

Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH



Instytut Informatyki

Studia Podyplomowe

Data Science - Programowanie w R i Python

Mikolaj Maslanka

Notatki do prezentacji na temat MLOps z wykorzystaniem AWS

Prowadzacy przedmiot:

Pawel Jamer

Warszawa, 2024

README

Dokument zawiera moje notatki, które przygotowałem pracując nad prezentacją. Jest to nieformalny dokument, który nie jest przeznaczony do publikacji. Gorąca prośba o zachowanie go wyłącznie do użytku własnego.

Oczywiście, istnieje możliwość jego edytowania i zmieniania według własnych potrzeb.

Instrukcja, jak uruchomić repozytorium, aby nie wyrządzić krzywdy swojemu PC, znajduje się w pliku `README.md` w głównym katalogu repozytorium.

Spis treści

README

Przydatne skróty

Wstęp do prezentacji

1 Wprowadzenie do MLOps	2
1.1 Zanim MLOps najpierw DevOps	2
1.2 Przykładowe technologie oraz kategorie w DevOps	3
1.3 Generalne komponenty MLOps	4
1.4 Przykład platformy typu Open Source - Kubeflow	5
1.5 Jak podejścić do budowania platformy MLOps?	6
2 Cykl Życia modeli ML	7
2.1 Podstawowe koncepcje	7
2.2 Cykl życia modeli ML na AWS	8
3 Przykładowa architektura MLOps - AWS	10
3.1 Wymagania które powinna spełnić platforma do ML	10
3.2 Przykładowa architektura nr1 z wykorzystaniem AWS	12
3.3 Przykładowa architektura nr2 z wykorzystaniem AWS	12
3.4 Zarządzanie danymi w uczeniu maszynowym	13
3.4.1 Katalogowanie danych	14
3.5 Feature store	15
4 Kluczowe narzędzia AWS dla MLOps	18
4.1 Usługi i serwisy AWS	18
5 AWS - CI/CD	20

5.1	Wartosciowe materiały	20
5.2	Przykładowe rozwiązanie 1	21
5.3	Przykładowe rozwiązanie 2	21
5.4	Przykładowe rozwiązanie 3	22
6	Najlepsze praktyki z MLOps na AWS	23
6.1	Rozpoczęcie projektu	23
6.2	Projektowanie i wdrożenie	24
6.3	Użytkowanie platformy i praktyki operacyjne	24
7	Wyzwania w MLOps na AWS	26
7.1	Rzut oka na wyzwania w MLOps	26
8	Podsumowanie oraz przyszłość dla MLOps	28
8.1	Ujednolicone platformy MLOps	28
8.2	Serverless GPUs	29
8.3	Innowacje i Przyszłość	29
	Załącznik 1 - Materiały dodatkowe	31

Wykaz rysunków

1.1	DevOps - flow pracy oraz projektu	3
1.2	DevOps - typowe narzędzia	3
1.3	MLOps - co to jest?	4
1.4	MLOps z wykorzystaniem Kubeflow	5
2.1	Cykl życia modeli ML	7
3.1	Kroki i przepływy logiczne czynności w ML	10
3.2	Architektura rozwiązania z wykorzystaniem platformy AWS	12
3.3	Architektura rozwiązania z wykorzystaniem platformy AWS	12
3.4	Podstawowe komponenty AWS Lake Formation	13
3.5	Feature store	15
5.1	Przykładowe rozwiązanie CI/CD - plus Terraform	21
5.2	Przykładowe rozwiązania CI/CD wraz z wykrywaniem zmiany	21
5.3	Przykładowe rozwiązania CI/CD AWS	22
8.1	End-to-end platformy do projektów typu MLOps	29

Przydatne skróty

AWS	Amazon Web Services
CI/CD	Continuous Integration/Continuous Delivery
ETL	Extract, Transform, Load
ML	Machine Learning
MLOps	Machine Learning Operations
S3	Simple Storage Service
EC2	Elastic Compute Cloud
IAM	Identity and Access Management
Lambda	Lambda (AWS)
Glue	Glue (AWS)
CDK	Cloud Development Kit
VPC	Virtual Private Cloud
IAM	Identity Access Management
SDK	Software Development Kit
OLTP	OnLine Transaction Processing
OLAP	OnLine Analytical Processing
DWH	Data WareHouse
CT	Continuous Training
IaC	Infrastructure as Code

Wstęp do prezentacji

Dlaczego praktyki MLOps są wymagające?

Na samym poczatku warto powiedzieć dlaczego praktyki MLOps są wymagające?

ODP: Ponieważ trzeba umieć sporo innych rozwiązań/technologii, które są bezpośrednio związane z wytwarzaniem oprogramowania.

Aby zrozumieć, dlaczego praktyki/podejście MLOps są tak wymagające, należy przyjrzeć się szeregowi technologii i narzędzi, które są fundamentem dla efektywnego zarządzania cyklem życia modeli ML.

Kluczowe technologie i narzędzia w MLOps

1. Zarządzanie Kontenerami i Wirtualizacja

- **Kubernetes:** Złożona platforma do orkiestracji kontenerów, umożliwiająca skalowanie i automatyzację wdrażania aplikacji.
- **Docker:** Narzędzie do tworzenia i zarządzania kontenerami, kluczowe w procesie standaryzacji środowisk uruchomieniowych.

2. Wersjonowanie i Integracja

- **Git:** System kontroli wersji, niezbędny do zarządzania kodem i śledzenia zmian.
- **CI/CD:** Narzędzia takie jak Jenkins lub GitHub Actions, umożliwiające automatyzację procesów testowania i wdrażania.

3. Zarządzanie Eksperymentami i Modelami

- **MLFlow**: Platforma do zarządzania eksperymentami ML, ułatwiająca śledzenie pracy nad modelami.
- **Seldon Core**: Narzędzie do zarządzania i skalowania modeli ML w produkcji.

4. Infrastruktura i Konfiguracja

- **Terraform**: Narzędzie do zarządzania infrastrukturą jako kodem (IaC).
- **Ansible**: Narzędzie do automatyzacji konfiguracji i zarządzania infrastrukturą.

5. Monitorowanie i Logowanie

- **Prometheus**: System monitorowania i alertyzacji.
- **Elastic Stack**: Zestaw narzędzi do logowania, monitorowania i analizy danych.

6. Testowanie

- **pytest**: Framework do testowania aplikacji napisanych w Pythonie.
-

Specyfika pracy z modelami uczenia maszynowego

W MLOps, oprócz typowych wyzwań związanych z praktykami DevOps, napotykamy na dodatkowe, unikalne wyzwania związane bezpośrednio z modelami uczenia maszynowego:

- **Data Versioning**: Zarządzanie wersjami danych, które są kluczowe dla nauki modeli.
 - **Feature Store**: Przechowywanie i wersjonowanie cech wykorzystywanych w modelach.
 - **Model Serving**: Zapewnienie możliwości replikacji środowiska, w którym model jest uruchamiany.
-

Role w zespołach MLOps

W zależności od wielkości i struktury organizacji, zespoły MLOps mogą być różnie skonfigurowane, często dzieląc się na podgrupy specjalizujące się w różnych aspektach systemu MLOps.

Umiejetności niezbędne dla MLOps Engineera

- **Python:** Python jest królem w świecie uczenia maszynowego, więc znajomość tego języka jest niezbędna.
 - **Docker i Kubernetes:** Znajomość zarządzania kontenerami i orkiestracji.
 - **Git:** Znajomość systemu kontroli wersji.
 - **Linux:** Podstawy obsługi i mechanizmy systemu.
 - **Podstawy uczenia maszynowego:** Zrozumienie kluczowych pojęć i technik.
 - **Podstawy DevOps:** Zrozumienie całego cyklu wytwarzania oprogramowania, w tym CI/CD, testowania, wdrażania, monitorowania i zarządzania.
-

Znaczenie platformy AWS w MLOps

Platforma AWS oferuje szeroki wachlarz usług i narzędzi, które wspierają praktyki MLOps, od zarządzania infrastrukturą po wdrażanie i skalowanie modeli ML. W kolejnych sekcjach szczegółowo omówimy, jak AWS umożliwia efektywne wdrażanie praktyk MLOps, podkreślając kluczowe usługi i najlepsze praktyki związane z tą platformą.

