AWS규범적 지침

프로덕션 준비 ML 파이프라인 생성AWS

AWS규범적 지침: 프로덕션 준비 ML 파이프라인 생성AWS

Table of Contents

프로덕션 준비 ML 파이프라인 생성 AWS

요시아 데이비스, 베르디 마치, 음과 송, 바이추안 선, 첸 우, 웨이 이 야프

Data Science 팀AWS전문 서비스

2021년 1월

기계 학습 (ML) 프로젝트는 비즈니스 가치를 제공하고 실제 문제를 해결하기 위해 모델링, 구현 및 생산을 포함하는 상당한 다단계 노력이 필요합니다. 각 단계에서 다양한 대안 및 사용자 지정 옵션을 사용할 수 있으며, 이로 인해 리소스 및 예산의 제약 조건 내에서 생산을 위한 ML 모델을 준비하는 것이 점점 더 어려워지고 있습니다. 지난 몇 년 동안 Amazon Web Services (AWS) 데이터 사이언스 팀은 ML 이니셔티브에 대한 다양한 산업 부문과 협력해 왔습니다. 많은 사람들이 공유하는 불만 사항을 확인했습니다.AWS고객은 조직의 문제와 기술적 과제 모두에서 비롯된 것이며 생산 준비가 가능한 ML 솔루션을 제공하기 위한 최적의 접근 방식을 개발했습니다.

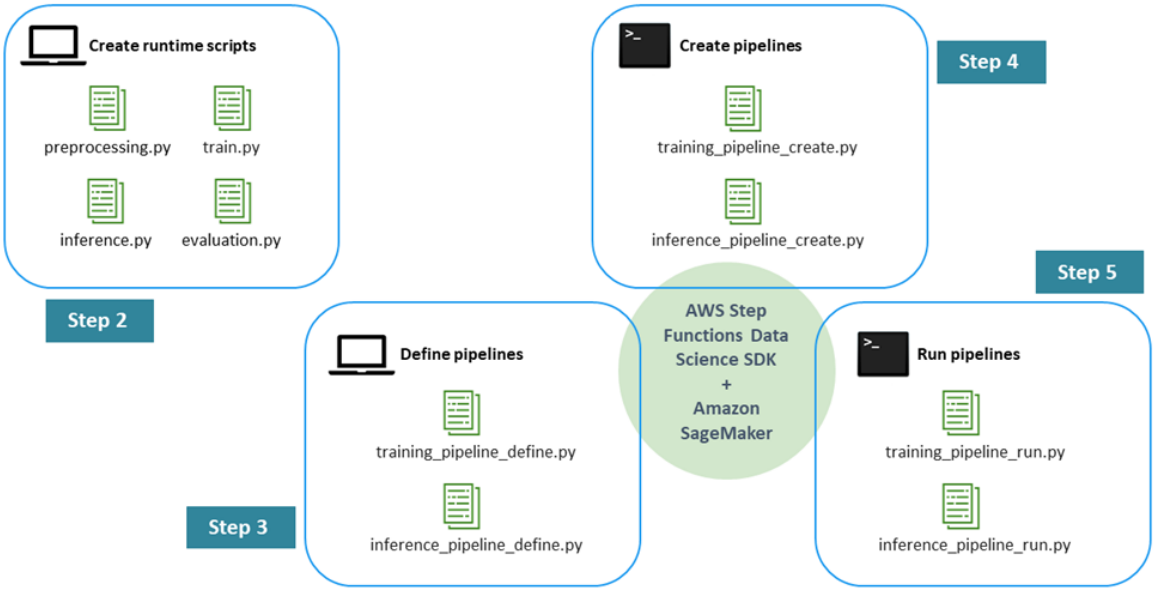
이 가이드는 ML 파이프라인 구현에 관여하는 데이터 과학자 및 ML 엔지니어를 위한 것입니다. 프로덕션 준비 ML 파이프라인을 제공하기 위한 당사의 접근 방식을 설명합니다. 이 가이드에서는 ML 모델을 대화형 방식으로 (개발 중) 실행하는 것에서 ML 사용 사례에 대한 파이프라인의 일부로 배포하는 방법 (프로덕션 중)으로 전환하는 방법에 대해 설명합니다. 이를 위해 예제 템플릿 집합도 개발했습니다 (ML Max 프로젝트 프로젝트사용자 지정 ML 솔루션을 프로덕션에 신속하게 제공하므로 설계 선택을 너무 많이 수행하지 않고도 신속하게 시작할 수 있습니다.

개요

프로덕션 준비가 완료된 ML 파이프라인을 생성하기 위한 프로세스는 다음 단계로 구성됩니다.

