بسم الله الرحمن الرحیم

**مستند سازی پروژه MLOps**

**شرکت آدین**

****

**اسفند ۱۴۰۱**

**نسخه ۰.۶**

فهرست

[۱. مشخصات کلی............................................................................................................................4](#_Toc127901742)

[نکات مثبت MLOP 5](#_Toc127901743)

[معایب MLOps 5](#_Toc127901744)

[Kubeflow چیست؟ 5](#_Toc127901745)

[معماری سرویس Kubeflow 6](#_Toc127901746)

[ویژگی های سرویس Kubeflow 6](#_Toc127901747)

[مقایسه Kubeflow و MLflow 7](#_Toc127901748)

[نمونه های جهانی استفاده از Kubeflow 7](#_Toc127901749)

[۲. معماری و تکنولوژی‌های مرتبط](#_Toc127901750) ...........................................................................................................8

[معماری مفهومی 8](#_Toc127901751)

[مرحله آماده سازی داده‌ها در چرخه حیات ML 9](#_Toc127901752)

[معماری منطقی 9](#_Toc127901753)

[معماری فیزیکی 10](#_Toc127901754)

[بهترین ابزار منبع باز برای جمع‌آوری داده ها 10](#_Toc127901755)

[ذخیره سازی داده‌ها 11](#_Toc127901756)

[بهترین دریاچه داده برای MLOps 14](#_Toc127901757)

[3. ویژگی‌های کارکردی](#_Toc127901758)

[شکاف‌های ارتباطی تیم‌های تجاری و فنی 16](#_Toc127901759)

[ارزیابی ریسک 16](#_Toc127901760)

[بازتاب تغییر اهداف کسب و کار در مدل 16](#_Toc127901761)

[۴. ویژگی‌های غیرکارکرد16.……………………………………………………………………………………………………………………](#_Toc127901762)

[آمادگی hybrid-cloud 16](#_Toc127901763)

[امنیت 16](#_Toc127901764)

[مقیاس پذیری 16](#_Toc127901765)

[تحمل خطا 16](#_Toc127901766)

[ارائه توان عملیاتی بالا 16](#_Toc127901767)

[پاسخ سریع 16](#_Toc127901768)

[۵. فرضیات و محدودیت‌ها ...................................................................................................................16](#_Toc127901769)

[۶. تست و ارزیابی MLOPs..................................................................................................................17](#_Toc127901770)

[تست مدل 17](#_Toc127901771)

[تست برنامه 17](#_Toc127901772)

[۷. آموزش و تحویل نصب و راه اندازی MLOPs............................................................................................17](#_Toc127901773)

[آموزش نصب Kubeflow 18](#_Toc127901774)

[راه اندازی Kubeflow 18](#_Toc127901775)

[داشبورد Kubeflow 22](#_Toc127901776)

[بخش های مختلف kubeflow به همراه نحوه کار آنها 22](#_Toc127901777)

[خط لوله kubeflow 25](#_Toc127901778)

[ایجاد و بارگذاری خط لوله 29](#_Toc127901779)

[تجربه اجرای یک خط لوله (Experiments kfp) 32](#_Toc127901780)

[Endpoint 33](#_Toc127901781)

[Kserve 33](#_Toc127901782)

[۸. منابع موردنیاز......................................................................................................................36](#_Toc127901783)

[۹. زمان‌بندی و شکست کارها........................................................................................................36](#_Toc127901784)

[۱۰. تحلیل هزینه.....................................................................................................................37](#_Toc127901785)

# مشخصات کلی

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 (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

## MLOPsبرای چه مشتریانی مناسب است؟

* بهبود سرعت ایجاد و استقرار مدل در نتیجه مدیریت جامع چرخه زندگی یادگیری ماشین، به دلیل افزایش ارزش یادگیری ماشین.
* MLOها همکاری و شفافیت را در میان تیم‌های dataOps، مهندسین یادگیری ماشین، تحلیلگران تجاری/مدیران محصول، مهندسین تضمین کیفیت و مهندسین زیرساخت افزایش می‌دهند.
* MLOps زمان بازاریابی جهت الگوریتم‌های یادگیری ماشین را کاهش می‌دهد و روش‌های یکپارچه‌سازی و تحویل مداوم، استقرار این سیستم‌ها را در تولید آسان‌تر می‌کند. 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.
* بهبود کیفیت مدل: با خودکارسازی آموزش، اعتبارسنجی و استقرار مدل‌ها، احتمال خطای انسانی کاهش می‌یابد که منجر به عملکرد بهتر مدل می‌شود.
* زمان سریع‌تر جهت بازاریابی: MLOps دانشمندان و مهندسان داده را قادر می‌سازد تا سریع‌تر تکرار کنند و مدل‌ها را سریع‌تر به تولید بفرستند.
* 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.
* 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.

