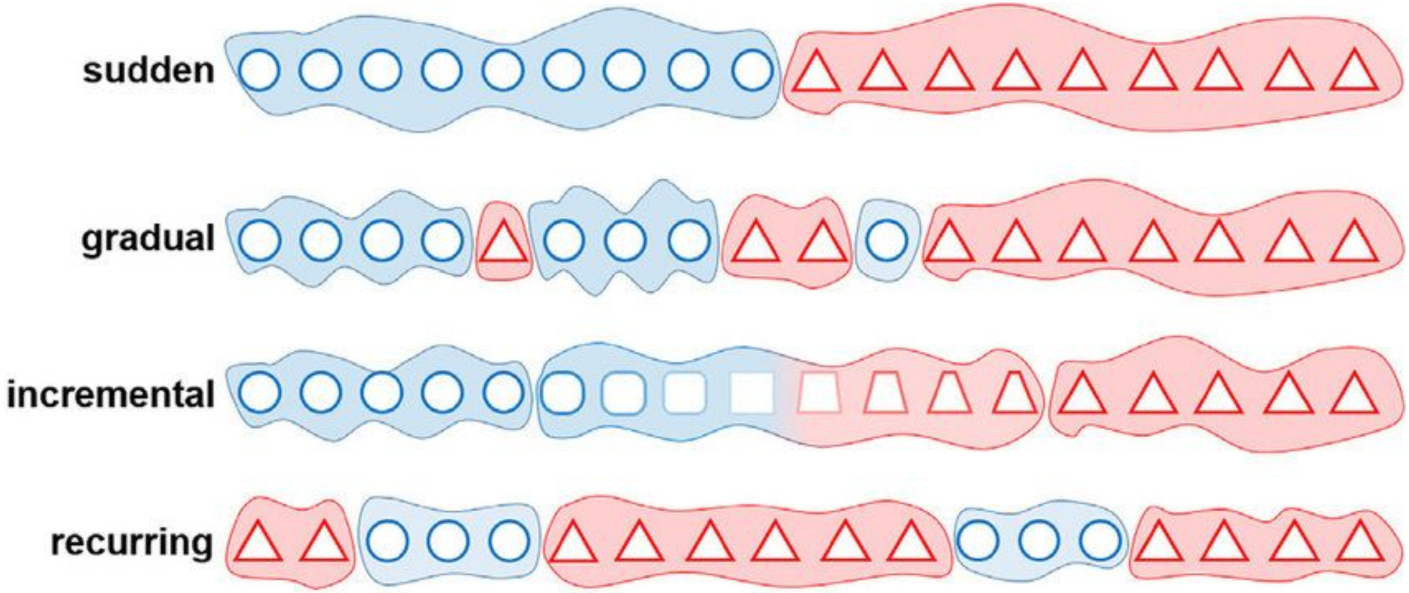


Sources:

https://dataz4s.com/statistics/chi-square-test/

https://towardsdatascience.com/machine-learning-in-production-why-you-should-care-about-data-and-concept-drift-d96d0bc907fb

Concept drift



Source: Krawczyk and Cano 2018. Online Ensemble Learning for Drifting and Noisy Data Streams

Drift types and actions to take

Drift Type Identified	Action
Feature Drift	• Investigate feature generation process • Retrain using new data
Label Drift	• Investigate label generation process • Retrain using new data
Prediction Drift	• Investigate model training process • Assess business impact of change in predictions
Concept Drift	• Investigate additional feature engineering • Consider alternative approach/solution • Retrain/tune using new data

What to Monitor?

What should I monitor?

- Basic summary statistics of features and target
- Distributions of features and target
- Model performance metrics
- Business metrics

Monitoring tests on data

Numeric Features

- Summary statistics:
  - Median / mean
  - Minimum
  - Maximum
  - Percentage of missing values
- Statistical tests:
  - Mean:
    - Two-sample Kolmogorov-Smirnov (KS) test with Bonferroni correction
    - Mann-Whitney (MW) test
  - Variance:
    - Levene test

Kolmogorov-Smirnov (KS) test with Bonferroni correction

Comparison of two continuous distributions

- Null hypothesis (H_0):
  Distributions x and y come from the same population
- If the KS statistic has a *p*-value lower than α, reject H_0
- Bonferroni correction:
  - Adjusts the α level to reduce false positives
  - $\alpha_{new} = \alpha_{original} / n$, where n = total number of feature comparisons

Levene test

Comparison of variances between two continuous distributions

- Null hypothesis (H_0):

$$\sigma^2_1 = \sigma^2_2 = \ldots = \sigma^2_n$$

- If the Levene statistic has a *p*-value lower than α, reject H_0

Monitoring tests on data

Numeric Features

- Summary statistics:
  - Median / mean
  - Minimum
  - Maximum
  - Percentage of missing values
- Statistical tests:
  - Mean:
    - Two-sample Kolmogorov-Smirnov (KS) test with Bonferroni correction
    - Mann-Whitney (MW) test
  - Variance:
    - Levene test

Categorical Features

- Summary statistics:
  - Mode
  - Number of unique levels
  - Percentage of missing values
- Statistical test:
  - One-way chi-squared test

One-way chi-squared test

Comparison of two categorical distributions

- Null hypothesis (H_0):
  Expected distribution = observed distribution
- If the Chi-squared statistic has a *p*-value lower than α, reject H_0

Monitoring tests on models

- Relationship between target and features
  - Numeric Target: Pearson Coefficient
  - Categorical Target: Contingency tables

- Model Performance
  - Regression models: MSE, error distribution plots etc
  - Classification models: ROC, confusion matrix, F1-score etc
  - Performance on data slices

- Time taken to train

How to Monitor?

Demo: Measuring models in production

- Logging and Versioning
  - MLflow (model)
  - Delta (data)
- Statistical Tests
  - SciPy
  - statsmodels
- Visualizations
  - seaborn

mlflow™

An open-source platform for ML lifecycle that helps with operationalizing ML

mlflow™ Tracking

Record and query experiments: code, metrics, parameters, artifacts, models

mlflow™ Projects

Packaging format for reproducible runs on any compute platform

mlflow™ Models

General model format that standardizes deployment options

mlflow™ Model Registry

Centralized and collaborative model lifecycle management

mlflow™

An open-source platform for ML lifecycle that helps with operationalizing ML

mlflow™ Tracking

Record and query experiments: code, metrics, parameters, artifacts, models

mlflow™ Projects

Packaging format for reproducible runs on any compute platform

mlflow™ Models

General model format that standardizes deployment options

mlflow™ Model Registry

Centralized and collaborative model lifecycle management

Demo Notebook

http://bit.ly/dais_2021_drifting_away

Conclusion

- Model measurement and monitoring are crucial when operationalizing ML models
- No one-size fits all
  - Domain & problem specific considerations
- Reproducibility
  - Enable rollbacks and maintain record of historic performance

Literature resources

- Paleyes et al 2021. Challenges in Deploying ML
- Klaise et al. 2020 Monitoring and explainability of models in production
- Rabanser et al 2019 Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift
- Martin Fowler: Continuous Delivery for Machine Learning

Emerging open-source monitoring packages

- EvidentlyAI
- Data Drift Detector
- Alibi Detect
- scikit-multiflow

Feedback

Your feedback is important to us.

Don't forget to rate and review the sessions.

DATA+AI SUMMIT 2021

FORMERLY SPARK+AI SUMMIT