

ML Wdrażanie modeli



ML Wdrażanie modeli

Agenda

1

Narzędzia do wdrażania i monitorowania modeli

2

Docker

3

Proces wdrażania i monitorowania modeli

info **Share**
ACADEMY



ML Wdrażanie modeli

Agenda

1

Narzędzia do wdrażania i monitorowania modeli

2

Docker

3

Proces wdrażania i monitorowania modeli

info **Share**
ACADEMY



ML Wdrażanie modeli

Agenda

1 Narzędzia do wdrażania i monitorowania modeli

2 Docker

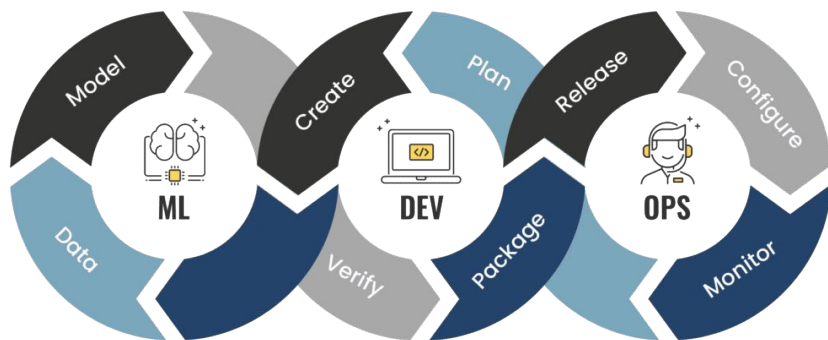
3 **Proces wdrażania i monitorowania modeli**



ML Wdrażanie modeli

MLOps

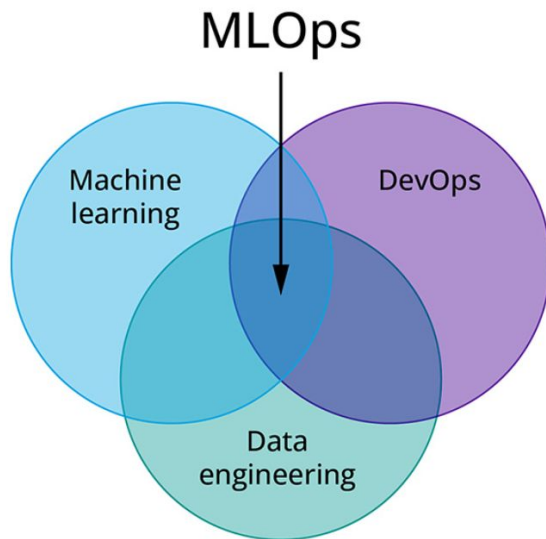
info **Share**
ACADEMY





ML Wdrażanie modeli

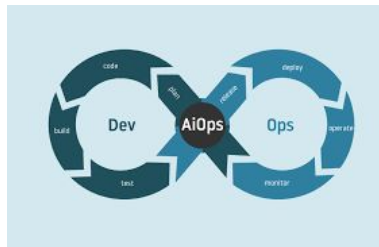
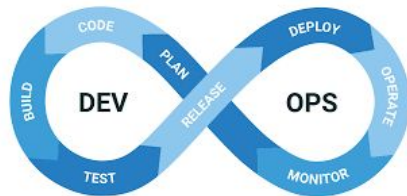
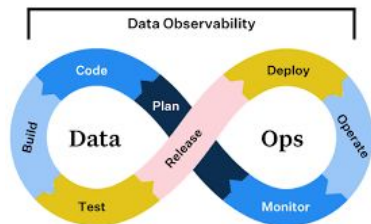
MLOps – dlaczego?



