MACHINE LEARNING OPERATIONS

WEEK 1

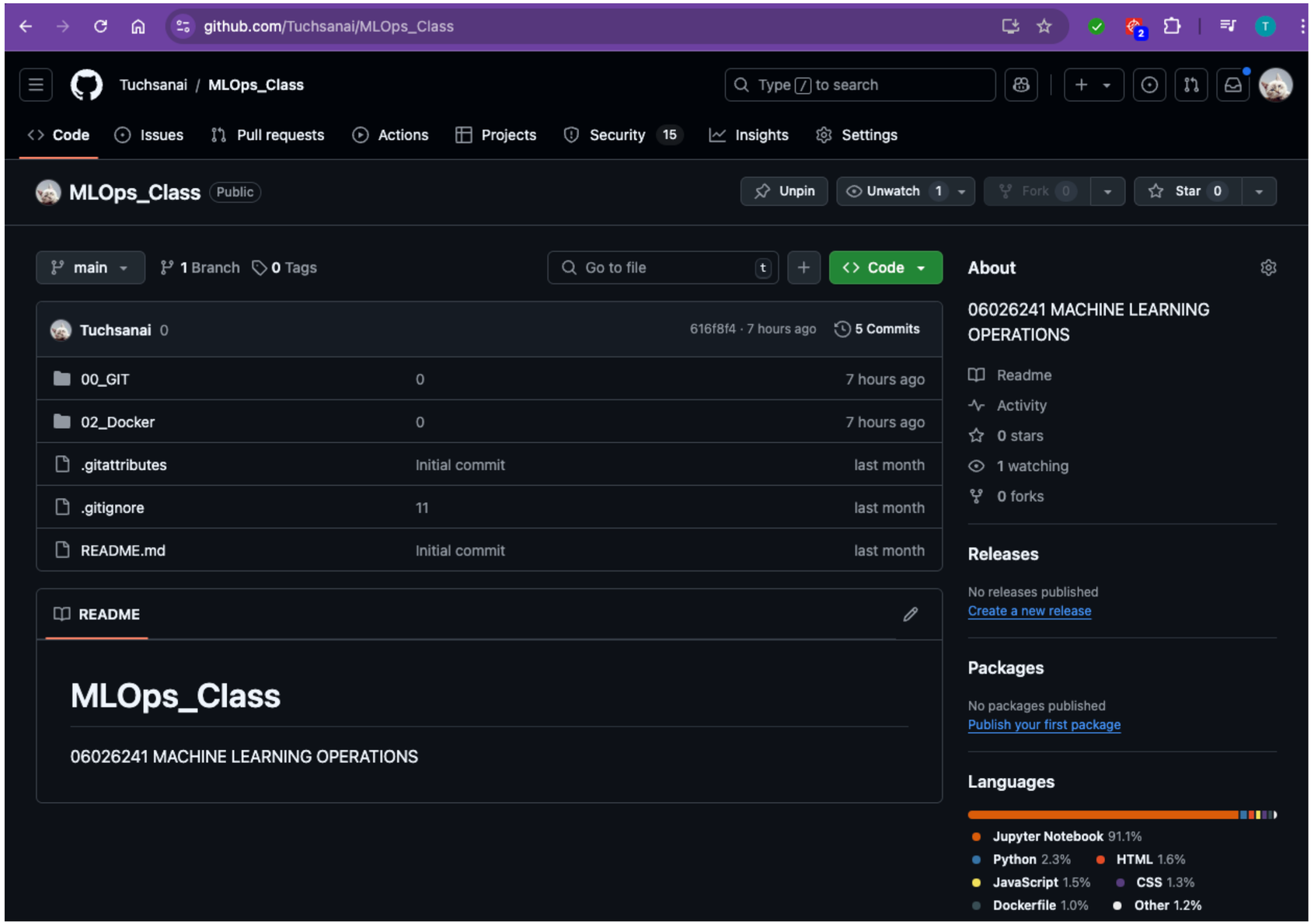
UNLIMITED

Presented by **Asst. Prof. Dr. Tuchsanai Ploysuwan**

MACHINE LEARNING OPERATIONS

Document

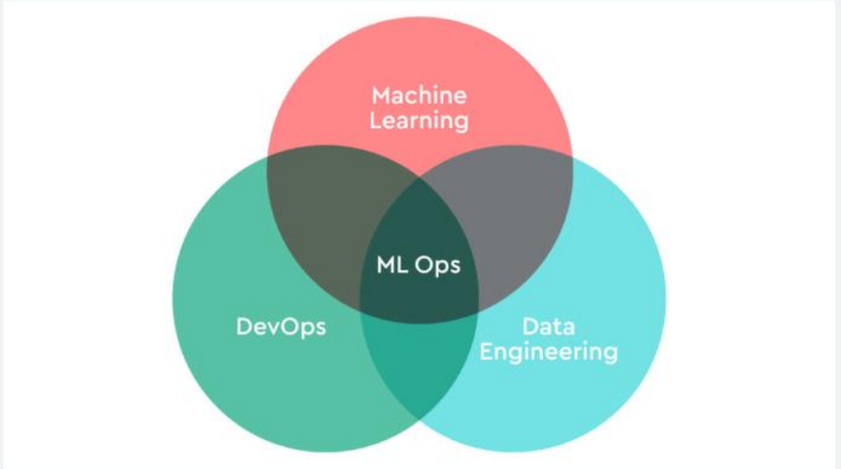
Group line


github.com/Tuchsanai/MLOps_Class
Tuchsanai / MLOps_Class
Type / to search
Code
Issues
Pull requests
Actions
Projects
Security 15
Insights
Settings
MLOps_Class
Public
Unpin
Unwatch 1
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
t
Code
Tuchsanai 0
616f8f4 · 7 hours ago
5 Commits
00_GIT 0 7 hours ago
02_Docker 0 7 hours ago
.gitattributes Initial commit last month
.gitignore 11 last month
README.md Initial commit last month
README
MLOps_Class
06026241 MACHINE LEARNING OPERATIONS
About
06026241 MACHINE LEARNING OPERATIONS
Readme
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
Jupyter Notebook 91.1%
Python 2.3%
HTML 1.6%
JavaScript 1.5%
CSS 1.3%
Dockerfile 1.0%
Other 1.2%

การให้คะแนน	
Midterm	35
Final	30
Homework and Exercise LAB	15
Mini project	20

MLOps is the practice of deploying machine learning models into production

"MLOps refers to the practice and discipline within machine learning that aims to unify and streamline the machine learning system development (Dev) and machine learning system operation (Ops). It involves collaboration between data scientists, ML engineers, and IT professionals to automate and optimize the end-to-end lifecycle of machine learning applications."



