A3 AiNex
AX/MAX Consulting Assistant & Analyst

ISO23053 ML Framework
Guide BooK (Vol.5)

A3 AiNex R&D

ISO 23053 ML 시스템 프레임워크 대시보드 가이드

개요

본 문서는 ISO 23053 ML 시스템 프레임워크 워크스페이스의 모든 항목에 대한 구체적인 설명을 제공합니다. ISO/IEC 23053:2022 기반 ML 시스템 프레임워크의 계층 구조와 구성 요소를 시각화하고 관리하는 대시보드로, 컨설턴트와 이용자가 이해하고 활용해야 하는 모든 내용을 포함합니다.

1. 워크스페이스 접근 및 화면 구성

1.1 접근 방법

위치: 사이드바 메뉴 → "국제표준 거버넌스" → "ISO 23053 ML" → "ML 대시보드"

접근 경로:
1. 프로젝트를 선택한 후 사이드바에서 "국제표준 거버넌스" 섹션을 확장한다.
2. "ISO 23053 ML" 메뉴를 클릭한다.
3. "ML 대시보드" 링크를 클릭한다.

주의사항:
- 프로젝트를 먼저 선택한 후 접근하는 것을 권장한다.
- 본 워크스페이스는 ML 시스템 프레임워크의 전체 구조를 보여주는 대시보드이므로, 다른 ISO 23053 관련 워크스페이스를 먼저 확인하는 것이 더 유용하다.

2. ML 기반 AI 시스템 계층 구조

2.1 목적

ISO 23053 표준에서 정의한 ML 기반 AI 시스템의 계층 구조를 시각화하여, 각 계층의 역할과 구성 요소를 명확히 이해할 수 있도록 한다.

2.2 계층 구조 개요

ML 기반 AI 시스템은 다음과 같은 5개 계층으로 구성된다:

1. 응용 계층 (Application Layer): 최상위 계층으로 사용자 인터페이스와 비즈니스 로직을 포함
2. ML 엔진 계층 (ML Engine Layer): 모델 서빙 및 추론을 담당하는 계층
3. ML 파이프라인 계층 (ML Pipeline Layer): 데이터 처리부터 모델 배포까지의 파이프라인을 관리하는 계층
4. 데이터 계층 (Data Layer): 데이터 저장 및 관리를 담당하는 계층
5. 인프라 계층 (Infrastructure Layer): 하드웨어 및 네트워크 인프라를 제공하는 최하위 계층

2.3 계층별 상세 설명

2.3.1 응용 계층 (Application Layer)

- 역할: 사용자와의 상호작용 및 비즈니스 로직 처리

구성 요소:
- 사용자 인터페이스: 웹, 모바일, API 등 다양한 인터페이스 제공
- 비즈니스 로직: 도메인별 비즈니스 규칙 및 프로세스 처리
- 통합 인터페이스: 외부 시스템과의 연동 인터페이스
- 모니터링 대시보드: 시스템 상태 및 성능 모니터링 화면

컨설턴트 가이드:
- 사용자 요구사항을 반영하여 적절한 인터페이스를 설계한다.
- 비즈니스 로직과 ML 추론 결과를 효과적으로 통합한다.

이용자 가이드:
- 각 구성 요소의 역할을 이해하고 활용한다.
- 모니터링 대시보드를 통해 시스템 상태를 확인한다.

2.3.2 ML 엔진 계층 (ML Engine Layer)

- 역할: 학습된 모델의 서빙 및 추론 수행

구성 요소:
- 모델 서빙: 학습된 모델을 서비스로 제공하는 컴포넌트
- 추론 엔진: 모델을 사용하여 예측을 수행하는 엔진
- 특성 스토어: 추론에 필요한 특성을 저장하고 제공하는 저장소
- 모델 레지스트리: 모델 버전 및 메타데이터를 관리하는 저장소

컨설턴트 가이드:
- 모델 서빙 인프라를 설계하고 구축한다.
- 추론 성능을 최적화하여 지연시간을 최소화한다.

이용자 가이드:
- 모델 레지스트리를 통해 모델 버전을 관리한다.
- 특성 스토어를 활용하여 일관된 특성을 사용한다.

2.3.3 ML 파이프라인 계층 (ML Pipeline Layer)

- 역할: 데이터 처리부터 모델 배포까지의 전체 파이프라인 관리

구성 요소:
- 데이터 처리: 데이터 수집, 정제, 변환 등의 처리 작업
- 모델 훈련: 모델 학습 및 최적화 작업
- 모델 평가: 모델 성능 평가 및 검증 작업

- 배포 관리: 모델 배포 및 버전 관리 작업

컨설턴트 가이드:
- 파이프라인을 자동화하여 효율성을 높인다.
- 각 단계의 품질을 보장하는 체크포인트를 설정한다.

이용자 가이드:
- 파이프라인의 각 단계를 이해하고 활용한다.
- 파이프라인 실행 결과를 모니터링한다.

2.3.4 데이터 계층 (Data Layer)

- 역할: 데이터 저장 및 관리

구성 요소:
- 데이터 레이크: 원시 데이터를 저장하는 저장소
- 데이터 웨어하우스: 구조화된 데이터를 저장하는 저장소
- 데이터 파이프라인: 데이터 이동 및 변환을 처리하는 파이프라인
- 메타데이터 카탈로그: 데이터 자산의 메타데이터를 관리하는 카탈로그

컨설턴트 가이드:
- 데이터 아키텍처를 설계하여 효율적인 데이터 관리를 지원한다.
- 메타데이터 카탈로그를 구축하여 데이터 검색 및 계보 추적을 가능하게 한다.

이용자 가이드:
- 데이터 레이크와 웨어하우스를 적절히 활용한다.
- 메타데이터 카탈로그를 통해 필요한 데이터를 검색한다.

2.3.5 인프라 계층 (Infrastructure Layer)

- 역할: 하드웨어 및 네트워크 인프라 제공

구성 요소:
- 컴퓨팅 (GPU/TPU): 모델 훈련 및 추론을 위한 컴퓨팅 리소스
- 스토리지: 데이터 및 모델 저장을 위한 스토리지 리소스
- 네트워크: 시스템 간 통신을 위한 네트워크 인프라
- 보안: 시스템 보안을 위한 보안 인프라

컨설턴트 가이드:
- 인프라 요구사항을 분석하여 적절한 인프라를 설계한다.
- 확장성과 안정성을 고려하여 인프라를 구성한다.

이용자 가이드:
- 인프라 리소스를 효율적으로 활용한다.
- 보안 정책을 준수하여 시스템을 운영한다.

3. 기능적 구성 요소 현황

3.1 데이터 관리

의미: 데이터 관리 관련 구성 요소의 구현 현황
- 표시 형식: 상태 표시 (예: 구현됨) 및 구성 요소 개수 (예: 6개 구성 요소)

구성 요소:
데이터 수집, 데이터 처리, 데이터 품질 검증, 데이터 버전 관리, 메타데이터 카탈로그, 데이터 보안/프라이버시

컨설턴트 가이드:
데이터 관리 구성 요소의 구현 현황을 확인하고 미구현 항목을 식별한다.

이용자 가이드:
데이터 관리 구성 요소의 현황을 확인하여 데이터 관리 체계를 파악한다.

3.2 모델 개발

의미: 모델 개발 관련 구성 요소의 구현 현황
- 표시 형식: 상태 표시 (예: 구현됨) 및 구성 요소 개수 (예: 6개 구성 요소)

구성 요소:
실험 추적 시스템, 모델 레지스트리, 자동화된 테스트, 모델 문서화, 하이퍼파라미터 최적화, 모델 해석성 도구

컨설턴트 가이드:
모델 개발 구성 요소의 구현 현황을 확인하고 개발 프로세스를 개선한다.

이용자 가이드:
모델 개발 구성 요소를 활용하여 효율적으로 모델을 개발한다.

3.3 모델 배포

의미: 모델 배포 관련 구성 요소의 구현 현황
- 표시 형식: 상태 표시 (예: 구현됨) 및 구성 요소 개수 (예: 6개 구성 요소)

구성 요소:
CI/CD 파이프라인, 모델 서빙 인프라, 배포 전략, 롤백 메커니즘, 컨테이너 오케스트레이션, API 게이트웨이/로드밸런서

컨설턴트 가이드:
모델 배포 구성 요소의 구현 현황을 확인하고 배포 프로세스를 자동화한다.

이용자 가이드:
모델 배포 구성 요소를 활용하여 안정적으로 모델을 배포한다.

3.4 모델 운영

의미: 모델 운영 관련 구성 요소의 구현 현황
- 표시 형식: 상태 표시 (예: 구현됨) 및 구성 요소 개수 (예: 6개 구성 요소)

구성 요소:
성능 모니터링 대시보드, 드리프트 감지 시스템, 알림 시스템, 재훈련 파이프라인, 감사 로깅, 비용 모니터링

컨설턴트 가이드:
모델 운영 구성 요소의 구현 현황을 확인하고 운영 프로세스를 최적화한다.

이용자 가이드:
모델 운영 구성 요소를 활용하여 지속적으로 모델을 모니터링하고 개선한다.

4. ML 유형별 프레임워크 지원

4.1 목적

ISO 23053 프레임워크가 지원하는 다양한 ML 유형을 제시하여, 조직이 어떤 ML 유형을 활용할 수 있는지 이해할 수 있도록 한다.