- 1단계 (p. 3). EDA 수행 및 초기 모델 개발— 데이터 과학자는 Amazon Simple Storage Service (Amazon S3) 에서 원시 데이터를 제공하고, EDA (탐색 데이터 분석) 를 수행하고, 초기 ML 모델을 개발하고, 추론 성능을 평가합니다. Jupyter 노트북을 통해 대화식으로 이러한 활동을 수행할 수 있습니다.
- 2단계 (p. 4). 런타임 스크립트 만들기— 모델을 런타임 Python 스크립트와 통합하여 ML 프레임워크 (이 경우 Amazon SageMaker) 로 관리하고 프로비저닝할 수 있습니다. 독립 실행형 모델의 대화식 개발에서 프로덕션으로 전환하는 첫 번째 단계입니다. 특히 전처리, 평가, 교육 및 추론에 대한 로직을 별도로 정의합니다.
- 3단계 (p. 7). 파이프라인 정의— 파이프라인의 각 단계에 대한 입력 및 출력 자리 표시자를 정의합니다. 이러한 값에 대한 구체적인 값은 나중에 런타임 중에 제공됩니다 (5단계). 교육, 추론, 교차 검증 및 역테스트를 위한 파이프라인에 중점을 둡니다.
- 단계 4 (p. 9). 파이프라인 생성— 다음과 같은 기본 인프라를 생성합니다.AWS Step Functions자동 (거의 한 번의 클릭) 방식으로 상태 머신 인스턴스AWS CloudFormation.
- 단계 5 (p. 11). 파이프라인 실행— 4단계에서 정의된 파이프라인을 실행합니다. 또한 3단계에서 정의한 입력/출력 자리 표시자에 대한 구체적인 값을 채우기 위해 메타데이터 및 데이터 또는 데이터 위치를 준비합니다. 여기에는 2단계에서 정의된 런타임 스크립트와 모델 하이퍼파라미터가 포함됩니다.
- 단계 6 (p. 12). 파이프라인 확장— 지속적 통합 및 지속적 배포 (CI/CD) 프로세스, 자동화된 재교육, 예정된 추론 및 이와 유사한 파이프라인 확장을 구현합니다.

다음 다이어그램에서는 이 프로세스의 주요 단계를 보여 줍니다.



Steps for creating the training and inference pipeline

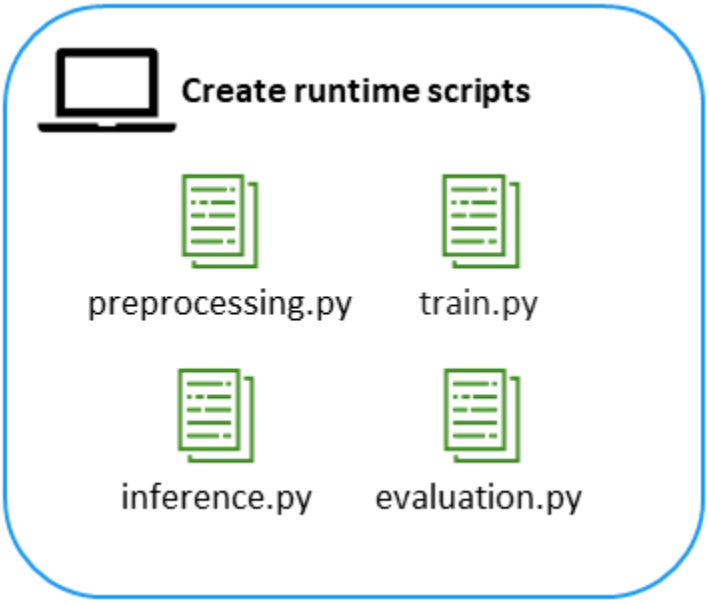
1단계. EDA 수행 및 초기 모델 개발

이 단계에서는 데이터 과학자가 ML 사용 사례와 데이터를 이해하기 위해 탐색적 데이터 분석 (EDA) 을 수행합니다. 그런 다음 ML 모델 (예: 분류 및 회귀 모델) 을 개발하여 주어진 사용 사례에서 문제를 해결합니다. 모델 개발 중에 데이터 과학자는 종종 데이터 형식, 데이터 수명 주기 및 중간 출력 위치와 같은 입력과 출력에 대한 가정을 합니다. 이러한 가정은 2단계의 단위 테스트 중에 검증에 사용될 수 있도록 문서화되어야 합니다.

이 단계는 모델 개발에 초점을 맞추고 있지만 데이터 과학자는 전처리, 교육, 평가 및 추론을 위해 최소한의 도우미 코드를 작성해야 하는 경우가 많습니다. 데이터 과학자는 개발 환경에서 이 코드를 실행할 수 있어야 합니다. 또한 광범위한 수동 변경 없이 다른 환경에서 실행되도록 이 도우미 코드를 동적으로 구성할 수 있도록 선택적 런타임 인수를 제공하는 것이 좋습니다. 이렇게 하면 2단계와 3단계에서 모델과 파이프라인 간의 통합을 가속화할 수 있습니다. 예를 들어 원시 데이터를 읽는 코드는 데이터를 일관성 있는 방식으로 선행 처리할 수 있도록 함수에 캡슐화해야 합니다.