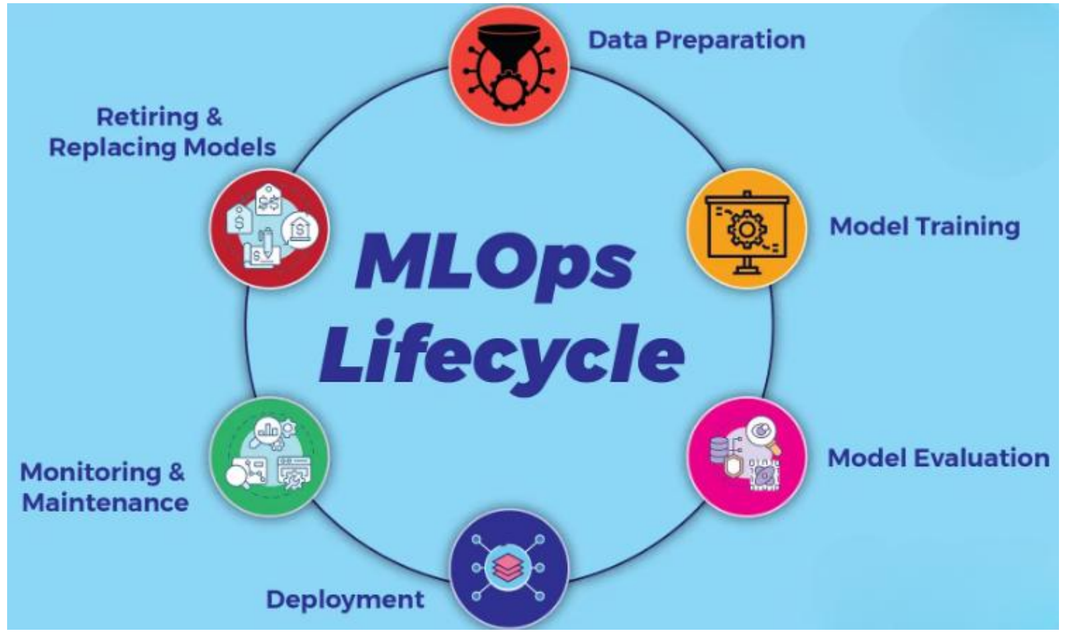
MLOps is to machine learning and data engineering what DevOps is to traditional software development.

💡 แนวคิดหลักของ MLOps

จากแผนภาพ Venn Diagram ด้านบนซ้าย MLOps คือจุดที่เกิดจากการรวมกันของสามสาขาวิชาหลัก:

- **Machine Learning (ML):** เกี่ยวข้องกับการสร้าง, ฝึกฝน, และประเมินผลโมเดลเรียนรู้ของเครื่อง
- **Data Engineering (DE):** เกี่ยวข้องกับการจัดการข้อมูล, การเตรียมข้อมูล, และการสร้างไปป์ไลน์ข้อมูลที่เชื่อถือได้
- **DevOps (Development Operations):** เกี่ยวข้องกับแนวปฏิบัติเพื่อลดวงจรชีวิตของการพัฒนาระบบและการส่งมอบซอฟต์แวร์คุณภาพสูงอย่างต่อเนื่อง (Continuous Integration/Continuous Delivery - CI/CD)

กล่าวโดยสรุป: **MLOps คือสิ่งที่ทำกับ Machine Learning และ Data Engineering คล้ายกับที่ DevOps ทำกับ Software Development ทั่วไป** โดยมีเป้าหมายเพื่อนำโมเดล ML ไปใช้ในการผลิตจริงได้อย่างมีประสิทธิภาพ, เชื่อถือได้, และปรับขนาดได้



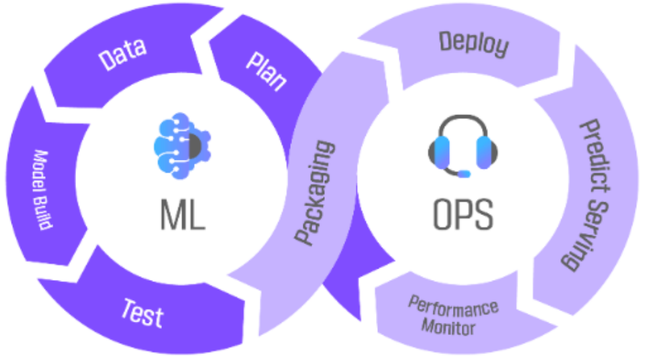
🔄 วงจรชีวิต MLOps (MLOps Lifecycle)

จากแผนภาพวงกลมด้านขวา **MLOps Lifecycle** แสดงให้เห็นขั้นตอนหลักที่ทำงานร่วมกันและเป็นวงจรต่อเนื่อง:

1. **Data Preparation (การเตรียมข้อมูล):** การรวบรวม, ทำความสะอาด, และจัดรูปแบบข้อมูลให้อยู่ในสภาพที่พร้อมสำหรับการฝึกโมเดล
2. **Model Training (การฝึกโมเดล):** การสร้าง, ปรับจูน, และฝึกโมเดล ML โดยใช้ข้อมูลที่เตรียมไว้
3. **Model Evaluation (การประเมินโมเดล):** การวัดประสิทธิภาพของโมเดลที่ฝึกฝนเพื่อตรวจสอบว่าตรงตามเกณฑ์ที่ต้องการหรือไม่
4. **Deployment (การนำไปใช้):** การทำให้โมเดลที่ผ่านการประเมินพร้อมใช้งานจริงในสภาพแวดล้อมการผลิต (Production) เช่น การติดตั้งเป็น API หรือฝังในแอปพลิเคชัน
5. **Monitoring & Maintenance (การเฝ้าระวังและการบำรุงรักษา):** การติดตามประสิทธิภาพของโมเดลที่ทำงานจริง (เช่น ความแม่นยำ, ความคลาดเคลื่อนของข้อมูล หรือ **Data Drift**, ความคลาดเคลื่อนของแนวคิด หรือ **Concept Drift**) และดำเนินการแก้ไขเมื่อจำเป็น
6. **Retiring & Replacing Models (การปลดระวางและเปลี่ยนโมเดล):** การตัดสินใจนำโมเดลเก่าออกและแทนที่ด้วยโมเดลใหม่ที่ดีกว่าที่ผ่านวงจรการฝึกฝนซ้ำ

What is an MLOps platform?

The MLOps platform provides a collaborative environment for software engineers and data scientists. It enables real-time collaboration and iterative data exploration to facilitate experiment tracking, model management, feature engineering, and more.



