بسمه تعالی

**پیشنهاد پروژه** MLOps

شرکت آدین



فروردین ۱۴۰۲

نسخه ۰.۹۷

# مشخصات کلی

MLOps (عملیات یادگیری ماشین) به مجموعه‌ای از فرایندها، ابزارها و شیوه‌ها جهت مدیریت چرخه مدل‌های یادگیری ماشین در یک محیط تولید، اشاره دارد. همچنین این چرخه شامل همکاری بین دانشمندان داده، مهندسان و تیم‌های DevOps است به‌نحوی‌که این اطمینان حاصل شود که مدل‌ها به طور مؤثر توسعه، استقرار، نظارت و به‌روزرسانی می‌شوند. هدف MLOها افزایش سرعت، قابلیت اطمینان و مقیاس‌پذیری مدل‌های یادگیری ماشین در تولید است، درحالی‌که خطرات ناشی از ریسک عدم موفقیت را نیز کاهش می‌دهد. همچنین به‌کارگیری MLOps فرایند مدیریت را ساده‌تر کرده، کیفیت را افزایش می‌دهد و استقرار مدل‌های یادگیری عمیق و یادگیری ماشین را در محیط‌های تولید در مقیاس بزرگ را خودکار می‌کند. هدف، بهبود خودکارسازی و ارتقای کیفیت مدل‌های تولید و درعین‌حال توجه به الزامات تجاری و نظارتی است.

MLOps به‌ویژه در مورد استقرار مدل‌های ML در تولید اهمیت دارد، زیرا به سازمان‌ها کمک می‌کند تا مطمئن شوند که مدل‌هایشان در طول زمان دقیق، قابل‌اعتماد و کارآمد هستند. به‌طورکلی، MLOps با خودکار کردن بسیاری از مراحل مربوط به استقرار و مدیریت مدل‌های ML، به دانشمندان و مهندسان داده اجازه می‌دهد تا با همکاری یکدیگر به ارائه سریع‌تر و کارآمدتر مدل‌های یادگیری ماشین دست یابند. برای رسیدن به اهداف فوق، تیم‌های MLOps معمولاً از ترکیبی از ابزارها، فرایندها و شیوه‌ها زیر را استفاده می‌کنند:

* کنترل نسخه برای مدل‌های ML و مجموعه‌داده‌ها
* آزمون خودکار و ادغام مداوم
* کانتینرسازی و ارکستراسیون برای استقرار مدل
* نظارت و ثبت گزارش برای ردیابی عملکرد مدل در تولید
* حاکمیت و انطباق برای مدل‌ها

مدیریت سیستم‌های هوش مصنوعی در مقیاس بزرگ کار آسانی نیست و در این مسیر چالش‌های مهمی وجود دارد که تیم‌ها باید با آن مواجه شوند. برخی از این چالش‌ها در کاربرد یادگیری ماشین (ML) در یک محصول مقیاس‌پذیر عبارتند از:

کیفیت و کمیت داده: اطمینان از وجود داده‌های آموزشی باکیفیت بالا برای آموزش مدل‌های ML یک چالش بزرگ است.

عملکرد مدل: مدل‌های ML گاهی اوقات ممکن است برای تعمیم به داده‌های جدید مشکل داشته باشند و ممکن است در صورت استقرار در تنظیمات دنیای واقعی ضعیف عمل کنند.

تفسیرپذیری مدل: مدل‌های ML می‌توانند پیچیده و غیرقابل‌درک باشند، و تفسیر پیش‌بینی‌ها و تشخیص مشکلات، آن‌ها را به چالش می‌کشد.

منابع محاسباتی: آموزش و به‌کارگیری مدل‌های ML بزرگ و پیچیده می‌تواند به مقدار قابل‌توجهی از منابع محاسباتی نیاز داشته باشد که می‌تواند در یک محصول مقیاس‌پذیر چالش‌برانگیز باشد.

استقرار مدل: استقرار مدل‌های ML در یک محیط تولید و ادغام آن‌ها در یک محصول می‌تواند چالش‌برانگیز باشد و نیاز به بررسی دقیق زیرساخت‌ها، حریم خصوصی داده‌ها و امنیت دارد.

نگهداری مدل: مدل‌های ML باید با تغییر داده‌ها و نیازمندی‌های محصول، به‌روزرسانی و بازآموزی شوند که می‌تواند در یک محصول مقیاس‌پذیر یک چالش باشد.

بایاس: مدل‌های ML گاهی اوقات می‌توانند سوگیری‌های موجود را در داده‌های آموزشی را تداوم بخشند و توجه به این مسائل هنگام توسعه یک محصول مقیاس‌پذیر ML بسیار مهم خواهد بود.

## MLOPs برای چه مشتریانی مناسب است؟

به طور کلی، MLOps یک چارچوب قدرتمند است که می تواند به تیم ها کمک کند تا پروژه های یادگیری ماشینی را به طور موثر مدیریت و مقیاس کنند. از سوی دیگر عواملی وجود دارد که ممکن است MLOps بهترین گزینه برای حل برخی از مسائل باشد. در اینجا چند نمونه از این دلایل ذکر می شود:

* **بهبود سرعت ایجاد و استقرار مدل:** در نتیجه مدیریت جامع چرخه زندگی یادگیری ماشین، به دلیل افزایش ارزش یادگیری ماشین.
* **زمان سریع‌تر جهت بازاریابی:** MLOps دانشمندان و مهندسان داده را قادر می‌سازد تا سریع‌تر تکرار کنند و مدل‌ها را سریع‌تر به تولید بفرستند. همچنین روش‌های یکپارچه‌سازی و تحویل مداوم، استقرار این سیستم‌ها را در تولید آسان‌تر می‌کند.
* **بهبود کیفیت مدل:** با خودکارسازی آموزش، اعتبارسنجی و استقرار مدل‌ها، احتمال خطای انسانی کاهش می‌یابد که منجر به عملکرد بهتر مدل می‌شود. همچنین، جهت اطمینان از ارائه پیش‌بینی‌های باکیفیت بالا، سیستم MLOps باید قادر به اندازه‌گیری دریفت مدل[[1]](#footnote-1) باشد. این امر احتمال دیدگاه‌های نادرست در فرضیه های مدل را کاهش می‌دهد.
* **همکاری بهتر:** MLOps همکاری بین دانشمندان داده، توسعه‌دهندگان و تیم‌های عملیاتی را تشویق می‌کند و تیم‌های متقابل را قادر می‌سازد تا به طور مؤثرتری با هم کار کنند. MLOها همکاری و شفافیت را در میان تیم‌های dataOps، مهندسین یادگیری ماشین، تحلیلگران تجاری/مدیران محصول، مهندسین تضمین کیفیت و مهندسین زیرساخت افزایش می‌دهند.
* **مقیاس‌پذیری:** خودکار کردن فرایند استقرار، مقیاس‌پذیری مدل‌ها را جهت رسیدگی به مقادیر بزرگ و افزایشی داده و جمعیت بیشتر کاربران آسان‌تر می‌کند.

## چالش‌های MLOps

* **پیچیدگی**: پیاده‌سازی MLOها می‌تواند پیچیده باشد و به مهارت‌های تخصصی و درک عمیق فرایندهای یادگیری ماشین و توسعه نرم‌افزار نیاز دارد. MLOps به ابزارها و مهارت‌های فراوانی نیاز دارد که به پیچیدگی آن می‌افزاید. جهت ساختن یک چارچوب موفق MLOps، به یک معمار باتجربه MLOps و مهندسان ML نیاز داریم که از قبل با اصول مفهوم DevOps آشنا هستند.
* **چالش‌های یکپارچه‌سازی**: ادغام جریان‌های کاری ML در فرایندهای توسعه نرم‌افزار موجود می‌تواند دشوار باشد و ممکن است نیاز به تغییرات قابل‌توجهی در جریان‌های کاری و ابزارهای موجود داشته باشد.
* **امنیت داده‌ها**: MLOها به مدیریت دقیق داده‌های حساس از جمله ذخیره‌سازی ایمن، کنترل دسترسی و نظارت بر استفاده از داده‌ها نیاز دارند.
* **محدودیت‌های منابع:** پیاده‌سازی MLOها نیازمند سرمایه‌گذاری قابل‌توجهی در فناوری، فرآیندها و پرسنل است که ممکن است جهت همه سازمان‌ها امکان‌پذیر نباشد. MLOps جهت استقرار مؤثر مدل ML در دنیای واقعی ضروری است، اما هزینه آن به دلیل پیچیدگی و منابع موردنیاز می‌تواند بسیار گران باشد. توانایی ایجاد و حفظ یک چرخه حیات MLOps به طور مؤثر به منابع کافی (افراد، زیرساخت‌ها و ابزارها) نیاز دارد. عامل اصلی تعیین‌کننده در استفاده یا عدم استفاده از MLO ها، بررسی کامل بازگشت سرمایه ابتکار یادگیری ماشین است.