다음과 같은 프레임워크부터 시작하는 것이 좋습니다.Scikit-learn,XGBoost,PyTorch,Keras또는TensorFlowML 모델 및 도우미 코드를 개발합니다. 예를 들어 scikit-learn은 파이썬으로 작성된 무료 ML 라이브러리입니다. 객체에 대한 균일 한 API 규칙을 제공하며 네 가지 주요 객체를 포함합니다.평가자,예언자,변신 로봇, 및모델—경량 데이터 변환, 라벨 및 피쳐 엔지니어링을 지원하며 전처리 및 모델링 단계를 캡슐화합니다. 이러한 객체는 상용구 코드 확산을 방지하고 검증 및 테스트 데이터가 훈련 데이터셋으로 유출되는 것을 방지하는 데 도움이 됩니다. 마찬가지로 모든 ML 프레임워크에는 자체 주요 ML 아티팩트가 구현되어 있으므로 ML 모델을 개발할 때 선택한 프레임워크의 API 규칙을 준수하는 것이 좋습니다.

2단계. 런타임 스크립트 만들기



이 단계에서는 1단계에서 개발한 모델과 관련 도우미 코드를 ML 플랫폼에 통합하여 프로덕션 준비 교육 및 추론을 수행할 수 있습니다. 구체적으로, 여기에는 모델을 SageMaker에 통합할 수 있도록 런타임 스크립트 개발이 포함됩니다. 이러한 독립형 파이썬 스크립트에는 미리 정의된 SageMaker 콜백 함수와 환경 변수가 포함되어 있습니다. 이러한 인스턴스는 Amazon Elastic Compute Cloud (Amazon EC2) 인스턴스에서 호스팅된 SageMaker 컨테이너 내에서 실행됩니다. 이Amazon SageMaker 파이썬 SDK 설명서에서는 이러한 콜백과 보조 설정이 교육 및 추론을 위해 함께 작동하는 방법에 대한 자세한 정보를 제공합니다. 예를 들어 단원을 참조하십시오.사이킷 학습 교육 스크립트 준비과Scikit-learn 모델 배포SageMaker 설명서에 있습니다.) 다음 섹션에서는 다음과 같은 작업 경험을 바탕으로 ML 런타임 스크립트 개발에 대한 추가 권장 사항을 제공합니다.AWS고객

처리 작업 사용

SageMaker는 배치 모드 모델 추론을 수행하기 위한 두 가지 옵션을 제공합니다. SageMaker 사용할 수 있습니다.처리 작업또는 a배치 변환 작업. 각 옵션에는 장점과 단점이 있습니다.

처리 작업은 SageMaker 컨테이너 내에서 실행되는 파이썬 파일로 구성됩니다. 처리 작업은 Python 파일에 넣은 로직으로 구성됩니다. 다음과 같은 장점이 있습니다.

- 교육 작업의 기본 로직을 이해하면 처리 작업을 쉽게 설정할 수 있으며 이해하기 쉽습니다. 교육 작업과 동일한 추상화를 공유합니다 (예: 인스턴스 수 및 데이터 배포 조정).
- 데이터 과학자와 ML 엔지니어는 데이터 조작 옵션을 완벽하게 제어할 수 있습니다.
- 데이터 과학자는 익숙한 읽기/쓰기 기능을 제외하고는 I/O 구성 요소 로직을 관리할 필요가 없습니다.
- SageMaker 이외의 환경에서 파일을 실행하는 것이 다소 쉬워서 신속한 개발과 로컬 테스트를 지원합니다.
- 오류가 있는 경우 스크립트가 실패하는 즉시 처리 작업이 실패하고 재시도를 기다리는 동안 예상치 못한 상황이 발생하지 않습니다.

반면에 배치 변환 작업은 SageMaker 엔드포인트의 개념의 확장입니다. 런타임에 이러한 작업은 콜백 함수를 가져온 다음 데이터 읽기, 모델 로드 및 예측을 위한 I/O를 처리합니다. Batch 변환 작업에는 다음과 같은 이점이 있습니다.

- 교육 작업에서 사용하는 추상화와 다른 데이터 배포 추상화를 사용합니다.
- 배치 추론과 실시간 추론 모두에 동일한 코어 파일과 함수 구조를 사용하므로 편리합니다.
- 재시도 기반 내결함성 메커니즘이 내장되어 있습니다. 예를 들어, 레코드 배치에서 오류가 발생하면 작업이 실패로 종료되기 전에 여러 번 다시 시도합니다.

투명성, 여러 환경에서의 사용 용이성, 교육 작업과의 공유 추상화 때문에 이 가이드에 제시된 참조 아키텍처에서 배치 변환 작업 대신 처리 작업을 사용하기로 결정했습니다.

Python 런타임 스크립트를 클라우드에 배포하기 전에 로컬로 실행해야 합니다. 특히, 파이썬 스크립트를 구조화하고 단위 테스트를 수행할 때 main guard 절을 사용하는 것이 좋습니다.

메인 가드 조항 사용

main guard 절을 사용하여 모듈 임포트를 지원하고 Python 스크립트를 실행하십시오. Python 스크립트를 개별적으로 실행하면 ML 파이프라인에서 문제를 디버깅하고 격리하는 데 유용합니다. 다음 단계를 수행하는 것이 좋습니다.