## Kubeflow چیست؟

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؟

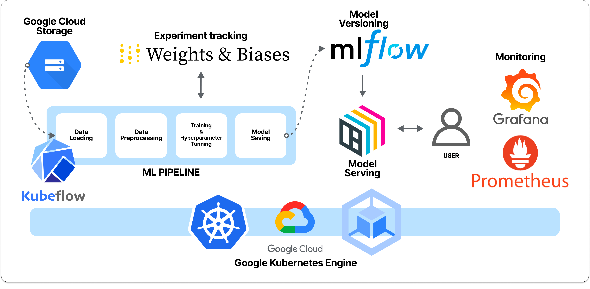
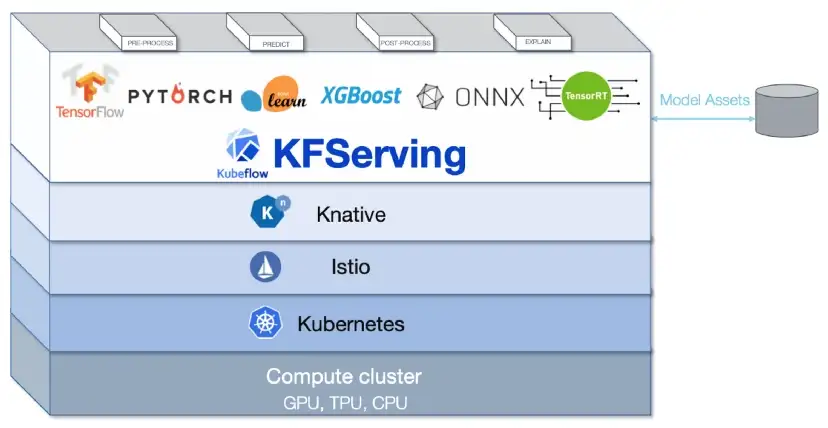
Kubeflow یک پلت فرم یادگیری ماشینی قدرتمند است که می تواند برای طیف وسیعی از وظایف مورد استفاده قرار گیرد. با این حال، انواع مختلفی از مشکلات وجود دارد که ممکن است Kubeflow بهترین انتخاب برای آنها نباشد. توجه به این نکته مهم است که مناسب بودن Kubeflow برای یک کار خاص به نیازهای آن بستگی دارد. اگر مطمئن 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:

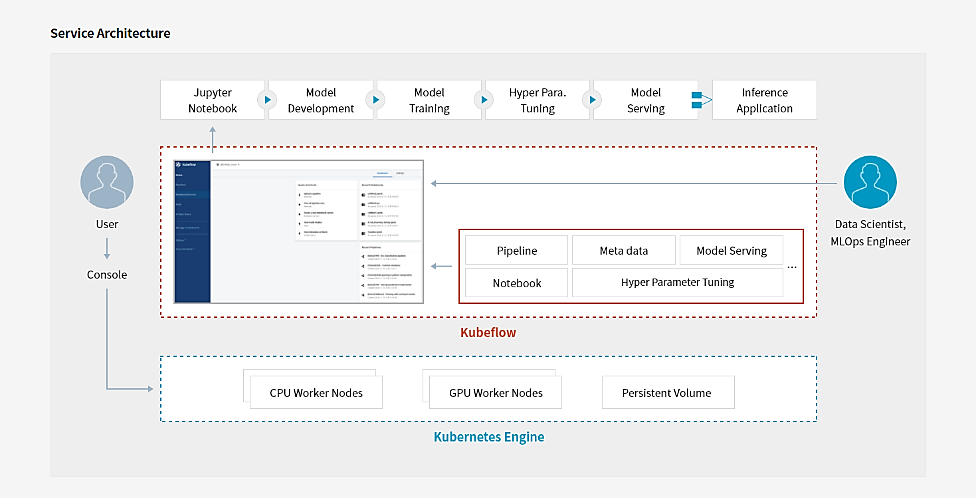
* **پروژه های در مقیاس کوچک:** اگر روی یک پروژه یادگیری ماشین کوچک کار می کنید، ممکن است Kubeflow زیاده روی کند. راه اندازی و مدیریت یک خوشه Kubeflow می تواند زمان بر باشد و ممکن است ارزش تلاش برای یک پروژه کوچک را نداشته باشد.
* **مدل‌های ساده:** اگر مدل شما ساده است و به منابع محاسباتی زیادی نیاز ندارد، 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.