## MLOps برای چه دسته مسائلی مناسب نیست؟

به طور کلی، MLOps یک چارچوب قدرتمند است که می تواند به تیم ها کمک کند تا پروژه های یادگیری ماشینی را به طور موثر مدیریت و مقیاس کنند. با این حال، همیشه برای هر مشکل یا سناریویی مناسب نیست، و تیم ها باید قبل از اتخاذ MLO به دقت نیازهای خود را در نظر بگیرند. مسائل خاصی وجود دارد که ممکن است MLOps بهترین گزینه برای آنها نباشد. در اینجا چند نمونه هستند:

**پروژه های در مقیاس کوچک:** MLOps برای کمک به تیم ها در مقیاس پروژه های یادگیری ماشینی طراحی شده است که می تواند مدل ها و منابع داده زیادی را شامل شود. اگر روی یک پروژه در مقیاس کوچک فقط با یک یا دو مدل کار می کنید، ممکن است MLO ها بیش از حد باشد.

**تیم های غیر فنی:** MLO ها به سطح خاصی از تخصص فنی برای پیاده سازی و نگهداری نیاز دارند. اگر تیم شما مهارت های لازم را ندارد، ممکن است اتخاذ و استفاده موثر از MLO ها چالش برانگیز باشد.

**نمونه سازی سریع:** اگر در حال آزمایش مدل ها و الگوریتم های مختلف یادگیری ماشین هستید، ممکن است به سختی و ساختار MLO ها نیاز نداشته باشید. در عوض، ممکن است رویکرد چابک‌تری را ترجیح دهید که به شما امکان می‌دهد ایده‌ها را سریع تکرار و آزمایش کنید.

**سیستم های قدیمی:** MLO ها ممکن است با برخی از سیستم های قدیمی سازگار نباشند، که می تواند ادغام در زیرساخت های موجود شما را دشوار کند.

**مدل‌های ساده:** MLOps برای مدل‌های پیچیده یادگیری ماشینی که نیاز به آموزش و بهینه‌سازی گسترده دارند، بسیار مفید است. برای مدل‌های ساده که به داده‌ها یا منابع محاسباتی زیادی نیاز ندارند، MLO‌ها ممکن است غیرضروری باشند.

**در دسترس بودن و کیفیت داده ها**: MLOps فرض می کند که شما به داده های باکیفیت و قابل اعتمادی دسترسی دارید که می توانند برای آموزش و آزمایش مدل های یادگیری ماشین خود استفاده شوند. اگر داده های شما ناقص، پر سر و صدا یا کیفیت پایین باشد، MLO ها ممکن است موثر نباشند و ممکن است لازم باشد قبل از استفاده از MLO ها، روی پاکسازی و آماده سازی داده ها سرمایه گذاری کنید.

**انطباق با مقررات:** MLO ها می توانند به تیم ها در مدیریت چرخه عمر مدل های یادگیری ماشین کمک کنند، اما ممکن است برای انطباق با مقررات در صنایع خاص با قیود سختگیرانه، مانند مراقبت های بهداشتی یا مالی کافی نباشد. در این موارد، ممکن است اقدامات بیشتری لازم باشد تا اطمینان حاصل شود که مدل‌ها اخلاقی، شفاف و مطابق با قوانین و مقررات قابل اجرا هستند.

**محدودیت های هزینه و منابع:** MLO ها می توانند به منابع زیادی دسترسی داشته و به سرمایه گذاری قابل توجهی در زیرساخت ها، ابزارها و پرسنل نیاز دارند. اگر تیم شما با منابع یا بودجه محدود کار می کند، ممکن است اتخاذ و نگهداری موثر MLO ها چالش برانگیز باشد.

**محیط های پویا:** MLOps فرض می کند که زیرساخت و منابع داده شما نسبتاً پایدار و قابل پیش بینی هستند. اگر در یک محیط بسیار پویا، مانند یک برنامه IoT یا یک خط لوله داده های جریانی کار می کنید، MLO ها ممکن است مناسب نباشند، و رویکردهای جایگزین مانند یادگیری آنلاین یا یادگیری تقویتی ممکن است مناسب تر باشند.

به طور خلاصه، در حالی که MLO ها می توانند برای مدیریت و مقیاس پروژه های یادگیری ماشینی بسیار موثر باشند، اما راه حل یکسانی نیست. تیم ها باید قبل از اتخاذ MLO ها نیازها، منابع و محدودیت های خود را به دقت در نظر بگیرند و اگر MLO برای مشکل آنها مناسب نیست، رویکردهای جایگزین را بررسی کنند.

## گزینه های مطرح MLOps و مزایای انتخابی Kubeflow

Kubeflow و MLflow هر دو پروژه های منبع باز برای مدیریت گردش کار یادگیری ماشین هستند. با این حال، بررسی آنها در ابعاد مختلف نشان‌گر تفاوت‌های اساسی میان این دو پروژه است:

* هدف: Kubeflow به طور خاص برای اجرا و استقرار مدل‌های یادگیری ماشین در Kubernetes، یک سیستم منبع باز محبوب برای خودکارسازی استقرار، مقیاس‌بندی و مدیریت برنامه‌های کاربردی کانتینری طراحی شده است. از سوی دیگر، MLflow یک پلت فرم کلی تر برای مدیریت کل چرخه زندگی یادگیری ماشینی است، از جمله آزمایشات ردیابی، کد بسته بندی در اجراهای تکرارپذیر، و به اشتراک گذاری و استقرار مدل ها.
* ویژگی‌ها: Kubeflow مجموعه‌ای از ابزارها و APIها را برای ساخت، استقرار و مدیریت مدل‌های یادگیری ماشین در Kubernetes ارائه می‌کند، از جمله نوت‌بوک‌های Jupyter، آموزش TensorFlow و سرویس‌دهی مدل. MLflow طیف گسترده‌تری از ویژگی‌ها را برای چرخه زندگی یادگیری ماشین ارائه می‌کند، از جمله ردیابی آزمایش، بسته‌بندی کد و اشتراک‌گذاری، و استقرار مدل در پلتفرم‌های مختلف از جمله Docker، AzureML، و Amazon SageMaker.
* یکپارچه سازی: Kubeflow به خوبی با سایر اجزای اکوسیستم Kubernetes، مانند Helm (یک مدیر بسته برای Kubernetes) ادغام می شود، و همچنین ابزارهایی را برای کاهش حجم کاری یادگیری ماشین در مجموعه ای از ماشین ها فراهم می کند. MLflow را می توان با انواع کتابخانه ها و پلتفرم های یادگیری ماشین استفاده کرد و یک REST API برای ادغام با ابزارهای دیگر ارائه می دهد.

به طور خلاصه، Kubeflow برای مدیریت بارهای کاری یادگیری ماشین در Kubernetes مناسب است، در حالی که MLflow یک پلتفرم کلی تر برای مدیریت کل چرخه زندگی یادگیری ماشین است.

## نمونه های جهانی استفاده از Kubeflow

* از جمله شرکت ها و سازمان هایی که از Kubeflow به طور جدی برای آسان سازی فرانید چرخه یادگیری ماشین استفاده می‌کنند میتوان به مجموعه ای در شرکت سامسونگ با نام Samsung SDS اشاره کرد. در واقع این مجموعه Kubeflow را به عنوان پلتفرم ابری منطبق باKubernetes انتخاب کرده و به مشتریان خود بر بستر Samsung Cloud Platform پیاده‌سازی و خدمات ارائه میدهد.[[2]](#footnote-2)
* این مجموعه که زیر مجموعه ای از شرکت سامسونگ میباشد در زمینه‌های مختلفی سرویس ، آموزش و خدمات به Enterpriseها ارائه میکند. در زمینه‌ یادگیری ماشین به طور جدی از Kubeflow استفاده کرده و به صورت آموزش و در کنار شرکت های بزرگ همکاری میکند. در سایت این شرکت درباره سرویس AI خود به شرکت های بزرگ، Kubeflow را پلتفرم یادگیری ماشین مبتنی بر Kubernetes معرفی می کند.
* از جمله شرکت های بزرگ دیگر میتوان به Arrikto اشاره کرد. این شرکت مستقر در امریکا در واقع برای یادگیری ماشین Kubeflow as a Service ارائه میدهد.
* Arrikto، برخلاف سایر شرکت‌های MLOps، تیم‌های MLOps را قادر می‌سازد تا مدل‌های یادگیری ماشین را 30 برابر سریع‌تر از پلتفرم‌های سنتی ML به بازار عرضه کنند. Arrikto به عنوان یک مشارکت‌کننده پیشرو در Kubeflow، گردش‌های کاری خودکار، خطوط لوله قابل تکرار، استقرار مداوم از دسک‌تاپ به ابر و دسترسی ایمن به داده‌ها را ارائه می‌کند. Arrikto's Enterprise Kubeflow به عنوان یک توزیع چند گره در AWS، GCP و Azure در دسترس است و پلتفرم ترجیحی MLOps است که امروزه توسط بسیاری از شرکت های Fortune 500 در تولید استفاده می شود. این شرکت بیش از 300 مشتری در 17 کشور جهان دارد و توسط Unusual Ventures و Odyssey VP حمایت می شود.