- Python 처리 파일에서 인수 파서를 사용하여 입력/출력 파일과 해당 위치를 지정합니다.
- 각 Python 파일에 대한 기본 가이드와 테스트 함수를 제공합니다.
- Python 파일을 테스트한 후 사용 중인지 여부에 관계없이 ML 파이프라인의 여러 단계에 통합하십시오.AWS Step Functions모델 또는 SageMaker 처리 작업입니다.
- 사용ASSERT테스트 및 디버깅을 용이하게 하기 위한 스크립트의 중요 섹션에 있는 명령문 예를 들어, 를 사용할 수 있습니다.ASSERT로드 후 데이터세트 기능의 수가 일관성을 유지하도록 하는 문을 참조하십시오.

단위 테스트

파이프라인에 대해 작성된 런타임 스크립트의 단위 테스트는 ML 파이프라인 개발에서 자주 무시되는 중요한 작업입니다. 기계 학습과 데이터 과학은 비교적 새로운 분야이며 단위 테스트와 같은 잘 정립 된 소프트웨어 엔지니어링 관행을 채택하는 속도가 느리기 때문입니다. ML 파이프라인은 프로덕션 환경에서 사용되기 때문에 실제 애플리케이션에 ML 모델을 적용하기 전에 파이프라인 코드를 테스트해야 합니다.

런타임 스크립트를 단위 테스트하면 ML 모델에 다음과 같은 고유한 이점이 있습니다.

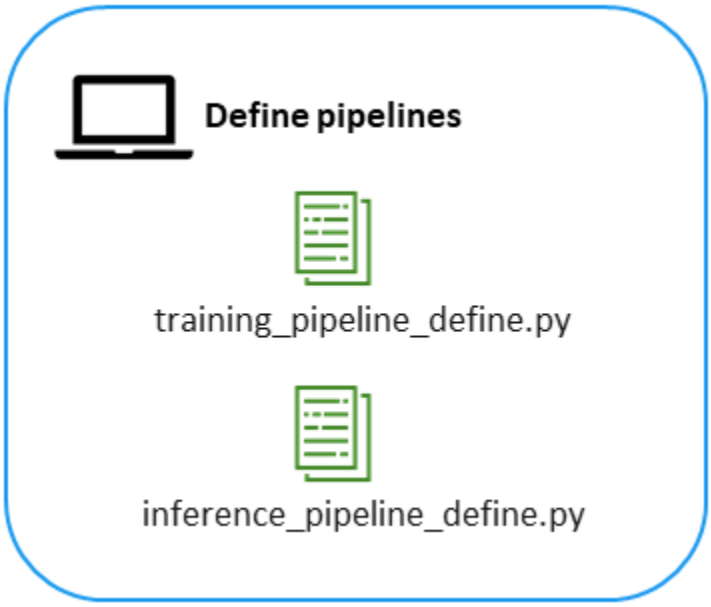
- 예상치 못한 데이터 변환을 방지합니다. 대부분의 ML 파이프라인에는 많은 데이터 변환이 포함되므로 이러한 변환이 예상대로 수행되는 것이 중요합니다.
- 코드의 재현성을 검증합니다. 코드의 모든 임의성은 다양한 사용 사례로 단위 테스트를 통해 감지할 수 있습니다.
- 코드의 모듈성을 적용합니다. 단위 테스트는 일반적으로 특정 테스트 세트 (테스트 케이스 모음) 가 프로그램의 소스 코드를 실행하는 정도인 테스트 커버리지 측정값과 연관됩니다. 높은 테스트 범위를 달성하기 위해 개발자는 코드를 모듈화합니다. 함수나 클래스로 분해하지 않고 대량의 코드에 대한 단위 테스트를 작성하기가 어렵 기 때문입니다.
- 저품질 코드 또는 오류가 프로덕션에 도입되는 것을 방지합니다.

다음과 같은 성숙한 단위 테스트 프레임워크를 사용하는 것이 좋습니다.pytest프레임 워크 내에서 광범위한 단위 테스트를 관리하는 것이 더 쉽기 때문에 단위 테스트 케이스를 작성합니다.

Important

단위 테스트는 모든 코너 케이스를 테스트한다고 보장할 수는 없지만 모델을 배포하기 전에 실수를 사전에 방지하는 데 도움이 될 수 있습니다. 운영 효율성을 보장하기 위해 배포 후 모델을 모니터링하는 것이 좋습니다.

3단계. 파이프라인 정의



이 단계에서는 파이프라인이 수행할 작업의 순서와 논리가 정의됩니다. 여기에는 개별 단계뿐만 아니라 논리적 입력 및 출력도 포함됩니다. 예를 들어 파이프라인 시작 시점의 데이터 상태는 어떻게 됩니까? 서로 다른 수준의 세분화 된 여러 파일이나 단일 플랫 파일에서 가져온 것입니까? 데이터가 여러 파일에서 가져온 경우 모든 파일에 대해 한 단계 또는 각 파일에 대해 별도의 단계가 필요합니까? 전처리 로직을 정의하려면? 결정은 데이터 원본의 복잡성과 선행 처리 정도에 따라 달라집니다.