Slajd 1

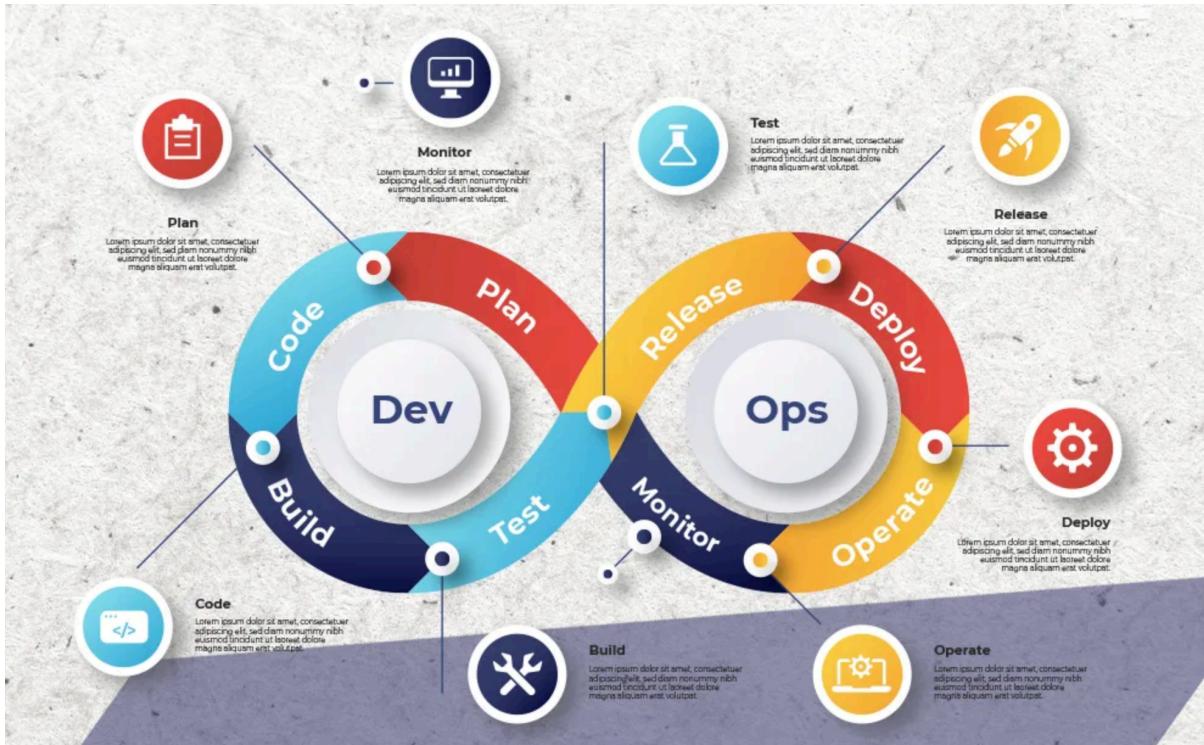
Wprowadzenie do MLOps

1.1 Zanim MLOps najpierw DevOps

DevOps to koncepcja, która zrewolucjonizowała świat IT, łącząc praktyki, narzędzia i filozofię pracy zorientowane na zwiększenie szybkości wytwarzania oprogramowania, poprawę jego jakości i bezpieczeństwa oraz wydajność zespołów.

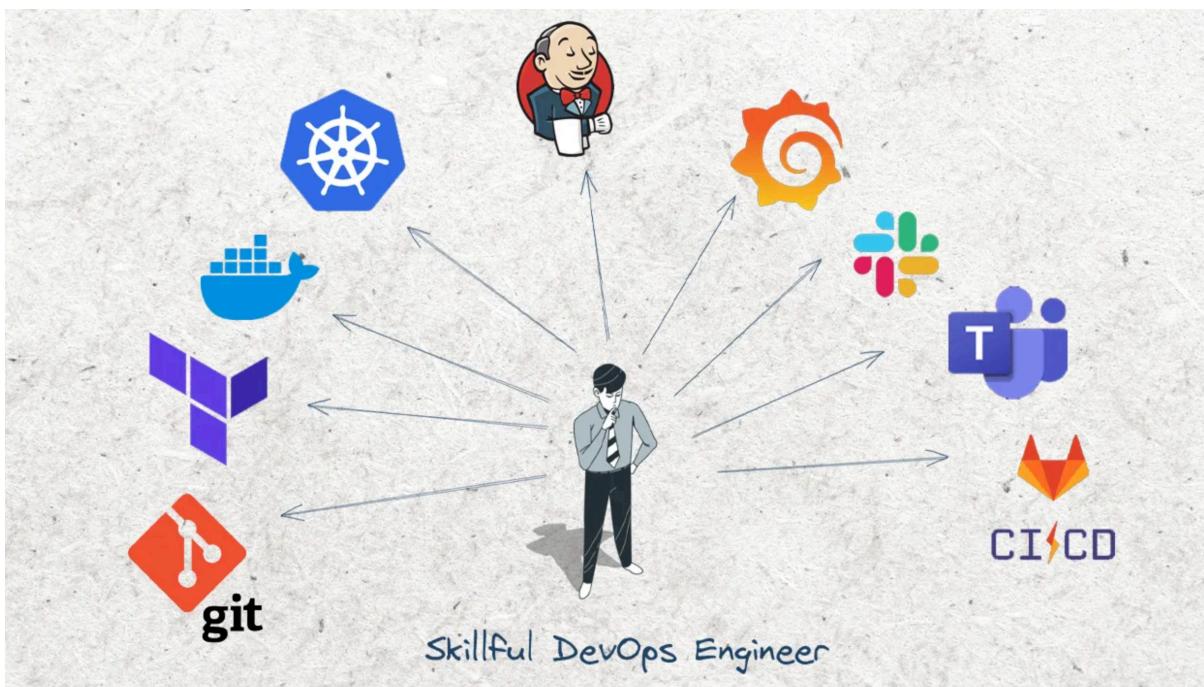
Cykl życia oprogramowania w DevOps jest zorientowany na ciągłą integrację, ciągłe dostarczanie i ciągłe ulepszanie produktu. Cykl ten składa się z etapów: Plan, Code, Build, Test, Release, Deploy, Operate i Monitor, tworząc nieprzerwany łańcuch.

Poniżej przedstawiony jest cykl życia oprogramowania w podejściu DevOps.



Rysunek 1.1: DevOps - flow pracy oraz projektu

1.2 Przykładowe technologie oraz kategorie w DevOps



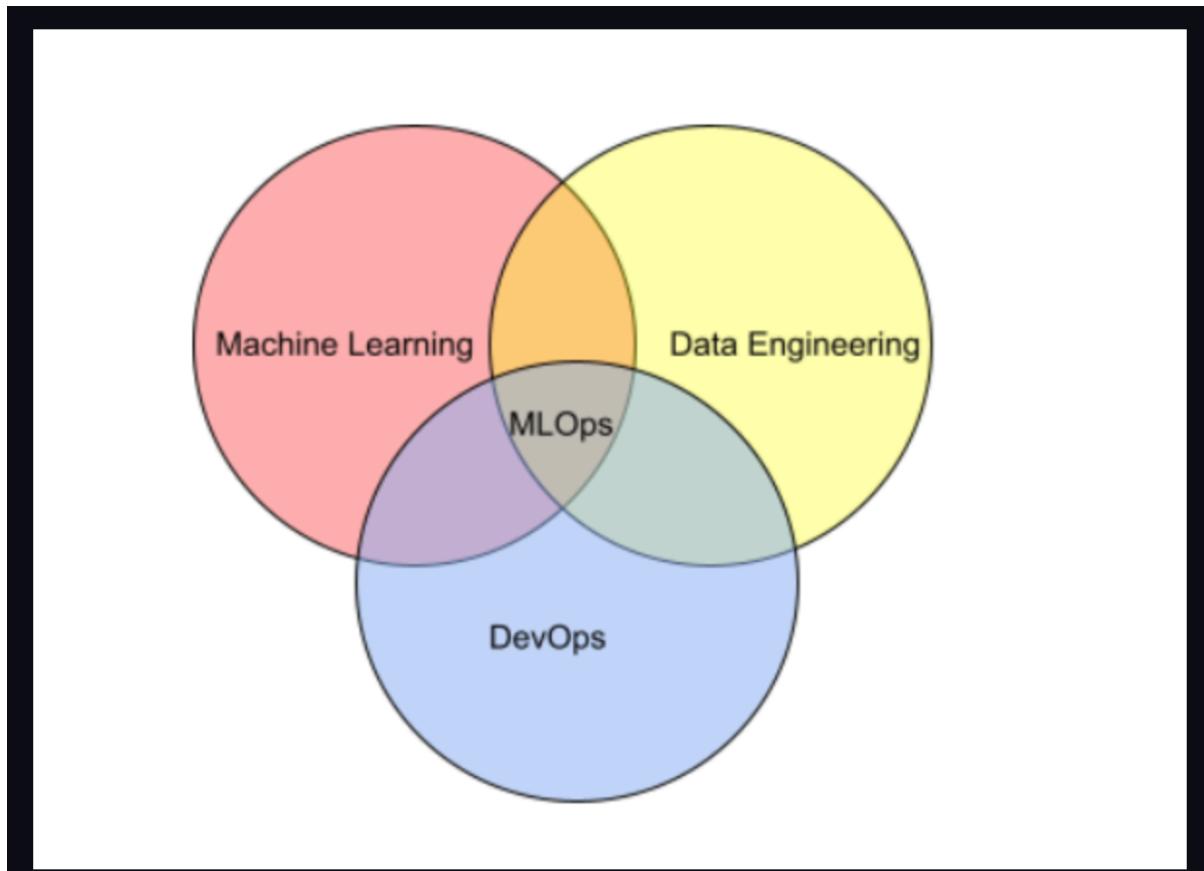
Rysunek 1.2: DevOps - typowe narzędzia

- Konteneryzacja i Orkiestracja:** Docker i Kubernetes to przykłady narzędzi, które zmieniły podejście do wdrażania i skalowania aplikacji.
- Systemy Kontroli Wersji:** Git jest nieodzownym elementem DevOps, umożliwiającym efektywne zarządzanie wersjami kodu.

wiąjącym efektywną współpracę i śledzenie zmian w kodzie.