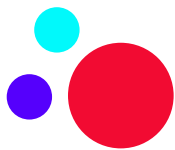


ML Wdrażanie modeli

MLOps a inne techniki



info **Share**
ACADEMY

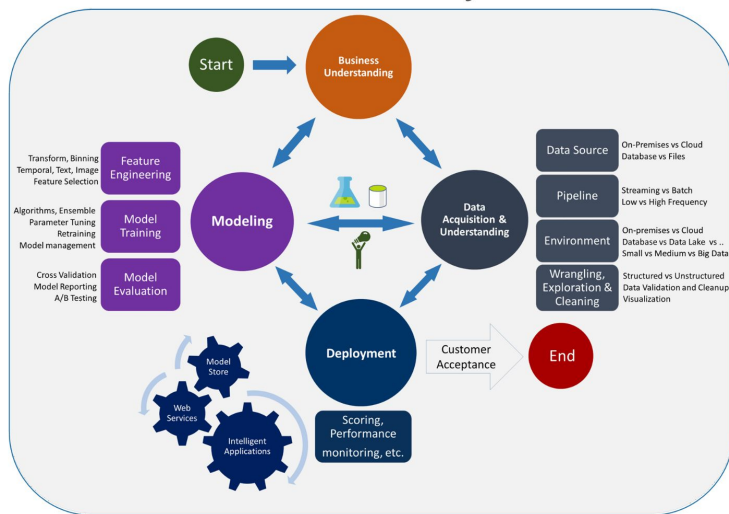


ML Wdrażanie modeli

MLOps a inne techniki

info **Share**
ACADEMY

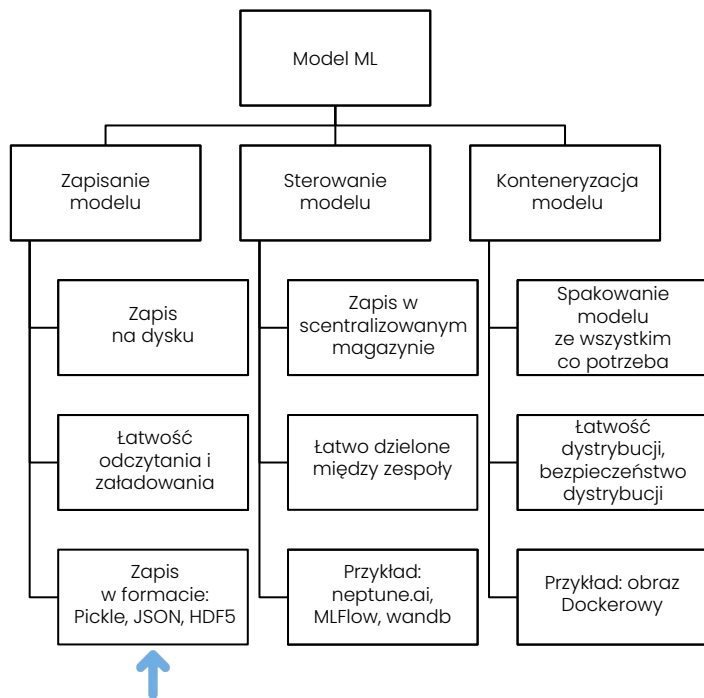
Data Science Lifecycle





ML Wdrażanie modeli

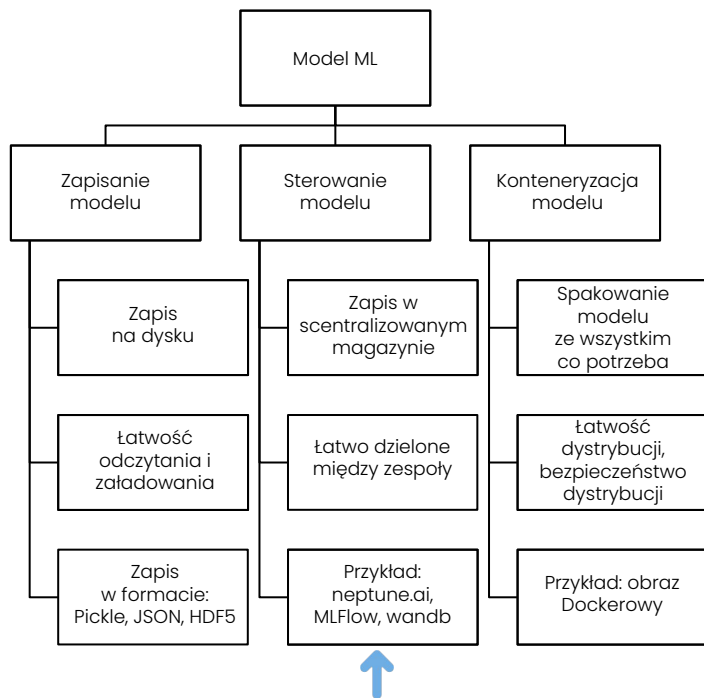
Zapisanie modelu





ML Wdrażanie modeli

Sterowanie modelem



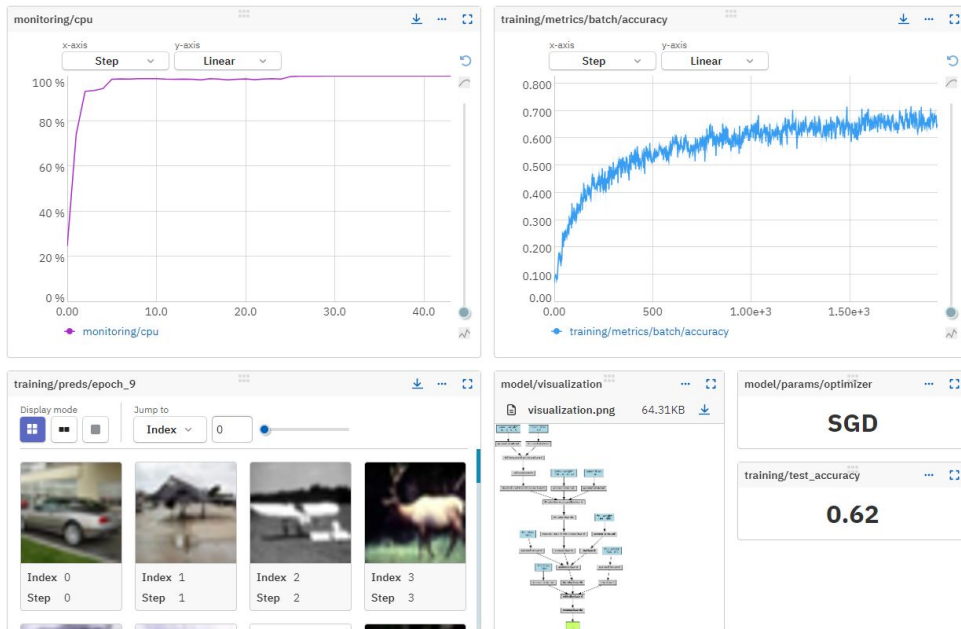


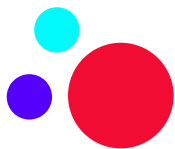
ML Wdrażanie modeli

Sterowanie modelem



info **Share**
ACADEMY





ML Wdrażanie modeli

Sterowanie modelem

mlflow

The screenshot shows the MLflow Experiments page for the 'boston-housing' experiment. It includes a search bar, a list of experiments, and a table of search runs. The table columns are: Start Time, Run Name, User, Source, Version, Features, min-sample-leaf, n-estimators, and three metrics: AZ-ul-10th-perce, AZ-ul-10th-perce, and AZ-ul-10th-perce.

	Start Time	Run Name	User	Source	Version	Features	min-sample-leaf	n-estimators	AZ-ul-10th-perce	AZ-ul-10th-perce	AZ-ul-10th-perce
<input type="checkbox"/>	2020-09-19 18:37:43	-	root	train.py	-	CRW.ZN/INDU...	7	71	0.226	1.375	4.913
<input type="checkbox"/>	2020-09-19 18:37:02	-	root	train.py	-	CRW.ZN/INDU...	7	86	0.226	1.525	4.469
<input type="checkbox"/>	2020-09-19 18:36:35	-	root	train.py	-	CRW.ZN/INDU...	5	21	0.274	1.37	4.026
<input type="checkbox"/>	2020-09-19 18:36:33	-	root	train.py	-	CRW.ZN/INDU...	10	66	0.271	1.435	4.341
<input type="checkbox"/>	2020-09-19 18:36:30	-	root	train.py	-	CRW.ZN/INDU...	8	78	0.27	1.4	4.348
<input type="checkbox"/>	2020-09-19 18:36:19	-	root	train.py	-	CRW.ZN/INDU...	10	110	0.332	1.391	4.909
<input type="checkbox"/>	2020-09-19 18:36:14	-	root	train.py	-	CRW.ZN/INDU...	10	20	0.266	1.401	4.635