참조 구현에서는 다음을 사용합니다.AWS Step Functions워크플로우 단계를 정의하는 서버리스 함수 오케스트레이터입니다. 그러나,ML 최대 프레임워크Apache AirFlow와 같은 다른 파이프라인 또는 상태 머신 시스템도 지원합니다 (파이프라인 오케스트레이션을 위한 다양한 엔진 (p. 13)섹션) 은 ML 파이프라인의 개발 및 배포를 촉진합니다.

Step Functions SDK 사용

ML 파이프라인을 정의하려면 먼저AWS Step FunctionsData Science SDK(Step Functions SDK): 파이프라인의 두 가지 주요 구성 요소를 정의합니다.단계과데이터. 파이프라인을 DAG (방향성 비순환 그래프) 라고 생각하면 단계는 그래프의 노드를 나타내며 데이터는 한 노드 (단계) 를 다음 노드 (단계) 에 연결하는 방향 간선으로 표시됩니다. ML 단계의 일반적인 예로는 전처리, 교육 및 평가가 있습니다. Step Functions SDK는 여러 기본 제공 단계를 제공합니다 (예:트레이닝스텝) 를 사용할 수 있습니다. 데이터의 예로는 입력, 출력 및 파이프라인의 일부 단계에 의해 생성되는 많은 중간 데이터셋이 있습니다. ML 파이프라인을 설계할 때는 데이터 항목의 구체적인 값을 알 수 없습니다. 정의할 수 있습니다.데이터 자리 표시자템플릿 (함수 매개 변수와 유사) 역할을 하며 데이터 항목 및 기본 데이터 유형의 이름만 포함합니다. 이렇게 하면 그래프에서 이동하는 데이터의 구체적인 값을 미리 알지 못해도 완전한 파이프라인 청사진을 설계할 수 있습니다. 이를 위해 Step Functions SDK의 자리 표시자 클래스를 사용하여 이러한 데이터 템플릿을 명시적으로 모델링할 수 있습니다.

ML 파이프라인에는 각 ML 단계의 동작을 세부적으로 제어할 수 있도록 구성 매개 변수가 필요합니다. 이러한 특수 데이터 자리 표시자가 호출됩니다.매개변수 자리 표시자. 파이프라인을 정의할 때 많은 값을 알 수 없습니다. 파라미터 자리 표시자의 예로는 파이프라인 설계 중에 정의한 인프라 관련 파라미터 (예: AWS 리

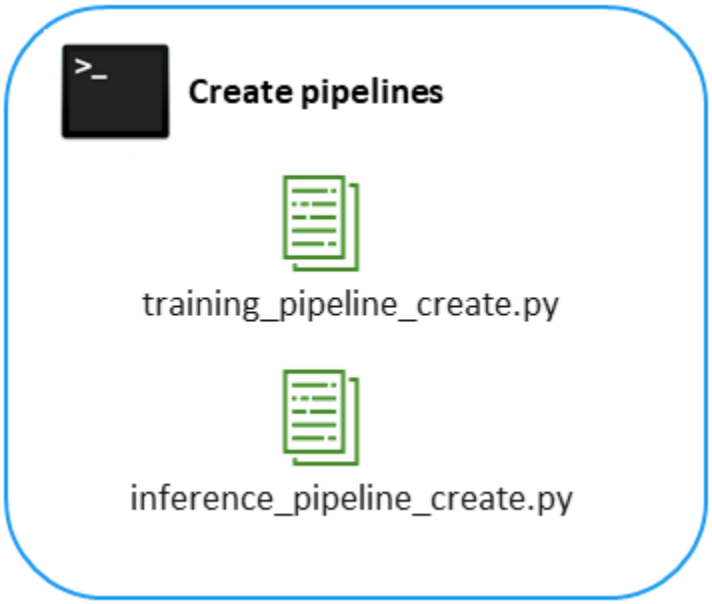
전 또는 컨테이너 이미지 URL) 와 파이프라인을 실행할 때 정의하는 ML 모델링 관련 파라미터 (예: 하이퍼파라미터) 가 있습니다.

Step Functions SDK 확장

참조 구현에서 한 가지 요구 사항은 특정 매개 변수 설정을 사용하여 ML 파이프라인 정의를 구체적인 ML 파이프라인 생성 및 배포와 분리하는 것이었습니다. 그러나 Step Functions SDK의 기본 제공 단계 중 일부는 이러한 자리 표시자 매개 변수를 모두 전달할 수 없었습니다. 대신 매개 변수 값은 파이프라인 설계 시간 동안 SageMaker 구성 API 호출을 통해 직접 얻어질 것으로 예상되었습니다. SageMaker 디자인 타임 환경이 SageMaker 런타임 환경과 동일한 경우 정상적으로 작동하지만 실제 설정에서는 거의 발생하지 않습니다. 파이프 라인 설계 시간과 런타임 간의 긴밀한 결합과 ML 플랫폼 인프라가 일정하게 유지된다는 가정은 설계된 파이프 라인의 적용 가능성을 크게 방해합니다. 실제로 ML 파이프라인은 기본 배포 플랫폼이 조금이라도 변경되면 즉시 중단됩니다.