องค์ประกอบของแพลตฟอร์ม MLOps

จากภาพด้านล่างซ้าย แสดงให้เห็นถึงการผนวกรวมกันของ **ML (Machine Learning) Pipeline** และ **OPS (Operations) Pipeline**:

ส่วน ML:

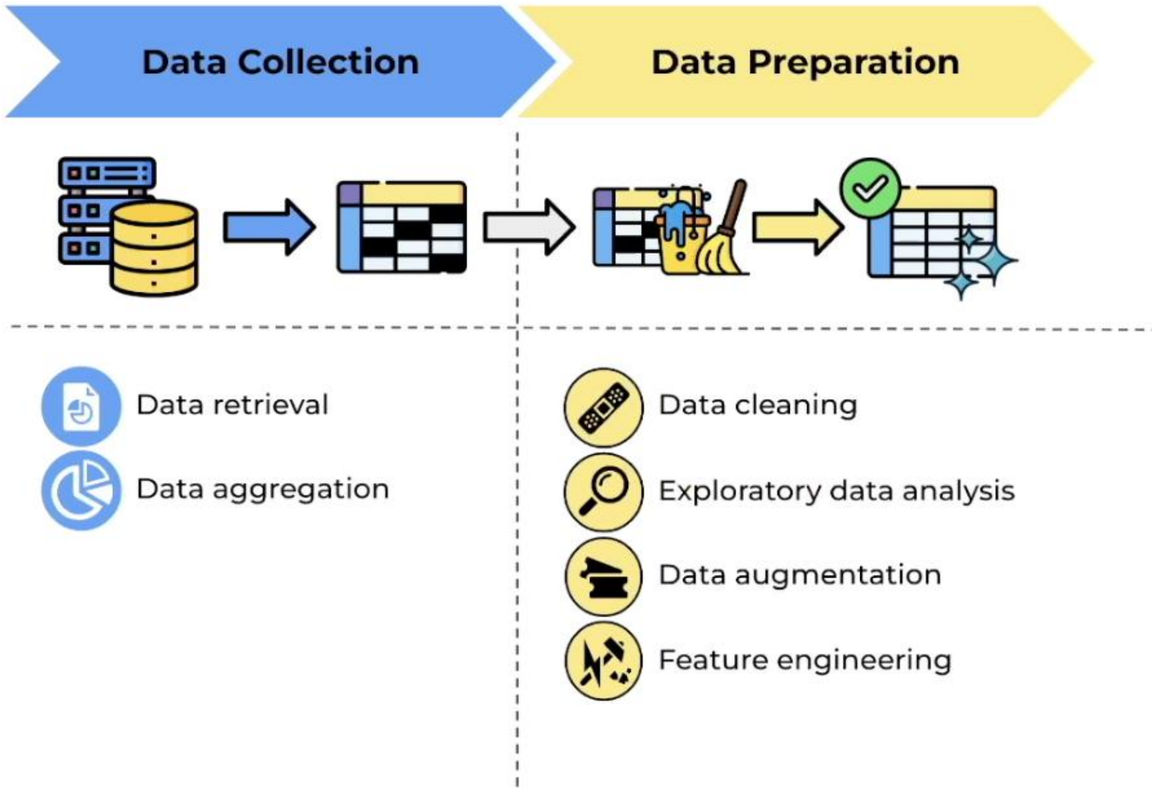
- **Data (ข้อมูล):** เริ่มต้นด้วยข้อมูล
- **Plan (การวางแผน):** การออกแบบการทดลองและกลยุทธ์
- **Test (การทดสอบ):** การทดสอบโมเดลและโค้ด

ส่วน OPS:

- **Packaging (การจัดแพ็คเกจ):** การบรรจุโค้ด, โมเดล, และทรัพยากรที่เกี่ยวข้องทั้งหมดให้อยู่ในรูปแบบที่สามารถนำไปใช้ได้ง่าย (เช่น Containerization)
- **Deploy (การปรับใช้):** การติดตั้งโมเดลและส่วนประกอบที่เกี่ยวข้องในสภาพแวดล้อมจริง
- **Product Serving (การให้บริการผลิตภัณฑ์):** การนำโมเดลที่ติดตั้งไปใช้ตอบสนองคำขอของผู้ใช้หรือระบบอื่น ๆ
- **Performance Monitor (การเฝ้าระวังประสิทธิภาพ):** การติดตามการทำงานของโมเดลในขณะให้บริการจริง (คล้ายกับ Monitoring ใน Lifecycle)

MLOps จึงเป็น **แพลตฟอร์มที่สนับสนุนสภาพแวดล้อมการทำงานร่วมกัน** สำหรับวิศวกรซอฟต์แวร์และนักวิทยาศาสตร์ข้อมูล เพื่อให้การทดลอง, การจัดการโมเดล, และการส่งมอบฟีเจอร์เป็นไปอย่างต่อเนื่องและสามารถติดตามผลได้

ML Product Lifecycle



1. 💾 Data Collection (การรวบรวมข้อมูล)

ขั้นตอนนี้คือการหาและนำข้อมูลดิบมาใช้สำหรับการฝึกฝนโมเดล

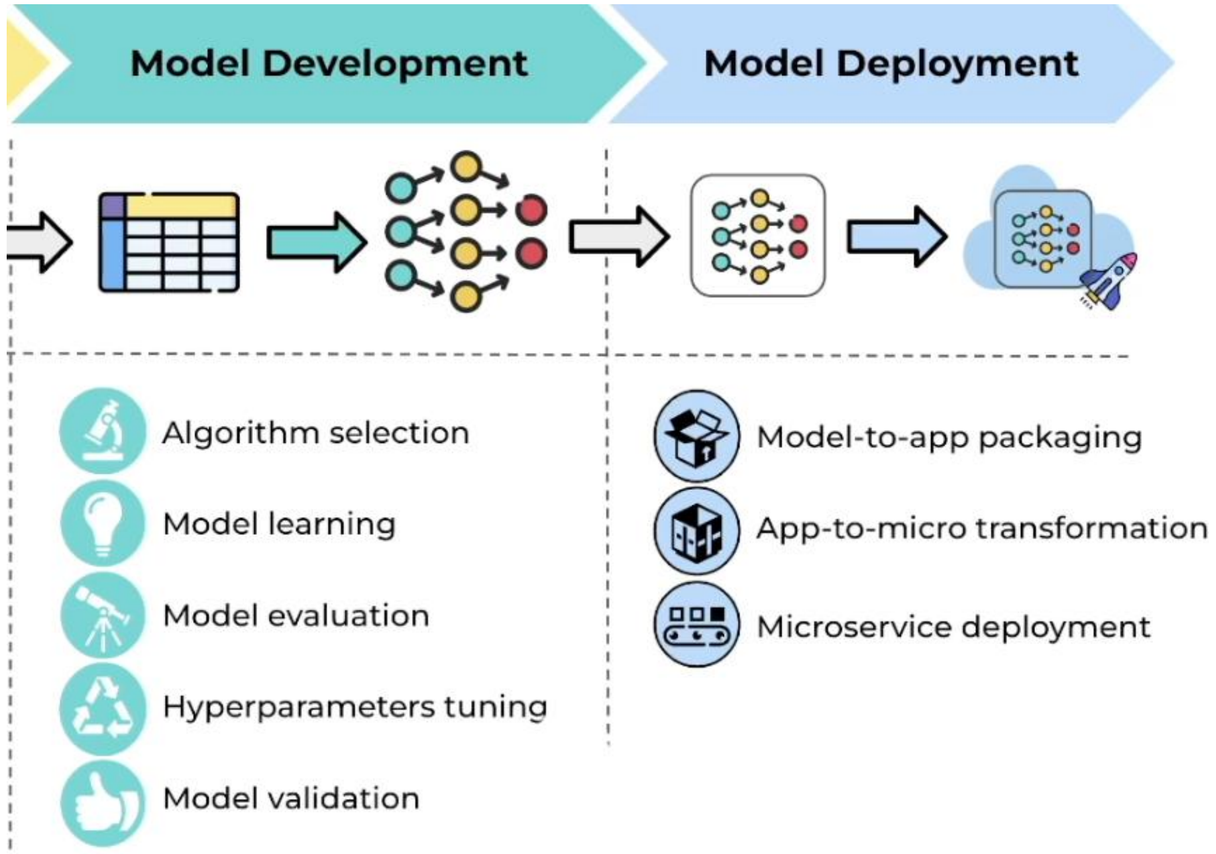
- **Data retrieval (การดึงข้อมูล):** ดึงข้อมูลที่จำเป็นจากแหล่งต่าง ๆ เช่น ฐานข้อมูล, API, หรือไฟล์
- **Data aggregation (การรวมข้อมูล):** นำข้อมูลที่ได้จากหลายแหล่งมารวมกันและจัดระเบียบ