3. **Automatyzacja i CI/CD:** Jenkins, Travis CI, GitHub Actions oraz inne narzędzia CI/CD automatyzują proces testowania i wdrażania, skracając czas dostarczania nowych funkcji i poprawek.

1.3 Generalne komponenty MLOps



Rysunek 1.3: MLOps - co to jest?

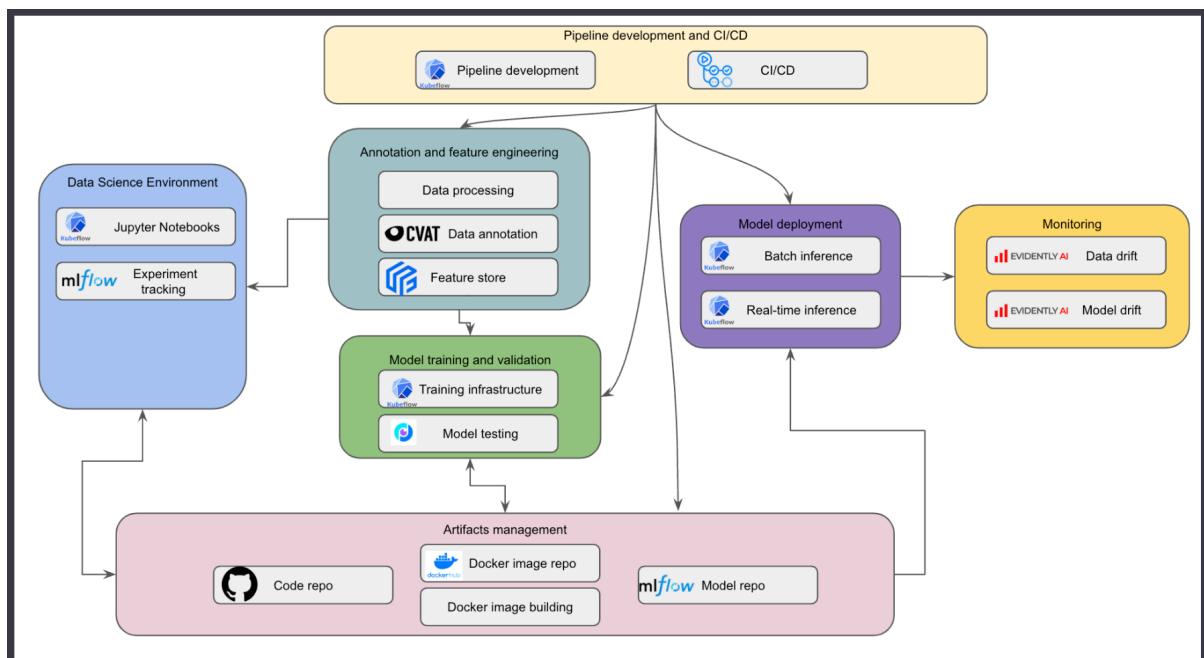
Platformy MLOps typu end-to-end łączą w sobie szereg niezbędnych funkcji i narzędzi, które powinny obejmować:

1. **Zarządzanie danymi:** Efektywne gromadzenie, wersjonowanie i przetwarzanie danych jest fundamentem dla modeli ML.
2. **Eksperymenty:** Narzędzia takie jak MLflow umożliwiają projektowanie, testowanie i optymalizację modeli.
3. **Wdrażanie modeli:** Konteneryzacja i zarządzanie API są kluczowe dla dostarczania modeli do produkcji.

4. **Monitorowanie:** Śledzenie wydajności modeli w czasie rzeczywistym jest niezbędne dla zapewnienia ich niezawodności.
5. **Współpraca i kontrola wersji:** Umożliwia dzielenie się pracą i zarządzanie zmianami w modelach i kodzie.
6. **Automatyzacja pracy:** Narzędzia orkiestracji, takie jak Kubeflow, upraszczają złożone przepływy pracy.
7. **Integracja z narzędziami ML:** Elastyczność w doborze narzędzi i bibliotek jest kluczowa dla wydajności i innowacyjności.

1.4 Przykład platformy typu Open Source - Kubeflow

Kubeflow jest przykładem platformy open source, która integruje się z Kuberntesem, dostarczając kompleksowe środowisko do zarządzania przepływami pracy związanymi z uczeniem maszynowym, od eksperymentacji po wdrożenie modeli i ich monitorowanie. Zapewnia integrację z narzędziami takimi jak Jupyter, MLflow oraz różnymi systemami zarządzania danymi i modelami, tworząc spójne środowisko dla data scientistów i inżynierów ML.



Rysunek 1.4: MLOps z wykorzystaniem Kubeflow

1.5 Jak podejsc do budowania platformy MLOps?

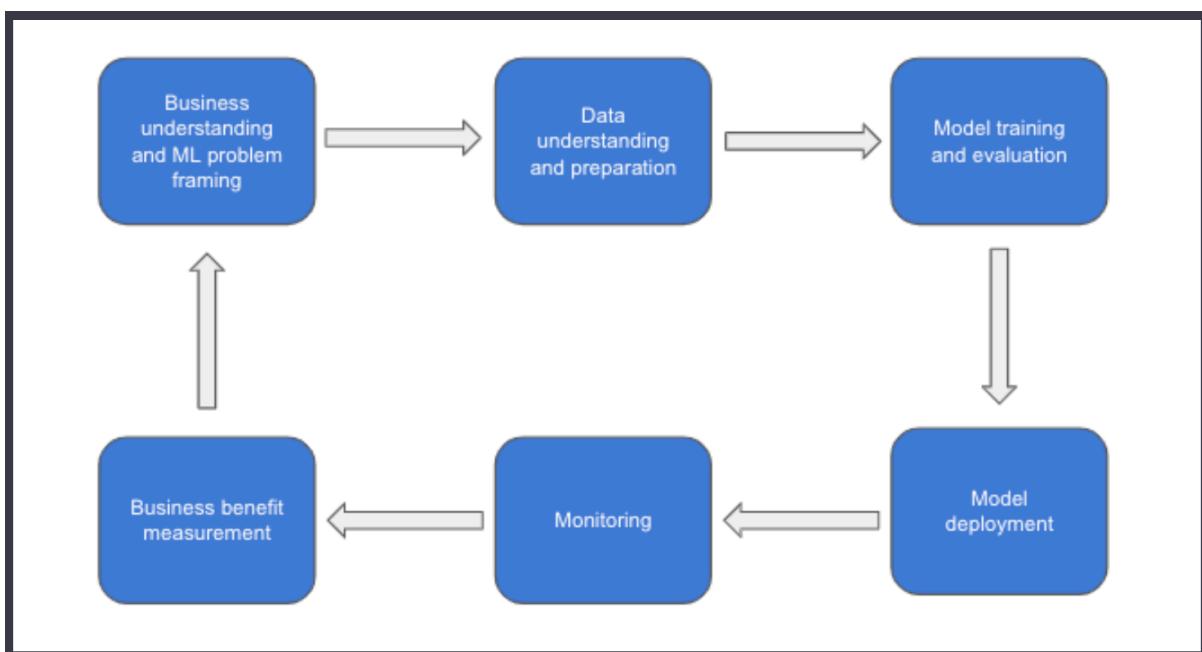
Bardzo fajny artykul:

- <https://neptune.ai/blog/ml-platform-guide>

Slajd 2

Cykl Życia modeli ML

2.1 Podstawowe koncepcje



Rysunek 2.1: Cykl życia modeli ML

Cykl życia modeli uczenia maszynowego (ML) to seria etapów, które należy przeprowadzić, aby skutecznie zaimplementować i zarządzać modelami ML. Każdy etap ma kluczowe znaczenie dla projektu i wymaga ścisłej współpracy między ekspertami, inżynierami ML i biznesem.

1. Zrozumienie biznesu i formułowanie problemu:

- Pierwszym krokiem jest zrozumienie celów biznesowych i określenie jak tech-

nologia ML może przyczynić się do ich realizacji. Na tym etapie ustala się metryki sukcesu, które mogą obejmować redukcję kosztów, łagodzenie ryzyka czy zwiększenie przychodów.

2. Zrozumienie i przygotowanie danych:

- Analiza i przygotowanie danych to fundament modelu ML. Obejmuje to gromadzenie danych, ich czyszczenie, eksplorację oraz inżynierię cech w celu zbudowania odpowiedniego zestawu danych do treningu modelu.