이러한 문제를 극복하고 강력한 ML 파이프라인 (한 번 설계하고 어디서나 실행하고자 함) 을 생성하기 위해 다음과 같은 기본 제공 단계를 확장하여 자체 사용자 지정 단계를 구현했습니다.트레이닝스텝,모델스텝, 및트랜스포머스텝. 이러한 확장은 에서 제공됩니다.ML 최대 프로젝트. 이러한 사용자 정의 단계의 인터페이스는 파이프라인 생성 중 또는 파이프라인이 실행 중일 때 구체적인 값으로 채울 수 있는 훨씬 더 많은 매개 변수 자리 표시자를 지원합니다.

4단계. 파이프라인 생성



파이프라인을 논리적으로 정의한 후에는 파이프라인을 지원할 인프라를 만들어야 합니다. 이 단계에는 최소한 다음과 같은 기능이 필요합니다.

- 코드, 모델 아티팩트, 훈련 및 추론 실행에 사용되는 데이터를 포함하여 파이프라인 입력 및 출력을 호스팅하고 관리하는 스토리지
- 모델링 및 추론은 물론 데이터 전처리 및 후처리를 위한 컴퓨팅 (GPU 또는 CPU) 입니다.
- 오케스트레이션, 사용 중인 리소스를 관리하고 정기적인 실행을 예약할 수 있습니다. 예를 들어, 새 데이터를 사용할 수 있게 되면 모델을 주기적으로 재교육할 수 있습니다.
- 로깅 및 경고, 파이프라인 모델 정확성 모니터링, 리소스 활용도 및 문제 해결

구현AWS CloudFormation

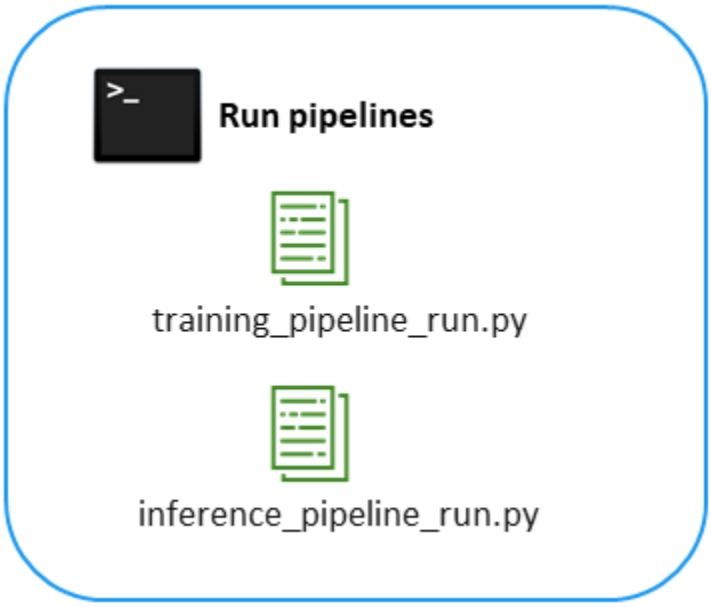
파이프라인을 생성하려면AWS CloudFormation()AWS인프라를 코드로 배포 및 관리하는 서비스입니다. 이AWS CloudFormation템플릿에는 Step Functions SDK를 사용하여 이전 단계인 Step Functions 정의가 포함됩니다. 이 단계에는 AWS 관리형 Step Functions 인스턴스의 생성이 포함됩니다.Step Functions 상태 머신. 교육 및 추론 작업은 필요할 때만 SageMaker 작업으로 필요에 따라 실행되므로 이 단계에서는 교육 및 추론을 위한 리소스가 생성되지 않습니다. 이 단계에는 생성도 포함됩니다.AWS Identity and Access ManagementStep Functions 실행하고, SageMaker를 실행하고, Amazon S3 읽고 쓰는 (IAM) 역할입니다.

Step Functions SDK에서 출력 수정

우리는 몇 가지 사소한 수정을해야했습니다.AWS CloudFormation이전 단원에서 출력합니다. 간단한 Python 문자열 매칭을 사용하여 다음을 수행했습니다.

- 만들기 위한 로직을 추가했습니다.`Parameters`의 단원AWS CloudFormation템플릿. 이는 두 가지 역할을 만들고 파이프라인 이름을 배포 환경과 함께 매개 변수로 정의하려고 하기 때문입니다. 이 단계에서는 6단계에서 설명한 대로 만들 수 있는 추가 리소스 및 역할도 다릅니다.
- 필수 필드를 갖도록 세 개의 필드를 다시 포맷했습니다.`!Sub`배포 프로세스의 일부로 동적으로 업데이트할 수 있도록 접두사와 따옴표:
  - 이`StateMachineName`속성으로 상태 시스템의 이름을 지정합니다.
  - 이`DefinitionString`속성으로 상태 머신을 정의합니다.
  - 이`RoleArn`속성으로, 상태 머신에서 반환됩니다.

5단계. 파이프라인을 실행합니다.