---

2. 🗂️ Data Preparation (การเตรียมข้อมูล)

ขั้นตอนนี้เป็นขั้นตอนที่สำคัญมากในการทำความสะอาด จัดรูปแบบ และแปลงข้อมูลให้อยู่ในสภาพที่พร้อมสำหรับการนำไปใช้ฝึกโมเดล

- **Data cleaning (การทำความสะอาดข้อมูล):** จัดการกับข้อมูลที่ไม่สมบูรณ์, ข้อมูลที่ผิดพลาด, หรือข้อมูลที่ซ้ำซ้อน
- **Exploratory data analysis (การวิเคราะห์ข้อมูลเชิงสำรวจ):** ทำความเข้าใจลักษณะของข้อมูล เช่น การกระจายตัวของข้อมูล, ความสัมพันธ์ระหว่างตัวแปร
- **Data augmentation (การเพิ่มข้อมูล):** สร้างข้อมูลเพิ่มเติมจากข้อมูลที่มีอยู่ (มักใช้กับรูปภาพหรือเสียง) เพื่อเพิ่มขนาดชุดข้อมูล
- **Feature engineering (วิศวกรรมฟีเจอร์):** การสร้างตัวแปร (ฟีเจอร์) ใหม่ ๆ หรือการแปลงฟีเจอร์เดิมเพื่อเพิ่มประสิทธิภาพให้กับโมเดล
- **Feature selection (การเลือกฟีเจอร์):** การเลือกชุดของฟีเจอร์ย่อยที่เหมาะสมที่สุดสำหรับการสร้างโมเดล



3. 🧠 Model Development (การพัฒนาโมเดล)

ขั้นตอนนี้คือการเลือกอัลกอริทึม การฝึกฝนโมเดล และการปรับปรุงประสิทธิภาพของโมเดล

- **Algorithm selection (การเลือกอัลกอริทึม):** เลือกระเบียบวิธี (เช่น Linear Regression, Decision Tree, Neural Networks) ที่เหมาะสมกับประเภทของปัญหาและข้อมูล
- **Model learning (การฝึกฝนโมเดล):** การใช้ชุดข้อมูลที่เตรียมไว้เพื่อฝึกฝนอัลกอริทึมให้เรียนรู้รูปแบบจากข้อมูล
- **Model evaluation (การประเมินโมเดล):** การวัดประสิทธิภาพของโมเดลโดยใช้ชุดข้อมูลทดสอบ (Test Set) ด้วยเมตริกที่เหมาะสม (เช่น Accuracy, Precision, Recall, F1-Score)
- **Hyperparameters tuning (การปรับจูนไฮเปอร์พารามิเตอร์):** การปรับค่าพารามิเตอร์ภายนอกของโมเดล (ที่ไม่ถูกเรียนรู้จากการฝึกฝน) เพื่อให้โมเดลมีประสิทธิภาพสูงสุด
- **Model validation (การตรวจสอบความถูกต้องของโมเดล):** การใช้ชุดข้อมูลการตรวจสอบ (Validation Set) เพื่อยืนยันว่าโมเดลทำงานได้ดีและไม่ได้เกิด Overfitting

---

4. 🚀 Model Deployment (การนำโมเดลไปใช้งาน)

ขั้นตอนนี้คือการนำโมเดลที่พัฒนาเสร็จแล้วไปใช้งานในสภาพแวดล้อมจริง เพื่อให้ผู้ใช้สามารถโต้ตอบกับโมเดลได้

- **Model-to-app packaging (การรวมโมเดลเข้ากับแอปพลิเคชัน):** การจัดแพ็กเกจโมเดลให้อยู่ในรูปแบบที่สามารถเรียกใช้ได้จากแอปพลิเคชันหรือระบบอื่น ๆ
- **App-to-micro transformation (การแปลงแอปพลิเคชันเป็นไมโครเซอร์วิส):** การแยกส่วนฟังก์ชันการทำงานของโมเดลออกมาเป็นบริการย่อย (Microservice)
- **Microservice deployment (การนำไมโครเซอร์วิสไปใช้งาน):** การติดตั้งและเรียกใช้งานบริการย่อยของโมเดลบนเซิร์ฟเวอร์หรือ Cloud Platform

5. 📉 Monitoring & Maintenance (การเฝ้าระวังและการบำรุงรักษา)