4.2 지원 ML 유형

4.2.1 지도 학습 (Supervised Learning)

의미: 레이블이 있는 데이터를 사용하여 모델을 학습시키는 방법

주요 유형:
- 분류: 입력 데이터를 미리 정의된 클래스로 분류
- 회귀: 연속적인 값을 예측
- 활용 사례: 이미지 분류, 스팸 필터링, 가격 예측, 고객 이탈 예측

컨설턴트 가이드:
레이블이 있는 데이터가 충분한 경우 지도 학습을 활용한다.

이용자 가이드:
분류나 회귀 문제에 지도 학습을 적용한다.

4.2.2 비지도 학습 (Unsupervised Learning)

의미: 레이블이 없는 데이터에서 패턴을 발견하는 방법

주요 유형:
- 클러스터링: 유사한 데이터를 그룹화
- 차원축소: 고차원 데이터를 저차원으로 변환

- 활용 사례: 고객 세그멘테이션, 이상 탐지, 데이터 시각화

컨설턴트 가이드:
레이블이 없거나 패턴 발견이 필요한 경우 비지도 학습을 활용한다.

이용자 가이드:
데이터 구조를 이해하거나 그룹화가 필요한 경우 비지도 학습을 적용한다.

4.2.3 강화 학습 (Reinforcement Learning)

의미: 에이전트가 환경과 상호작용하며 보상을 최대화하는 방법

주요 유형:
- 에이전트: 환경과 상호작용하는 주체
- 정책: 에이전트의 행동을 결정하는 정책
- 활용 사례: 게임 AI, 로봇 제어, 자율주행, 추천 시스템

컨설턴트 가이드:
순차적 의사결정이 필요한 경우 강화 학습을 활용한다.

이용자 가이드:
에이전트가 환경과 상호작용하며 학습하는 시스템에 강화 학습을 적용한다.

4.2.4 딥러닝 (Deep Learning)

의미: 다층 신경망을 사용하여 복잡한 패턴을 학습하는 방법

주요 유형:
- CNN (Convolutional Neural Network): 이미지 처리에 특화된 신경망
- RNN (Recurrent Neural Network): 시계열 데이터 처리에 특화된 신경망
- Transformer: 자연어 처리에 특화된 신경망
- 활용 사례: 이미지 인식, 자연어 처리, 음성 인식, 번역

컨설턴트 가이드:
복잡한 패턴 학습이 필요한 경우 딥러닝을 활용한다.

이용자 가이드:
대용량 데이터와 복잡한 문제에 딥러닝을 적용한다.

5. 활용 가이드

5.1 컨설턴트 활용

- 시스템 설계: 계층 구조를 참고하여 ML 시스템 아키텍처를 설계한다.
- 구성 요소 평가: 기능적 구성 요소 현황을 확인하여 구현 상태를 평가한다.

- ML 유형 선택: 지원되는 ML 유형을 참고하여 적절한 ML 유형을 선택한다.

5.2 이용자 활용

- 시스템 이해: 계층 구조를 통해 ML 시스템의 전체 구조를 이해한다.
- 구성 요소 확인: 기능적 구성 요소 현황을 확인하여 구현된 기능을 파악한다.
- ML 유형 활용: 지원되는 ML 유형을 참고하여 프로젝트에 적합한 ML 유형을 선택한다.

6. 주의사항

6.1 계층 간 의존성

- 각 계층은 하위 계층에 의존하므로, 하위 계층을 먼저 구축해야 한다.
- 계층 간 인터페이스를 명확히 정의하여 결합도를 낮춘다.

6.2 구성 요소 구현

- 모든 구성 요소를 한 번에 구현하기보다는 우선순위에 따라 단계적으로 구현한다.
- 필수 구성 요소를 먼저 구현하고, 선택적 구성 요소는 점진적으로 추가한다.

6.3 ML 유형 선택

- 문제의 특성과 데이터의 특성을 고려하여 적절한 ML 유형을 선택한다.
- 여러 ML 유형을 조합하여 사용할 수도 있다.

7. 참고 자료

- ISO/IEC 23053:2022 - Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)
- ISO 23053 구성 요소 매핑 가이드
- ML 시스템 프레임워크 구현 체크리스트 가이드
- ML 파이프라인 아키텍처 가이드

8. 문의 및 지원

ML 대시보드 사용 과정에서 문의사항이 있으시면 컨설턴트에게 문의하시기 바랍니다.

ISO 23053 구성 요소 → 플랫폼 기능 매핑 가이드

개요

본 문서는 ISO 23053 구성 요소 → 플랫폼 기능 매핑 워크스페이스의 모든 항목에 대한 구체적인 설명을 제공합니다. **ISO/IEC 23053:2022 표준**에서 정의한 ML 시스템 구성 요소와 플랫폼의 기능/모듈을 매핑하여, 각 구성 요소에 해당하는 플랫폼 기능을 명확히 제시하는 워크스페이스로, 컨설턴트와 이용자가 이해하고 활용해야 하는 모든 내용을 포함합니다.

1. 워크스페이스 접근 및 화면 구성

1.1 접근 방법

위치: 사이드바 메뉴 → "국제표준 거버넌스" → "ISO 23053 ML" → "구성 요소 매핑"
접근 경로:
1. 프로젝트를 선택한 후 사이드바에서 "국제표준 거버넌스" 섹션을 확장한다.
2. "ISO 23053 ML" 메뉴를 클릭한다.
3. "구성 요소 매핑" 링크를 클릭한다.

주의사항:
- 프로젝트를 먼저 선택한 후 접근하는 것을 권장한다.
- 본 워크스페이스는 ISO 23053 구성 요소와 플랫폼 기능 간의 매핑 관계를 보여주므로, 각 구성 요소의 의미를 이해한 후 확인하는 것이 더 유용하다.

2. ISO 23053 구성 요소 - 플랫폼 기능 매핑

2.1 목적

ISO 23053 표준에서 정의한 ML 시스템 구성 요소와 플랫폼의 기능/모듈을 매핑하여, 각 구성 요소에 해당하는 플랫폼 기능을 명확히 제시한다. 이를 통해 구성 요소별로 어떤 플랫폼 기능을 활용해야 하는지 쉽게 파악할 수 있다.

2.2 매핑 테이블 구성

매핑 테이블은 다음 5개 컬럼으로 구성된다:

1. ISO 23053 구성 요소: ISO 23053 표준에서 정의한 ML 시스템 구성 요소
2. 세부 기능: 해당 구성 요소의 구체적인 기능
3. 플랫폼 모듈/기능: 해당 구성 요소에 대응하는 플랫폼의 모듈 또는 기능
4. 구현 상태: 플랫폼 기능의 구현 상태 (연결됨/미연결)
5. 바로가기: 해당 플랫폼 기능으로 이동하는 버튼

2.3 구성 요소별 상세 매핑

2.3.1 데이터 관리 (Data Management)

2.3.1.1 데이터 수집/처리

- ISO 23053 구성 요소: 데이터 관리
- 세부 기능: 데이터 수집/처리
- 플랫폼 모듈/기능: Stage 2: 데이터 아키텍처

의미:
- 데이터 수집 및 처리는 ML 시스템의 기초가 되는 작업이다.
- Stage 2의 데이터 아키텍처 설계에서 데이터 수집 방법과 처리 프로세스를 정의한다.

컨설턴트 가이드:
- Stage 2의 데이터 아키텍처 설계를 통해 데이터 수집 및 처리 방식을 정의한다.
- 배치 및 스트리밍 데이터 수집 방식을 고려한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 2 워크스페이스로 이동한다.
- 데이터 아키텍처 설계에서 데이터 수집 및 처리 방식을 정의한다.

2.3.1.2 데이터 품질 검증

- ISO 23053 구성 요소: 데이터 관리
- 세부 기능: 데이터 품질 검증
- 플랫폼 모듈/기능: Stage 2: 데이터 거버넌스

의미:
- 데이터 품질 검증은 ML 모델의 성능에 직접적인 영향을 미친다.
- Stage 2의 데이터 거버넌스에서 데이터 품질 검증 프로세스를 수립한다.

컨설턴트 가이드:
- Stage 2의 데이터 거버넌스를 통해 데이터 품질 검증 프로세스를 수립한다.
- 데이터 품질 지표를 정의하고 자동화된 검증 체계를 구축한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 2 워크스페이스로 이동한다.
- 데이터 거버넌스에서 데이터 품질 검증 프로세스를 수립한다.

2.3.1.3 메타데이터 카탈로그

- ISO 23053 구성 요소: 데이터 관리
- 세부 기능: 메타데이터 카탈로그
- 플랫폼 모듈/기능: ISO 42001: 데이터 시트

의미:
- 메타데이터 카탈로그는 데이터 자산의 검색 및 계보 추적을 가능하게 한다.
- ISO 42001의 데이터 시트를 통해 데이터셋의 메타데이터를 체계적으로 관리한다.

컨설턴트 가이드:
- ISO 42001의 데이터 시트를 활용하여 데이터셋의 메타데이터를 관리한다.
- 메타데이터 카탈로그를 구축하여 데이터 검색 및 계보 추적을 지원한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 데이터 시트 워크스페이스로 이동한다.
- 데이터 시트를 작성하여 데이터셋의 메타데이터를 관리한다.

2.3.2 모델 개발 (Model Development)

2.3.2.1 실험 관리/추적

- ISO 23053 구성 요소: 모델 개발
- 세부 기능: 실험 관리/추적
- 플랫폼 모듈/기능: Stage 3: PoC 계획

의미:
- 실험 관리 및 추적은 모델 개발 과정에서 중요한 역할을 한다.
- Stage 3의 PoC 계획에서 실험 설계 및 추적 방식을 정의한다.