3. Trening i ocena modelu:

- Następnie przeprowadza się eksperymenty z różnymi algorytmami, dobiera hiperparametry i ocenia modele za pomocą zdefiniowanych metryk.

4. Wdrażanie modelu:

- Po treningu model jest gotowy do wdrożenia. Oznacza to umieszczenie go w środowisku produkcyjnym.

5. Monitorowanie modelu:

- Po wdrożeniu modelu konieczne jest ciągłe monitorowanie jego wydajności, aby wykryć i zaradzić potencjalnym problemom, takim jak dryf danych, który może wpływać na dokładność modelu.

6. Śledzenie metryk biznesowych:

- Ważne jest, aby stale mierzyć wpływ modelu na kluczowe wskaźniki biznesowe i, jeśli to konieczne, dokonywać jego rekalibracji lub optymalizacji, aby zapewnić ciągłe dostarczanie wartości dla firmy.

2.2 Cykl życia modeli ML na AWS

Amazon Web Services zapewnia kompleksowy zestaw usług, które wspierają cały cykl życia modeli uczenia maszynowego.

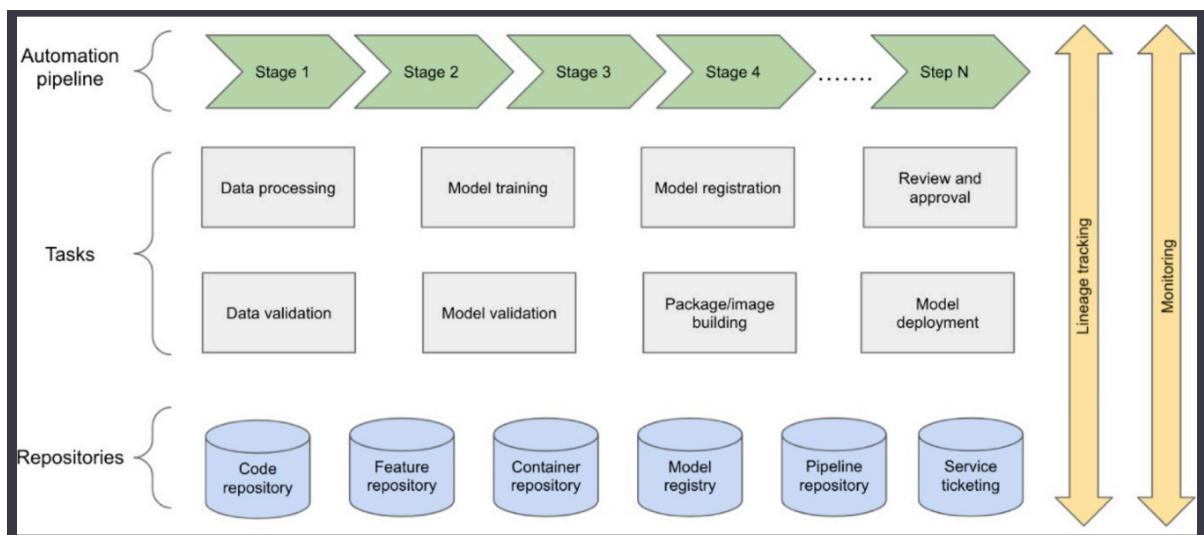
- **Amazon S3:** Bezpieczne i skalowalne przechowywanie danych.

- **AWS Glue:** ETL (Extract, Transform, Load) do przygotowania i transformacji danych.
- **AWS Lake Formation:** Budowa bezpiecznego data lake.
- **Amazon SageMaker:** Kompletna usługa, która umożliwia szybkie budowanie, trenowanie i wdrażanie modeli ML.
- **Amazon Elastic Container Service:** Orkiestracja kontenerów w dużych skalach.
- **AWS Elastic Beanstalk:** Proste wdrażanie i skalowanie aplikacji i usług.
- **Amazon CloudWatch:** Monitorowanie zasobów i aplikacji z alarmami i powiadomieniami.
- **Amazon QuickSight:** Business Intelligence dla zaawansowanej wizualizacji i zrozumienia metryk biznesowych.
- **AWS Data Pipeline:** Orkiestracja przetwarzania danych między różnymi usługami AWS.

Slajd 3

Przykładowa architektura MLOps - AWS

3.1 Wymagania które powinna spełnić platforma do ML



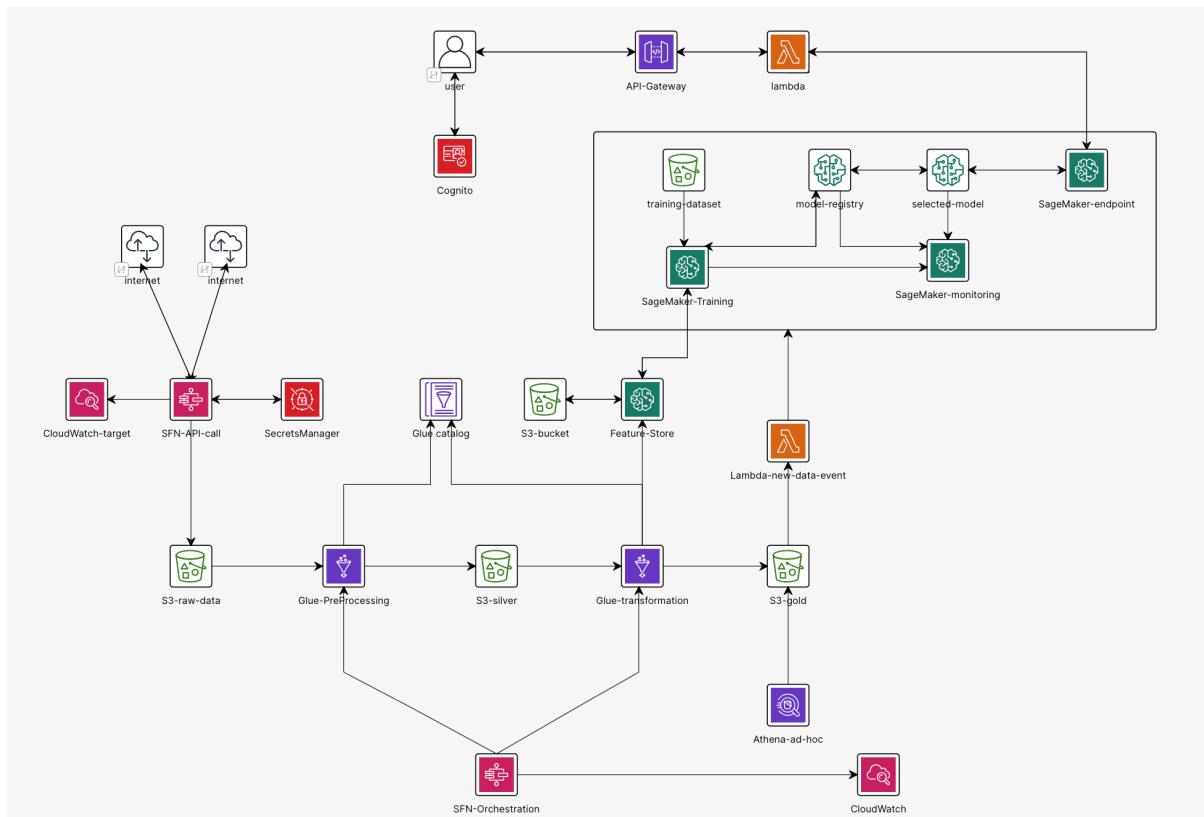
Rysunek 3.1: Kroki i przepływ logicznych czynności w ML

- Wsparcie dla Cyklu Życia ML od Początku do Końca:** Musi obsługiwać zarówno eksperymentalną, jak i operacyjną fazę życia modelu, włączając w to trening modeli na dużą skalę i ich wdrażanie.
- CI/CT/CD Pipeline:** Rozszerzenie tradycyjnej integracji ciągłej o elementy danych i modeli, umożliwiające ciągły trening i wdrażanie modeli ML.
- Nadzór Operacyjny:** Funkcje monitorowania przepływów pracy, wydajności mo-

deli, dryftu (zmiany) danych oraz metryk infrastruktury, w połączeniu z automatycznym systemem alertów i mechanizmami odzyskiwania (recovery) po awariach.