# معماری و تکنولوژی‌های مرتبط

## معماری مفهومی و منطقی

به‌طورکلی مراحل اصلی در معماری مفهومی MLOps عبارت‌اند از:

1. Data Preparation - آماده سازی داده‌ها:

این مرحله شامل جمع‌آوری، تمیز کردن و پیش پردازش داده‌هایی است که برای آموزش و ارزیابی مدل‌های ML استفاده می‌شود. آماده‌سازی داده‌ها ممکن است شامل کارهایی مانند عادی‌سازی داده‌ها، تشخیص نقاط پرت و مهندسی ویژگی‌ها باشد.

1. Model Selection - انتخاب مدل:

در این مرحله، دانشمندان و مهندسان داده، معماری، الگوریتم و فراپارامترهای مدل ML مناسب را بر اساس ویژگی‌های داده‌ها و مسئله‌ای که باید حل شود، انتخاب می‌کنند.

Model Training - آموزش مدل:

این مرحله شامل استفاده از داده‌های آماده شده برای آموزش مدل انتخاب شده، با تنظیم پارامترهای مدل برای به حداقل رساندن خطا بین خروجی‌های پیش بینی شده و خروجی‌های واقعی است.

1. Model Evaluation - ارزیابی مدل:

این مرحله شامل ارزیابی عملکرد مدل آموزش‌دیده با مقایسه پیش‌بینی‌های آن با مجموعه‌ای از نتایج شناخته‌شده (که داده‌های آزمایشی نیز نامیده می‌شود)، و تعیین میزان تعمیم مدل به داده‌های نادیده جدید است.

1. Model Deployment - استقرار مدل:

این مرحله شامل در دسترس قرار دادن مدل برای کاربران نهایی است، خواه از طریق استقرار مدل در یک محیط تولید یا با ارائه یک API برای دسترسی دیگران به مدل باشد.

1. Model Monitoring and Maintenance - نظارت و نگهداری مدل:

این مرحله شامل نظارت بر عملکرد مدل مستقر در طول زمان و انجام به روز رسانی‌ها یا تنظیمات موردنیاز برای حفظ دقت و عملکرد مدل است.

Model Retainment- بازنشستگی مدل:

این مرحله شامل بازنشستگی مدل‌هایی است که دیگر موردنیاز نیستند یا دیگر عملکرد خوبی ندارند، تا اطمینان حاصل شود که فقط مدل‌هایی با عملکرد خوب به کار گرفته می‌شوند.

معماری منطقی MLOps معمولاً شامل اجزای زیر است:

* **مدیریت کد منبع:** این مؤلفه کد منبع مدل‌های یادگیری ماشین، الگوریتم‌ها و خطوط لوله را ذخیره می‌کند.
* **یکپارچه سازی مداوم/ استقرار مستمر (CI/CD**): این جزء ادغام و استقرار مدل‌های یادگیری ماشین را خودکار می‌کند.
* **آموزش مدل:** این جزء با استفاده از داده‌ها و الگوریتم‌های ذخیره شده در جزء مدیریت کد منبع، مدل‌های یادگیری ماشین را آموزش می‌دهد.
* **استقرار:** این جزء، مدل‌های آموزش دیده را در یک محیط تولید مستقر می‌کند، جایی که می‌توان از آنها برای پیش بینی استفاده کرد.
* **مانیتورینگ**: این مؤلفه عملکرد مدل‌های مستقر شده را نظارت می‌کند و در مورد دقت و قابلیت اطمینان آنها بازخورد ارائه می‌کند.
* **مدیریت مدل:** این مؤلفه چرخه عمر مدل‌های یادگیری ماشین را مدیریت می‌کند، از جمله نسخه‌سازی، به‌روزرسانی و بازنشستگی مدل‌ها در صورت لزوم. همچنین مدل‌های آموزش‌دیده و فراداده‌های مربوط به مدل‌ها مانند نسخه، دقت و داده‌های آموزشی را ذخیره می‌کند.

آماده سازی داده‌ها اولین مرحله در چرخه حیات یادگیری ماشینی (ML) است و شامل جمع‌آوری، تمیز کردن و پیش پردازش داده‌هایی است که برای آموزش و ارزیابی مدل‌های ML استفاده می‌شود. هدف از این مرحله آماده سازی داده‌ها در قالبی است که به راحتی توسط مدل ML مصرف شود و اطمینان حاصل شود که داده‌ها از کیفیت بالایی برخوردار هستند.

## Kubeflow

Kubeflow یک پلتفرم منبع باز برای توسعه، استقرار و مدیریت مدل‌های یادگیری ماشینی (ML) در Kubernetes (سیستم ارکستراسیون کانتینر) است به نحوی که توسعه، آزمایش و استقرار مدل‌های ML را در محیط‌های تولید برای سازمان‌ها آسان کرده و کارایی و مقیاس‌پذیری گردش‌های کاری ML خود را بهبود بخشند.

ویژگی‌های کلیدی Kubeflow عبارتند از:

* یک رابط کاربر پسند برای آموزش و استقرار مدل‌های ML با استفاده از چارچوب‌های مطرح مانند TensorFlow، PyTorch و scikit-learn و ...
* پشتیبانی از آموزش توزیع‌شده مدل با استفاده از Kubernetes، که امکان آموزش سریع‌تر و استفاده بهتر از منابع را فراهم می‌کند.
* ابزارهایی برای ثبت و تجسم پیشرفت آموزش مدل، و همچنین برای تنظیم هایپرپارامتر و استقرار مدل.
* ادغام با کتابخانه‌ها و ابزارهای معروف یادگیری ماشین، مانند TensorFlow Extended (TFX) و Seldon و ...
* توانایی اجرای در سیستم شخصی، در فضای ابری یا در محیط های ترکیبی.
* به هدف استقرار کلیه کامپوننت های چرخه یادگیری ماشین بر بستر کوبرنتیز ساخته شده است. از این رو میتوان گفت یک مجموعه ابزار کاملا منطبق و همگام با زیست بوم ابری میباشد. همچنین برای CNCF و دریافت گواهینامه معتبر زیست بوم ابری این نهاد درخواست فرستاده است.

به‌طورکلی، Kubeflow یک پلت فرم قدرتمند و منعطف برای مدیریت چرخه یادگیری ماشین است که به دانشمندان و مهندسان داده اجازه می‌دهد تا با هم کار کنند تا مدل‌های یادگیری ماشینی خود را به شیوه‌ای کارآمدتر و مؤثرتر بسازند، مستقر کنند و مدیریت کنند.

## Kubeflow برای چه دسته مسائلی مناسب نیست؟

Kubeflow یک پلت فرم یادگیری ماشینی قدرتمند است که می تواند برای طیف وسیعی از وظایف مورد استفاده قرار گیرد. با این حال، انواع مختلفی از مشکلات وجود دارد که ممکن است Kubeflow بهترین انتخاب برای آنها نباشد. توجه به این نکته مهم است که مناسب بودن Kubeflow برای یک کار خاص به نیازهای آن بستگی دارد. اگر مطمئن نیستید که Kubeflow گزینه مناسبی برای پروژه شماست، بهتر است با یک متخصص در زیرساخت یادگیری ماشین مشورت کنید. برخی مسائل که احتمالا Kubeflow برای آنها مناسب نیست، در ذیل آمده است:

* **پروژه های در مقیاس کوچک:** اگر روی یک پروژه یادگیری ماشین کوچک کار می کنید، ممکن است Kubeflow زیاده روی کند. راه اندازی و مدیریت یک خوشه Kubeflow می تواند زمان بر باشد و ممکن است ارزش تلاش برای یک پروژه کوچک را نداشته باشد.
* **مدل‌های ساده:** اگر مدل شما ساده است و به منابع محاسباتی زیادی نیاز ندارد، ممکن است به مقیاس‌پذیری و انعطاف‌پذیری که Kubeflow ارائه می‌کند نیاز نداشته باشید.
* **یادگیری ماشینی غیر توزیعی:** اگر نیازی به اجرای حجم کار یادگیری ماشینی خود در یک خوشه توزیع شده ندارید، ممکن است ابزارهای ساده تر و سبک تری در دسترس باشند.
* **بارهای کاری غیر کانتینری:** Kubeflow برای کار با حجم کاری کانتینری طراحی شده است، بنابراین اگر حجم کاری شما کانتینری نباشد، ممکن است ادغام آن با Kubeflow دشوار باشد.
* **محیط های غیر کوبرنتیز:** Kubeflow بر روی Kubernetes ساخته شده است، بنابراین اگر از Kubernetes به عنوان پلت فرم ارکستراسیون خود استفاده نمی کنید، ممکن است بهترین انتخاب برای شما نباشد. به طور کلی، اگر روی یک پروژه یادگیری ماشینی پیچیده و در مقیاس بزرگ کار می کنید که به محاسبات توزیع شده و کانتینری نیاز دارد، Kubeflow یک انتخاب عالی است. با این حال، اگر پروژه شما کوچک، ساده است یا به این ویژگی ها نیاز ندارد، ممکن است ابزارهای دیگری وجود داشته باشند که با نیازهای شما سازگارتر باشند.
* **یادگیری ماشین بی‌درنگ:** Kubeflow برای پردازش دسته‌ای طراحی شده است و ممکن است بهترین انتخاب برای برنامه‌های یادگیری ماشین بلادرنگ نباشد. برنامه های بلادرنگ نیاز به تأخیر کم و توان عملیاتی بالا دارند و Kubeflow ممکن است برای این نیازها بهینه نشده باشد.
* **بارهای کاری محاسباتی با کارایی بالا (HPC):** اگرچه Kubeflow را می توان برای برخی از بارهای کاری HPC استفاده کرد، اما ممکن است بهترین انتخاب برای همه انواع بارهای کاری HPC نباشد. Kubeflow برای کار با کلاسترهای توزیع شده طراحی شده است و ممکن است برای بارهای کاری HPC تک گره بهینه نشده باشد.
* **پژوهش در حوزه یادگیری ماشین:** Kubeflow در درجه اول برای تولید گردش کار یادگیری ماشین طراحی شده است و ممکن است بهترین انتخاب برای تحقیقات یادگیری ماشین نباشد. محققان اغلب به انعطاف‌پذیری و کنترل بیشتری بر محیط‌های یادگیری ماشینی خود نیاز دارند و Kubeflow ممکن است سطح سفارشی‌سازی مورد نیاز را ارائه ندهد.
* **پیش پردازش داده:** Kubeflow بر آموزش و استقرار یادگیری ماشین متمرکز است و ممکن است بهترین انتخاب برای کارهای پیش پردازش داده نباشد. ابزارهای دیگری مانند Apache Spark و Apache Beam وجود دارند که برای پیش پردازش داده ها مناسب تر هستند.

## ویژگی های بستر Kubeflow

* **محیط MLOps مبتنی بر ابر**

Kubeflow محیط‌های توسعه مدل ML را بهینه‌سازی شده برای ابر ارائه می‌دهد و امکان پیوند مبتنی بر Kubernetes را با نرم‌افزارهای منبع باز مختلف فراهم می‌کند.

* **راحتی استفاده برای داده های بزرگ**

محیط های استاندارد از طیف وسیعی از چارچوب های یادگیری ماشین از TensorFlow، PyTorch، scikit-learn و Keras پشتیبانی می کنند. خط لوله برای کل فرایندهای توسعه، یادگیری و استقرار مدل‌های یادگیری ماشینی خودکار است تا از پیکربندی/ایجاد ساده و همچنین استفاده مجدد از مدل‌ها اطمینان حاصل شود.

* **ویژگی های افزودنی**

Kubeflow منبع باز گسترده‌ای از اجرای کار یادگیری توزیعی/نظارت تا مدیریت خدمات استنتاج/تحلیل و مدیریت صف شغل در هر پلتفرمی ارائه می‌دهد. کاربران همچنین می‌توانند از زمان‌بندی‌های کار (FIFO، Bin-packing و Gang-based)، نظارت بر منابع GPU، گزارش موتور Kubeflow و ویژگی‌های اضافی دیگری که معمولاً در نرم‌افزارهای منبع باز در دسترس نیستند، استفاده کنند.

* **سایر ویژگی ها**
* نوت بوک Jupyter (توسعه مدل، یادگیری و استنتاج)
* اتوماسیون گردش کار (بر اساس خطوط لوله یادگیری ماشین)
* مدیریت زمانبندی کار GPU و صف کار
* نظارت بر منابع GPU
* مانیتورینگ/ ثبت نام Kubeflow و اجرای/نظارت کار یادگیری توزیع شده
* ساخت و مدیریت چارچوب ML (TensorFlow، PyTorch و غیره)
* مدیریت/تجزیه و تحلیل خدمات استنتاج و مدیریت آزمایش های مدل/گره های یادگیری

به‌طورکلی، هر یک از این مراحل نقش مهمی در چرخه زندگی یادگیری ماشین ایفا می‌کند و برای سازمان‌ها مهم است که یک فرآیند MLOps قوی و کارآمد را برای مدیریت چرخه عمر ML از ابتدا تا انتها اجرا کنند.

MLOps (عملیات یادگیری ماشین) مجموعه‌ای از شیوه‌ها و ابزارهایی است که به سازمان‌ها کمک می‌کند تا چرخه زندگی یادگیری ماشین (ML) را به طور موثر مدیریت کنند. اجزای مفهومی MLO ها به هم مرتبط هستند و برای دستیابی به هدف ارائه سیستم های ML قابل اعتماد، مقیاس پذیر و قابل نگهداری با هم کار می کنند. در زیر شرح مختصری از رابطه بین اجزای مفهومی MLO ها ارائه شده است:

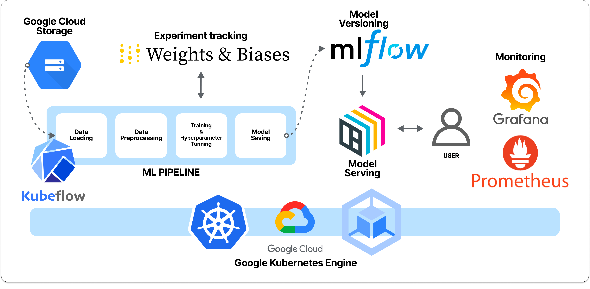
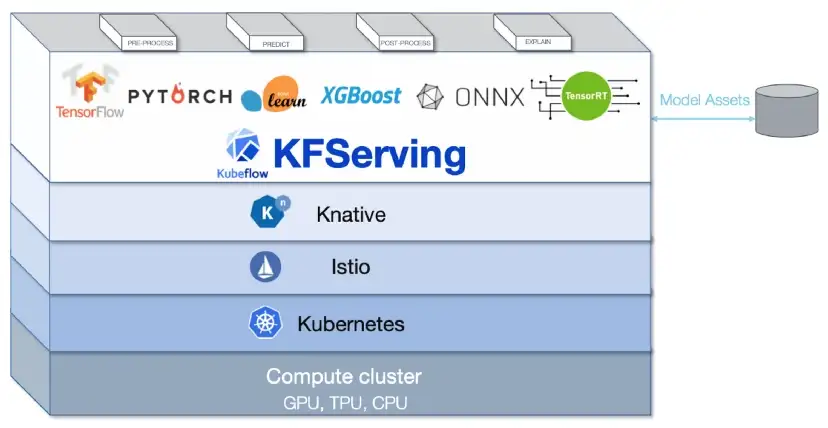
## معماری فیزیکی

تکمیل شده سند معماری فیزیکی در انتهای فاز یک تحویل میگردد.

## ساختار شماتیک بستر Kubeflow و تعامل با سایر سرویس‌ها در لایه فیزیکی

اجزای اصلی Kubeflow عبارتند از:

* Kubernetes: Kubeflow مبتنی بر Kubernetes ساخته شده و یک پلت فرم مقیاس پذیر و قابل اعتماد برای استقرار و مدیریت کانتینرها فراهم می کند.
* خطوط لوله: پلت فرمی برای ساخت، استقرار و مدیریت گردش کار یادگیری ماشینی سرتاسری است. این یک رابط بصری برای ایجاد و مدیریت گردش‌های کاری پیچیده و همچنین ابزارهایی برای آماده‌سازی داده‌ها، آموزش مدل و استقرار فراهم می‌کند.
* Kubeflow Serving: Serving بستری برای استقرار و مدیریت مدل های یادگیری ماشین در تولید است. این یک پلت فرم مقیاس پذیر و قابل اعتماد برای ارائه مدل ها به عنوان میکروسرویس، با پشتیبانی از استنتاج بلادرنگ و دسته ای فراهم می کند.
* اپراتورهای آموزشی Kubeflow: اپراتورهای آموزشی Kubeflow راهی برای استقرار و مدیریت مشاغل آموزشی توزیع شده در Kubernetes ارائه می دهند. آنها به شما اجازه می دهند با استفاده از چارچوب های یادگیری ماشینی محبوب مانند TensorFlow، PyTorch و XGBoost، مدل ها را آموزش دهید.
* Kubeflow Metadata: Kubeflow Metadata راهی برای ردیابی و مدیریت فراداده های مرتبط با گردش کار یادگیری ماشین ارائه می دهد. این به شما امکان می دهد اطلاعات مربوط به داده های مورد استفاده برای آموزش، مدل های آموزش دیده و فراپارامترهای مورد استفاده را ذخیره کنید.
* نوت بوک های Kubeflow Jupyter: نوت بوک های Kubeflow Jupyter راهی برای اجرا و مدیریت نوت بوک های Jupyter در Kubernetes ارائه می دهند. این به شما امکان می‌دهد گردش‌های کاری یادگیری ماشین خود را در یک محیط آشنا توسعه و آزمایش کنید.
* Kubeflow Katib: Kubeflow Katib یک چارچوب تنظیم هایپرپارامتر است که به شما امکان می دهد فرآیند یافتن هایپرپارامترهای بهینه را برای مدل های یادگیری ماشین خود به طور خودکار انجام دهید. از انواع الگوریتم های تنظیم پشتیبانی می کند و می تواند با فریم ورک های معروف یادگیری ماشینی مانند TensorFlow، PyTorch و XGBoost استفاده شود.
* داشبورد Kubeflow: داشبورد Kubeflow یک رابط مبتنی بر وب برای مدیریت و نظارت بر استقرار Kubeflow شما ارائه می دهد. این یک نمای جامع از تمام اجزای موجود در استقرار شما، و همچنین معیارها و گزارش‌های بلادرنگ برای هر جزء ارائه می‌کند.



معماری فیزیکی MLOها به نیازهای خاص سازمان و پروژه‌های یادگیری ماشینی در حال توسعه بستگی دارد. زیرساخت باید مقیاس پذیر، انعطاف پذیر و قادر به رسیدگی به نیازهای جریان کار یادگیری ماشین و ادغام و استقرار مداوم مدل‌های یادگیری ماشین باشد. معماری فیزیکی MLOps به زیرساخت فیزیکی، سخت افزار و اجزای نرم افزاری مورد استفاده برای پیاده‌سازی معماری منطقی MLOها اشاره دارد. این می‌تواند شامل موارد زیر باشد:

**زیرساخت محاسباتی:** این می‌تواند شامل سرورهای داخلی، ماشین‌های مجازی مبتنی بر ابر یا ترکیبی از هر دو باشد.

**ذخیره‌سازی داده‌ها:** این می‌تواند شامل پایگاه‌های داده، دریاچه‌های داده یا سیستم‌های فایل مورد استفاده برای ذخیره داده‌های آموزشی، داده‌های آزمایشی و خروجی‌های مدل باشد.

**شبکه سازی:** این می‌تواند شامل اجزای سخت افزاری و نرم افزاری باشد که از ارتباط بین اجزای مختلف معماری MLOps پشتیبانی می‌کند.

**نظارت:** این می‌تواند شامل ابزارهای نظارت، تحلیلگرهای گزارش و داشبوردهایی باشد که برای نظارت بر عملکرد مدل‌های یادگیری ماشین و گردش کار MLOps استفاده می‌شوند.

**امنیت:** این می‌تواند شامل فایروال‌ها، سیستم‌های کنترل دسترسی، رمزگذاری و سایر اقدامات امنیتی برای محافظت از داده‌ها و مدل‌های حساس درگیر در MLOها باشد.

**ابزارها و پلتفرم‌ها:** این می‌تواند شامل ابزارها و پلتفرم‌های نرم‌افزاری منبع باز و تجاری باشد که برای یادگیری ماشین استفاده می‌شوند، مانند TensorFlow، PyTorch و غیره.

برخی ابزار/پلتفرم برتر MLOps جهت مدیریت چرخه یادگیری ماشین عبارتنداز:

* Amazon SageMaker
* Azure Machine Learning
* TensorFlow Extended (TFX)
* MLFlow
* Google Cloud ML Engine
* Data Version Control (DVC)
* H2O Driverless AI
* Kubeflow
* و ...

# ویژگی‌های کارکردی

Kubeflow یک پلت فرم منبع باز است که برای ساده سازی توسعه، استقرار و مدیریت گردش کار یادگیری ماشین در Kubernetes طراحی شده است. ویژگی های کاربردی را می توان به صورت زیر خلاصه کرد:

* **مدیریت داده ها شامل Data gathering و Data analysis:** MLOps ویژگی های مدیریت داده مانند کنترل نسخه، خطوط لوله داده و پیش پردازش داده را ارائه می دهد که به شما امکان می دهد فرآیند مدیریت داده ها را ساده کنید و اطمینان حاصل کنید که داده های شما به درستی برچسب گذاری و ذخیره می شوند.
* **:Data transformation-preparation** با سیستم های ذخیره سازی داده های مختلف مانند ذخیره سازی اشیا، سیستم های فایل و پایگاه های داده ادغام می شود. این به کاربران اجازه می دهد تا به راحتی به داده های خود در محیط Kubeflow دسترسی داشته باشند و آنها را مدیریت کنند.
* **Offline/ Online feature store:**
* **سرویس‌دهی مدل:** MLOpsیک زیرساخت سرویس‌دهی مدل مقیاس‌پذیر ارائه می‌دهد که به شما امکان می‌دهد مدل‌های یادگیری ماشین را به‌عنوان ریز سرویس‌ها در Kubernetes استقرار دهید. این ویژگی به شما امکان می دهد مدل های خود را از طریق REST API در معرض دید قرار دهید و ادغام آنها در برنامه های خود را آسان می کند.
* **تنظیم Hyperparameter**: MLOpsمجموعه‌ای از ابزارها را برای خودکارسازی فرآیند تنظیم پارامترها ارائه می‌کند که به شما امکان می‌دهد مجموعه بهینه‌ای از پارامترها را برای مدل خود به سرعت پیدا کنید.
* **Model monitoring**: MLOps به شما امکان می‌دهد آزمایش‌های مختلف را ردیابی و مقایسه کنید و به شما امکان می‌دهد عملکرد مدل‌های خود را در طول زمان پیگیری کنید و تصمیم‌گیری آگاهانه در مورد اینکه کدام مدل‌ها را اجرا کنید، بگیرید. همچنین ابزارهای تجسم را برای کمک به شما در نظارت بر پیشرفت گردش کار یادگیری ماشینی، تجسم نتایج آزمایشات و اشکال زدایی مشکلات مدل های خود ارائه می دهد.
* **استقرار خودکار**  MLOpsمی‌تواند استقرار خطوط لوله یادگیری ماشین را با استفاده از ابزارهایی مانند Argo Workflows خودکار کند و کاربران را قادر می‌سازد تا به راحتی گردش‌های کاری یادگیری ماشینی پیچیده را ایجاد، اجرا و مدیریت کنند.
* **نسخه‌سازی مدل:** MLOps یک سیستم کنترل نسخه برای مدل‌های یادگیری ماشین ارائه می‌کند، که به کاربران امکان می‌دهد نسخه‌های مختلف مدل‌های خود را پیگیری کنند و به راحتی بین آنها جابجا شوند.

# ویژگی‌های غیرکارکردی

به طور کلی، ویژگی‌های غیر کاربردی Kubeflow برای اطمینان از اینکه جریان‌های کاری ML می‌توانند به شیوه‌ای قابل اعتماد، ایمن و مقیاس‌پذیر مستقر و مدیریت شوند، حیاتی هستند. Kubeflow با ارائه این ویژگی‌ها، سازمان‌ها را قادر می‌سازد تا بر توسعه و استقرار مدل‌های ML با کیفیت بالا بدون نگرانی در مورد زیرساخت‌های اساسی و پیچیدگی‌های عملیاتی تمرکز کنند. Kubeflow علاوه بر ارائه طیف گسترده‌ای از ویژگی‌های کاربردی، تعدادی ویژگی غیرعملکردی را نیز ارائه می‌کند که برای اطمینان از قابلیت اطمینان، مقیاس‌پذیری و امنیت گردش‌های کاری ML حیاتی هستند.