컨설턴트 가이드:
- Stage 3의 PoC 계획을 통해 실험 설계 및 추적 방식을 정의한다.
- 실험 결과를 체계적으로 관리하여 재현성을 보장한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 3 워크스페이스로 이동한다.
- PoC 계획에서 실험 설계 및 추적 방식을 정의한다.

2.3.2.2 모델 평가/해석

- ISO 23053 구성 요소: 모델 개발
- 세부 기능: 모델 평가/해석
- 플랫폼 모듈/기능: Stage 3: 플랫폼 구축

의미:
- 모델 평가 및 해석은 모델의 성능과 신뢰성을 확인하는 중요한 단계이다.
- Stage 3의 플랫폼 구축에서 모델 평가 및 해석 도구를 구축한다.

컨설턴트 가이드:
- Stage 3의 플랫폼 구축을 통해 모델 평가 및 해석 도구를 구축한다.
- 다양한 평가 지표와 해석 도구를 활용하여 모델을 평가한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 3 워크스페이스로 이동한다.
- 플랫폼 구축에서 모델 평가 및 해석 도구를 구축한다.

2.3.2.3 모델 문서화

- ISO 23053 구성 요소: 모델 개발
- 세부 기능: 모델 문서화
- 플랫폼 모듈/기능: ISO 42001: 모델 카드

의미:
- 모델 문서화는 모델의 성능, 제한사항, 사용 가이드를 명확히 하는 중요한 작업이다.
- ISO 42001의 모델 카드를 통해 모델 정보를 체계적으로 문서화한다.

컨설턴트 가이드:
- ISO 42001의 모델 카드를 활용하여 모델 정보를 문서화한다.
- 모델의 성능, 제한사항, 사용 가이드를 명확히 기록한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 모델 카드 워크스페이스로 이동한다.
- 모델 카드를 작성하여 모델 정보를 문서화한다.

2.3.3 모델 배포 (Model Deployment)

2.3.3.1 모델 패키징/서빙

- ISO 23053 구성 요소: 모델 배포
- 세부 기능: 모델 패키징/서빙
- 플랫폼 모듈/기능: Stage 3: 시스템 통합

의미:
- 모델 패키징 및 서빙은 학습된 모델을 운영 환경에 배포하는 중요한 단계이다.
- Stage 3의 시스템 통합에서 모델 패키징 및 서빙 방식을 정의한다.

컨설턴트 가이드:
- Stage 3의 시스템 통합을 통해 모델 패키징 및 서빙 방식을 정의한다.
- 컨테이너화 및 API 서빙 방식을 고려한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 3 워크스페이스로 이동한다.
- 시스템 통합에서 모델 패키징 및 서빙 방식을 정의한다.

2.3.3.2 배포 전략 (카나리, A/B)

- ISO 23053 구성 요소: 모델 배포

- 세부 기능: 배포 전략 (카나리, A/B)
- 플랫폼 모듈/기능: MLOps 표준 5.4

의미:
- 배포 전략은 모델을 안전하게 배포하고 성능을 검증하는 중요한 요소이다.
- MLOps 표준 5.4에서 카나리 배포 및 A/B 테스트 전략을 정의한다.

컨설턴트 가이드:
- MLOps 표준 5.4를 참고하여 배포 전략을 수립한다.
- 카나리 배포 및 A/B 테스트를 통해 모델 성능을 검증한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 MLOps 표준 워크스페이스로 이동한다.
- MLOps 표준 5.4를 확인하여 배포 전략을 수립한다.

2.3.3.3 모델 레지스트리

- ISO 23053 구성 요소: 모델 배포
- 세부 기능: 모델 레지스트리
- 플랫폼 모듈/기능: MLOps 표준 5.3

의미:
- 모델 레지스트리는 모델 버전 및 메타데이터를 관리하는 저장소이다.
- MLOps 표준 5.3에서 모델 레지스트리 구축 방식을 정의한다.

컨설턴트 가이드:
- MLOps 표준 5.3을 참고하여 모델 레지스트리를 구축한다.
- 모델 버전 및 메타데이터를 체계적으로 관리한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 MLOps 표준 워크스페이스로 이동한다.
- MLOps 표준 5.3을 확인하여 모델 레지스트리를 구축한다.

2.3.4 모델 운영 (Model Operations)

2.3.4.1 성능 모니터링

- ISO 23053 구성 요소: 모델 운영
- 세부 기능: 성능 모니터링
- 플랫폼 모듈/기능: Stage 5: 모니터링, 성과 평가

의미:
- 성능 모니터링은 배포된 모델의 성능을 지속적으로 추적하는 중요한 작업이다.
- Stage 5의 모니터링 및 성과 평가에서 성능 모니터링 체계를 구축한다.

컨설턴트 가이드:
- Stage 5의 모니터링 및 성과 평가를 통해 성능 모니터링 체계를 구축한다.
- 모델 성능 지표를 정의하고 자동화된 모니터링을 구현한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 5 워크스페이스로 이동한다.
- 모니터링 및 성과 평가에서 성능 모니터링 체계를 구축한다.

2.3.4.2 드리프트 감지

- ISO 23053 구성 요소: 모델 운영
- 세부 기능: 드리프트 감지
- 플랫폼 모듈/기능: MLOps 표준 5.5

의미:
- 드리프트 감지는 데이터나 모델의 변화를 자동으로 탐지하는 중요한 기능이다.
- MLOps 표준 5.5에서 드리프트 감지 방식을 정의한다.

컨설턴트 가이드:
- MLOps 표준 5.5를 참고하여 드리프트 감지 시스템을 구축한다.
- 데이터 드리프트 및 모델 드리프트를 자동으로 탐지한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 MLOps 표준 워크스페이스로 이동한다.
- MLOps 표준 5.5를 확인하여 드리프트 감지 시스템을 구축한다.

2.3.4.3 재훈련/거버넌스

- ISO 23053 구성 요소: 모델 운영
- 세부 기능: 재훈련/거버넌스
- 플랫폼 모듈/기능: Stage 5: 지속적 개선

의미:
- 재훈련 및 거버넌스는 모델 성능을 유지하고 개선하는 중요한 프로세스이다.
- Stage 5의 지속적 개선에서 재훈련 및 거버넌스 프로세스를 수립한다.

컨설턴트 가이드:
- Stage 5의 지속적 개선을 통해 재훈련 및 거버넌스 프로세스를 수립한다.
- 재훈련 트리거 조건을 정의하고 자동화된 재훈련 파이프라인을 구축한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 5 워크스페이스로 이동한다.
- 지속적 개선에서 재훈련 및 거버넌스 프로세스를 수립한다.

2.3.5 품질 속성 (Quality Attributes)

2.3.5.1 성능/확장성/신뢰성

- ISO 23053 구성 요소: 품질 속성
- 세부 기능: 성능/확장성/신뢰성
- 플랫폼 모듈/기능: Stage 2: 비기능적 요구사항

의미:
- 성능, 확장성, 신뢰성은 ML 시스템의 품질을 결정하는 중요한 속성이다.
- Stage 2의 비기능적 요구사항에서 이러한 품질 속성을 정의한다.

컨설턴트 가이드:
- Stage 2의 비기능적 요구사항을 통해 성능, 확장성, 신뢰성 요구사항을 정의한다.
- 각 품질 속성에 대한 구체적인 지표를 설정한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 2 워크스페이스로 이동한다.
- 비기능적 요구사항에서 성능, 확장성, 신뢰성 요구사항을 정의한다.

2.3.5.2 보안

- ISO 23053 구성 요소: 품질 속성
- 세부 기능: 보안
- 플랫폼 모듈/기능: 보안 설정, 거버넌스 체계

의미:
- 보안은 ML 시스템의 중요한 품질 속성이다.
- 보안 설정 및 거버넌스 체계를 통해 보안 요구사항을 구현한다.
-

컨설턴트 가이드:
- 보안 설정 및 거버넌스 체계를 통해 보안 요구사항을 구현한다.
- 접근 제어, 암호화, 감사 로깅 등의 보안 기능을 구축한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 보안 설정 워크스페이스로 이동한다.
- 보안 설정 및 거버넌스 체계를 통해 보안 요구사항을 구현한다.

2.3.6 인프라 (Infrastructure)

2.3.6.1 컴퓨팅/스토리지/네트워크

- ISO 23053 구성 요소: 인프라
- 세부 기능: 컴퓨팅/스토리지/네트워크
- 플랫폼 모듈/기능: Stage 2: 아키텍처 설계

의미:
- 인프라는 ML 시스템의 기반이 되는 하드웨어 및 네트워크 리소스이다.
- Stage 2의 아키텍처 설계에서 인프라 요구사항을 정의한다.

컨설턴트 가이드:
- Stage 2의 아키텍처 설계를 통해 인프라 요구사항을 정의한다.
- 컴퓨팅, 스토리지, 네트워크 리소스를 계획한다.

이용자 가이드:
- "이동" 버튼을 클릭하여 Stage 2 워크스페이스로 이동한다.
- 아키텍처 설계에서 인프라 요구사항을 정의한다.

3. 기능적 구성 요소 상세

3.1 데이터 관리 구성 요소

3.1.1 데이터 수집

의미:
배치/스트리밍 수집, API 연동, 크롤링 등을 통한 데이터 수집

구현 방법:
- 배치 수집: 주기적으로 대량의 데이터를 수집
- 스트리밍 수집: 실시간으로 연속적인 데이터를 수집
- API 연동: 외부 API를 통해 데이터를 수집
- 크롤링: 웹 크롤러를 통해 데이터를 수집

컨설턴트 가이드:
데이터 소스의 특성에 따라 적절한 수집 방식을 선택한다.