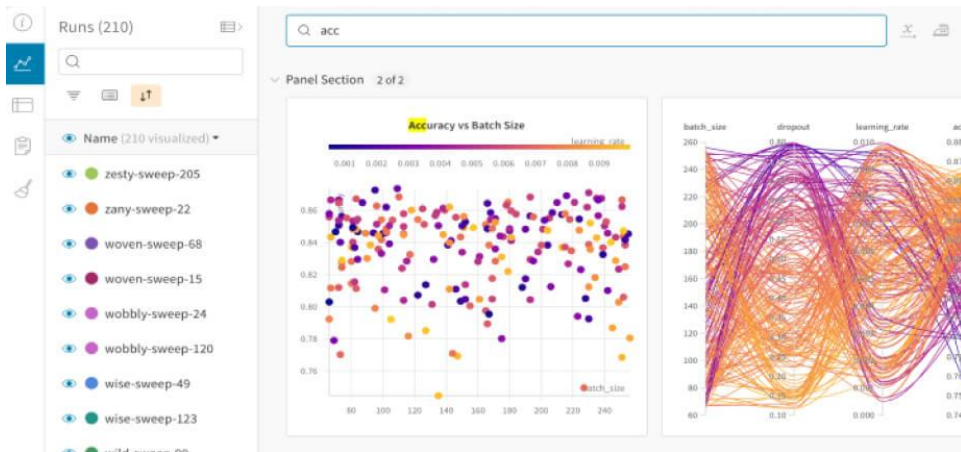
info **Share**
ACADEMY



ML Wdrażanie modeli

Sterowanie modelem

W&B

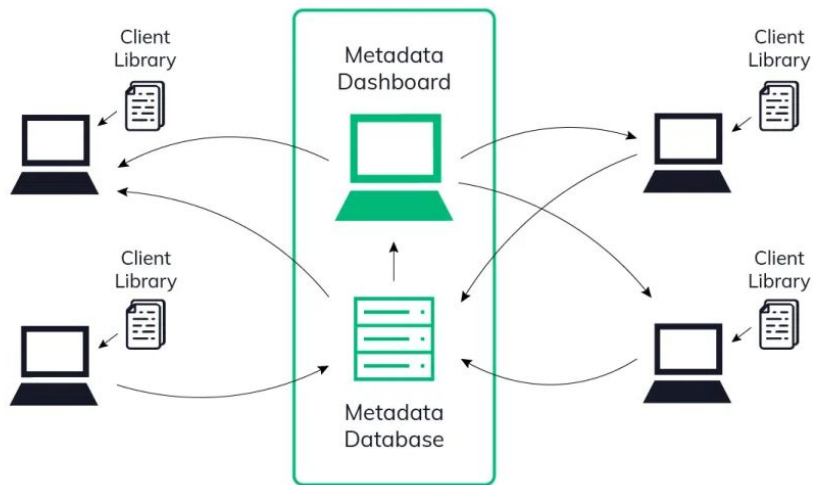


infoShare
ACADEMY



ML Wdrażanie modeli

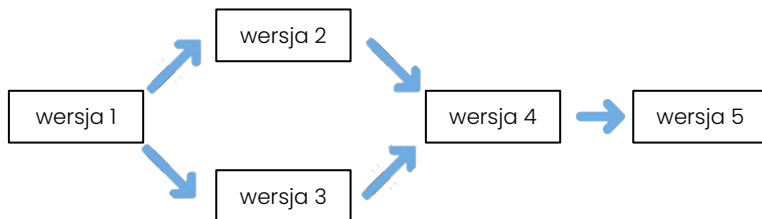
Scentralizowany magazyn eksperymentów



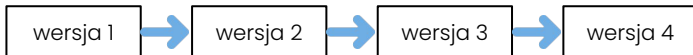


ML Wdrażanie modeli

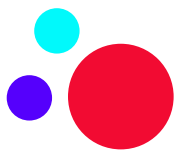
Dlaczego kontrola wersji modeli jest kluczowa?



?