برخی از ویژگی های کلیدی غیر کاربردی Kubeflow عبارتند از:

* **مقیاس پذیری:** به کاربران امکان می‌دهد تا بسته به تقاضا، جریان‌های کاری ML خود را افزایش یا کاهش دهند، که کمک می‌کند تا اطمینان حاصل شود که گردش‌های کاری می‌توانند حجم زیادی از داده و قدرت پردازش را مدیریت کنند.
* **امنیت:** تعدادی از ویژگی های امنیتی، از جمله مکانیسم های احراز هویت و مجوز، رمزگذاری داده ها در حال انتقال و در حالت استراحت، و ذخیره امن اسرار و اعتبار را ارائه می دهد.
* **قابلیت اطمینان**MLOps: برای ارائه یک پلت فرم بسیار قابل اعتماد برای گردش کار ML طراحی شده است. این ویژگی هایی مانند تحمل خطا، خود ترمیم و مقیاس خودکار را ارائه می دهد که به اطمینان از اینکه جریان های کاری ML می توانند بدون وقفه یا خرابی کار کنند کمک می کند.
* **قابلیت حمل:** کاربران می توانند به راحتی گردش کار ML خود را از یک محیط به محیط دیگر منتقل کنند. این امر از طریق استفاده از Containerization و Kubernetes به دست می آید، که یک پلت فرم ثابت برای اجرای گردش های کاری ML بدون توجه به زیرساخت های اساسی فراهم می کند.
* **تحمل خطا:** MLOps از مکانیسم‌های تحمل خطا داخلی Kubernetes مانند کنترل‌کننده‌های تکرار و مجموعه‌های حالتی استفاده می‌کند تا اطمینان حاصل کند که بارهای کاری یادگیری ماشین در صورت خرابی گره یا مشکلات شبکه مختل نمی‌شود.
* **سفارشی سازی:** MLOps با قابل تنظیم با طیف وسیعی از API ها و نقاط افزونه به شما امکان می دهد عملکرد خود را با سیستم های موجود ادغام کنید. این به شما امکان می دهد پلتفرم را با نیازهای خاص خود تنظیم کنید و راه حل های سفارشی ایجاد کنید که نیازهای منحصر به فرد شما را برآورده کند.
* **مانیتورینگ و**  **نظارت و مدیریت لاگ‌ها:** مانیتورینگ فرآیند ردیابی عملکرد مدل‌های ML در تولید است. نظارت شامل جمع‌آوری داده‌ها در مورد پیش‌بینی‌های مدل، مقایسه آن‌ها با نتایج واقعی، و شناسایی هر گونه اختلاف یا خطا است. مانیتورینگ به شناسایی مشکلات احتمالی سیستم، مانند جابجایی مدل کمک می‌کند و پاسخ سریع برای حفظ عملکرد سیستم را ممکن می‌سازد.

# فرضیات و محدودیت‌ها

با توجه به محدودیت ها در موارد زیر، حداکثر متناسب با سخت افزار تامین شده در فضای زیرساخت فضای Deployment تعهد از سمت پیمانکار صورت می‌گیرد که پس از تست در فضای stage قابلیت تخمین وجود خواهد داشت.

* تعداد پاپلاین‌ها
* تعداد مدل‌ها
* انواع مدل‌ها مورد پشتیبانی
* نرخ داده ورودی مورد استفاده در یادگیری ماشین
* نرخ و تعداد سرو مدل
* نرخ استخراج و تعداد فیچرهای موجود در مخزن آنلاین و آفلاین

همچنین در تحویل دهی پرو‌ژه MLOPs،‌ محدودیت های زیرساختی زیر لحاظ خواهند شد:

* حداکثر تعداد کلاستر
* حداکثر تعداد پاد
* حداکثر تعداد مدل برای یادگیری
* حداکثر تعداد مدل برای استنتاج
* حداکثر تعداد درخواست

## مشکل مقیاس پذیری استقرار مدل

در حالی که با ساختن مدل‌هایی برای هر مورد استفاده از دقت استنتاج و حفظ حریم خصوصی داده بهتر بهره می‌برید، استقرار هزاران تا صدها هزار مدل در یک خوشه Kubernetes چالش‌برانگیزتر است. علاوه بر این، تعداد فزاینده‌ای از موارد استفاده از سرویس‌دهی مدل‌های مبتنی بر شبکه عصبی وجود دارد. برای دستیابی به تأخیر معقول، این مدل‌ها بهتر روی پردازنده‌های گرافیکی ارائه می‌شوند. با این حال، از آنجایی که پردازنده‌های گرافیکی منابع گران قیمتی هستند، سرویس دهی به بسیاری از مدل‌های مبتنی بر GPU پرهزینه است.طراحی اصلی KFServing یک مدل را برای هر InferenceService مستقر می کند. اما، وقتی با تعداد زیادی مدل سروکار داریم، پارادایم «یک مدل، یک سرور» چالش‌هایی را برای یک خوشه Kubernetes ایجاد می‌کند. برای مقیاس‌بندی تعداد مدل‌ها، باید تعداد InferenceServices را مقیاس کنیم، چیزی که می‌تواند به سرعت محدودیت‌های خوشه را به چالش بکشد. سرویس چند مدل برای رسیدگی به سه نوع محدودیت طراحی شده است که KFServing با آن مواجه خواهد شد: 1) محدودیت منابع محاسبه، 2) حداکثر محدودیت pods، 3) حداکثر محدودیت آدرس IP.

## محاسبه محدودیت منابع

هر InferenceService یک سربار منبع دارد، به دلیل اینکه در هر پادکارهای جانبی تزریق می شود. این معمولاً حدود 0.5 CPU و منبع حافظه 0.5 G را به ازای هر نسخه InferenceService اضافه می کند. به عنوان مثال، اگر 10 مدل را مستقر کنیم، هر کدام با 2 کپی، سربار منبع 10 \* 2 \* 0.5 = 10 CPU و 10 \* 2 \* 0.5 = 10 گیگابایت حافظه است. سربار منبع هر مدل 1 CPU و 1 گیگابایت حافظه است. استقرار بسیاری از مدل ها با استفاده از رویکرد فعلی، به سرعت از منابع محاسباتی خوشه استفاده می کند. با سرویس چند مدل، این مدل ها را می توان در یک InferenceService بارگذاری کرد، سپس میانگین سربار هر مدل 0.1 CPU و 0.1 گیگابایت حافظه است. برای مدل‌های مبتنی بر GPU، تعداد GPUهای مورد نیاز به صورت خطی با افزایش تعداد مدل‌ها افزایش می‌یابد، که مقرون به صرفه نیست. اگر بتوان چندین مدل را در یک سرور مدل مجهز به GPU مانند TritonServer بارگذاری کرد، به GPUهای بسیار کمتری در کلاستر نیاز داریم.

## حداکثر محدودیت Pod

Kubelet دارای حداکثر تعداد پادها در هر گره با محدودیت پیش‌فرض روی 110 است. طبق بهترین روش Kubernetes، یک گره نباید بیش از 100 پاد را اجرا کند.

با این محدودیت، یک خوشه معمولی 50 گره با محدودیت Pod پیش‌فرض می‌تواند حداکثر 1000 مدل را اجرا کند با این فرض که هر InferenceService به طور متوسط دارای 4 Pod است (دو کپی ترانسفورماتور و دو نسخه پیش‌بینی‌کننده).

## حداکثر محدودیت آدرس IP

خوشه های Kubernetes همچنین دارای محدودیت آدرس IP در هر خوشه هستند. هر پاد در InferenceService به یک IP مستقل نیاز دارد. به عنوان مثال، یک خوشه با 4096 آدرس IP می تواند حداکثر 1024 مدل را با فرض اینکه هر InferenceService به طور متوسط دارای 4 پاد باشد (دو ماکت ترانسفورماتور و دو نسخه پیش بینی کننده) استقرار دهد.

## مزایای استفاده از سرویس چند مدل

ویژگی سرویس دهی چند مدل برای رفع سه محدودیت بالا طراحی شده است. میانگین سربار منابع در هر مدل را کاهش می دهد بنابراین استقرار مدل مقرون به صرفه تر می شود. و تعداد مدل‌هایی که می‌توانند در یک خوشه مستقر شوند دیگر با حداکثر محدودیت pods و حداکثر محدودیت آدرس IP محدود نخواهد شد.

# تست و ارزیابی MLOPs

تکمیل شده سناریوی تست ویژگی های کارکردی و غیرکارکردی در انتهای فاز یک تحویل میگردد.

## ارزیابی واحد، تجمیعی و کارایی بخش های کارکردی MLOPs

**Unit test**

در این تست دو مرحله کلی pipeline و serving وجود دارد:

* Pipeline: تمام مراحل یادگیری و در بعضی مواقع ارزیابی های مدل صورت میگیرد در مرحله اول مدل طراحی میشود. در مثال فعلی سه بخش داریم preprocess, model\_fit و evaluation که در بخش preprocess داده ها از دیتاست یا دیتابیس دریافت، آماده سازی و به داده های آموزش و آزمون تقسیم و در نهایت در data wherhouse/data lake ذخیره می شوند. تمامی مراحل در یک تابع انجام می شود. از آنجایی که هیچ ورودی و خروجی نداریم و تمام دیتاها در پایگاه داده ذخیره می شوند، برای آزمایش این مولفه با چک کردن دیتای موجود آزمایش میکنیم که خروجی ها وجود دارند یا خیر.