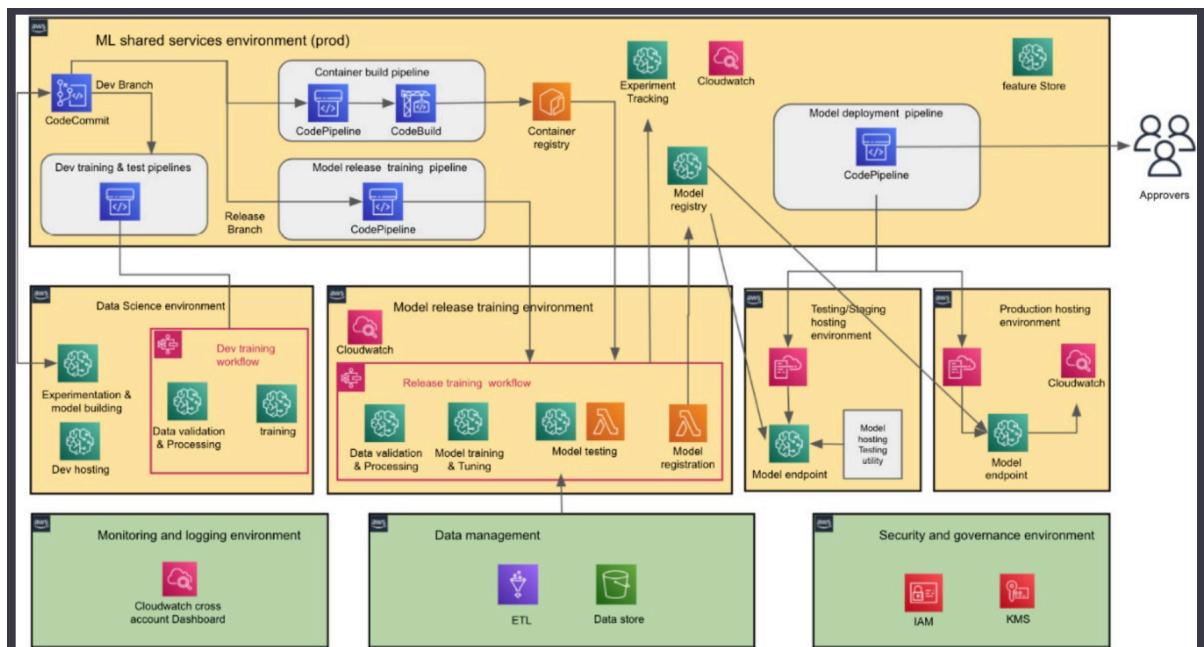
4. **Elastyczność Języka i Frameworków:** Wsparcie dla popularnych języków programowania i frameworków ML, aby sprostać różnorodnym preferencjom rozwojowym i wymogom projektów.
5. **Zarządzanie Zasobami Obliczeniowymi:** Efektywne zarządzanie różnorodnymi zasobami obliczeniowymi, w tym specjalistycznym sprzętem ML, w celu zrównoważenia wydajności i kosztów.
6. **Integracje z Systemami Zewnętrznymi:** Solidne połączenia z zewnętrznym oprogramowaniem i platformami, aby umożliwić płynną orkiestrację pracy i współpracę.
7. **Środki Bezpieczeństwa:** Silne mechanizmy uwierzytelniania, kontroli dostępu i szyfrowania danych w celu zapewnienia bezpiecznego dostępu do danych i zgodności ze standardami branżowymi.
8. **Zarządzanie Artefaktami:** Śledzenie i wersjonowanie artefaktów ML, takich jak kod i modele, w celu utrzymania możliwości odtworzenia i zgodności.
9. **Standaryzacja Pakietów:** Zarządzanie centralną biblioteką pakietów ML w celu utrzymania standardów organizacyjnych i polityk bezpieczeństwa.
10. **Dostęp do Różnych Magazynów Danych:** Łatwy dostęp do różnych magazynów danych w celu usprawnienia tworzenia i treningu modeli.
11. **Możliwości Samoobsługi:** Opcje samoobsługi dla procesów takich jak wprowadzanie nowych użytkowników i konfiguracja środowiska, aby zwiększyć autonomię i wydajność.
12. **Narzędzia Oceny Modeli:** Kompletny zestaw narzędzi do testowania i walidacji modeli w celu zapewnienia ich niezawodności i dokładności.

3.2 Przykładowa architektura nr1 z wykorzystaniem AWS



Rysunek 3.2: Architektura rozwiązania z wykorzystaniem platformy AWS

3.3 Przykładowa architektura nr2 z wykorzystaniem AWS

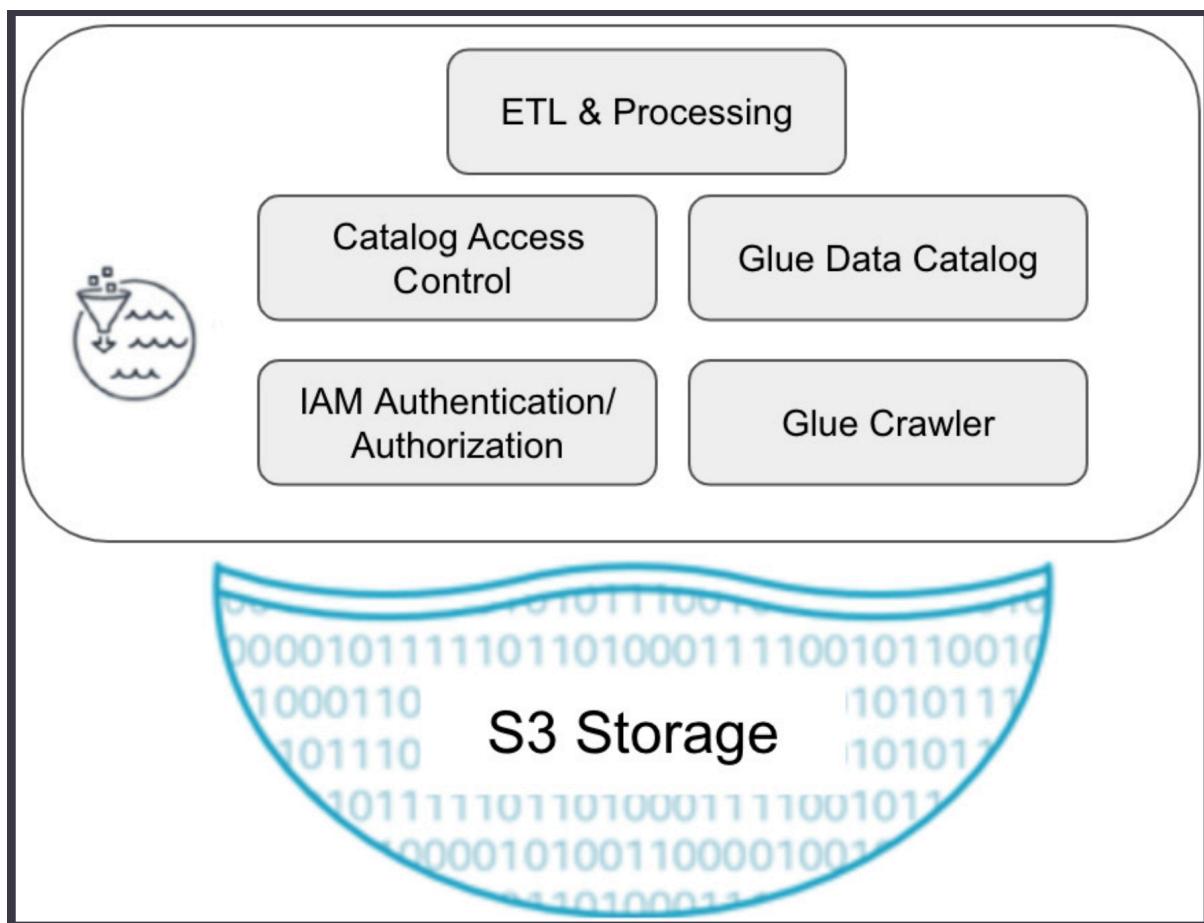


Rysunek 3.3: Architektura rozwiązania z wykorzystaniem platformy AWS

3.4 Zarządzanie danymi w uczeniu maszynowym

Na wysokim poziomie, zarządzanie danymi przecina się z cyklem życia ML na trzech etapach: zrozumienie i przygotowanie danych, szkolenie modeli i ich ocena oraz wdrażanie modelu.

Przykładowo - AWS Lake Formation to kompleksowa usługa zarządzania danymi oferowana przez AWS, która usprawnia proces budowania i utrzymania data lake na platformie AWS. Poniższy rysunek ilustruje podstawowe komponenty AWS Lake Formation:



Rysunek 3.4: Podstawowe komponenty AWS Lake Formation

Ogólnie rzecz biorąc, AWS Lake Formation oferuje podstawowe funkcje, które poprawiają zarządzanie danymi:

Tworzenie i Utrzymanie Katalogu Danych: AWS Lake Formation ułatwia tworzenie i bieżące zarządzanie katalogiem danych, dostarczając scentralizowane repozytorium dla metadanych, co umożliwia łatwe odkrywanie i eksplorację danych w jeziorze danych.