이용자 가이드:
데이터 수집 방식을 이해하고 활용한다.

3.1.2 데이터 처리

의미:
정제/변환, 특성 추출, 라벨링, 증강 등의 데이터 처리 작업

구현 방법:
- 정제/변환: 데이터의 품질을 개선하고 형식을 변환
- 특성 추출: 모델에 필요한 특성을 추출
- 라벨링: 지도 학습을 위한 레이블 생성
- 증강: 데이터 증강을 통해 데이터셋 확장

컨설턴트 가이드:
데이터 처리 파이프라인을 자동화하여 효율성을 높인다.

이용자 가이드:
데이터 처리 프로세스를 이해하고 활용한다.

3.1.3 데이터 품질

의미:
완전성, 일관성, 정확성 등의 데이터 품질 검증

구현 방법:
- 완전성 검증: 결측값 확인 및 처리
- 일관성 검증: 데이터 간 일관성 확인
- 정확성 검증: 데이터의 정확성 확인

컨설턴트 가이드:
데이터 품질 검증을 자동화하여 품질을 보장한다.

이용자 가이드:
데이터 품질 검증 프로세스를 이해하고 활용한다.

3.1.4 버전 관리

의미:
스냅샷, 변경 이력, 롤백 등의 데이터 버전 관리

구현 방법:
- 스냅샷: 특정 시점의 데이터 상태를 저장
- 변경 이력: 데이터 변경 내역을 추적
- 롤백: 이전 버전으로 데이터 복원

컨설턴트 가이드:
데이터 버전 관리를 통해 재현성을 보장한다.

이용자 가이드:
데이터 버전 관리 시스템을 이해하고 활용한다.

3.2 모델 개발 구성 요소

3.2.1 실험 관리

의미:
실험 설계, 추적, 결과 비교, 재현성 보장

구현 방법:
- 실험 설계: 실험 목적 및 방법 설계
- 추적: 실험 과정 및 결과 추적
- 결과 비교: 여러 실험 결과 비교
- 재현성: 실험 재현을 위한 환경 및 파라미터 저장

컨설턴트 가이드:
실험 관리 도구를 활용하여 실험을 체계적으로 관리한다.

이용자 가이드:
실험 관리 프로세스를 이해하고 활용한다.

3.2.2 모델 훈련

의미:
분산 훈련, 증분 학습, 전이 학습 등의 모델 훈련 방법

구현 방법:
- 분산 훈련: 여러 GPU/서버를 활용한 병렬 훈련
- 증분 학습: 새로운 데이터를 추가하여 모델 업데이트
- 전이 학습: 사전 학습된 모델을 활용한 학습

컨설턴트 가이드:
모델의 특성에 따라 적절한 훈련 방법을 선택한다.

이용자 가이드:
모델 훈련 방법을 이해하고 활용한다.

3.2.3 하이퍼파라미터 튜닝

의미:
그리드/랜덤 서치, 베이지안 최적화, AutoML 등을 통한 하이퍼파라미터 최적화

구현 방법:
- 그리드 서치: 하이퍼파라미터 조합을 체계적으로 탐색
- 랜덤 서치: 무작위로 하이퍼파라미터 조합 탐색
- 베이지안 최적화: 이전 결과를 활용하여 효율적으로 탐색
- AutoML: 자동으로 최적의 모델 및 하이퍼파라미터 탐색

컨설턴트 가이드:
하이퍼파라미터 튜닝을 자동화하여 효율성을 높인다.

이용자 가이드:
하이퍼파라미터 튜닝 방법을 이해하고 활용한다.

3.2.4 모델 평가/해석

의미:
성능 메트릭, SHAP/LIME, 편향 평가 등을 통한 모델 평가 및 해석

구현 방법:
- 성능 메트릭: 정확도, 정밀도, 재현율, F1 점수 등
- SHAP/LIME: 모델 예측에 대한 설명 제공
- 편향 평가: 모델의 편향 여부 평가

컨설턴트 가이드:
다양한 평가 지표와 해석 도구를 활용하여 모델을 평가한다.

이용자 가이드:
모델 평가 및 해석 방법을 이해하고 활용한다.

3.3 모델 배포 구성 요소

3.3.1 모델 패키징

의미:
컨테이너화, 직렬화, 종속성 관리 등을 통한 모델 패키징

구현 방법:
- 컨테이너화: Docker 등을 활용한 모델 컨테이너화
- 직렬화: 모델을 파일로 저장
- 종속성 관리: 모델 실행에 필요한 라이브러리 관리

컨설턴트 가이드:
모델 패키징을 표준화하여 배포 효율성을 높인다.

이용자 가이드:
모델 패키징 방법을 이해하고 활용한다.

3.3.2 모델 서빙

의미:
REST API, gRPC, 배치/스트리밍 추론 등을 통한 모델 서빙

구현 방법:
- REST API: HTTP 기반 API 서빙
- gRPC: 고성능 RPC 서빙
- 배치 추론: 대량의 데이터를 일괄 처리
- 스트리밍 추론: 실시간으로 연속적인 추론 수행

컨설턴트 가이드:
요구사항에 따라 적절한 서빙 방식을 선택한다.

이용자 가이드:
모델 서빙 방법을 이해하고 활용한다.

3.3.3 배포 전략

의미:
블루-그린, 카나리, A/B 테스트 등의 배포 전략

구현 방법:
- 블루-그린 배포: 새 버전과 기존 버전을 동시에 운영
- 카나리 배포: 소규모 트래픽으로 새 버전 테스트
- A/B 테스트: 여러 버전을 동시에 테스트하여 성능 비교

컨설턴트 가이드:
배포 전략을 통해 안전하게 모델을 배포한다.

이용자 가이드:
배포 전략을 이해하고 활용한다.

3.3.4 롤백 관리

의미:
자동/수동 롤백, 롤백 기준 등을 통한 롤백 관리

구현 방법:
- 자동 롤백: 성능 저하 시 자동으로 이전 버전으로 복원
- 수동 롤백: 관리자가 수동으로 이전 버전으로 복원
- 롤백 기준: 롤백을 수행할 조건 정의

컨설턴트 가이드:
롤백 메커니즘을 구축하여 배포 위험을 최소화한다.

이용자 가이드:
롤백 관리 방법을 이해하고 활용한다.

3.4 모델 운영 구성 요소

3.4.1 성능 모니터링

의미:
지연시간, 처리량, 정확도, 리소스 사용 등의 성능 모니터링

구현 방법:
- 지연시간 모니터링: 요청 처리 시간 추적
- 처리량 모니터링: 초당 처리 요청 수 추적
- 정확도 모니터링: 모델 예측 정확도 추적
- 리소스 사용 모니터링: CPU, 메모리, GPU 사용량 추적

컨설턴트 가이드:
성능 모니터링을 자동화하여 지속적으로 모델 성능을 추적한다.

이용자 가이드:
성능 모니터링 방법을 이해하고 활용한다.

3.4.2 드리프트 감지

의미:
데이터/개념/예측/특성 드리프트 등의 드리프트 감지

구현 방법:
- 데이터 드리프트: 입력 데이터 분포의 변화 감지
- 개념 드리프트: 목표 변수의 분포 변화 감지
- 예측 드리프트: 모델 예측 분포의 변화 감지
- 특성 드리프트: 특성 분포의 변화 감지

컨설턴트 가이드:
드리프트 감지를 자동화하여 모델 성능 저하를 사전에 감지한다.

이용자 가이드:
드리프트 감지 방법을 이해하고 활용한다.

3.4.3 알림/경고

의미:
임계값 기반, 이상 탐지, 에스컬레이션 등의 알림 및 경고 시스템

구현 방법:
- 임계값 기반: 성능 지표가 임계값을 초과하면 알림
- 이상 탐지: 이상 패턴을 자동으로 탐지하여 알림
- 에스컬레이션: 심각한 문제 발생 시 상위 관리자에게 알림

컨설턴트 가이드:
알림 및 경고 시스템을 구축하여 문제를 신속하게 대응한다.

이용자 가이드:
알림 및 경고 시스템을 이해하고 활용한다.

3.4.4 재훈련 트리거

의미:
일정/성능/드리프트 기반 등의 재훈련 트리거

구현 방법:
- 일정 기반: 정기적으로 모델 재훈련
- 성능 기반: 성능 저하 시 모델 재훈련
- 드리프트 기반: 드리프트 감지 시 모델 재훈련

컨설턴트 가이드:
재훈련 트리거를 자동화하여 모델 성능을 유지한다.

이용자 가이드:
재훈련 트리거 방법을 이해하고 활용한다.

4. 활용 가이드

4.1 컨설턴트 활용

- 매핑 활용: ISO 23053 구성 요소와 플랫폼 기능 간의 매핑을 활용하여 구현 계획을 수립한다.
- Gap 분석: 매핑되지 않은 구성 요소를 식별하여 추가 기능 개발이 필요한지 판단한다.
- 구현 지원: 매핑된 플랫폼 기능을 활용하여 각 구성 요소를 구현하도록 지원한다.

4.2 이용자 활용

- 구성 요소 이해: ISO 23053 구성 요소의 의미와 역할을 이해한다.
- 플랫폼 기능 활용: 매핑된 플랫폼 기능을 활용하여 각 구성 요소를 구현한다.
- 빠른 이동: "이동" 버튼을 클릭하여 관련 플랫폼 기능으로 빠르게 이동한다.