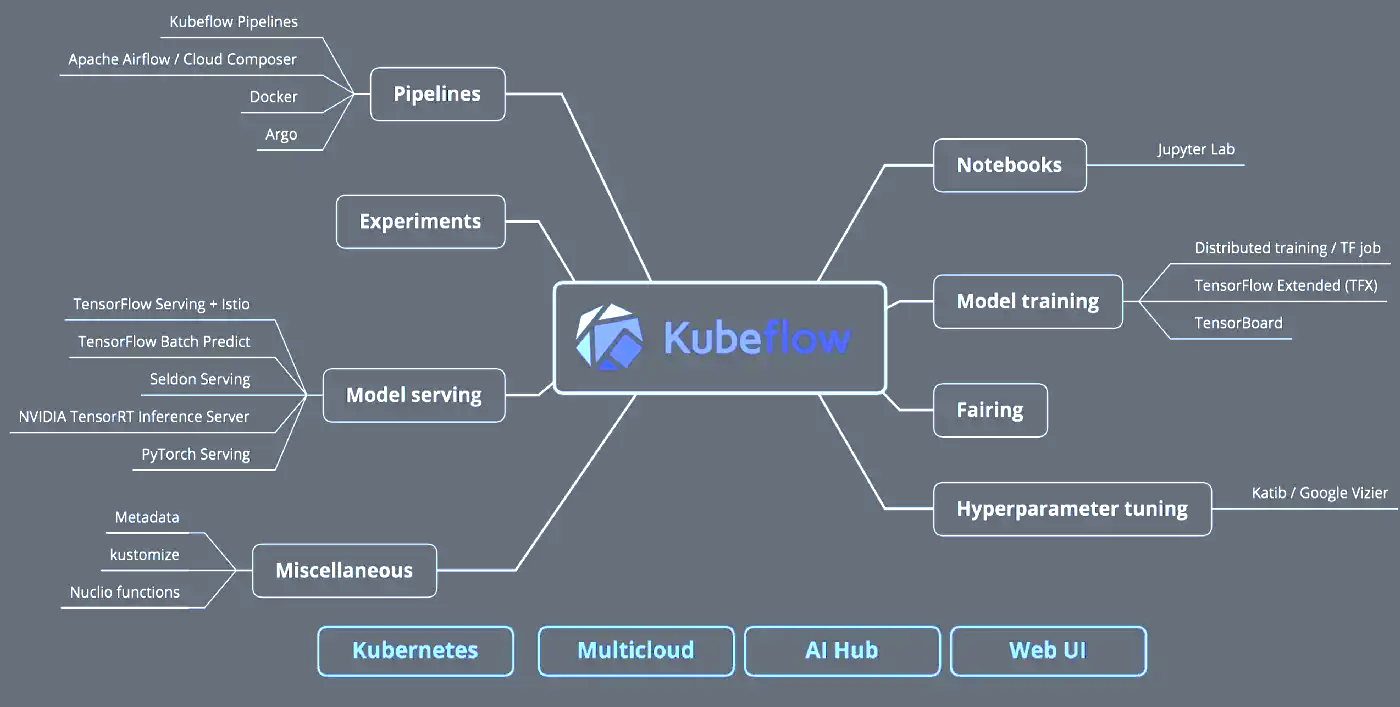
## ساختار شماتیک بستر Kubeflow و تعامل با سایر سرویس‌ها

اجزای اصلی Kubeflow عبارتند از:

* Kubernetes: Kubeflow در بالای Kubernetes ساخته شده است، که یک پلت فرم مقیاس پذیر و قابل اعتماد برای استقرار و مدیریت کانتینرها فراهم می کند.
* خطوط لوله Kubeflow: خطوط لوله Kubeflow 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: Kubeflow 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.