3.4.1 KATALOGOWANIE DANYCH

Katalog danych odgrywa kluczową rolę w zarządzaniu danymi i umożliwia analitykom zajmującym się danymi odkrywanie i dostęp do danych przechowywanych w scentralizowanym magazynie. Staje się szczególnie ważny podczas fazy zrozumienia i eksploracji danych w cyklu życia ML, gdy trzeba wyszukiwać i rozumieć dostępne dane dla swoich projektów. Podczas oceny technologii katalogu danych należy wziąć pod uwagę następujące kluczowe czynniki:

- Katalog metadanych: Technologia powinna wspierać scentralizowany katalog danych dla efektywnego zarządzania metadanymi. Obejmuje to obsługę metadanych, takich jak nazwy baz danych, schematy tabel i tagi tabel.
- Automatyczne katalogowanie danych: Możliwość automatycznego odkrywania i katalogowania zestawów danych oraz wnioskowania schematów danych z różnych źródeł danych, takich jak Amazon S3, bazy danych relacyjnych, bazy danych NoSQL i logi. Zwykle funkcjonalność ta jest implementowana za pomocą crawlera, który skanuje źródła danych, identyfikuje elementy metadanych (np. nazwy kolumn, typy danych) i dodaje je do katalogu.
- Elastyczność tagowania: Możliwość przypisywania niestandardowych atrybutów lub tagów do jednostek metadanych, takich jak bazy danych, tabele i pola. Ta elastyczność wspiera zaawansowane możliwości wyszukiwania i odkrywania danych w katalogu.
- Integracja z innymi narzędziami: Bezproblemowa integracja katalogu danych z szeroką gamą narzędzi do przetwarzania danych, umożliwiająca łatwy dostęp do danych. Ponadto korzystna jest natywna integracja z platformami zarządzania data lake.

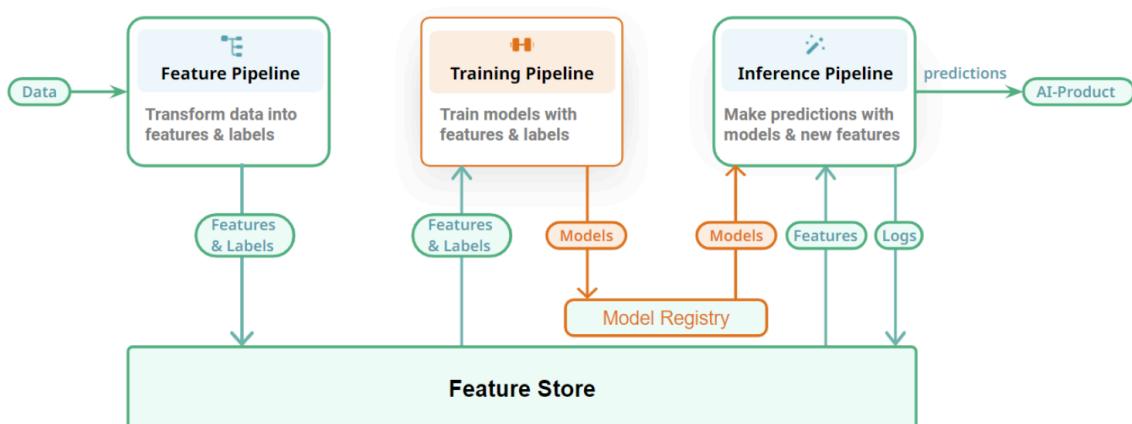
Narzędzia do zarządzania wersjami danych

Zamiast opracowywać niestandardowe rozwiązania do kontroli wersji danych, dostępne są narzędzia specjalnie zaprojektowane do efektywnego zarządzania wersjami danych. Oto kilka godnych uwagi opcji:

- Git LFS (Large File Storage): Git LFS rozszerza możliwości Gita o obsługę dużych plików, w tym zbiorów danych. Przechowuje te pliki poza repozytorium Gita, zachowując informacje o wersjonowaniu. Git LFS bezproblemowo integruje się z Gitem i jest powszechnie używany do wersjonowania dużych plików w projektach skoncentrowanych na danych.
- DataVersionControl (DVC): DVC to narzędzie open-source zaprojektowane specjalnie do wersjonowania i zarządzania danymi. Integruje się z Gitem i oferuje funkcje do śledzenia i zarządzania dużymi zbiorami danych. DVC umożliwia lekkie linki do rzeczywistych plików danych przechowywanych w zdalnym magazynie, takim jak Amazon S3 lub współdzielony system plików. To podejście pozwala na utrzymanie historii zmian i łatwe przełączanie między różnymi wersjami zestawów danych, eliminując potrzebę duplikowania danych.
- Pachyderm: Pachyderm to open-source'owe narzędzie do wersjonowania danych i śledzenia ich pochodzenia. Oferuje kontrolę wersji dla danych, umożliwiając śledzenie zmian w danych, kodzie i plikach konfiguracyjnych. Pachyderm wspiera rozproszone ramy przetwarzania danych, takie jak Apache Spark, i oferuje funkcje takie jak powtarzalność, śledzenie pochodzenia danych i oparte na pochodzeniu danych rozgałęzianie.

Link do repo: <https://github.com/pachyderm/pachyderm>

3.5 Feature store



Rysunek 3.5: Feature store

Przykład open source:

- <https://github.com/feast-dev/feast>

Linki do dodatkowych informacji:

- <https://www.hopsworks.ai/dictionary/feature-store>
- <https://github.com/visenger/awesome-mlops?tab=readme-ov-file#mlops-feature-stores>

Feature Store jest centralnym repozytorium służącym do przechowywania, dzielenia się i zarządzania cechami danych wykorzystywanymi do tworzenia modeli uczenia maszynowego. Jego głównym zadaniem jest zwiększenie wydajności pracy poprzez zapewnienie im szybkiego dostępu do już przetworzonych i gotowych do użycia cech danych, co usprawnia tworzenie i wdrożenie modeli ML.

Kluczowe korzyści wynikające z wdrożenia Feature Store obejmują:

- **Standaryzacja cech danych:** Ujednolicona definicja cech zapewnia spójność w różnych modelach i zespołach, eliminując duplikację pracy i potencjalne błędy.
- **Ułatwienie współdzielenia:** Feature Store umożliwia naukowcom z różnych zespołów na dzielenie się i ponowne wykorzystanie cech, co przyspiesza rozwój nowych modeli.
- **Śledzenie i wersjonowanie cech:** Wszystkie cechy są śledzone i wersjonowane, co zapewnia pełną przejrzystość procesu i umożliwia łatwe odtworzenie poprzednich eksperymentów.
- **Optymalizacja obliczeń:** Przechowywanie wcześniej przetworzonych cech zmniejsza redundantne obliczenia, oszczędzając czas i zasoby obliczeniowe.
- **Integracja z przepływami pracy ML:** Feature Store łatwo integruje się z istniejącymi narzędziami i platformami ML, ułatwiając automatyzację treningu i wdrażania modeli.

AWS oferuje te funkcjonalności poprzez różne usługi, takie jak AWS Glue do katalogowania danych i Amazon S3 do przechowywania, umożliwiając inżynierom ML efektywne zarządzanie cechami i przyspieszając cały cykl życia uczenia maszynowego.

Slajd 4

Kluczowe narzędzia AWS dla MLOps

4.1 Uslugi i serwisy AWS

Krotki opis tego co juz zoatalo pokazane i omowionme wczesniej.

1. **Amazon SageMaker:** Usługa umożliwia kompleksowe zarządzanie cyklem życia ML, w tym budowanie, trening, strojenie, wdrażanie i monitorowanie modeli. Oferuje narzędzia do automatyzacji i uproszczenia tych procesów.
2. **AWS Glue:** Zapewnia potężne narzędzia ETL, które są kluczowe do przetwarzania i przygotowania danych. Umożliwia integrację różnorodnych źródeł danych i automatyzację przepływów danych.
3. **AWS Step Functions:** Pozwala na orkiestrację złożonych sekwencji zadań, integrując różne usługi AWS w spójny przepływ pracy.
4. **AWS Lambda:** Umożliwia uruchamianie kodu bez zarządzania serwerami, idealne do krótkotrwałych, event-driven procesów w ramach MLOps.
5. **AWS CodePipeline i AWS CodeBuild:** Ułatwiają implementację ciągłej integracji i dostarczania, automatyzując procesy budowania, testowania i wdrażania modeli.

6. **Amazon CloudWatch:** Oferuje monitorowanie aplikacji i infrastruktury, umożliwiając śledzenie wydajności, logów i metryk.
7. **Amazon EMR:** Zapewnia elastyczne środowisko do przetwarzania dużych zbiorów danych, idealne do zadań takich jak przetwarzanie Big Data, analiza danych, czy uczenie maszynowe.
8. **Amazon Redshift:** Jest to usługa magazynu danych w chmurze, która pozwala na łatwe i szybkie analizowanie wszystkich danych przy użyciu standardowego języka SQL.
9. **AWS Data Pipeline:** Usługa do orkiestracji przetwarzania danych, która pozwala na przenoszenie i przekształcanie danych między różnymi usługami AWS.
10. **Amazon Athena:** Interaktywne zapytania SQL do analizy danych w Amazon S3, bez potrzeby zarządzania infrastrukturą.
11. **Amazon QuickSight:** Usługa Business Intelligence umożliwiająca szybkie tworzenie i publikowanie interaktywnych dashboardów.
12. **Amazon SageMaker Model Monitor:** Usługa do ciągłego monitorowania i utrzymania jakości modeli ML w produkcji.
13. **Amazon SageMaker Debugger:** Usługa do debugowania modeli ML, która automatycznie wykrywa i usuwa problemy związane z wydajnością i jakością modelu.