5. 주의사항

5.1 매핑의 정확성

- 각 ISO 23053 구성 요소가 적절한 플랫폼 모듈과 연결되어 있는지 확인해야 한다.
- 매핑이 부정확한 경우, 추가 기능 개발이 필요할 수 있다.

5.2 기능 활용

- 매핑된 플랫폼 기능을 적극적으로 활용하여 ISO 23053 표준 준수를 달성해야 한다.
- 플랫폼 기능의 한계를 이해하고 필요시 추가 도구를 활용한다.

5.3 Gap 분석

- 매핑되지 않은 구성 요소가 있는 경우, Gap 분석을 수행하여 추가 기능 개발 계획을 수립한다.

6. 참고 자료

- ISO/IEC 23053:2022 - Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)
- ISO 23053 ML 대시보드 가이드
- ML 시스템 프레임워크 구현 체크리스트 가이드
- ML 파이프라인 아키텍처 가이드

7. 문의 및 지원

구성 요소 매핑 사용 과정에서 문의 사항이 있으시면 컨설턴트에게 문의하시기 바랍니다.

ML 시스템 프레임워크 구현 체크리스트 가이드

개요

본 문서는 ML 시스템 프레임워크 구현 체크리스트 워크스페이스의 모든 설정 항목에 대한 구체적인 설명을 제공합니다. ISO 23053 기반 ML 시스템 프레임워크 구현을 위한 체크리스트로, 컨설턴트와 이용자가 이해하고 활용해야 하는 모든 내용을 포함합니다.

1. 워크스페이스 접근 및 화면 구성

1.1 접근 방법

위치: 사이드바 메뉴 → "국제표준 거버넌스" → "ISO 23053 ML" → "구현 체크리스트"
접근 경로:
1. 프로젝트를 선택한 후 사이드바에서 "국제표준 거버넌스" 섹션을 확장한다.
2. "ISO 23053 ML" 메뉴를 클릭한다.
3. "구현 체크리스트" 링크를 클릭한다.

주의사항:
- 프로젝트를 먼저 선택한 후 접근하는 것을 권장한다.
- 체크리스트 항목을 완료하기 전에 관련 자료를 준비하는 것이 좋다.

2. 데이터 계층 (Data Layer)

2.1 데이터 수집 파이프라인 구축

의미:
배치/스트리밍 수집 인프라 구성

구현 내용:
- 배치 수집: 주기적으로 대량의 데이터를 수집하는 파이프라인 구축
- 스트리밍 수집: 실시간으로 연속적인 데이터를 수집하는 파이프라인 구축
- 데이터 소스 연동: 다양한 데이터 소스(DB, API, 파일 등)와의 연동

컨설턴트 가이드:
- 데이터 소스의 특성에 따라 적절한 수집 방식을 선택한다.
- 수집 파이프라인을 자동화하여 효율성을 높인다.

이용자 가이드:
- 데이터 수집 파이프라인이 구축되어 있는지 확인한다.

- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

2.2 데이터 품질 검증 프로세스

의미:
완전성, 일관성, 정확성 검증 자동화

구현 내용:
- 완전성 검증: 결측값 확인 및 처리 자동화
- 일관성 검증: 데이터 간 일관성 확인 자동화
- 정확성 검증: 데이터의 정확성 확인 자동화
- 품질 지표 정의: 데이터 품질 지표 정의 및 모니터링

컨설턴트 가이드:
- 데이터 품질 검증 프로세스를 자동화하여 품질을 보장한다.
- 품질 지표를 정의하고 임계값을 설정한다.

이용자 가이드:
- 데이터 품질 검증 프로세스가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

2.3 특성 스토어 구현

의미:
온/오프라인 특성 저장소 구축

구현 내용:
- 온라인 특성 스토어: 실시간 추론에 필요한 특성 저장소
- 오프라인 특성 스토어: 배치 추론 및 훈련에 필요한 특성 저장소
- 특성 버전 관리: 특성의 버전 관리 및 추적
- 특성 검색: 특성 검색 및 재사용 기능

컨설턴트 가이드:
- 특성 스토어를 구축하여 특성의 재사용성을 높인다.
- 온/오프라인 특성 스토어를 통합하여 일관성을 보장한다.

이용자 가이드:
- 특성 스토어가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

2.4 데이터 버전 관리

의미:
데이터셋 버전 추적 및 롤백 기능

구현 내용:

- 데이터셋 버전 관리: 데이터셋의 버전 추적 및 관리
- 변경 이력: 데이터셋 변경 내역 추적
- 롤백 기능: 이전 버전으로 데이터셋 복원
- 버전 비교: 데이터셋 버전 간 비교 기능

컨설턴트 가이드:
- 데이터 버전 관리를 통해 실험의 재현성을 보장한다.
- 버전 관리 시스템을 구축하여 데이터 변경을 추적한다.

이용자 가이드:
- 데이터 버전 관리 시스템이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

2.5 메타데이터 카탈로그

의미:
데이터 자산 검색 및 계보 추적

구현 내용:
- 메타데이터 저장: 데이터 자산의 메타데이터 저장
- 검색 기능: 메타데이터 기반 데이터 검색
- 계보 추적: 데이터의 출처 및 변환 과정 추적
- 데이터 맵: 데이터 자산 간의 관계 시각화

컨설턴트 가이드:
- 메타데이터 카탈로그를 구축하여 데이터 검색 및 계보 추적을 지원한다.
- 메타데이터를 표준화하여 일관성을 보장한다.

이용자 가이드:
- 메타데이터 카탈로그가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

2.6 데이터 보안/프라이버시

의미:
암호화, 접근 제어, 익명화

구현 내용:
- 암호화: 저장 및 전송 중 데이터 암호화
- 접근 제어: 역할 기반 접근 제어 (RBAC)
- 익명화: 개인정보 익명화 및 가명화
- 감사 로깅: 데이터 접근 및 사용 로그 기록

컨설턴트 가이드:
- 데이터 보안 및 프라이버시를 보장하는 체계를 구축한다.
- 규제 요구사항을 준수하는 보안 정책을 수립한다.

이용자 가이드:
- 데이터 보안/프라이버시 체계가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

3. 모델 개발 (Model Development)

3.1 실험 추적 시스템

의미:
MLflow, W&B 등 실험 관리 도구

구현 내용:
- 실험 추적: 실험 파라미터, 메트릭, 아티팩트 추적
- 실험 비교: 여러 실험 결과 비교 및 분석
- 실험 재현: 실험 재현을 위한 환경 및 파라미터 저장
- 실험 공유: 실험 결과 공유 및 협업

컨설턴트 가이드:
- 실험 추적 시스템을 구축하여 실험을 체계적으로 관리한다.
- MLflow, W&B 등의 도구를 활용하여 실험 추적을 자동화한다.

이용자 가이드:
- 실험 추적 시스템이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

3.2 모델 레지스트리

의미:
모델 버전 관리 및 메타데이터 저장

구현 내용:
- 모델 버전 관리: 모델의 버전 추적 및 관리
- 메타데이터 저장: 모델 성능, 파라미터, 학습 데이터 등 메타데이터 저장
- 모델 검색: 모델 검색 및 필터링 기능
- 모델 승인: 모델 배포 전 승인 프로세스

컨설턴트 가이드:
- 모델 레지스트리를 구축하여 모델을 체계적으로 관리한다.
- 모델 메타데이터를 표준화하여 일관성을 보장한다.

이용자 가이드:
- 모델 레지스트리가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

3.3 자동화된 테스트

의미:
단위/통합/성능 테스트 파이프라인

구현 내용:
- 단위 테스트: 모델 구성 요소의 단위 테스트
- 통합 테스트: 모델과 시스템의 통합 테스트
- 성능 테스트: 모델 성능 및 처리량 테스트
- 회귀 테스트: 모델 변경 시 기존 기능 테스트

컨설턴트 가이드:
- 자동화된 테스트 파이프라인을 구축하여 모델 품질을 보장한다.
- CI/CD 파이프라인에 테스트를 통합한다.

이용자 가이드:
- 자동화된 테스트 파이프라인이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

3.4 모델 문서화 (모델 카드)

의미:
모델 성능, 제한사항, 사용 가이드

구현 내용:
- 모델 성능: 모델의 성능 지표 및 평가 결과
- 제한사항: 모델의 제한사항 및 사용 시 주의사항
- 사용 가이드: 모델 사용 방법 및 예시
- 편향 정보: 모델의 편향 정보 및 완화 방법

컨설턴트 가이드:
- 모델 카드를 작성하여 모델 정보를 체계적으로 문서화한다.
- ISO 42001의 모델 카드 표준을 참고하여 문서화한다.

이용자 가이드:
- 모델 문서화가 이루어지고 있는지 확인한다.
- 이루어지고 있다면 체크하고, 없다면 문서화 계획을 수립한다.

3.5 하이퍼파라미터 최적화

의미:
자동 튜닝 파이프라인 구축

구현 내용:
- 그리드 서치: 하이퍼파라미터 조합을 체계적으로 탐색

- 랜덤 서치: 무작위로 하이퍼파라미터 조합 탐색
- 베이지안 최적화: 이전 결과를 활용하여 효율적으로 탐색
- AutoML: 자동으로 최적의 모델 및 하이퍼파라미터 탐색

컨설턴트 가이드:
- 하이퍼파라미터 최적화 파이프라인을 자동화하여 효율성을 높인다.
- 최적화 알고리즘을 선택하여 탐색 효율성을 최적화한다.