이 단계에서는 에서 생성된 훈련 또는 추론 파이프라인을 실행합니다.AWS CloudFormation4단계의 스택입니다. 내부 자리 표시자 매개변수가 구체적인 값으로 채워질 때까지 파이프라인을 실행할 수 없습니다. 자리 표시자 매개 변수에 값을 할당하는 이 작업은 5단계의 주요 활동입니다. 자리 표시자 매개 변수의 예는 다음과 같습니다.

- 입력, 출력 및 중간 데이터세트의 위치
- 런타임 스크립트와 2단계에서 개발된 기타 전처리 또는 평가 코드의 Amazon S3 위치 (예:`sm_submit_url`교육 파이프라인의 경우)
- 의 이름입니다.AWS리전

파이프라인을 실행하기 전에 이러한 경로 값이 유효한 데이터나 코드를 가리키는지 확인해야 합니다. 예를 들어, Python 런타임 스크립트의 Amazon S3 URL을 나타내는 자리 표시자 파라미터를 채우는 경우 해당 스크립트를 해당 URL에 업로드해야 합니다. 파이프라인을 실행하는 사람은 일관성 확인 및 데이터 업로드를 담당합니다. 파이프라인을 정의하거나 생성하는 사람은 이 문제에 대해 걱정할 필요가 없습니다.

파이프라인의 완성도에 따라 이 단계는 정기적으로 (매주 또는 매월) 실행되도록 자동화될 수 있습니다. 자동화에는 강력한 모니터링이 필요합니다. 이는 중요한 영역이지만 이 설명서에서는 다루지 않습니다. 교육 파이프라인 실행의 경우 평가 지표를 모니터링하는 것이 적절할 것입니다. 추론 파이프라인의 경우 입력 데이터 분포 드리프트를 모니터링하고 가능한 경우 주기적으로 라벨을 수집하고 예측 정확도로 드리프트를 측정하는 것이 적절할 것입니다. 훈련 및 추론 실행의 이러한 레코드는 나중에 분석을 위해 데이터베이스에 기록되어야 합니다.

6단계. 파이프라인 확장

이 안내서에서는 에서 ML 파이프라인 구축을 시작할 수 있는 방법에 대해 설명합니다.AWS콘크리트 아키텍처와 함께 신속하게. 메타데이터 관리, 실험 추적 및 모니터링과 같이 파이프라인 성숙을 위한 추가 고려 사항이 있습니다. 다음은 이 설명서에서는 다루지 않는 중요한 주제입니다. 다음 섹션에서는 파이프라인 자동화인 파이프라인 관리의 또 다른 측면에 대해 설명합니다.

다양한 수준의 자동화

SageMaker 콘솔에서 교육 파이프라인을 수동으로 설정할 수 있지만 실제로는 ML 교육 파이프라인 배포에서 수동 접점을 최소화하여 ML 모델이 일관되고 반복적으로 배포되도록 하는 것이 좋습니다. 요구 사항 및 해결하려는 비즈니스 문제에 따라 반자동, 완전 자동화 및 완전 관리라는 세 가지 수준에서 배포 전략을 결정하고 구현할 수 있습니다.

- 반자동 — 기본적으로 이전 섹션에서 설명한 단계는 다음을 사용하여 교육 및 추론 파이프라인을 배포하므로 반자동화된 접근 방식을 따릅니다.AWS CloudFormation템플릿. 이렇게 하면 파이프라인의 재현성을 보장하고 쉽게 변경하고 업데이트할 수 있습니다.
- 완전 자동화 — 보다 향상된 옵션은 개발, 스테이징 및 프로덕션 환경에 지속적 통합 및 지속적 배포 (CI/CD) 를 사용하는 것입니다. 교육 파이프라인 배포에 CI/CD 사례를 통합하면 품질 게이트뿐만 아니라 추적성을 자동화하도록 보장할 수 있습니다.
- 완전 관리형 — 궁극적으로 완전 관리형 시스템을 개발하여 간단한 매니페스트 세트로 ML 교육 파이프라인을 배포할 수 있으며, 시스템은 자체 구성 및 필요에 맞게 조정할 수 있습니다.AWS서비스.

이 안내서에서는 구체적인 아키텍처를 제시하기로 결정했습니다. 그러나 고려할 수 있는 대체 기술이 있습니다. 다음 두 섹션에서는 플랫폼과 오케스트레이션 엔진에 대한 몇 가지 대체 선택에 대해 설명합니다.

ML 워크로드를 위한 다양한 플랫폼

Amazon SageMaker입니다.AWSML 모델 교육 및 제공을 위한 관리형 서비스입니다. 많은 사용자가 다양한 기본 제공 기능과 ML 워크로드를 실행하기 위해 제공하는 다양한 옵션을 높이 평가합니다. SageMaker는 클라우드에서 ML 구현을 시작하는 경우 특히 유용합니다. SageMaker의 주요 기능은 다음과 같습니다.