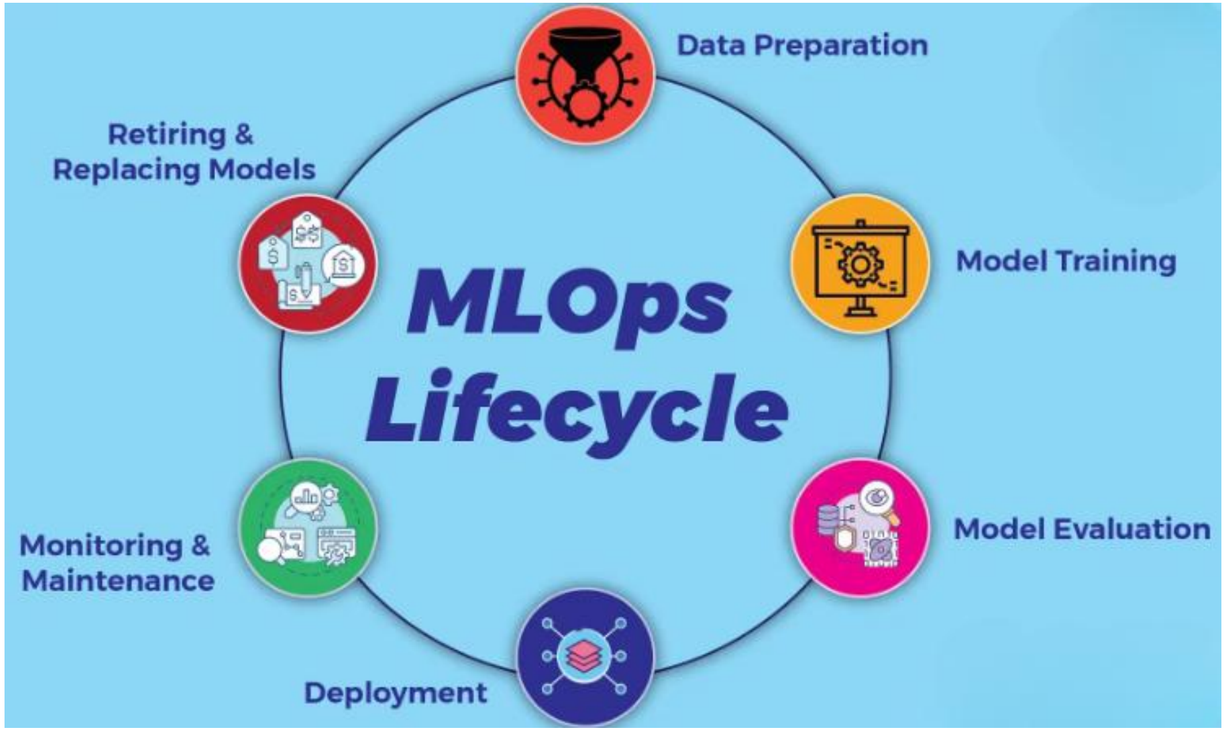
หลังจากโมเดลถูกนำไปใช้งานจริงแล้ว จำเป็นต้องติดตามประสิทธิภาพอย่างต่อเนื่อง

- **วัตถุประสงค์:** ตรวจสอบว่าโมเดลยังคงทำงานได้ถูกต้องและมีประสิทธิภาพตามที่คาดหวัง
- **กิจกรรมหลัก:**
  - **Model Performance Monitoring:** ตรวจสอบ Metrics ทางสถิติและความแม่นยำของโมเดล
  - **Data Drift & Concept Drift Detection:** ตรวจจับความคลาดเคลื่อนของข้อมูลขาเข้า หรือการเปลี่ยนแปลงความสัมพันธ์ระหว่างข้อมูลและผลลัพธ์
  - **System Monitoring:** ตรวจสอบความพร้อมใช้งาน (Uptime), ความหน่วง (Latency), และการใช้ทรัพยากรของบริการ (CPU/RAM)
- **MLOps Focus:** กำหนด **Thresholds** (เกณฑ์) สำหรับ Metrics หากประสิทธิภาพของโมเดลต่ำกว่าเกณฑ์ที่กำหนดไว้ ระบบจะต้องแจ้งเตือน (Alert) และที่สำคัญที่สุดคือ **Trigger** ให้เกิดการฝึกซ้ำ (Retraining) อัตโนมัติ

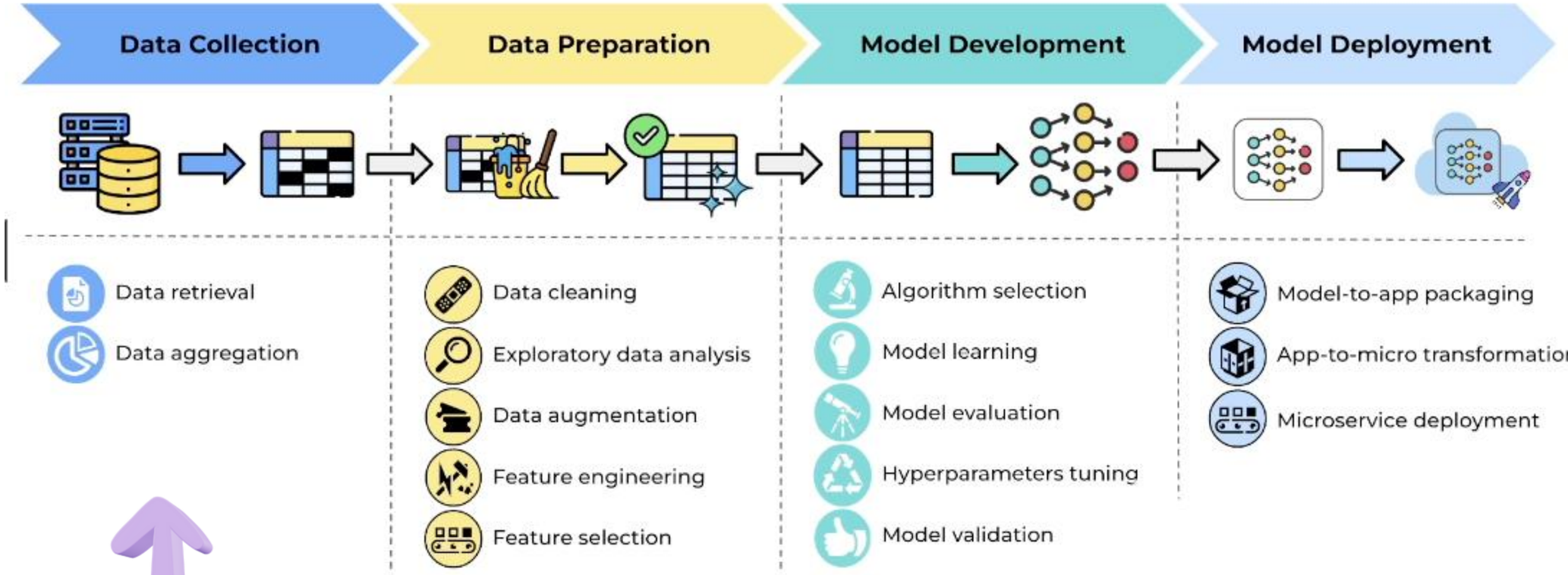
6. ♻️ Retiring & Replacing Models (การปลดระวางและเปลี่ยนโมเดล)