## 

## ویژگی های سرویس 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

## چرا 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

به‌طورکلی مراحل اصلی در معماری مفهومی MLOpsعبارت‌اند 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.

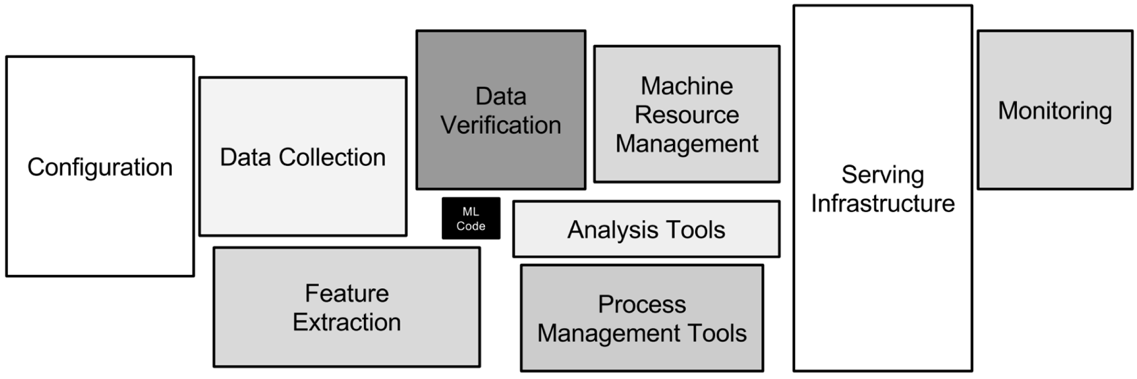
1. **Model Monitoring and Maintenance - نظارت و نگهداری مدل:**

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.

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.



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.

## مرحله آماده سازی داده‌ها در چرخه حیات ML

برخی از وظایف متداول درگیر در تهیه داده‌ها عبارتند از:

1. **Data Collection - جمع‌آوری داده‌ها**: جمع‌آوری داده‌ها از منابع مختلف مانند پایگاه‌های داده، فایل‌ها و APIهای خارجی.
2. **Data Cleaning - پاکسازی داده‌ها**: شناسایی و حذف هر گونه خطا، ناسازگاری یا موارد پرت در داده‌ها و اطمینان از کامل و دقیق بودن داده‌ها
3. **Data Integration - یکپارچه سازی داده‌ها:** ترکیب چندین منبع داده در یک مجموعه داده واحد، و حل هر گونه تضاد یا ناسازگاری که ممکن است ایجاد شود.
4. **Data Transformation - تبدیل داده‌ها:** تبدیل داده‌ها به قالبی که می‌تواند توسط مدل ML مصرف شود، مانند تبدیل متغیرهای طبقه بندی به مقادیر عددی.
5. **Data Normalization - عادی سازی داده‌ها**: مقیاس بندی داده‌ها به گونه ای که محدوده مقادیر یکسانی داشته باشند، که می‌تواند به بهبود عملکرد برخی از مدل‌های ML کمک کند.
6. **Feature Engineering - مهندسی ویژگی:** ایجاد ویژگی‌ها یا متغیرهای جدید که می‌تواند به بهبود عملکرد مدل ML کمک کند.
7. **Data Splitting - تقسیم داده‌ها:** تقسیم داده‌ها به مجموعه‌های آموزشی، اعتبار سنجی و تست، که برای آموزش، ارزیابی و آزمایش مدل ML استفاده می‌شود.

به‌طورکلی، آماده‌سازی داده‌ها مرحله‌ای حیاتی در چرخه حیات ML است، زیرا کیفیت و قالب داده‌ها به شدت بر دقت و عملکرد مدل نهایی ML تأثیر می‌گذارد. یک مجموعه داده به خوبی آماده شده می‌تواند به یک مدل دقیق و کارآمد منجر شود، و بنابراین، سرمایه‌گذاری زمان و منابع برای اطمینان از پاکسازی، تبدیل و آماده شدن داده‌ها برای مراحل بعدی چرخه حیات ML مهم است.