- 내장된 추적성 (라벨링, 교육, 모델 추적, 최적화 및 추론 포함).
- 최소한의 Python 및 ML 경험으로 교육 및 추론을 위한 원클릭 옵션이 내장되어 있습니다.
- 고급 하이퍼파라미터 튜닝.
- 모든 주요 인공 지능 및 기계 학습 (ML/AI) 프레임워크 및 맞춤형 Docker 컨테이너를 Support.
- 기본 제공 모니터링 기능.
- 교육 작업, 처리 작업, 일괄 변환 작업, 모델, 엔드포인트 및 검색 가능성을 포함한 기록 추적 기능이 내장되어 있습니다. 훈련, 처리 및 배치 변환과 같은 일부 이력은 불변이며 추가 전용입니다.

SageMaker를 사용하는 대안 중 하나는 다음과 같습니다.AWS Batch.AWS Batch사용자 환경에 대한 컴퓨팅 및 오케스트레이션에 대한 보다 낮은 수준의 제어를 제공하지만 머신 러닝을 위해 맞춤 제작되지는 않습니다. 주요 기능 중 일부는 다음과 같습니다.

- 워크로드를 기반으로 컴퓨팅 리소스의 기본 자동 확장.
- 작업 우선 순위, 재시도 및 작업 종속성에 대한 기본 지원.

- 반복 및 온디맨드 작업 구축을 지원하는 대기열 기반 접근 방식입니다.
- CPU 및 GPU 워크로드 Support. GPU는 특히 딥 러닝 모델의 경우 훈련 프로세스의 속도를 크게 높일 수 있기 때문에 ML 모델 구축에 GPU를 사용하는 기능이 중요합니다.
- 사용자 정의 정의 기능Amazon 머신 이미지(AMI) 컴퓨팅 환경을 위한 것입니다.

파이프라인 오케스트레이션을 위한 다양한 엔진

두 번째 주요 구성 요소는 파이프라인 오케스트레이션 레이어입니다.AWS에서 제공합니다Step Functions완전 관리형 오케스트레이션 환경을 제공합니다. Step Functions 인기 있는 대안은 아파치 에어플로우입니다. 둘 사이의 결정을 내릴 때 다음 사항을 고려하십시오.

- 필수 인프라 —AWS Step Functions완전 관리형 서비스이며 서버리스 서비스이지만 Airflow는 자체 인프라를 관리해야 하며 오픈 소스 소프트웨어를 기반으로 합니다. 따라서 Step Functions 즉시 고가용성을 제공하는 반면 Apache Airflow를 관리하려면 추가 단계가 필요합니다.
- 스케줄링 기능 — Step Functions 공기 흐름 모두 유사한 기능을 제공합니다.
- 시각화 기능 및 UI — Step Functions 공기 흐름 모두 유사한 기능을 제공합니다.
- 계산 그래프 내에서 변수 전달 — Step Functions는 사용을 위한 제한된 기능을 제공합니다.AWS Lambda 기능이지만 Airflow는 XCom 인터페이스를 제공합니다.
- 사용법 — Step Functions 매우 인기가 있습니다.AWS고객 및 Airflow는 데이터 엔지니어링 커뮤니티에서 널리 채택되었습니다.

결론

기계 학습이 연구 분야에서 응용 분야로 전환됨에 따라 다양한 산업 분야에서 ML 파이프라인 개발, 배포 및 운영에서 매년 25% 의 성장세를 보였습니다. ML의 비즈니스 가치는 매일 ML 운영 및 파이프라인을 통해 실현되며, 이는 ML 모델 및 알고리즘의 연구 개발을 주도합니다. 그럼에도 불구하고 ML을 프로덕션에 배포하는 것은 데이터 관리, 처리, 분석, 모델링, 검증 및 보안과 같이 상당히 다른 활동과 아티팩트가 서로 얽혀 있기 때문에 수많은 과제를 안고 있습니다. 다양한 AI/ML 계약을 통해AWS데이터 사이언스 팀은 다양한 ML DevOps 활동과 아티팩트를 최적으로 융합하거나 분리하기 위한 템플릿 세트를 제공하는 엔드 투 엔드 워크플로우가 부족하다는 점을 확인했습니다. 이 안내서에서는ML 최대 워크플로이 긴급한 문제를 해결합니다. ML Max는 단계별 지침과 일련의 프로그래밍 템플릿을 제공합니다. 목표는 대화식 모델 개발 단계에서 프로덕션 준비가 된 완전하고 확장 가능한 ML 파이프라인 구성으로 빠르고 비용 효율적으로 전환할 수 있도록 하는 것입니다.

추가 리소스

관련 가이드 및 패턴

- 딥 러닝 시스템의 불확실성 정량화
- 다음을 사용하여 ML 빌드, 교육 및 워크로드를 Amazon SageMaker SageMaker로 마이그레이션AWS개발자 도구
- AWS규범 지침

AWS서비스

- AWS Batch
- AWS CloudFormation
- IAM
- Amazon SageMaker
- AWS Step Functions

AWS일반 리소스

- AWS선적 서류 비치
- AWS일반 참조
- AWS용어집

문서 기록

아래 표에 이 설명서의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대해 알림을 받으려면RSS 피드.

update-history-change	update-history-description	update-history-date
첫 게시 (p. 16)	—	2021년 1월 29일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.