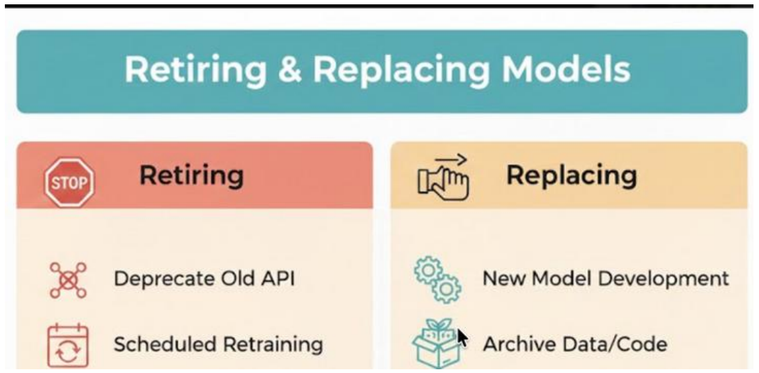
ขั้นตอนนี้เป็นจุดเชื่อมต่อกลับไปสู่จุดเริ่มต้นของวงจร

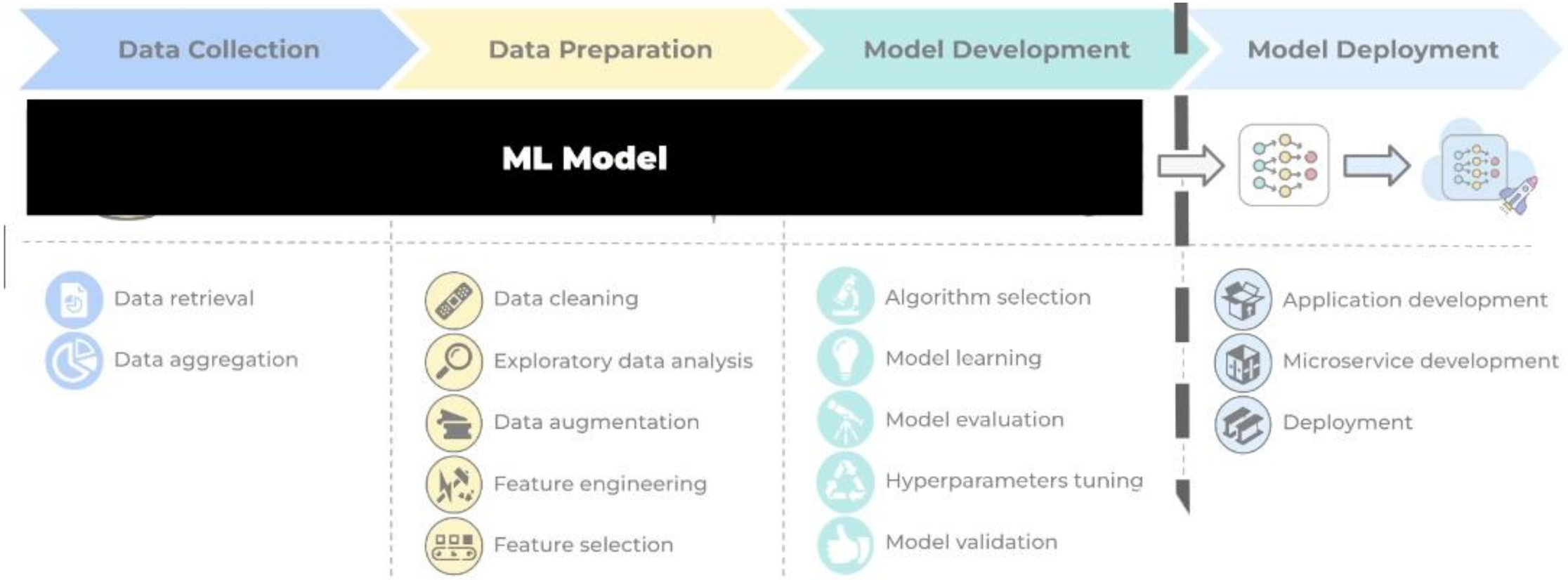
- **วัตถุประสงค์:** จัดการวงจรชีวิตของโมเดลเก่า และนำโมเดลใหม่ที่ฝึกซ้ำมาแทนที่
- **กิจกรรมหลัก:**
  - **Decision to Retrain:** การตัดสินใจฝึกโมเดลใหม่มักเกิดจากการแจ้งเตือนจากขั้นตอน Monitoring
  - **Model Retirement:** การนำโมเดลเวอร์ชันเก่าออกจาก Production อย่างเป็นระเบียบ
  - **Pipeline Triggering:** การรันไปป์ไลน์ทั้งหมด (ตั้งแต่ Data Preparation ใหม่) เพื่อสร้างโมเดลเวอร์ชันใหม่ที่เรียนรู้จากข้อมูลล่าสุด
- **MLOps Focus:** ความสามารถในการทำซ้ำ (Repeatability) และความเป็นอัตโนมัติ (Automation) คือกุญแจสำคัญ เพราะการอัปเดตโมเดลควรเกิดขึ้นบ่อยครั้งและเป็นไปตามกลไกที่กำหนด