## معماری منطقی

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.
* مدیریت مدل: این مؤلفه مدل‌های آموزش‌دیده و فراداده‌های مربوط به مدل‌ها مانند نسخه، دقت و داده‌های آموزشی را ذخیره می‌کند.
* استقرار مدل: این جزء، مدل‌های آموزش دیده را در یک محیط تولید مستقر می‌کند، جایی که می‌توان از آنها برای پیش بینی استفاده کرد.
* مانیتورینگ مدل: 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.

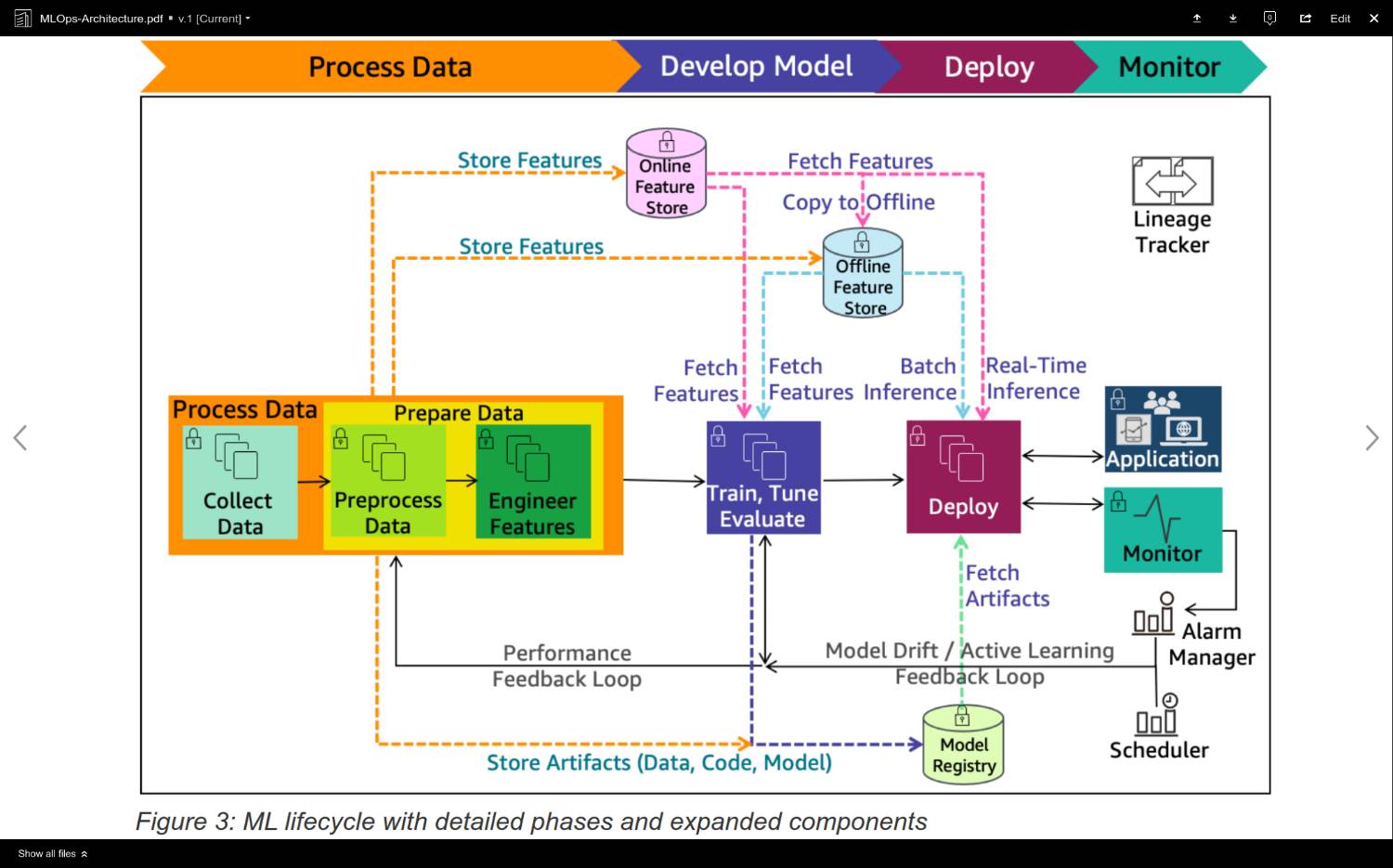
## معماری فیزیکی

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 Kubeflow را می توان به صورت زیر خلاصه کرد:

* **آموزش توزیع‌شده:** Kubeflow ابزارهایی را برای آموزش مدل‌های یادگیری ماشین در مجموعه‌ای از پردازنده‌های گرافیکی فراهم می‌کند، که به شما این امکان را می‌دهد تا فرآیند آموزش را برای مدیریت مجموعه داده‌های بزرگتر و مدل‌های پیچیده‌تر افزایش دهید.
* **سرویس‌دهی مدل:** Kubeflow یک 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 Kubeflow**: مجموعه‌ای 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.
* **مدیریت داده ها:** Kubeflow 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.
* **ردیابی آزمایش**: Kubeflow 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.
* **نمایش**: Kubeflow 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: Kubeflow** می‌تواند 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.

**نظارت و ثبت: Kubeflow** با ابزارهای نظارت و گزارش گیری Kubernetes یکپارچه می شود تا بینش هایی را در زمان واقعی از وضعیت بارهای کاری یادگیری ماشین ارائه دهد. این شامل معیارهایی مانند استفاده از پردازنده و حافظه و همچنین گزارش‌های اشکال‌زدایی می‌شود.

**یکپارچه سازی ذخیره سازی: Kubeflow** 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.

**نسخه‌سازی مدل: Kubeflow** 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.

**سرویس‌دهی مدل: Kubeflow** یک مؤلفه سرویس‌دهی مدل را ارائه می‌کند که به کاربران امکان می‌دهد مدل‌های یادگیری ماشین را به‌عنوان REST API استقرار و ارائه کنند. این مولفه می تواند به صورت افقی مقیاس شود و ویژگی هایی مانند مقیاس خودکار، تعادل بار و بررسی سلامت را ارائه می دهد.

**نظارت بر عملکرد مدل:**

**ابزارهای موجود نظارت بر عملکرد مدل در Kubeflow:**

Kubeflow ابزارهای مختلفی را برای نظارت بر عملکرد و سلامت گردش‌های کاری یادگیری ماشینی در حال اجرا بر روی خوشه‌های Kubernetes ارائه می‌کند این ابزارها به توسعه دهندگان و دانشمندان داده کمک می کنند تا مسائل را شناسایی کرده و عملکرد گردش کار ML خود را بهینه کنند.

برخی از ابزارهای نظارتی کلیدی موجود در Kubeflow عبارتند از:

1. Prometheus: Prometheus یک سیستم نظارتی محبوب است که معیارها را از منابع مختلف جمع‌آوری می‌کند و آنها را در یک پایگاه داده سری زمانی ذخیره می‌کند.
2. Grafana: Grafana یک پلت فرم منبع باز برای تجسم و تجزیه و تحلیل معیارها است.
3. TensorBoard: TensorBoard یک ابزار محبوب برای تجسم و تجزیه و تحلیل عملکرد مدل های یادگیری ماشین است. Kubeflow با TensorBoard ادغام می شود تا یک داشبورد متمرکز برای تجسم پیشرفت آموزش، عملکرد مدل و سایر معیارها ارائه دهد.
4. Kibana: Kibana یک پلت فرم تجسم داده منبع باز است که می تواند برای تجزیه و تحلیل گزارش و نظارت استفاده شود. Kubeflow با Kibana ادغام می شود تا یک داشبورد متمرکز برای نظارت بر گزارش های تولید شده توسط اجزای Kubeflow فراهم کند.
5. Jaeger: Jaeger یک سیستم ردیابی توزیع شده منبع باز است که می تواند برای نظارت بر عملکرد و سلامت سیستم های توزیع شده استفاده شود. Kubeflow با Jaeger ادغام می شود تا قابلیت های ردیابی توزیع شده را برای گردش های کاری ML در حال اجرا بر روی خوشه های Kubernetes ارائه دهد.
6. Argo: Argo یک موتور گردش کار منبع باز است که می تواند برای مدیریت و نظارت بر گردش کار ML استفاده شود. Kubeflow با Argo یکپارچه می شود تا یک داشبورد متمرکز برای نظارت بر وضعیت و پیشرفت گردش کار ML ارائه دهد.
7. داشبورد Kubernetes: داشبورد Kubernetes یک رابط کاربری مبتنی بر وب برای نظارت و مدیریت کلاسترهای Kubernetes است. Kubeflow با داشبورد Kubernetes ادغام می شود تا یک داشبورد متمرکز برای نظارت بر سلامت خوشه Kubernetes ارائه دهد.
8. Metrics Server: Metrics Server یک مؤلفه Kubernetes است که معیارهای استفاده از منابع را از گره ها و پادهای موجود در خوشه جمع آوری می کند. Kubeflow با Metrics Server ادغام می شود تا معیارهای مربوط به CPU، حافظه و شبکه استفاده از اجزای Kubeflow را جمع آوری و تجسم کند.
9. Istio Mixer: Istio Mixer جزء شبکه سرویس Istio است که داده های تله متری را از میکروسرویس های در حال اجرا بر روی خوشه های Kubernetes جمع آوری می کند. Kubeflow با Istio Mixer یکپارچه می شود تا معیارهای مربوط به ترافیک شبکه و ارتباطات بین میکروسرویس ها را جمع آوری و تجزیه و تحلیل کند.
10. داشبورد Pipeline: Kubeflow Pipelines یک داشبورد مبتنی بر وب برای نظارت و مدیریت گردش کار یادگیری ماشین ارائه می دهد. داشبورد یک نمایش بصری از گردش کار، از جمله ورودی ها، خروجی ها و وضعیت هر مرحله در گردش کار ارائه می دهد.