이용자 가이드:
- 하이퍼파라미터 최적화 파이프라인이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

3.6 모델 해석성 도구

의미:
SHAP, LIME 등 설명가능 AI

구현 내용:
- SHAP: 모델 예측에 대한 기여도 분석
- LIME: 지역적 모델 해석
- 특성 중요도: 특성의 중요도 분석
- 의사결정 트리: 모델의 의사결정 과정 시각화

컨설턴트 가이드:
- 모델 해석성 도구를 구축하여 모델의 설명가능성을 향상시킨다.
- 다양한 해석 도구를 활용하여 모델을 다각도로 분석한다.

이용자 가이드:
- 모델 해석성 도구가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

4. 모델 배포 (Model Deployment)

4.1 CI/CD 파이프라인

의미:
자동 빌드, 테스트, 배포 파이프라인

구현 내용:
- 자동 빌드: 코드 변경 시 자동으로 모델 빌드
- 자동 테스트: 빌드 후 자동으로 테스트 실행
- 자동 배포: 테스트 통과 시 자동으로 배포
- 롤백: 배포 실패 시 자동 롤백

컨설턴트 가이드:
- CI/CD 파이프라인을 구축하여 배포 프로세스를 자동화한다.
- 파이프라인의 각 단계에 품질 게이트를 설정한다.

이용자 가이드:
- CI/CD 파이프라인이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

4.2 모델 서빙 인프라

의미:
TensorFlow Serving, TorchServe, Triton 등

구현 내용:
- TensorFlow Serving: TensorFlow 모델 서빙
- TorchServe: PyTorch 모델 서빙
- Triton: 다양한 프레임워크 모델 서빙
- API 게이트웨이: API 요청 라우팅 및 관리

컨설턴트 가이드:
- 모델 프레임워크에 따라 적절한 서빙 인프라를 선택한다.
- 서빙 인프라를 확장 가능하도록 설계한다.

이용자 가이드:
- 모델 서빙 인프라가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

4.3 배포 전략 (카나리, A/B)

의미:
점진적 배포 및 실험 인프라

구현 내용:
- 카나리 배포: 소규모 트래픽으로 새 버전 테스트
- A/B 테스트: 여러 버전을 동시에 테스트하여 성능 비교
- 블루-그린 배포: 새 버전과 기존 버전을 동시에 운영
- 트래픽 분산: 트래픽을 여러 버전에 분산

컨설턴트 가이드:
- 배포 전략을 수립하여 안전하게 모델을 배포한다.
- 실험 인프라를 구축하여 배포 전 성능을 검증한다.

이용자 가이드:
- 배포 전략이 수립되어 있는지 확인한다.
- 수립되어 있다면 체크하고, 없다면 수립 계획을 세운다.

4.4 롤백 메커니즘

의미:
자동/수동 롤백 프로세스

구현 내용:
- 자동 롤백: 성능 저하 시 자동으로 이전 버전으로 복원
- 수동 롤백: 관리자가 수동으로 이전 버전으로 복원
- 롤백 기준: 롤백을 수행할 조건 정의
- 롤백 모니터링: 롤백 후 모델 성능 모니터링

컨설턴트 가이드:
- 롤백 메커니즘을 구축하여 배포 위험을 최소화한다.
- 롤백 기준을 명확히 정의하여 자동 롤백을 구현한다.

이용자 가이드:
- 롤백 메커니즘이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

4.5 컨테이너 오케스트레이션

의미:
Kubernetes 기반 배포 관리

구현 내용:
- Kubernetes 클러스터: 컨테이너 오케스트레이션 플랫폼
- 자동 스케일링: 트래픽에 따라 자동으로 스케일링
- 서비스 디스커버리: 서비스 자동 발견 및 라우팅
- 헬스 체크: 컨테이너 상태 모니터링 및 자동 복구

컨설턴트 가이드:
- Kubernetes를 활용하여 컨테이너를 효율적으로 관리한다.
- 자동 스케일링을 설정하여 리소스를 효율적으로 활용한다.

이용자 가이드:
- 컨테이너 오케스트레이션이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

4.6 API 게이트웨이/로드밸런서

의미:
트래픽 관리 및 분산

구현 내용:
- API 게이트웨이: API 요청 라우팅 및 관리
- 로드밸런서: 트래픽을 여러 서버에 분산

- 인증/인가: API 요청 인증 및 인가
- 레이트 리미팅: API 요청 속도 제한

컨설턴트 가이드:
- API 게이트웨이 및 로드밸런서를 구축하여 트래픽을 효율적으로 관리한다.
- 인증 및 보안 기능을 통합한다.

이용자 가이드:
- API 게이트웨이/로드밸런서가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5. 모델 운영 (Model Operations)

5.1 성능 모니터링 대시보드

의미:
지연시간, 처리량, 정확도 추적

구현 내용:
- 지연시간 모니터링: 요청 처리 시간 추적 및 시각화
- 처리량 모니터링: 초당 처리 요청 수 추적 및 시각화
- 정확도 모니터링: 모델 예측 정확도 추적 및 시각화
- 리소스 모니터링: CPU, 메모리, GPU 사용량 추적 및 시각화

컨설턴트 가이드:
- 성능 모니터링 대시보드를 구축하여 모델 성능을 지속적으로 추적한다.
- 주요 지표를 시각화하여 한눈에 파악할 수 있도록 한다.

이용자 가이드:
- 성능 모니터링 대시보드가 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5.2 드리프트 감지 시스템

의미:
데이터/모델 드리프트 자동 탐지

구현 내용:
- 데이터 드리프트: 입력 데이터 분포의 변화 감지
- 모델 드리프트: 모델 예측 분포의 변화 감지
- 개념 드리프트: 목표 변수의 분포 변화 감지
- 특성 드리프트: 특성 분포의 변화 감지

컨설턴트 가이드:
- 드리프트 감지 시스템을 구축하여 모델 성능 저하를 사전에 감지한다.
- 드리프트 임계값을 설정하여 알림을 자동화한다.

이용자 가이드:
- 드리프트 감지 시스템이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5.3 알림 시스템

의미:
이상 탐지 및 에스컬레이션

구현 내용:
- 이상 탐지: 성능 저하, 드리프트 등 이상 상황 자동 탐지
- 알림 전송: 이메일, 슬랙 등 다양한 채널로 알림 전송
- 에스컬레이션: 심각한 문제 발생 시 상위 관리자에게 알림
- 알림 규칙: 알림 조건 및 규칙 정의

컨설턴트 가이드:
- 알림 시스템을 구축하여 문제를 신속하게 대응한다.
- 알림 규칙을 정의하여 적절한 알림을 전송한다.

이용자 가이드:
- 알림 시스템이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5.4 재훈련 파이프라인

의미:
자동화된 모델 업데이트

구현 내용:
- 재훈련 트리거: 일정, 성능, 드리프트 기반 재훈련 트리거
- 자동 재훈련: 재훈련 트리거 시 자동으로 모델 재훈련
- 재훈련 검증: 재훈련된 모델의 성능 검증
- 자동 배포: 검증 통과 시 자동으로 배포

컨설턴트 가이드:
- 재훈련 파이프라인을 자동화하여 모델 성능을 유지한다.
- 재훈련 트리거 조건을 정의하여 적절한 시점에 재훈련한다.

이용자 가이드:
- 재훈련 파이프라인이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5.5 감사 로깅

의미:
예측/사용/변경 기록

구현 내용:
- 예측 로깅: 모델 예측 결과 및 입력 데이터 로깅
- 사용 로깅: 모델 사용 내역 및 사용자 정보 로깅
- 변경 로깅: 모델 변경 내역 및 변경자 정보 로깅
- 로그 분석: 로그 분석 및 리포트 생성

컨설턴트 가이드:
- 감사 로깅을 구축하여 모델 사용 내역을 추적한다.
- 로그를 보안 저장소에 저장하여 무결성을 보장한다.

이용자 가이드:
- 감사 로깅이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

5.6 비용 모니터링

의미:
인프라/추론 비용 추적

구현 내용:
- 인프라 비용: 컴퓨팅, 스토리지, 네트워크 비용 추적
- 추론 비용: 모델 추론에 소요되는 비용 추적
- 비용 분석: 비용 분석 및 리포트 생성
- 비용 최적화: 비용 최적화 방안 제시

컨설턴트 가이드:
- 비용 모니터링을 구축하여 ML 시스템의 비용을 추적한다.
- 비용 분석을 통해 비용 최적화 방안을 도출한다.

이용자 가이드:
- 비용 모니터링이 구축되어 있는지 확인한다.
- 구축되어 있다면 체크하고, 없다면 구축 계획을 수립한다.

6. 거버넌스 (Governance)

6.1 접근 제어 정책

의미:
역할 기반 접근 제어 (RBAC)

구현 내용:
- 역할 정의: 역할 및 권한 정의
- 사용자 할당: 사용자에게 역할 할당
- 접근 제어: 리소스 접근 제어
- 권한 검토: 정기적인 권한 검토

컨설턴트 가이드:
- 접근 제어 정책을 수립하여 보안을 강화한다.
- 최소 권한 원칙을 적용하여 접근 권한을 최소화한다.

이용자 가이드:
- 접근 제어 정책이 수립되어 있는지 확인한다.
- 수립되어 있다면 체크하고, 없다면 수립 계획을 세운다.