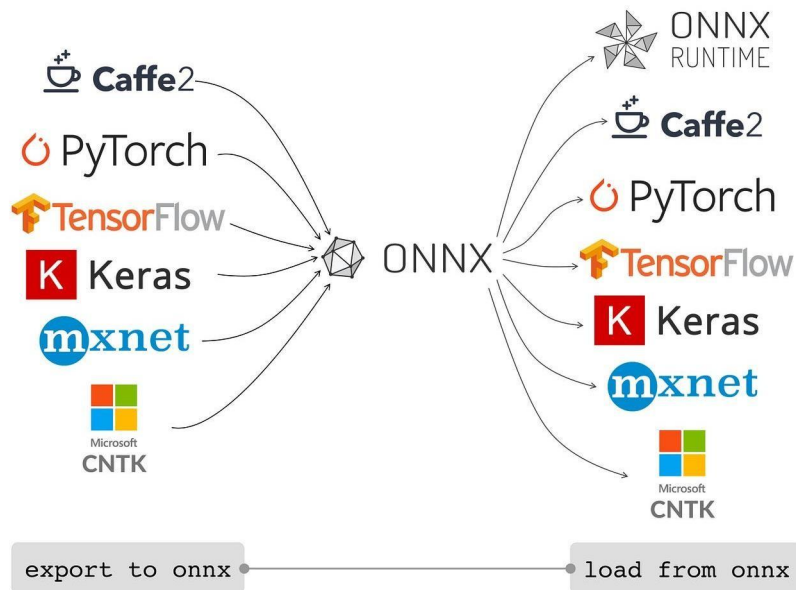


info **Share**
ACADEMY



ML Wdrażanie modeli

Standaryzacja frameworków dla modeli DL

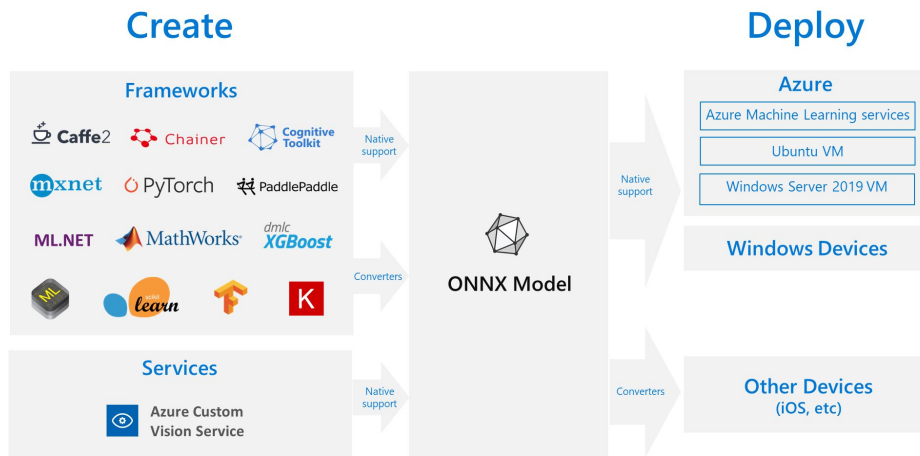


info **Share**
ACADEMY



ML Wdrażanie modeli

Standaryzacja frameworków dla modeli DL



info **Share**
ACADEMY

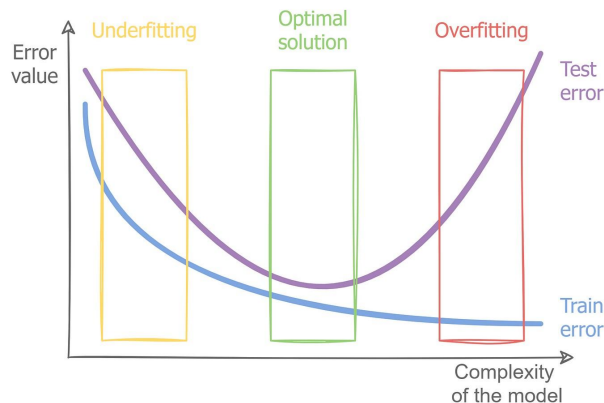


ML Wdrażanie modeli

Kontrola złożoności modelu

Złożoność modelu odnosi się do poziomu skomplikowania lub pojemności modelu, czyli zdolności modelu do dopasowania się do danych treningowych.

Kontrola złożoności ma na celu zapobieganie nadmiernemu dopasowaniu (overfitting) oraz niedostatecznemu dopasowaniu (underfitting) modelu.

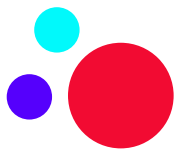




ML Wdrażanie modeli

Jak postępować?

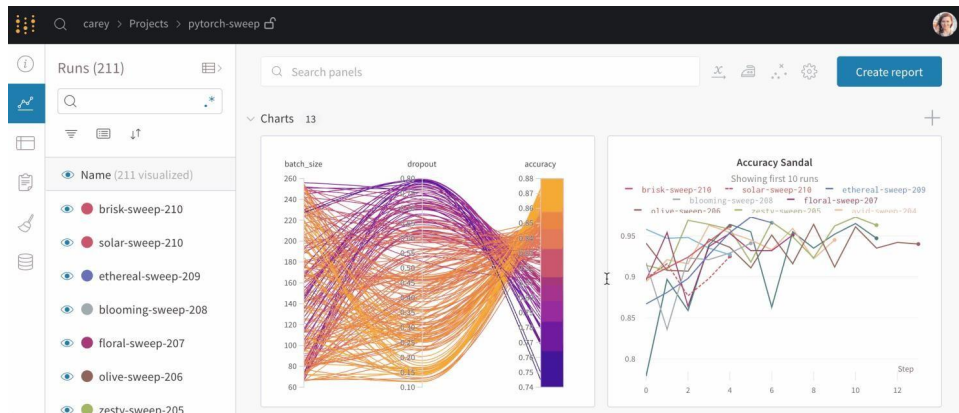
- Wybór odpowiedniego modelu
- Ograniczanie liczby cech
- Kontrola hiperparametrów
- Cross-Validation
- Regularyzacja
- Przykłady danych
- Early Stopping
- Ensemble Learning
- Ocena błędu
- Interpretacja modelu



ML Wdrażanie modeli

Monitorowanie modeli

infoShare
ACADEMY





ML Wdrażanie modeli

Monitorowanie rozwiązań produkcyjnych



AWS CloudWatch

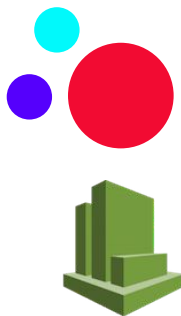


Application Insights



infoShareAcademy.com

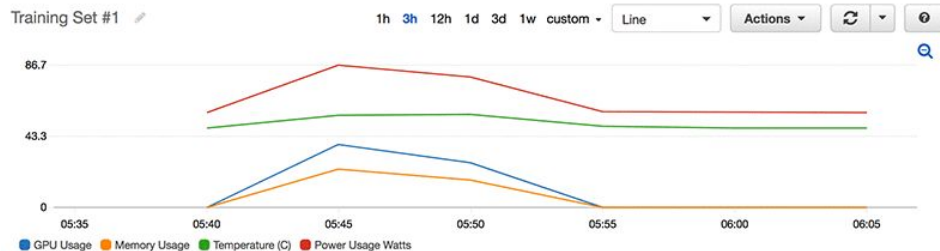
info **Share**
ACADEMY



ML Wdrażanie modeli

Monitorowanie rozwiązań produkcyjnych

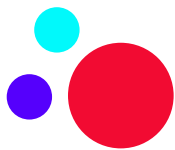
AWS CloudWatch



All metrics Graphed metrics (4) Graph options

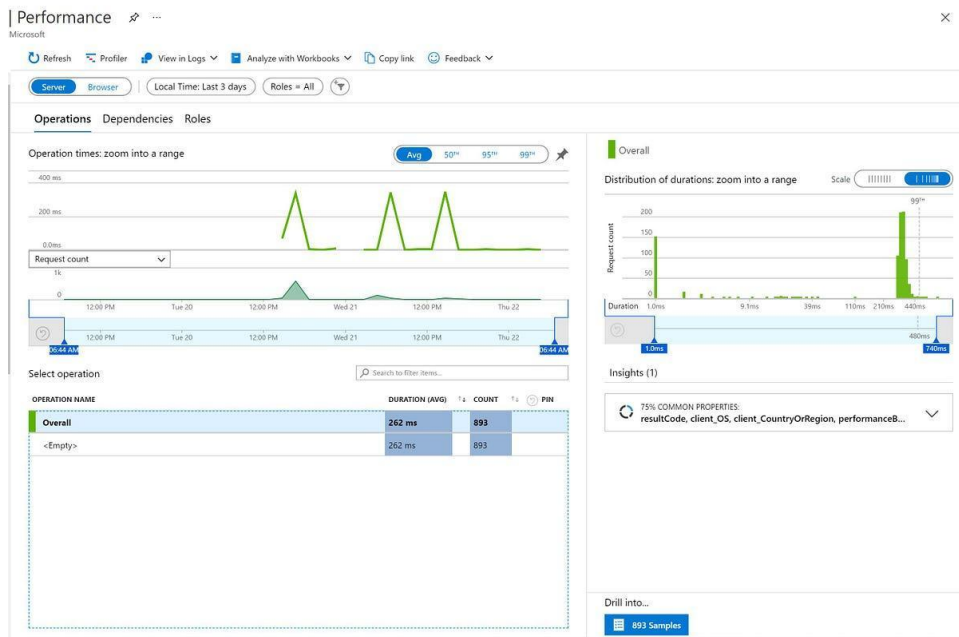
All > DeepLearningTrain > GPUNumber, ImageId, InstanceId, InstanceType