Slajd 5

AWS - CI/CD

5.1 Wartosciowe merytury

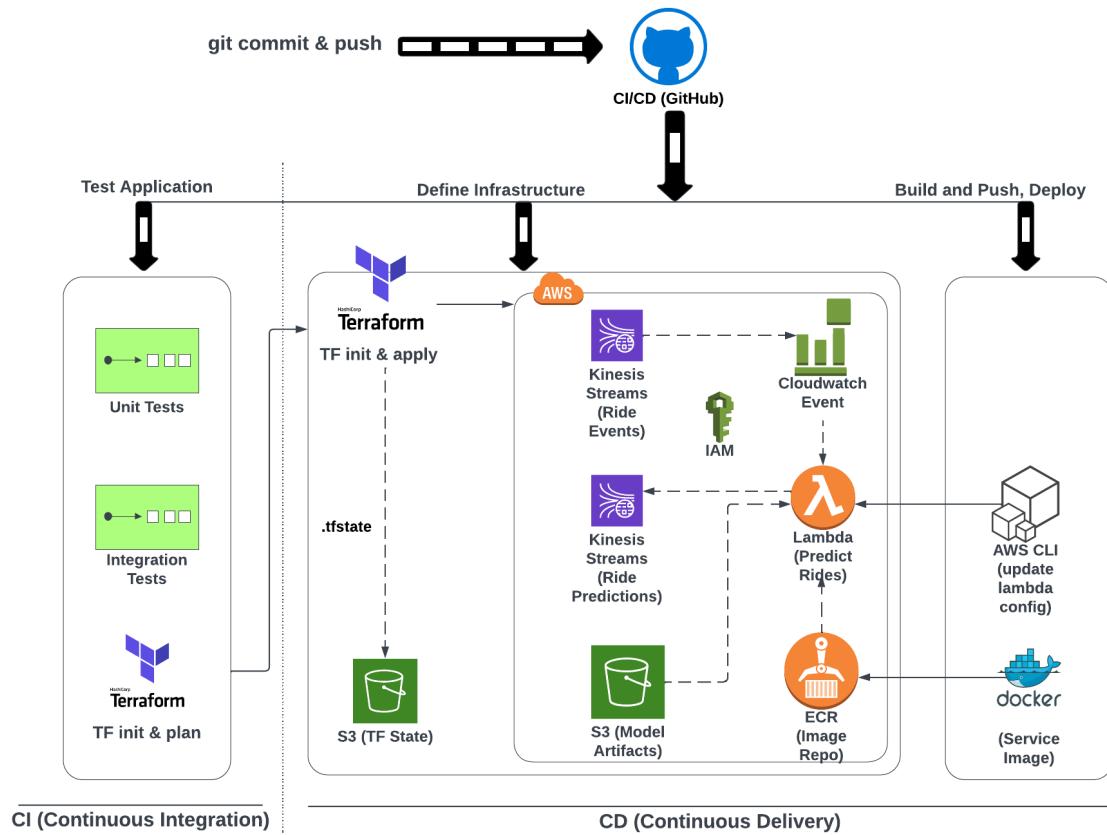
Link do dobrego artykułu na temat CI/CD w MLOps

- <https://neptune.ai/blog/ways-ml-teams-use-ci-cd-in-production>

Link do artykułu na temat CI/CD w AWS

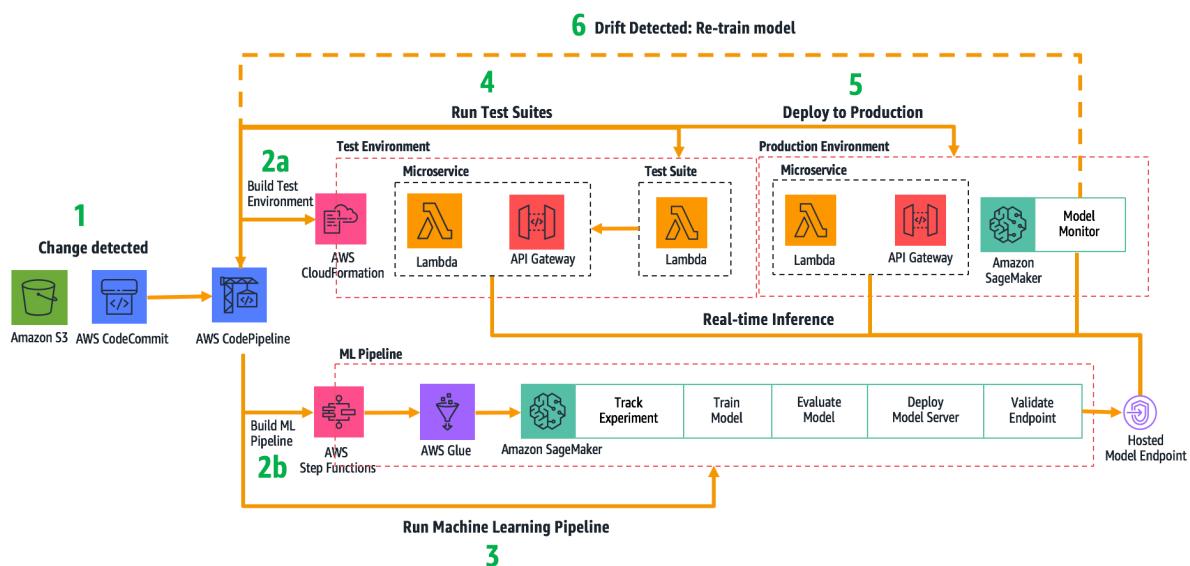
- <https://aws.amazon.com/blogs/machine-learning/build-a-ci-cd-pipeline-for-deploying-custom-machine-learning-models-using-aws-services/>

5.2 Przykładowe rozwiązanie 1



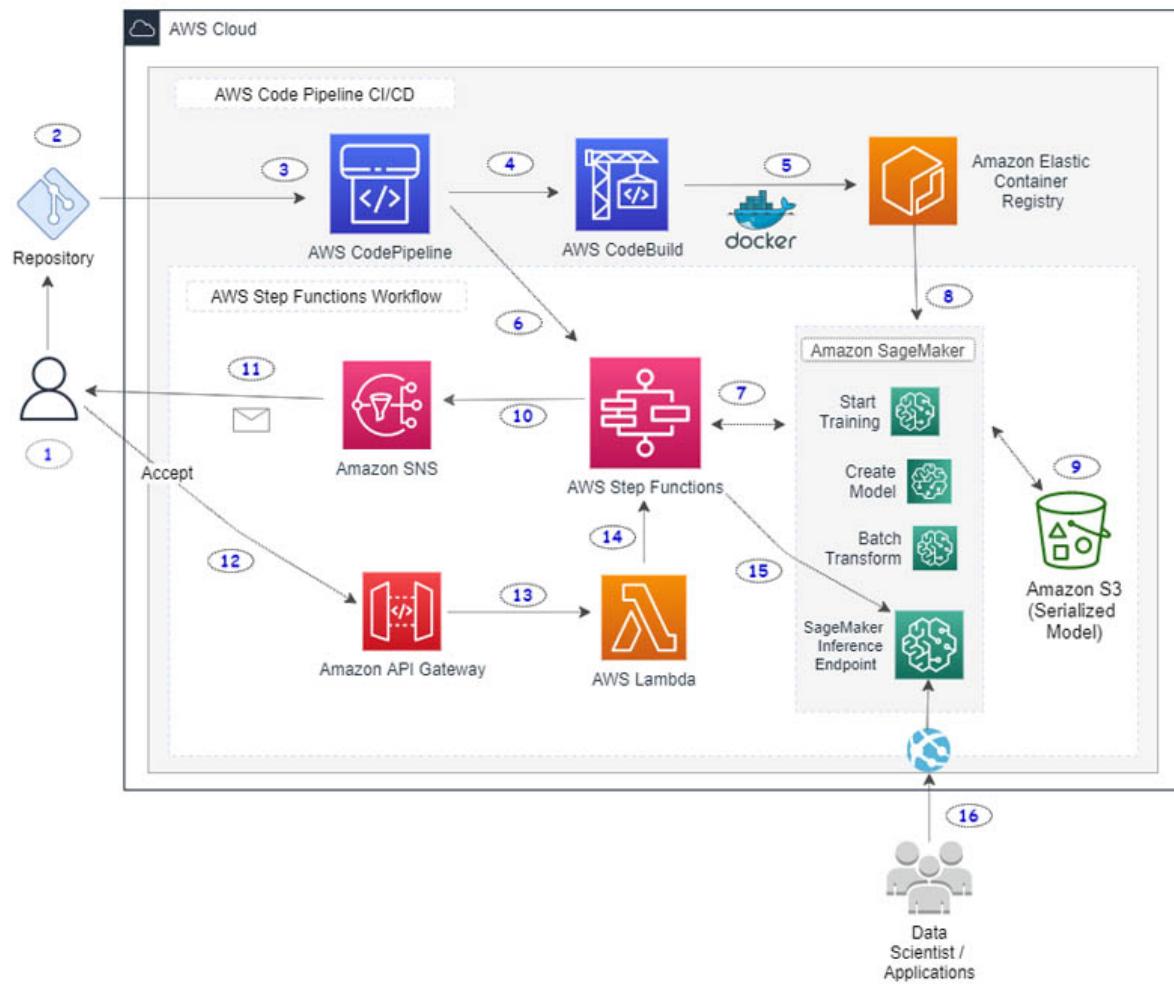
Rysunek 5.1: Przykładowe rozwiązanie CI/CD - plus Terraform