6.2 규정 준수 검증

의미:
GDPR, AI Act 등 규제 준수

구현 내용:
- 규제 모니터링: 관련 규제 변경 사항 모니터링
- 준수 검증: 규제 준수 여부 검증
- 문서화: 준수 증빙 자료 문서화
- 정기 검토: 정기적인 준수 검토

컨설턴트 가이드:
- 규정 준수 검증 프로세스를 수립하여 규제를 준수한다.
- 관련 규제를 모니터링하여 변경 사항에 대응한다.

이용자 가이드:
- 규정 준수 검증 프로세스가 수립되어 있는지 확인한다.
- 수립되어 있다면 체크하고, 없다면 수립 계획을 세운다.

6.3 모델 승인 워크플로우

의미:
배포 전 승인 프로세스

구현 내용:
- 승인 단계: 승인 단계 및 승인자 정의
- 승인 요청: 모델 배포 전 승인 요청
- 승인 검토: 승인자에 의한 모델 검토
- 승인 기록: 승인 내역 기록 및 추적

컨설턴트 가이드:
- 모델 승인 워크플로우를 수립하여 모델 배포를 관리한다.
- 승인 기준을 명확히 정의하여 일관성을 보장한다.

이용자 가이드:
- 모델 승인 워크플로우가 수립되어 있는지 확인한다.
- 수립되어 있다면 체크하고, 없다면 수립 계획을 세운다.

6.4 모델 폐기 정책

의미:
모델 수명주기 종료 관리

구현 내용:
- 폐기 기준: 모델 폐기 기준 정의
- 폐기 프로세스: 모델 폐기 프로세스 정의
- 데이터 보관: 폐기된 모델의 데이터 보관 정책
- 문서화: 폐기 내역 문서화

컨설턴트 가이드:
- 모델 폐기 정책을 수립하여 모델 수명주기를 관리한다.
- 폐기 기준을 명확히 정의하여 일관성을 보장한다.

이용자 가이드:
- 모델 폐기 정책이 수립되어 있는지 확인한다.
- 수립되어 있다면 체크하고, 없다면 수립 계획을 세운다.

7. 체크리스트 저장 및 내보내기

7.1 체크리스트 저장

- 기능: "체크리스트 저장" 버튼을 클릭하면 현재 체크리스트 상태를 저장한다.
- 저장 내용: 각 항목의 체크 상태

컨설턴트 가이드:
체크리스트를 정기적으로 저장하여 진행 상황을 추적한다.

이용자 가이드:
체크리스트를 완료한 후 저장하여 진행 상황을 보존한다.

7.2 체크리스트 내보내기

- 기능: "체크리스트 내보내기" 버튼을 클릭하면 현재 체크리스트를 파일로 내보낸다.
- 내보내기 형식: PDF, Excel 등
-

컨설턴트 가이드:
체크리스트를 내보내어 보고서나 문서에 활용한다.

이용자 가이드:
체크리스트를 내보내어 공유하거나 보관한다.

8. 활용 가이드

8.1 컨설턴트 활용

- 구현 계획 수립: 체크리스트를 참고하여 ML 시스템 프레임워크 구현 계획을 수립한다.
- 진행 상황 추적: 체크리스트를 정기적으로 확인하여 구현 진행 상황을 추적한다.
- Gap 분석: 미완료 항목을 식별하여 추가 구현이 필요한 항목을 파악한다.

8.2 이용자 활용

- 구현 현황 확인: 체크리스트를 통해 현재 구현 현황을 확인한다.
- 구현 계획 수립: 미완료 항목을 확인하고 구현 계획을 수립한다.
- 진행 상황 추적: 체크리스트를 정기적으로 업데이트하여 진행 상황을 추적한다.

9. 주의사항

9.1 정확한 평가

- 체크리스트 항목에 대해 조직의 실제 구현 상태를 정확히 반영해야 한다.
- 과대평가나 과소평가를 피하고 정확한 상태를 반영한다.

9.2 우선순위 관리

- 모든 항목을 한 번에 구현하기보다는 우선순위에 따라 단계적으로 구현한다.
- 필수 항목을 먼저 구현하고, 선택적 항목은 점진적으로 추가한다.

9.3 정기적 업데이트

- 구현 현황이 변경되면 체크리스트를 정기적으로 업데이트해야 한다.
- 권장 업데이트 주기: 월별 또는 분기별

10. 참고 자료

- ISO/IEC 23053:2022 - Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)
- ISO 23053 ML 대시보드 가이드
- ISO 23053 구성 요소 매핑 가이드
- ML 파이프라인 아키텍처 가이드

11. 문의 및 지원

구현 체크리스트 사용 과정에서 문의사항이 있으시면 컨설턴트에게 문의하시기 바랍니다.

ML 파이프라인 아키텍처 가이드

개요

본 문서는 ML 파이프라인 아키텍처 워크스페이스의 모든 항목에 대한 구체적인 설명을 제공합니다. ISO 23053 기반 ML 시스템의 파이프라인 아키텍처를 시각화하고 관리하는 워크스페이스로, 컨설턴트와 이용자가 이해하고 활용해야 하는 모든 내용을 포함합니다.

1. 워크스페이스 접근 및 화면 구성

1.1 접근 방법

위치: 사이드바 메뉴 → "국제표준 거버넌스" → "ISO 23053 ML" → "ML 파이프라인"

접근 경로:
1. 프로젝트를 선택한 후 사이드바에서 "국제표준 거버넌스" 섹션을 확장한다.
2. "ISO 23053 ML" 메뉴를 클릭한다.
3. "ML 파이프라인" 링크를 클릭한다.

주의사항:
- 프로젝트를 먼저 선택한 후 접근하는 것을 권장한다.
- 본 워크스페이스는 ML 파이프라인의 전체 흐름을 보여주므로, 각 파이프라인의 역할을 이해한 후 확인하는 것이 더 유용하다.

2. 데이터 파이프라인 (Data Pipeline)

2.1 목적

데이터 소스부터 특성 스토어까지의 데이터 처리 흐름을 시각화하여, 데이터 파이프라인의 각 단계를 명확히 이해할 수 있도록 한다.

2.2 파이프라인 단계

2.2.1 데이터 소스

의미: 데이터의 출처
- 유형: DB, API, 파일 등

컨설턴트 가이드:
다양한 데이터 소스를 고려하여 파이프라인을 설계한다.

이용자 가이드:
데이터 소스의 특성을 이해하고 적절한 수집 방식을 선택한다.

2.2.2 데이터 수집

의미: 데이터 소스로부터 데이터를 수집하는 단계
- 유형: 배치/스트리밍

컨설턴트 가이드:
데이터 소스의 특성에 따라 배치 또는 스트리밍 수집 방식을 선택한다.

이용자 가이드:
데이터 수집 방식을 이해하고 활용한다.

2.2.3 데이터 처리

의미: 데이터를 정제하고 변환하는 단계
- 작업: 정제/변환

컨설턴트 가이드:
데이터 처리 파이프라인을 자동화하여 효율성을 높인다.

이용자 가이드:
데이터 처리 프로세스를 이해하고 활용한다.

2.2.4 특성 엔지니어링

의미: 모델에 필요한 특성을 추출하고 변환하는 단계
- 작업: 추출/변환

컨설턴트 가이드:
특성 엔지니어링을 자동화하여 재사용성을 높인다.

이용자 가이드:
특성 엔지니어링 프로세스를 이해하고 활용한다.

2.2.5 특성 스토어

의미: 특성을 저장하고 제공하는 저장소
- 유형: 온/오프라인

컨설턴트 가이드:
특성 스토어를 구축하여 특성의 재사용성을 높인다.

이용자 가이드:
특성 스토어를 활용하여 일관된 특성을 사용한다.

2.3 파이프라인 유형

2.3.1 배치 파이프라인

특징:
- 대용량 데이터 처리
- 주기적 실행
- 높은 처리량
- 도구: Spark, Airflow, dbt
- 활용 사례: 일일 리포트 생성, 대량 데이터 분석

컨설턴트 가이드:
대용량 데이터를 주기적으로 처리해야 하는 경우 배치 파이프라인을 활용한다.

이용자 가이드:
배치 파이프라인의 특징을 이해하고 활용한다.

2.3.2 스트리밍 파이프라인

특징:
- 실시간 처리
- 낮은 지연시간
- 연속적 흐름
- 도구: Kafka, Flink
- 활용 사례: 실시간 추천, 이상 탐지

컨설턴트 가이드:
실시간 처리가 필요한 경우 스트리밍 파이프라인을 활용한다.

이용자 가이드:
스트리밍 파이프라인의 특징을 이해하고 활용한다.

2.3.3 하이브리드 (Lambda)

특징:
- 배치 + 실시간 조합
- 정확성 + 신속성
- 서빙 레이어 통합
- 유연한 쿼리
- 활용 사례: 실시간 대시보드, 배치 분석

컨설턴트 가이드:
정확성과 신속성을 모두 필요로 하는 경우 하이브리드 파이프라인을 활용한다.

이용자 가이드:
하이브리드 파이프라인의 특징을 이해하고 활용한다.

3. 훈련 파이프라인 (Training Pipeline)

3.1 목적

데이터 로드부터 모델 저장까지의 모델 훈련 흐름을 시각화하여, 훈련 파이프라인의 각 단계를 명확히 이해할 수 있도록 한다.

3.2 파이프라인 단계

3.2.1 데이터 로드

의미: 데이터를 메모리로 로드하는 단계
- 구현: 효율적 배치 생성, 데이터 증강
- 도구: PyTorch DataLoader, tf.data

컨설턴트 가이드:
데이터 로더를 최적화하여 훈련 효율성을 높인다.

이용자 가이드:
데이터 로드 방식을 이해하고 활용한다.