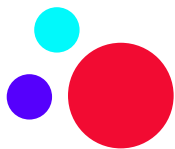
	GPUNumber (4)	ImageId	InstanceId	InstanceType	Metric Name
<input checked="" type="checkbox"/>	0	ami-4b44745d	i-026a45573df762ac2	p2.xlarge	GPU Usage
<input checked="" type="checkbox"/>	0	ami-4b44745d	i-026a45573df762ac2	p2.xlarge	Memory Usage
<input checked="" type="checkbox"/>	0	ami-4b44745d	i-026a45573df762ac2	p2.xlarge	Temperature (C)
<input checked="" type="checkbox"/>	0	ami-4b44745d	i-026a45573df762ac2	p2.xlarge	Power Usage Watts



ML Wdrażanie modeli

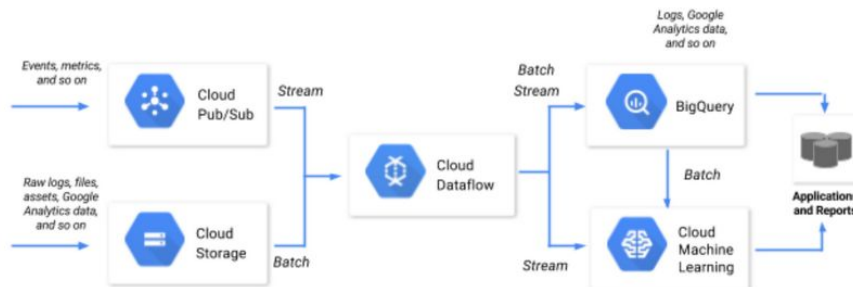
Monitorowanie rozwiązań produkcyjnych





ML Wdrażanie modeli

Monitorowanie rozwiązań produkcyjnych





ML Wdrażanie modeli

Wytyczne dotyczące bezpieczeństwa
w kontekście MLOps



info **Share**
ACADEMY



ML Wdrażanie modeli

**Znaczenie zgodności z przepisami
prawno-etycznymi**

Jak radzić sobie z przypadkowością:

1. Ochrona danych osobowych.
2. Transparentność i wyjaśnialność modeli.
3. Unikanie dyskryminacji i niesprawiedliwości.
4. Zarządzanie bezpieczeństwem.
5. Uprawnienia i zgody.
6. Monitorowanie i audyt.
7. Dostosowywanie się do zmian w przepisach.



ML Wdrażanie modeli

Skrypt z wprowadzonymi parametrami

Narzędzia takie jak notebooki są praktyczne gdy pracujemy nad modelami bądź chcemy zaprezentować wyniki.

Gdy chcemy zautomatyzować proces potrzebujemy skryptów, które umożliwiają nam wprowadzanie różnych zmiennych takich jak zakres dat, ścieżki odczytu czy zapisu itd.



**JUPYTER
NOTEBOOK**

**PYTHON
SCRIPT**

info Share
ACADEMY



ML Wdrażanie modeli

Flask



Flask

Stosunkowo łatwym narzędziem w Pythonie do postawienia API serwującego modele jest pakiet Flask.

Po uruchomieniu programu możemy na wskazanym hoście i porcie odpytać się o wynik predykcji wysyłając metodą POST dane w formacie json.



ML Wdrażanie modeli

Instalacja Flask

pip install Flask ← instalacja

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('index.html')
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

python app.py ← uruchomienie aplikacji



ML Wdrażanie modeli

Routing we Flasku

Dekorator @app.route:

```
from flask import Flask
```

```
app = Flask(__name__)
```

```
@app.route('/')
```

```
def home():
```

```
    return 'Strona główna'
```

```
@app.route('/about')
```

```
def about():
```

```
    return 'Strona informacyjna'
```



ML Wdrażanie modeli

Routing we Flasku

Parametry:

```
@app.route('/user/<username>')  
def show_user(username):  
    return f'Profil użytkownika {username}'
```



ML Wdrażanie modeli

Routing we Flasku

Metody HTTP:

```
@app.route('/submit', methods=['POST', 'GET'])
def submit_form():
    if request.method == 'POST':
        return 'Dane zostały przesłane!'
    else:
        return 'Formularz do przesyłania danych'
```



Zadanie 17.1 (instrukcja)

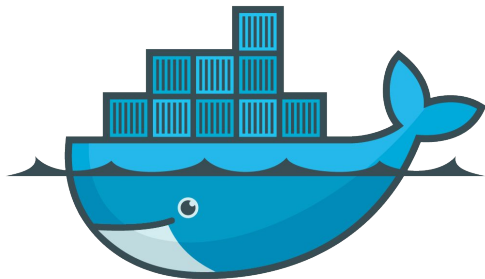
Zadaniem jest stworzenie prostego interfejsu programistycznego (API) przy użyciu frameworku Flask w języku Python.

Twoje API powinno obsługiwać zapytania HTTP GET i POST, umożliwiając użytkownikowi pobieranie informacji oraz dodawanie nowych danych.



ML Wdrażanie modeli

Docker



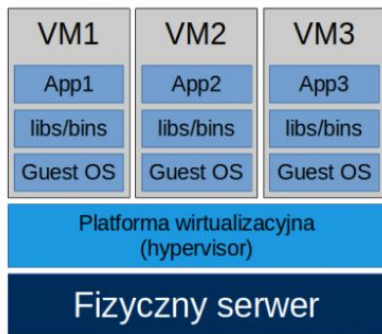
docker



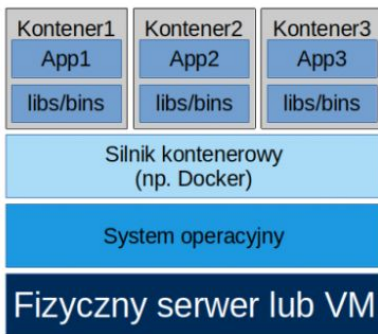
ML Wdrażanie modeli

Co to jest konteneryzacja i dlaczego warto jej używać?