Machine Learning Product Lifecycle



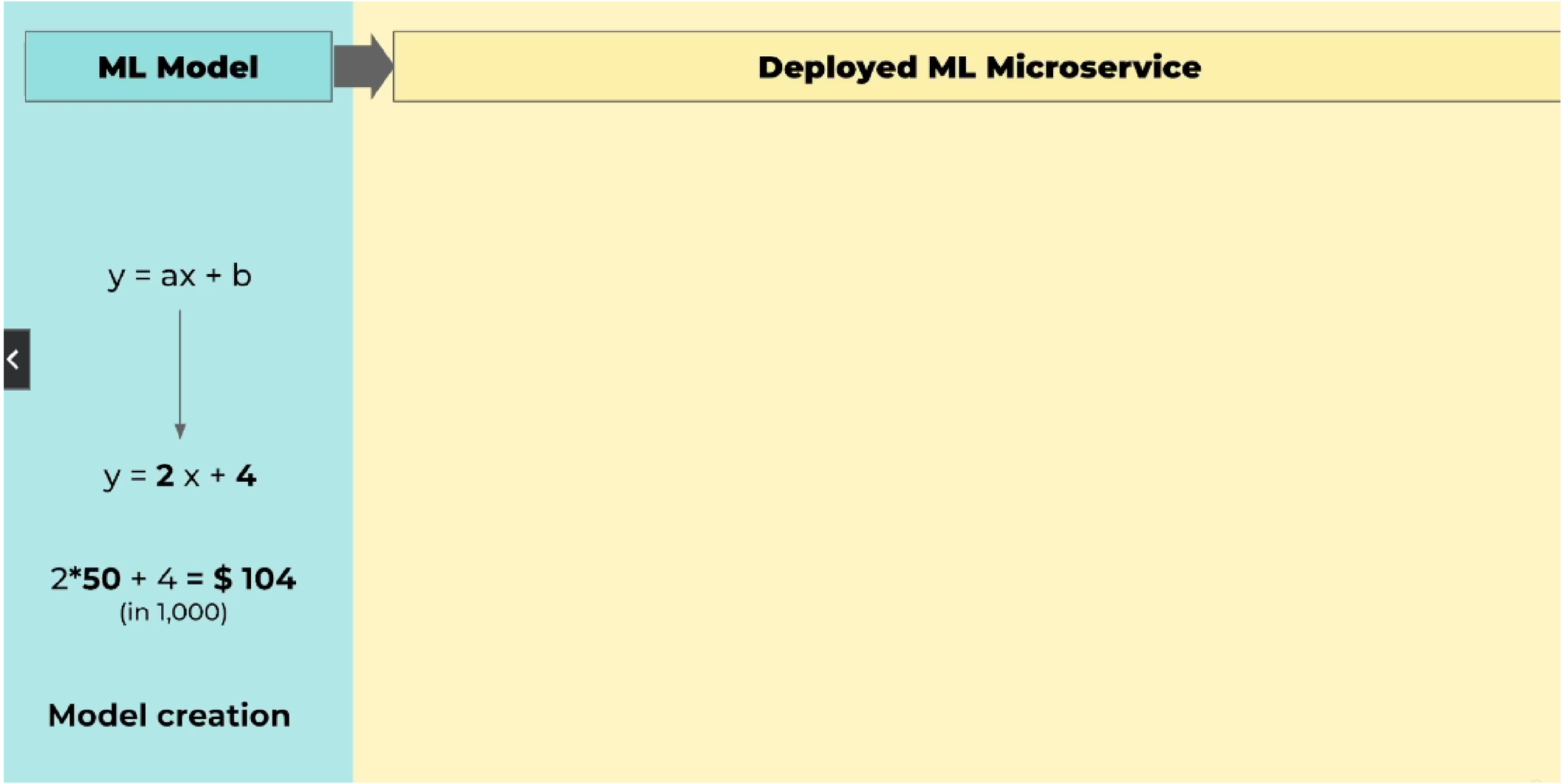


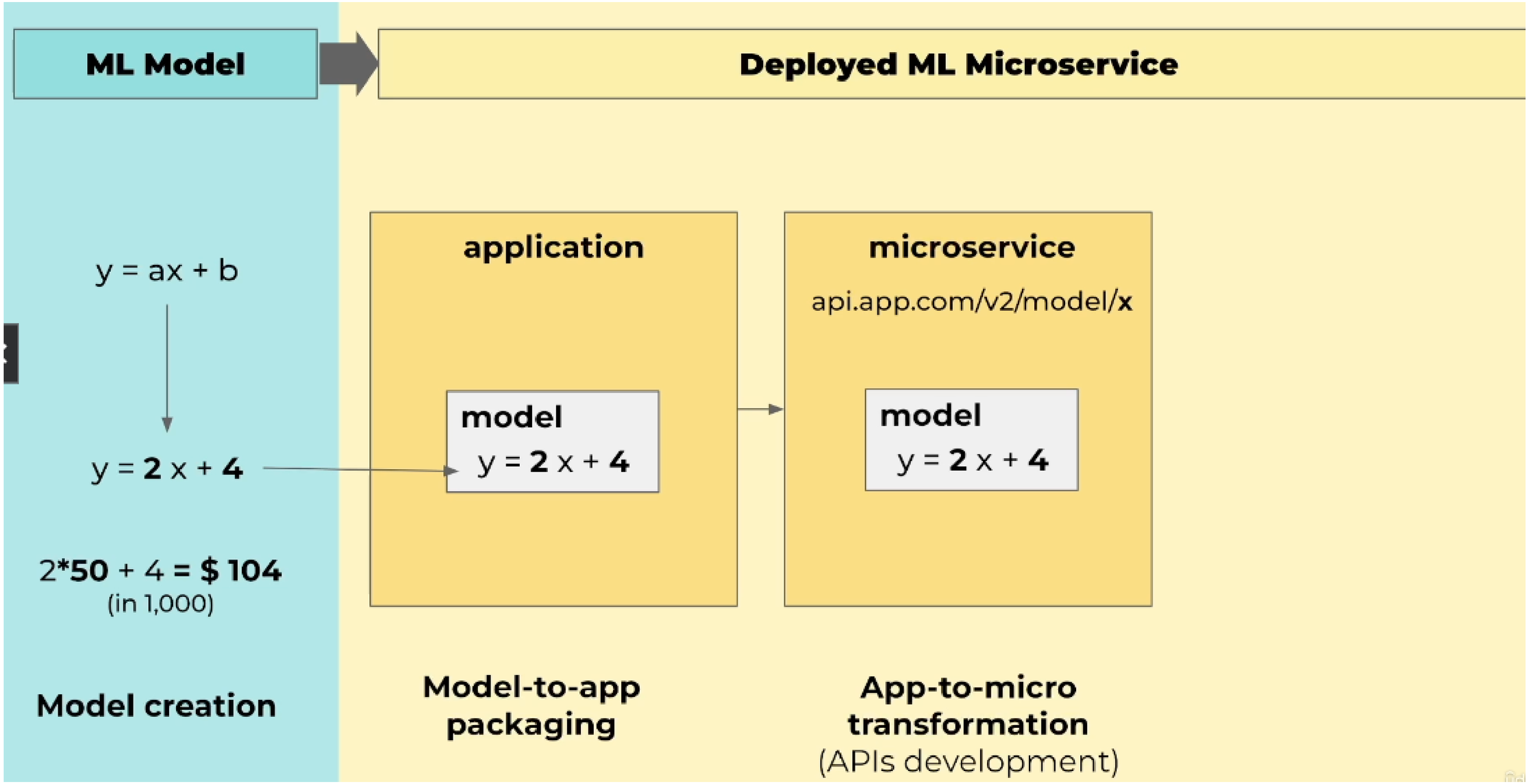

Data Collection
Data Preparation
Model Development
Model Deployment
ML Model
Data retrieval
Data aggregation
Data cleaning
Exploratory data analysis
Data augmentation
Feature engineering
Feature selection
Algorithm selection
Model learning
Model evaluation
Hyperparameters tuning
Model validation
Application development
Microservice development
Deployment