3.2.2 데이터 분할

의미: 데이터를 훈련/검증/테스트 세트로 분할하는 단계
- 구현: 무작위 분할, 계층적 분할

컨설턴트 가이드:
데이터 분할 방식을 선택하여 모델 성능을 최적화한다.

이용자 가이드:
데이터 분할 방식을 이해하고 활용한다.

3.2.3 모델 선택

의미: 모델 아키텍처를 선택하는 단계
- 구현: 아키텍처 설계, 레이어 구성
- 도구: PyTorch, TensorFlow, Keras

컨설턴트 가이드:
문제의 특성에 따라 적절한 모델 아키텍처를 선택한다.

이용자 가이드:
모델 선택 방식을 이해하고 활용한다.

3.2.4 훈련 루프

의미: 모델을 학습시키는 단계
- 구현: 옵티마이저 설정, 학습률 스케줄링
- 도구: torch.optim, tf.optimizers

컨설턴트 가이드:
훈련 루프를 최적화하여 모델 성능을 향상시킨다.

이용자 가이드:
훈련 루프를 이해하고 활용한다.

3.2.5 검증

의미: 모델 성능을 검증하는 단계
- 구현: 검증 세트로 성능 평가

컨설턴트 가이드:
검증을 통해 모델의 일반화 성능을 확인한다.

이용자 가이드:
검증 방식을 이해하고 활용한다.

3.2.6 모델 저장

의미: 학습된 모델을 저장하는 단계
- 구현: 모델 가중치 및 메타데이터 저장

컨설턴트 가이드:
모델을 체계적으로 저장하여 재사용성을 높인다.

이용자 가이드:
모델 저장 방식을 이해하고 활용한다.

3.3 훈련 파이프라인 구성 요소

3.3.1 데이터 로더

의미: 효율적 배치 생성, 데이터 증강
- 도구: PyTorch DataLoader, tf.data

컨설턴트 가이드:
데이터 로더를 최적화하여 훈련 효율성을 높인다.

이용자 가이드:
데이터 로더를 이해하고 활용한다.

3.3.2 모델 정의

의미: 아키텍처 설계, 레이어 구성
- 도구: PyTorch, TensorFlow, Keras

컨설턴트 가이드:
모델 아키텍처를 설계하여 문제를 해결한다.

이용자 가이드:
모델 정의 방식을 이해하고 활용한다.

3.3.3 옵티마이저

의미: SGD, Adam, 학습률 스케줄링
- 도구: torch.optim, tf.optimizers

컨설턴트 가이드:
옵티마이저를 선택하여 모델 성능을 최적화한다.

이용자 가이드:
옵티마이저를 이해하고 활용한다.

3.3.4 실험 추적

의미: 메트릭 로깅, 하이퍼파라미터 기록
- 도구: MLflow, W&B, Neptune

컨설턴트 가이드:
실험 추적을 통해 실험을 체계적으로 관리한다.

이용자 가이드:
실험 추적 도구를 이해하고 활용한다.

3.3.5 분산 훈련

의미: 데이터/모델 병렬화
- 도구: Horovod, DeepSpeed, Ray

컨설턴트 가이드:
분산 훈련을 통해 대규모 모델을 효율적으로 훈련한다.

이용자 가이드:
분산 훈련 방식을 이해하고 활용한다.

4. 추론 파이프라인 (Inference Pipeline)

4.1 목적

요청부터 응답까지의 모델 추론 흐름을 시각화하여, 추론 파이프라인의 각 단계를 명확히 이해할 수 있도록 한다.

4.2 파이프라인 단계

4.2.1 요청

의미: 사용자 또는 시스템으로부터 추론 요청을 받는 단계

컨설턴트 가이드:
요청 형식을 표준화하여 일관성을 보장한다.

이용자 가이드:
요청 방식을 이해하고 활용한다.

4.2.2 전처리

의미: 입력 데이터를 모델에 맞게 전처리하는 단계

컨설턴트 가이드:
전처리 로직을 표준화하여 일관성을 보장한다.

이용자 가이드:
전처리 방식을 이해하고 활용한다.

4.2.3 특성 조회

의미: 특성 스토어에서 필요한 특성을 조회하는 단계

컨설턴트 가이드:
특성 스토어를 활용하여 일관된 특성을 사용한다.

이용자 가이드:
특성 조회 방식을 이해하고 활용한다.

4.2.4 추론

의미: 모델을 사용하여 예측을 수행하는 단계

컨설턴트 가이드:
추론 성능을 최적화하여 지연시간을 최소화한다.

이용자 가이드:
추론 방식을 이해하고 활용한다.

4.2.5 후처리

의미: 모델 출력을 사용자에게 제공하기 위해 후처리하는 단계

컨설턴트 가이드:
후처리 로직을 표준화하여 일관성을 보장한다.

이용자 가이드:
후처리 방식을 이해하고 활용한다.

4.2.6 응답

의미: 처리 결과를 사용자 또는 시스템에 반환하는 단계

컨설턴트 가이드:
응답 형식을 표준화하여 일관성을 보장한다.

이용자 가이드:
응답 방식을 이해하고 활용한다.

4.3 추론 유형

4.3.1 실시간 추론

특징:
- 지연시간: < 100ms
- REST API, gRPC
- 활용 사례: 추천, 사기탐지

컨설턴트 가이드:
낮은 지연시간이 필요한 경우 실시간 추론을 활용한다.

이용자 가이드:
실시간 추론의 특징을 이해하고 활용한다.

4.3.2 배치 추론

특징:
- 높은 처리량
- 주기적/이벤트 트리거
- 활용 사례: 리포트, 세그먼트

컨설턴트 가이드:
대량의 데이터를 처리해야 하는 경우 배치 추론을 활용한다.

이용자 가이드:
배치 추론의 특징을 이해하고 활용한다.

4.3.3 스트리밍 추론

특징:
- 준실시간 응답
- 이벤트 기반
- 활용 사례: IoT, 로그 분석

컨설턴트 가이드:
연속적인 데이터 스트림을 처리해야 하는 경우 스트리밍 추론을 활용한다.

이용자 가이드:
스트리밍 추론의 특징을 이해하고 활용한다.

4.3.4 엣지 추론

특징:
- 디바이스에서 실행
- 경량 모델 필요
- 활용 사례: 모바일, IoT

컨설턴트 가이드:
네트워크 지연을 최소화해야 하는 경우 엣지 추론을 활용한다.

이용자 가이드:
엣지 추론의 특징을 이해하고 활용한다.

5. 모델 서빙 인프라

5.1 모델 서버

의미: 모델을 서비스로 제공하는 서버
- 기능: 모델 로딩, 요청 처리, 응답 반환

- 도구/서비스: TensorFlow Serving, TorchServe, Triton

컨설턴트 가이드:
모델 프레임워크에 따라 적절한 모델 서버를 선택한다.

이용자 가이드:
모델 서버를 이해하고 활용한다.

5.2 API 게이트웨이

의미: API 요청을 라우팅하고 관리하는 게이트웨이
- 기능: 요청 라우팅, 인증/인가, 레이트 리미팅
- 도구/서비스: Kong, AWS API Gateway, Azure API Management

컨설턴트 가이드:
API 게이트웨이를 구축하여 API를 효율적으로 관리한다.

이용자 가이드:
API 게이트웨이를 이해하고 활용한다.

5.3 로드밸런서

의미: 트래픽을 여러 서버에 분산하는 로드밸런서
- 기능: 트래픽 분산, 헬스 체크, 자동 스케일링
- 도구/서비스: NGINX, HAProxy, AWS ELB

컨설턴트 가이드:
로드밸런서를 구축하여 트래픽을 효율적으로 분산한다.

이용자 가이드:
로드밸런서를 이해하고 활용한다.

6. 활용 가이드

6.1 컨설턴트 활용

- 파이프라인 설계: 각 파이프라인의 단계를 참고하여 ML 시스템 파이프라인을 설계한다.
- 최적화: 파이프라인의 각 단계를 최적화하여 효율성을 높인다.
- 모니터링: 파이프라인의 각 단계를 모니터링하여 문제를 신속하게 파악한다.

6.2 이용자 활용

- 파이프라인 이해: 각 파이프라인의 흐름을 이해하여 ML 시스템의 동작을 파악한다.
- 문제 해결: 파이프라인의 각 단계를 확인하여 문제가 발생한 단계를 식별한다.

- 최적화: 파이프라인의 병목 지점을 식별하여 최적화한다.

7. 주의사항

7.1 파이프라인 단계 간 의존성

- 각 파이프라인 단계는 이전 단계에 의존하므로, 단계 간 인터페이스를 명확히 정의해야 한다.
- 단계 간 데이터 형식을 표준화하여 일관성을 보장한다.

7.2 성능 최적화

- 파이프라인의 각 단계를 최적화하여 전체 파이프라인의 성능을 향상시킨다.
- 병목 지점을 식별하여 우선적으로 최적화한다.

7.3 모니터링

- 파이프라인의 각 단계를 모니터링하여 문제를 신속하게 파악한다.
- 각 단계의 성능 지표를 추적하여 최적화 방안을 도출한다.

8. 참고 자료

- ISO/IEC 23053:2022 - Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)
- ISO 23053 ML 대시보드 가이드
- ISO 23053 구성 요소 매핑 가이드
- ML 시스템 프레임워크 구현 체크리스트 가이드

9. 문의 및 지원

ML 파이프라인 아키텍처 사용 과정에서 문의사항이 있으시면 컨설턴트에게 문의하시기 바랍니다.