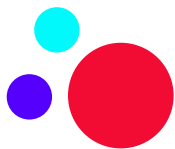
Wirtualne maszyny:



Kontenery:



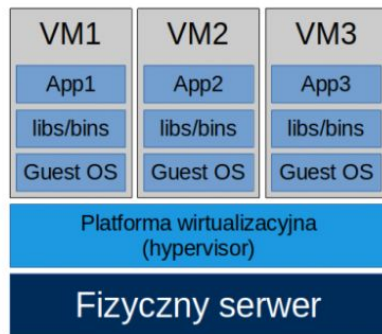
info **Share**
ACADEMY



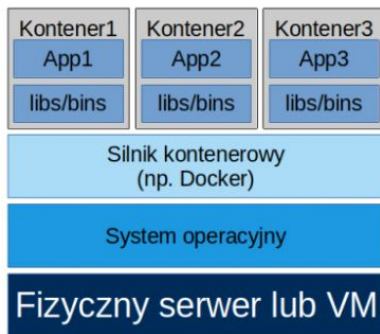
ML Wdrażanie modeli

Różnice między wirtualizacją a konteneryzacją

Wirtualne maszyny:



Kontenery:

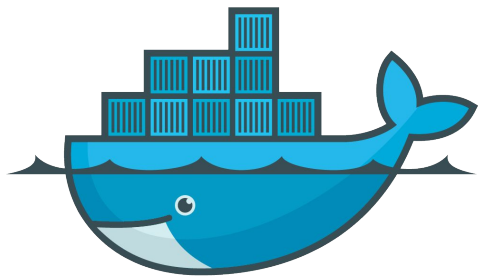


info **Share**
ACADEMY



ML Wdrażanie modeli

Instalacja Dockera na różnych
systemach operacyjnych

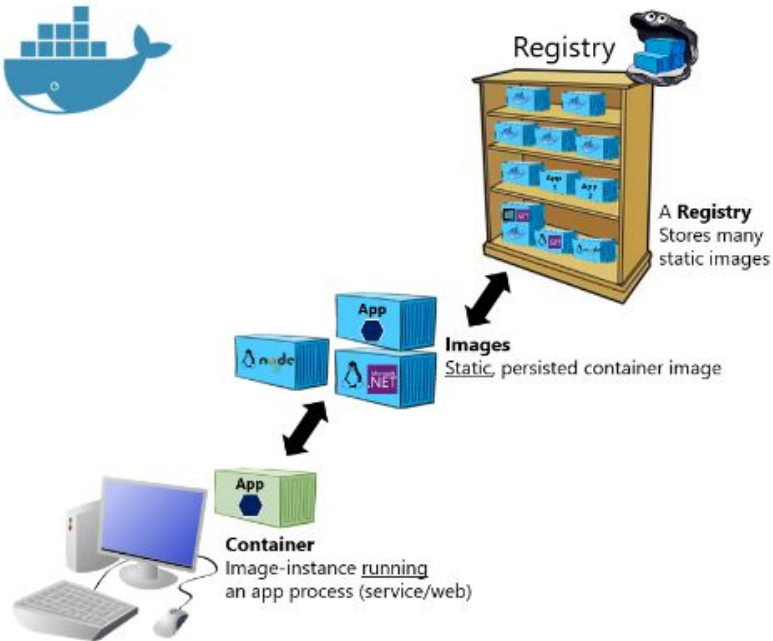


docker



ML Wdrażanie modeli

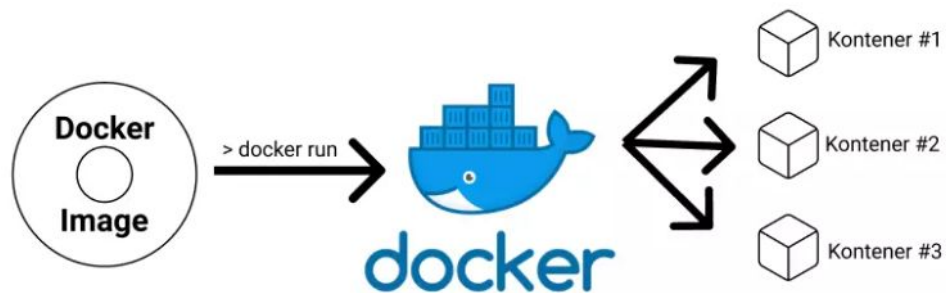
Docker – podstawowa taksonomia





ML Wdrażanie modeli

Kontenery w Dockerze

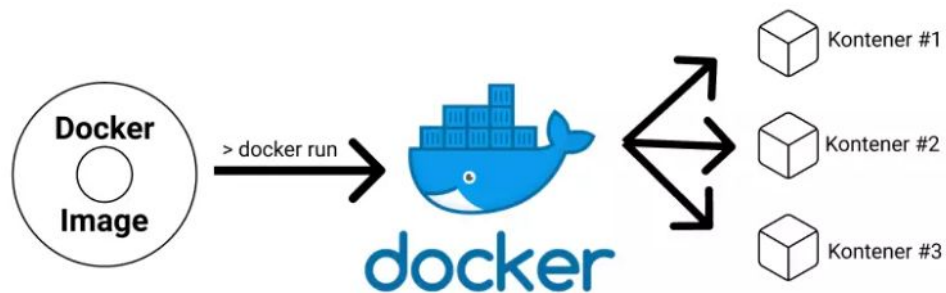


info **Share**
ACADEMY

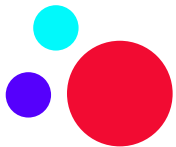


ML Wdrażanie modeli

Obrazy w Dockerze



info **Share**
ACADEMY



ML Wdrażanie modeli

Dockerfile

Ustalamy bazowy obraz, w tym przypadku używamy obrazu z Pythonem:

FROM python:3.8

Tworzymy katalog roboczy w kontenerze:

WORKDIR /app

Kopiujemy pliki z aktualnego katalogu (gdzie znajduje się Dockerfile) do katalogu roboczego w kontenerze:

COPY . /app

Instalujemy zależności (w tym przypadku używamy pip, narzędzia do zarządzania pakietami w Pythonie):

RUN pip install --no-cache-dir -r requirements.txt

Wskazujemy, że aplikacja będzie dostępna na porcie 5000:

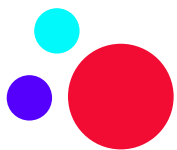
EXPOSE 5000

Uruchamiamy aplikację przy pomocy komendy CMD:

CMD ["python", "app.py"]

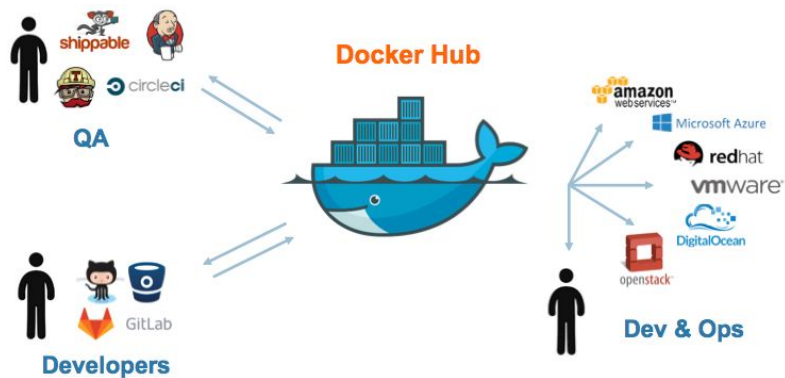
infoShareAcademy.com

info Share
ACADEMY



ML Wdrażanie modeli

Docker Hub





info **Share**
ACADEMY



ML Wdrażanie modeli

Docker Hub



python 


Updated 4 days ago


Python is an interpreted, interactive, object-oriented, open-source programming language.

Windows Linux ARM 64 386 PowerPC 64 LE x86-64 IBM Z mips64le ARM

📦 1B+ · ⭐ 9.4K

Pulls: 7,707,661
Last week



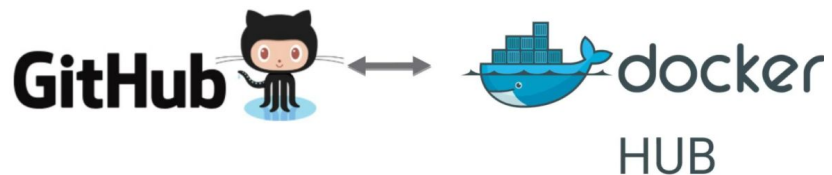
[Learn more](#) 

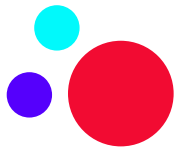
info **Share**
ACADEMY



ML Wdrażanie modeli

Docker Hub





ML Wdrażanie modeli

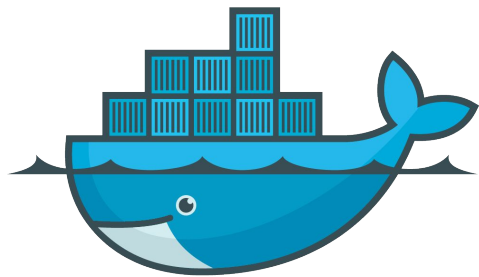
Podstawowe polecenia Docker CLI

1. `docker --version`
2. `docker pull <image>`
3. `docker images` lub `docker image ls`
4. `docker ps` lub `docker container ls`
5. `docker ps -a` lub `docker container ls -a`
6. `docker run <options> <image>`
7. `docker stop <container_id or name>`
8. `docker rm <container_id or name>`



ML Wdrażanie modeli

Pierwszy kontener w Dockerze



docker



ML Wdrażanie modeli

Docker – przypisywanie portów

1. Mapowanie portów:

```
docker run -d -p 8080:80 nazwa_obrazu
```

2. Mapowanie wielu portów:

```
docker run -d -p 8080:80 -p 3000:3000 nazwa_obrazu
```

3. Domyślne porty wewnątrz kontenera:

```
docker run -d -p 8080:5000 nazwa_obrazu
```

`docker run -d -p 8080:80 nazwa_obrazu` ← Jaki port?

info **Share**
ACADEMY



ML Wdrażanie modeli

Korzystanie z publicznych obrazów z Docker Hub

1. Rejestracja na Docker Hub:



Create your account



2. Wyszukiwanie obrazu:

docker search nginx

3. Pobieranie obrazu:

docker pull nginx



ML Wdrażanie modeli

Docker Compose: co to jest i dlaczego warto używać?





ML Wdrażanie modeli

compose.yml

version: '3' # Wersja Docker Compose

services:

web:

image: "tiangolo/uwsgi-nginx-flask:python3.8"

container_name: my-flask-app

ports:

- "5000:80" # Mapowanie portu - Port 5000 na hoście przekierowany na port 80 wewnątrz kontenera

volumes:

- ./app:/app # Mapowanie woluminu - Montowanie lokalnego katalogu './app' do '/app' wewnątrz

kontenera

depends_on:

- redis # Zdefiniowanie zależności - Kontener "web" zależy od kontenera "redis"

redis:

image: "redis:alpine"

container_name: my-redis-server

ports:

- "6379:6379" # Mapowanie portu - Port 6379 na hoście przekierowany na port 6379 wewnątrz kontenera

info **Share**
ACADEMY



ML Wdrażanie modeli

compose.yml

docker-compose up

docker-compose down



Zadanie 17.2 (instrukcja)

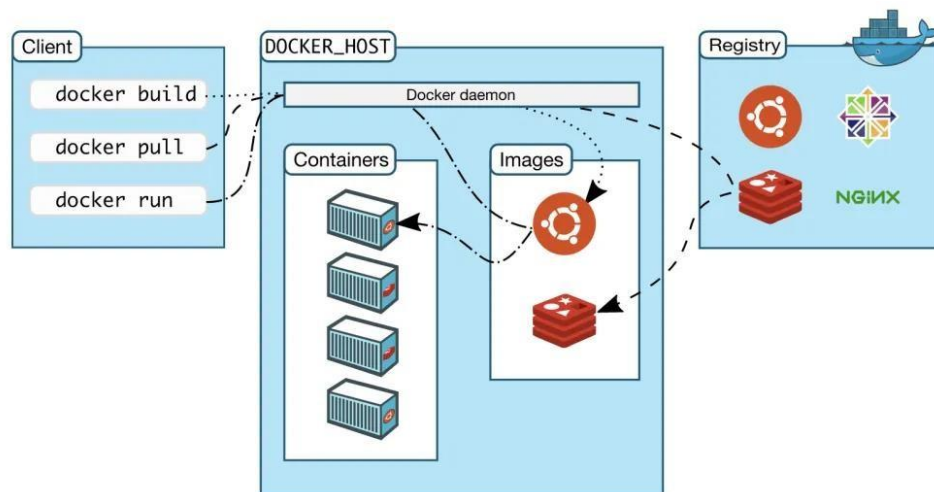
Zadaniem jest stworzenie pliku docker-compose.yml do konteneryzacji dwóch usług: aplikacji Pythona opartej na Flask oraz bazy danych Redis. Aplikacja Pythona powinna korzystać z bazy danych Redis do przechowywania prostych wiadomości.