با توجه به اینکه هر مولفه یک تابع است، ما برای هر کدام یک unit test طراحی میکنیم و در کنار pipeline در هنگام آموزش یا اجرای experiment آنها آزمایش های لازم را از توابع میگیرند و برای هر یک از مولفه های موجود در pipeline تست به صورت جداگانه نوشته میشود و نتایج هر تست نیز در database ذخیره میشود.

* Serving: در این مرحله آزمون برای هر کدام از متد های کلاس استنتاج می باشد. کلاس استنتاج دارای دو متد اصلی load data و predict می باشد. برای هر کدام از این بخش ها unit test جداگانه طراحی می شود تا عملکرد استنتاج را ارزیابی کند. برای load data عملکرد به این گونه می باشد که آیا مدل واقعا در مرحله استنتاج کار می کند یا خیر. برای predict نیز یک ورودی به صورت آزمایشی به مدل داده شده و خروجی مورد نظر بررسی خواهد شد. نتایج این آزمایش ها نیز در database ذخیره می شود.
* Feast:در این ارزیابی روند اجرای جریان داده و اتصال بخش های مختلف Feast چک خواهد شد.

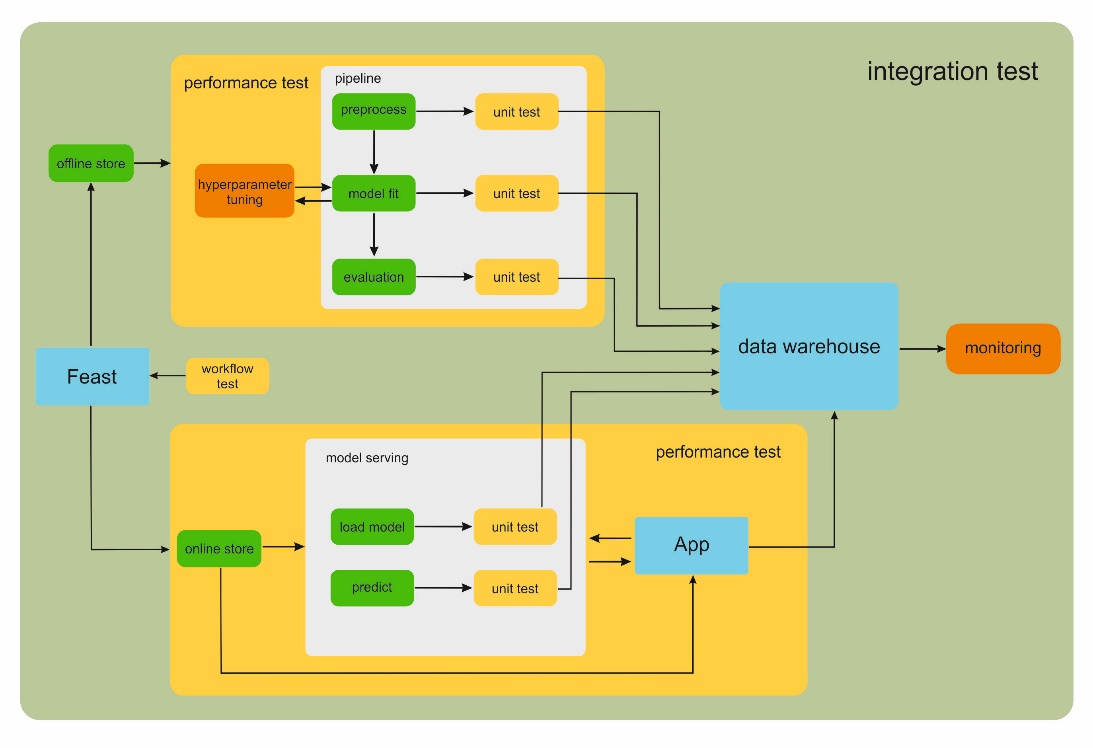
**Performance test**

در بخش ارزیابی عملکرد نیز ما دو مرحله کلی داریم که کاملا مجزا عمل میکنند:

* Pipeline: همانطور که در شکل مشاهده میکنید perforamnce test در این بخش عملکرد و همچنین مدت زمان اجرای مدل را آزمایش میکند. به اینگونه که مدل، داده های مورد نیاز را دریافت کرده و pipeline شروع به اجرا میکند در مرحله model fit بهترین پارامترها به وسیله hyperparamet tuning انتخاب می شود و مدل فراید آموزش خود را به اتمام میرساند. در این فرایند بررسی کردن آنکه باید بهترین خروجی ممکن از مدل دریافت شود و بالاترین کارایی را داشته باشد، وظیفه perfomance test می باشد.
* Serving: در این استنتاج همانطور که مشاهده میکنید سه بخش کلی داریم. بخش اول online store، بخش دوم که serving نهایی و بخش سوم که یک app برای serve کردن مدل نهایی تعبیه شده است. این آزمون به این گونه عمل میکند که داده های آنلاین از Feast دریافت شده و به دو بخش app و serving ارسال میشود. داده های online کاملا دارای برچسب یا جواب صحیح می باشد. وظیفه app درخواست از serving برای پاسخ داده ها می باشد و خروجی نهایی را با خروجی واقعی مقایسه کرده و کارایی مدل نهایی را مورد ارزیابی قرار میدهد. این امر باعث می شود تا مدل همواره در حال performance test باشد.

**Integration test**

در این بخش آزمایش کلی برای تمام بخش های MLops اعمال خواهد شد. برای integration test ابتدا شروع خط لوله و اتمام آن را مورد ارزیابی قرار میدهیم. به این شکل که کل pipeline اجرا شده دیتاهای لازم در data wherhouse/data lake ذخیره شوند و ارزیابی مدل صورت گرفته باشد.



# آموزش و تحویل

**شرایط ، آزمون و تحویل گیری**

1. پیمانکار موظف به ارائه طرح آزمون پروژه است. این طرح آزمون معرف مجموعه تست‌هایی است که پیمانکار اعلام می‌نماید که خروجی خود را با آن تست کرده و موظف است نتایج تست را ارائه نماید. شایان ذکر است که در مراحل اجرایی قرارداد، نتایج حاصل از طرح آزمون می‌بایست همراه محصول تحویل کارفرما گردد. طرح آزمون پروژه باید به تایید ناظر کارفرما برسد.
2. لازم به ذکر است که در صورت تست سناریو اعلامی توسط کارفرما و عدم دستیابی به نتایج اعلام شده، محصول/خروجی معیوب تلقی شده و پیمانکار موظف است در زمان پروژه، عیوب را رفع کرده تا در نهایت، به خروجی مدنظر دست یابند.
3. مراحل آزمون و تحویل اولیه تجهیزات/سامانه‌ها در محل پیمانکار یا شرکت ثالث مورد تأیید کارفرما و مطابق با طرح آزمون ارائه شده و با هزینه پیمانکار انجام خواهد شد. تأمین کلیه دستگاه‌ها/سامانه‌های اندازه‌گیری و آزمون بر عهده پیمانکار است.

**نصب، راه‌اندازی و تحويل سيستم**

پیمانکار موظف است خروجی‌های هر فاز از سیستم را جداگانه راه‌اندازی، تحویل و آموزش دهد و تایید نهایی خروجی هر فاز منوط به تایید عملکرد آن فاز خواهد بود.

تبصره: تحویل‌گیری نهایی سامانه منوط به یکپارچگی بخش‌های مختلف و تایید کارکرد نهایی سامانه است.

* ارائه مستنداتی پیرامون مراحل نصب و راه اندازی کلیه مولفه‌های اصلی بستر Mlops
* به روزرسانی مستندات به ازای هر نسخه از مولفه‌های اصلی بستر Mlops
* کلیه مراحل مندرج در مستندات توسط تیم سحاب با حضور نماینده شرکت آدین، اجرا و صحت سنجی خواهد شد.