## ابزار انتخاب شده برای نظارت بر عملکرد مدل

* Grafana
* Prometheus

**دلایل انتخاب ابزارها**

Kubeflow با Prometheus ادغام می شود تا معیارها را از اجزای مختلف پشته Kubeflow جمع آوری و تجسم کند، مانند خوشه Kubernetes، سرویس مش Istio و خطوط لوله ML. همچنین Kubeflow با داشبوردهای از پیش پیکربندی شده Grafana ارائه می شود که بینش هایی را در مورد سلامت و عملکرد اجزای Kubeflow، مانند مشاغل آموزشی، نوت بوک ها و خطوط لوله ارائه می دهد.

**ابزارهای ارائه/استنتاج مدل در داشبورد Kubeflow:**

Kubeflow ابزارهای مختلفی را برای ارائه مدل ارائه می دهد. برخی از ابزارهای ارائه مدل در Kubeflow هستند که راه حل های کارآمد و مقیاس پذیری را برای ارائه مدل های یادگیری ماشین در Kubernetes ارائه می دهند. برخی از ابزارهای ارائه مدل در Kubeflow عبارتند از:

1. TensorFlow Serving: این یک سیستم سرویس دهی با کارایی بالا برای مدل های یادگیری ماشینی است که می تواند چندین مدل و نسخه را مدیریت کند.
2. Seldon Core: این یک پلت فرم منبع باز برای استقرار و مدیریت مدل های یادگیری ماشین در Kubernetes است که از چارچوب های مختلفی مانند TensorFlow، PyTorch و scikit-learn پشتیبانی می کند.
3. KServing: این یک چارچوب بدون سرور مبتنی بر Kubernetes برای ارائه مدل‌های یادگیری ماشین است که از چارچوب‌های مختلف یادگیری ماشین مانند TensorFlow، PyTorch و scikit-learn پشتیبانی می‌کند.
4. MLflow: این یک پلتفرم منبع باز برای چرخه زندگی کامل یادگیری ماشین است که شامل سرویس دهی مدل است. این یک API REST برای استقرار مدل‌های یادگیری ماشینی ارائه می‌کند که می‌تواند با استفاده از ابزارهای مختلف ارائه شود.
5. TFX Serving: این یک پلتفرم ارائه مدل مبتنی بر TensorFlow است که از چارچوب‌های مختلف یادگیری ماشین پشتیبانی می‌کند و زیرساخت سرویس‌دهی مقیاس‌پذیر و انعطاف‌پذیر را فراهم می‌کند.
6. اینها برخی از ابزارهای ارائه مدل در Kubeflow هستند که راه حل های کارآمد و مقیاس پذیر را برای ارائه مدل های یادگیری ماشین در Kubernetes ارائه می دهند.
7. KubeServe: این یک پلت فرم ارائه مدل منبع باز است که از چارچوب های مختلف یادگیری ماشین مانند TensorFlow، PyTorch و ONNX پشتیبانی می کند. این یک زیرساخت خدمات انعطاف پذیر و مقیاس پذیر را ارائه می دهد که می تواند به راحتی در Kubernetes مستقر شود.
8. Clipper: این یک سیستم ارائه مدل با کارایی بالا است که از چارچوب های مختلف یادگیری ماشین مانند TensorFlow، PyTorch و scikit-learn پشتیبانی می کند. پیش‌بینی‌های با تأخیر کم را ارائه می‌کند و می‌تواند سرویس‌دهی بی‌درنگ مدل‌های یادگیری ماشین را انجام دهد.