Redis: oprogramowanie działające jako nierelacyjna baza danych przechowująca dane w strukturze klucz-wartość w pamięci operacyjnej serwera, przeznaczona do działania jako klasyczna baza danych, miejsce przechowywania pamięci podręcznej oraz broker wiadomości.



ML Wdrażanie modeli

Docker – podsumowanie





ML Wdrażanie modeli

Jak możemy użyć modelu?

Batch (czyli w partiach, co jakiś okres):

- przewidywanie przychodu na koniec miesiąca,
- symulacja zużycia prądu na następny okres,
- estymacja liczby użytkowników na koniec dnia.



ML Wdrażanie modeli

Jak możemy użyć modelu?

Batch:

- uruchamianie skryptu przy pomocy harmonogramu Zada (job scheduler).



ML Wdrażanie modeli

Technologie – przykłady



argo

info **Share**
ACADEMY



ML Wdrażanie modeli

Jak możemy użyć modelu?

Real-time (czyli na bieżąco, na żądany request):

- wyświetlenie rekomendacji po kliknięciu w produkt,
- ocena zdolności kredytowej klienta po wprowadzeniu danych do systemu,
- wykrywanie twarzy w aparacie przed zrobieniem zdjęcia.



ML Wdrażanie modeli

Jak możemy użyć modelu?

Real-time:

- "Hard code" w aplikacji lub wbudowane w aplikację dedykowane metody,
- odpytanie bazy danych o wcześniej wyliczone wyniki modelu,
- odpytanie dedykowanego API serwującego modele.



ML Wdrażanie modeli

Technologie – przykłady



TORCHSERVE
TORCHELASTIC





ML Wdrażanie modeli

Jak możemy użyć modelu?

Stream (czyli na strumieniu danych):

- podobnie jak przy batch, gdy potrzebujemy wyników bez opóźnień,
- rozpoznawanie obiektów przez kamerę.



ML Wdrażanie modeli

Technologie – przykłady



OpenVINO™

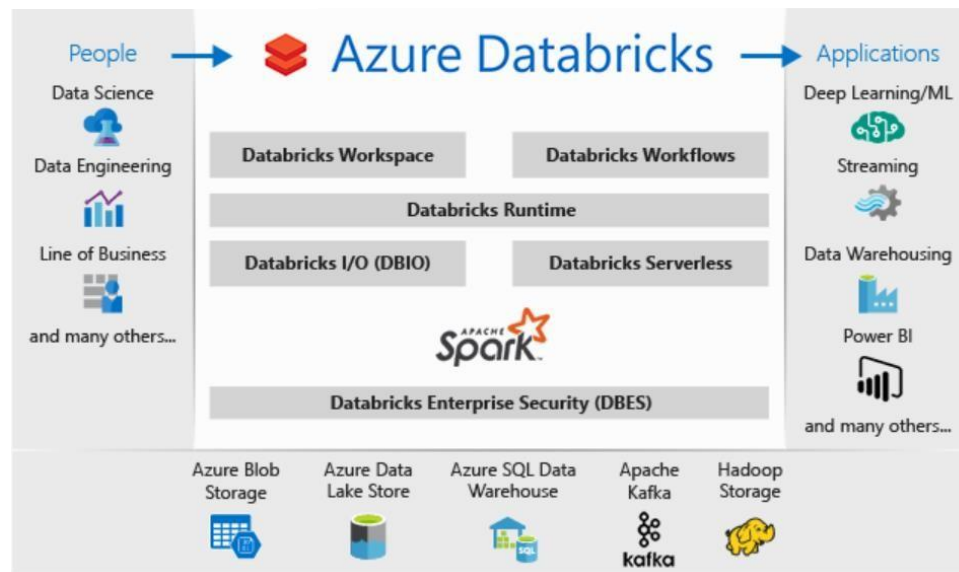
info **Share**
ACADEMY



ML Wdrażanie modeli

Databricks

Azure Databricks 





ML Wdrażanie modeli

Zakładanie konta i konfiguracja
środowiska pracy w Databricks

Azure Databricks

The data and AI service from Databricks available through Microsoft Azure to store all your data on a simple open lakehouse and unify all your analytics and AI workloads.

Get started

Schedule a demo





ML Wdrażanie modeli

Tworzenie notatników w Databricks



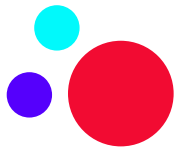
databricks



ML Wdrażanie modeli

mlflow w Databricks





ML Wdrażanie modeli

Budowanie i szkolenie modeli uczenia maszynowego

1. Instalacja mlflow:

```
pip install mlflow
```

2. Importowanie bibliotek i przygotowanie danych.

3. Definiowanie modelu.

4. Uruchamianie eksperymentów z mlflow.

5. Trenowanie modelu:

```
with mlflow.start_run():  
    model = ... # Inicjalizacja modelu  
    model.fit(X_train, y_train)  
    mlflow.sklearn.log_model(model, "model")
```

6. Ewaluacja modelu.

7. Zapisywanie modelu:

```
mlflow.sklearn.save_model(model, "model")
```



ML Wdrażanie modeli

Monitorowanie, śledzenie i zarządzanie
eksperymentami mlflow

`mlflow.log_param("param_name", param_value)`

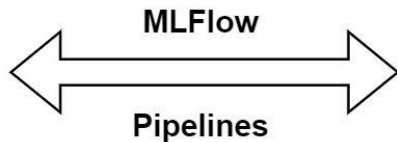


logowanie
parametrów

`mlflow.log_metric("metric_name", metric_value)`



logowanie
metrk





ML Wdrażanie modeli

Szkolenie modeli uczenia maszynowego
mlflow

mlflow™



Zadanie 17.3 (instrukcja)

Zadaniem jest wdrożenie modelu mlflow:

- skorzystaj z wybranego zbioru danych biblioteki np. sklearn,
- przygotuj dane,
- zaimplementuj wybrany model,
- użyj mlflow do zapisania modelu po treningu,
- zapisz model mlflow i zweryfikuj poprawność modelu na zbiorze testowym.



ML Wdrażanie Modeli

Podsumowanie