5.3 Przykładowe rozwiązanie 2



Rysunek 5.2: Przykładowe rozwiązania CI/CD wraz z wykrywaniem zmiany

5.4 Przykładowe rozwiązanie 3



Rysunek 5.3: Przykładowe rozwiązania CI/CD AWS

Slajd 6

Najlepsze praktyki z MLOps na AWS

6.1 Rozpoczęcie projektu

- **Zespoły wielofunkcyjne:** Tworzenie zintegrowanych zespołów składających się z inżynierów danych, badaczy ML, inżynierów DevOps, programistów i ekspertów biznesowych.
- **Rozwój wymagań i procesów zarządzania:** Wczesne ustalanie procesów dla weryfikacji modeli, wyjaśnialności i zatwierdzeń przed wdrożeniem produkcyjnym.
- **Wskaźniki sukcesu (KPIs):** Określenie KPIs biznesowych i monitorowanie wpływu modelu i platformy na te wskaźniki.
- **Pilotowe projekty ML:** Wybór kilku pilotowych projektów ML do wdrożenia i nauki na platformie.
- **Stan docelowy i realizacja etapami:** Definiowanie długoterminowej wizji i celu platformy ML i stopniowe wdrażanie.

6.2 Projektowanie i wdrożenie

- **Zarządzane możliwości:** Wykorzystanie wbudowanych możliwości gotowych platform jak na przykład SageMaker - jako domyślnych.
- **Infrastruktura jako kod:** Użycie CloudFormation lub Terraform do zarządzania infrastrukturą ML.
- **CI/CD:** Implementacja ciągłej integracji i wdrażania z wykorzystaniem na przykład: CodePipeline, CodeBuild, CodeDeploy i SageMaker.
- **Automatyzacja śledzenia eksperymentów:** Automatyczne logowanie metadanych modeli.
- **Zatwierdzone repozytorium bibliotek:** Tworzenie scentralizowanego repozytorium bibliotek.
- **Skalowalność:** Projektowanie platformy do obsługi różnych wzorców użycia i szczytów ruchu.
- **Bezpieczeństwo:** Wprowadzenie najlepszych praktyk bezpieczeństwa od samego początku.
- **Samoobsługa:** Rozwój funkcji samoobsługi dla użytkowników.
- **Repozytorium modeli:** Użycie pojedynczego repozytorium dla modeli.
- **Centralne repozytorium cech:** Wdrażanie scentralizowanego repozytorium cech.

6.3 Użytkowanie platformy i praktyki operacyjne

- **Ograniczenie dostępu do produkcji:** Ograniczenie dostępu do systemów produkcyjnych do minimum.
- **Optymalizacja kosztów:** Wykorzystanie auto-scalingu i instancji spotowych do optymalizacji kosztów chmury.

- **Monitorowanie i obserwowalność:** Aktywne monitorowanie dokładności modeli i wydajności systemu.
- **Zarządzanie zmianą:** Definiowanie procesów zarządzania zmianami.
- **Zarządzanie incydentami:** Ustanowienie planu reagowania na incydenty.
- **Wdrożenia Multi-AZ i regionów:** Poprawa odporności i minimalizacja opóźnień przez wdrożenie w wielu strefach dostępności i regionach.
- **Planowanie pojemności:** Proaktywna ocena i projektowanie potrzeb infrastruktury.

Slajd 7

Wyzwania w MLOps na AWS

Bardzo dobry artykuł:

- MLOps Challenges and How to Face Them

7.1 Rzut oka na wyzwania w MLOps

1. **Zarządzanie Danych i Jakość Danych:** Wyzwanie w utrzymaniu jakości i spójności danych.
2. **Integracja i Orkiestracja:** Problemy związane z integracją różnorodnych narzędzi i technologii, oraz orkiestracją złożonych przepływów pracy.
3. **Skalowalność i Wydajność:** Trudności w skalowaniu infrastruktury i modeli do obsługi rosnących wymagań.
4. **Bezpieczeństwo i Prywatność Danych:** Zagadnienia ochrony danych, w tym zarządzanie dostępem i szyfrowanie.
5. **Monitoring i Utrzymanie Modeli:** Wyzwania związane z monitorowaniem wydajności modeli i ich ciągłym doskonaleniem.
6. **Automatyzacja i Samoobsługa:** Trudności w automatyzacji procesów oraz zapewnieniu samoobsługowych możliwości dla użytkowników.

7. **Współpraca i Zarządzanie Wiedzą:** Problemy związane z współpracą między zespołami oraz zarządzaniem wiedzą i umiejętnościami.
8. **Zgodność i Regulacje:** Wyzwania związane z przestrzeganiem przepisów branżowych i zgodnością.

Slajd 8

Podsumowanie oraz przyszłość dla MLOps

8.1 Ujednolicone platformy MLOps

W najbliższych latach możemy oczekwać wzrostu popularności platform MLOps typu end-to-end, które oferują zintegrowane środowiska umożliwiające pokrycie całego cyklu życia modelu ML - od eksperymentów po wdrożenie i monitorowanie. Te platformy zwiększą efektywność i skracają czas potrzebny na dostarczenie modeli do produkcji, jednocześnie upraszczając zarządzanie złożonymi przepływami pracy.

Przykład takich platform:



Rysunek 8.1: End-to-end platformy do projektow typu MLOps

Przykład platformy (bardzo rozbudowanej) ale związanej bezpośrednio z software developmentem:

- <https://www.harness.io/>

8.2 Serverless GPUs

Serverless computing, a w szczególności wykorzystanie GPU w modelu serverless, to kolejny trend, który będzie ewoluował. Możliwość uruchamiania zadań obliczeniowych wymagających GPU bez konieczności zarządzania fizyczną infrastrukturą jest odpowiedzią na rosnące zapotrzebowanie na moc obliczeniową przy jednoczesnej kontroli kosztów. Dostawcy usług chmurowych oferujący GPU w modelu serverless pozwolą na bardziej elastyczne i skalowalne przetwarzanie danych.

8.3 Innowacje i Przyszłość

MLOps jest obszarem, który z pewnością będzie nadal dynamicznie się rozwijał w nadchodzących latach.

1. **Znaczenie MLOps:** Narzędzia open-source, wzrost inwestycji w start-upy skoncentrowane na MLOps to wyraźne sygnały wzrostu znaczenia MLOps.
2. **Innowacje technologiczne:** Specjalistyczne układy przetwarzające dane, takie jak ASIC i FPGA (Field Programmable Gate Arrays), oraz postępy w automatyzacji DataOps, są kluczowe dla przyszłości MLOps.
3. **Standardy i praktyki:** Rozwijające się standardy takie jak Open Neural Network Exchange (ONNX) umożliwiają lepszą interoperacyjność między platformami ML, co jest kluczowe dla przyszłego rozwoju branży.
4. **Zastosowanie DevOps w ML:** Metodyki zaczerpnięte z DevOps, takie jak Continuous Training, Continuous Integration i Continuous Deployment, zyskują na znaczeniu w projektach ML, tworząc podstawy dla bardziej zwinnych i odpornych systemów ML.
5. **LLMOPs - zarządzanie dużymi modelami językowymi:** Wyzwania związane z LLMOPs rozniez zyskają dzięki zastosowaniu automatyzacji.

Załącznik 1 - Materiały dodatkowe

Książki

- The Machine Learning Solutions Architect Handbook - Second Edition
- Automated Machine Learning on AWS
- Machine Learning Engineering with Python - Second Edition
- System Design Interview volume 1
- System Design Interview volume 2
- Practical Machine Learning on Databricks

Kursy - Udemy

- Kubernetes Hands-On - Deploy Microservices to the AWS Cloud
- AWS EKS Kubernetes-Masterclass | DevOps, Microservices
- Taking Python to Production

Artykuly

- MLOps Landscape in 2024
- Is It Still Smart to Learn DevOps in 2024?
- Why Do You Need a Feature Store?
- MLOps learning resources

- End-to-End MLOps on AWS

Dev Containers & Python

- DevContainers
- Docker and poetry

Darmowe szkolenia

- mlops-zoomcamp
- The Full Stack 7-Steps MLOps Framework

GitHub repos

- Awesome MLOps

YouTube videos

- A CI/CD Framework for Production Machine Learning at Massive Scale

Blogs

- SwirlAI