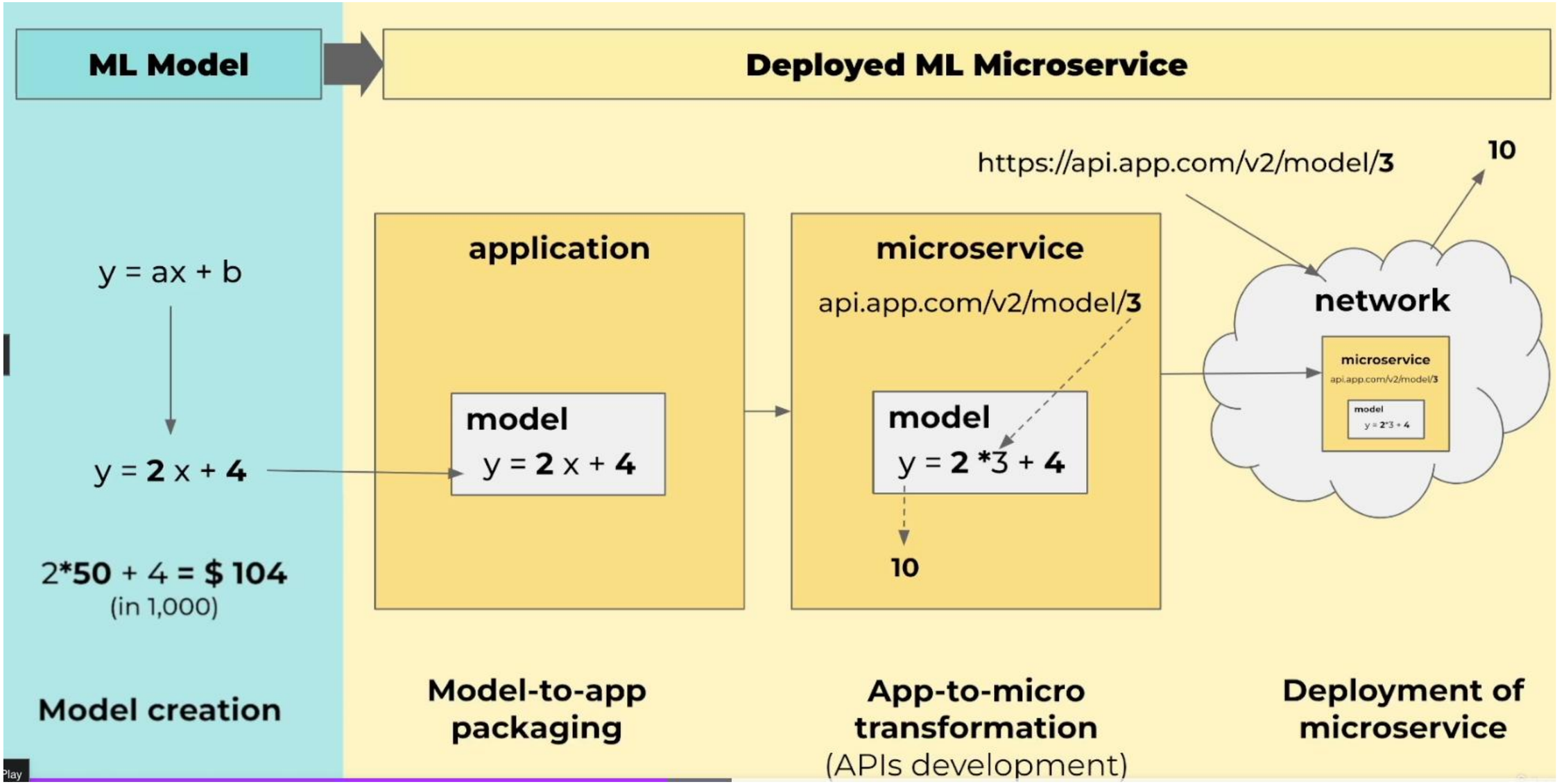
Data Collection → Data Preparation → Model Development → Model Deployment

ML Model

Deployed Microservice


ML Model
Deployed ML Microservice
y = ax + b
y = 2 x + 4
2*50 + 4 = $ 104
(in 1,000)
Model creation

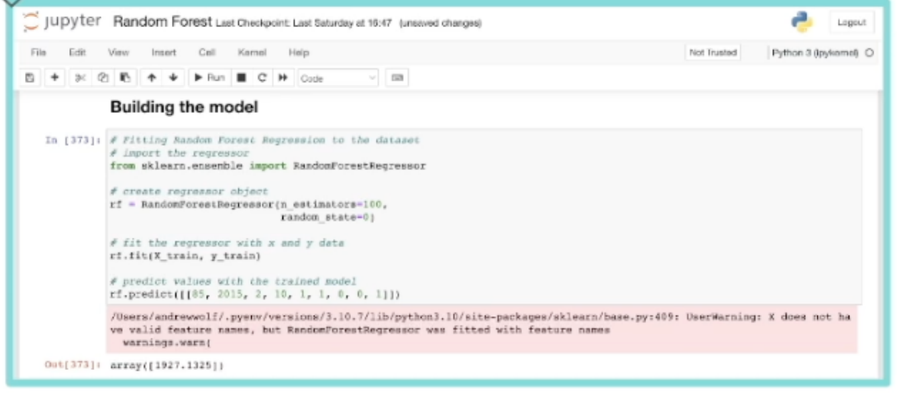

ML Model
Deployed ML Microservice
y = ax + b
y = 2 x + 4
2*50 + 4 = $ 104
(in 1,000)
Model creation
application
model
y = 2 x + 4
Model-to-app packaging
microservice
api.app.com/v2/model/x
model
y = 2 x + 4
App-to-micro transformation
(APIs development)


ML Model
Deployed ML Microservice
https://api.app.com/v2/model/3
10
y = ax + b
y = 2 x + 4
2*50 + 4 = $ 104
(in 1,000)
application
model
y = 2 x + 4
microservice
api.app.com/v2/model/3
model
y = 2 *3 + 4
10
network
microservice
api.app.com/v2/model/3
model
y = 2*3 + 4
Model creation
Model-to-app packaging
App-to-micro transformation
(APIs development)
Deployment of microservice

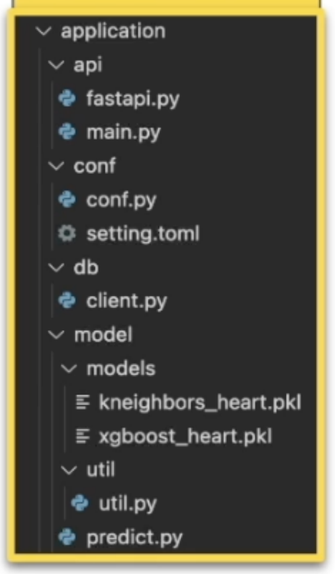
ML Model

Deployed ML Microservice

1 Model creation in Jupyter Notebook



2 Model-to-app packaging



3 App-to-micro transformation via APIs

```
api.app.com/v2/model/84,23,53,43/prediction
```

4 Deploying ML microservice

```
version: '3.10'
services:
    app:
        build: app/.
        restart: always
        environment:
        - DOCKER_HOME=/home
        - LOAD_EX=n
        - EXECUTOR=Local
        ports:
        - "5050:5050"
        volumes:
        - ./app:/home
        networks:
        - main_net
```

Popular tools for MLOps Process

Version Control

GitHub

MLOPS