**آموزش**

شرکت آدین موظف به برگزاری دوره آموزشی با شرایط ذیل است:‌

1. آموزش جامع بهره‌برداری و استفاده از محصول پروژه (مواردی همچون نحوه نصب، راه‌اندازی، نگهداری، توسعه و نیز بهره‌برداری برای تجهیزاتی/سامانه‌هایی که نیاز به آموزش دارند) می‌بایست حتماً در طرح آموزش پروپوزال دیده شود.
2. شرکت آدین موظف به تحویل سری کامل از مستندات آموزشی تمامی اجزاء سیستم با جزئیات کامل به‌صورت نسخه چاپی (Hard Copy) و نسخه الکترونیکی (Soft Copy-CD)، به تعداد شرکت کنندگان دوره آموزشی است.
3. آموزش‌های راهبری و کاربری برای کارشناسان ذیربط در محل مورد تایید نماینده فنی سحاب در خصوص تکمیل فاز آموزش، باید به‌صورت کارگاه و یا کلاس آموزشی ارائه گردد. آموزش ‌می‌بایست در دو سطح اپراتوری و مدیر سیستم برای حداقل 100 نفر-ساعت برگزار شود.
4. برگزاری جلسات آموزش مفاهیم کلی و نصب و مدیریت بستر Mlops
5. تحویل تمامی مستندات مربوط به فاز توسعه
6. تاییدیه نماینده فنی سحاب در خصوص تایید صحت مستندات تولید شده

**لیست مستندات قابل تحویل**

تحویل دادنی‌های مرسوم به عنوان مستندات، در مستند ضمائم پروژه به طور کامل ذکر شده است. مستندات و قالب تهیه آن می‌بایست مورد تایید کارفرما قرارگیرد. به طور مثال مستند نصب و راه‌اندازی باید در دو فرمت مستندات مکتوب و ویدیویی فراهم گردد. برخی از مستندات مهم به شرح زیر است:

* مستند نیازمندی و معماری
* مستند راه اندازی و استقرار
* مستند آزمون
* مستند نحوه استفاده

# منابع موردنیاز

منابع مورد نیاز در production به گونه‌ای در نظر گرفته شود که برای حل مسئله‌ای با ابعاد داده‌ای حداکثر ۱۰ برابر مسئله کشف رفتارهای مخاطره‌آمیز در دفاتر نمایندگی اپراتورهای همراه مناسب باشد. پس از تعیین منابع production توسط آدین ابعاد stage نیز معادل ۲۰ درصد محیط production توسط کارفرما فراهم خواهد شد. حداقل تعداد GPU در فضای stage حداقل ۲ می باشد.

**منابع پیشنهادی در محیط Production**

* 128 x 20 TB RAM
* 2 x 20 TB HDD
* 40 x 20 Core CPU
* 4 GPU, RTX 4090

# زمان‌بندی و شکست کارها

مراحل اجرای Kubeflow معمولاً شامل موارد زیر است:

* نصب: Kubeflow را روی زیرساخت هدف، از جمله خوشه‌های Kubernetes و هر گونه وابستگی لازم، نصب و پیکربندی کنید.
* آماده‌سازی داده‌ها: داده‌ها را جهت استفاده در Kubeflow آماده و سازماندهی کنید، از جمله داده‌ها، پاک‌سازی و عادی‌سازی.
* توسعه مدل: مدل‌های یادگیری ماشین را با استفاده از ابزارهای داخلی Kubeflow یا ادغام با سایر چارچوب‌های ML مانند TensorFlow، PyTorch و Jupyter توسعه دهید.
* آموزش مدل: مدل‌ها را روی داده‌های مقیاس بزرگ با استفاده از قابلیت‌های کوچک‌سازی Kubeflow، از جمله آموزش توزیع‌شده و شتاب GPU، آموزش دهید.
* استقرار مدل: با استفاده از ابزارهای استقرار Kubeflow، از جمله TensorFlow Serving، Seldon، و KFServing، مدل‌ها را جهت تولید مستقر کنید.
* نظارت و نگهداری: مدل‌های در حال تولید، از جمله عملکرد، دقت و جابجایی داده‌ها را پایش کنید و مدل‌ها را در صورت لزوم به‌روزرسانی کنید.
* یکپارچه سازی مداوم و استقرار مستمر (CI/CD): فرآیند یکپارچه سازی، آزمایش و استقرار مدل‌ها را با استفاده از ابزارهایی مانند جنکینز یا GitLab به صورت خودکار انجام دهید.

# گانت چارت کلی پروژه

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **کلیات هر فاز** | فاز یک  ( اردیبهشت- خرداد) | | فاز دو  ( تیر، مرداد، شهریور) | | | فاز سه  ( مهر- آبان ـ آذر) | | | فاز چهار  (دی- بهمن- اسفند) | | | فاز پنج  (فروردین-اردیبهشت-خرداد) | | |
|  | **اردیبهشت** | **خرداد** | **تیر** | **مرداد** | **شهریور** | **مهر** | **آبان** | **آذر** | **دی** | **بهمن** | **اسفند** | **فروردین** | **اردیبهشت** | **خرداد** |
| **استقرار kubeflow بر روی بستر کوبرنتیز به همراه ماژول های ML** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **تعریف دقیق صورت مسئله شاهکار** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **آماده‌سازی داده‌ها** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **توسعه مدل** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **طراحی ازمون** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ایجاد پایپلاین CI/CD یادگیری مدل مسئله** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **استنتاج مدل در داشبورد Kubeflow** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **مانیتورینگ و نظارت داشبورد Kubeflow** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Off-line feature store** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **On-line feature store** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **مستندسازی** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **تعریف دقیق صورت مسئله دوم** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **آماده‌سازی داده‌ها** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **توسعه مدل** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ایجاد پایپلاین CI/CD یادگیری مدل مسئله** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **استنتاج مدل در داشبورد Kubeflow** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **مانیتورینگ و نظارت داشبورد Kubeflow** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Off-line feature store** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **On-line feature store** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **مستندسازی** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# شکست کار پروژه

**جدول شکست کار پروژه**

| WBS | شرح | مدت‌ زمان (ماه) | پیش‌نیاز | مسئول |
| --- | --- | --- | --- | --- |
| فاز اول | | | | |
| ۱.۱ | **تدوین سند نیازمندی** | ۲ |  | پیمانکار |
| ۱.۲ | **ارایه نیازمندی سخت‌افزار موردنیاز آزمایشگاه کارفرما** |  | پیمانکار |
| ۱.۳ | **تحویل نمونه داده مسئله شاهکار** |  | کارفرما |
| ۱.۴ | **ارایه معماری سامانه** |  | پیمانکار |
| ۱.۵ | **راه‌اندازی بستر اجرای stage** | ۱.۱و ۱.۴ | کارفرما |
| ۱.۶ | **تعریف دقیق صورت مسئله شاهکار** | ۱.۳ | پیمانکار |
| ۱.۷ | **استقرار kubeflow بر روی بستر کوبرنتیز به همراه ماژول های ML** | ۱.۵ |  |
| ۱.۸ | **آماده‌سازی داده‌ها** | ۱.۳ | پیمانکار |
| ۱.۹ | **توسعه مدل** | ۱.۸ | پیمانکار |
| ۱.۱۰ | **طراحی ازمون** |  |  |
|  |  |  |  |
| فاز دوم | | | | |
|  |  |  |  |  |
| 2.۱ | **ایجاد پایپلاین CI/CD یادگیری مدل مسئله شاهکار** |  | ۱.۸ | پیمانکار |
| 2.۲ | **استنتاج مدل در داشبورد Kubeflow** | ۳ | ۱.۴ | پیمانکار |
| 2.۳ | **تعریف مسئله دوم ML** |  | کارفرما |
| ۲.۴ | **تحویل داده مسئله دوم ML** |  | کارفرما |
| ۲.۵ | **تعریف دقیق صورت مسئله دوم ML** |  | ۲.۴ | پیمانکار |
| فاز سوم | | | | |
| ۳.۱ | **مانیتورینگ و نظارت داشبورد Kubeflow** | ۳ | ۲.۱ | پیمانکار |
| ۳.۲ | **Off-line feature store** | ۲.۲ | پیمانکار |
| 3.۴ | **آماده‌سازی داده‌ها** | ۲.۵ | پیمانکار |
| 3.۵ | **توسعه مدل** | ۳.۴ | پیمانکار |
|  |  |  |  |
| فاز چهارم | | | | |
| ۴.۱ | **On-line feature store** | ۳ | ۲.۲ | پیمانکار |
| ۴.۲ | **مستندسازی** | ۴.۱ | پیمانکار |
| ۴.۳ | **ایجاد پایپلاین CI/CD یادگیری مدل مسئله دوم** | ۳.۵ | پیمانکار |
| ۴.۴ | **استنتاج مدل در داشبورد Kubeflow** |  | ۴.۳ | پیمانکار |
|  |  |  |  |  |
| فاز پنجم |  |  |  |  |
| ۵.۱ | **مانیتورینگ و نظارت داشبورد Kubeflow** |  | ۴.۴ | پیمانکار |
| ۵.۲ | **Off-line feature store** |  | ۴.۴ | پیمانکار |
| ۵.۳ | **On-line feature store** |  | ۴.۴ | پیمانکار |
| ۵.۴ | **مستندسازی مسئله اول و دوم** |  | ۵.۳ و ۴.۲ | پیمانکار |
|  |  |  |  |  |

1. Model drift [↑](#footnote-ref-1)
2. <https://www.samsungsds.com/vn/ai-kubeflow/kubeflow.html> [↑](#footnote-ref-2)