9. Cortex: این یک پلت فرم منبع باز برای استقرار مدل های یادگیری ماشین در Kubernetes است که از چارچوب های مختلف یادگیری ماشین مانند TensorFlow، PyTorch و scikit-learn پشتیبانی می کند. این یک زیرساخت خدمات مقیاس پذیر و کارآمد را فراهم می کند که می تواند پیش بینی های بازده بالا و تأخیر کم را مدیریت کند.
10. Kubemlops: این یک پلتفرم MLOps منبع باز است که از چارچوب های مختلف یادگیری ماشین پشتیبانی می کند و یک زیرساخت خدمات مقیاس پذیر و انعطاف پذیر را ارائه می دهد. این یک رابط یکپارچه برای مدیریت گردش کار یادگیری ماشین، از جمله ارائه مدل ارائه می دهد.

## ابزار انتخاب شده برای ارائه/استنتاج مدل

* KServe

**دلایل انتخاب ابزار** KServe

kserve یک پلت فرم سرویس منبع باز برای مدل های یادگیری ماشینی است که قابلیت های مختلفی را برای استقرار و مدیریت مدل ها ارائه می دهد. معیار عملیات kserve برای اندازه گیری عملکرد پلت فرم و اطمینان از اینکه می تواند حجم کاری مورد انتظار را مدیریت کند، ضروری است.Kserve یک سیستم سرویس دهی با کارایی بالا است که می تواند برای استقرار و ارائه مدل های یادگیری ماشین در مقیاس استفاده شود. هنگامی که با Kubeflow استفاده می شود، Kserve چندین مزیت را ارائه می دهد:

* **مقیاس پذیری:** Kserve به شما این امکان را می دهد که به راحتی زیرساخت های سرویس دهی مدل خود را در صورت نیاز برای پاسخگویی به تقاضای در حال تغییر مقیاس بندی کنید. این امر به ویژه برای برنامه هایی با حجم کاری متغیر یا نوسانات فصلی مهم است.
* **در دسترس بودن بالا:** Kserve دارای مکانیزم های داخلی تحمل خطا و متعادل کننده بار است تا اطمینان حاصل شود که زیرساخت ارائه مدل شما همیشه برای رسیدگی به درخواست ها در دسترس است. این به کاهش زمان خرابی و بهبود قابلیت اطمینان کلی سیستم کمک می کند.
* **عملکرد:** Kserve برای ارائه سرویس مدل با تأخیر کم و بازده بالا طراحی شده است، که برای برنامه های بلادرنگ که زمان پاسخگویی باید به حداقل برسد، حیاتی است.
* **انعطاف‌پذیری**: Kserve از انواع مدل‌ها و چارچوب‌های مختلف پشتیبانی می‌کند، که به شما امکان می‌دهد از ابزارها و کتابخانه‌هایی استفاده کنید که برای موارد استفاده خاص شما بهترین کار را دارند. این کار ادغام Kserve را در جریان کار یادگیری ماشین موجود شما آسان‌تر می‌کند.
* **یکپارچه سازی**: Kserve به طور یکپارچه با سایر اجزای پلت فرم Kubeflow، مانند ابزار Kustomize برای مدیریت پیکربندی های Kubernetes و مش سرویس Istio برای مدیریت ترافیک شبکه، ادغام می شود.

## شکاف‌های ارتباطی تیم‌های تجاری و فنی

چالش اصلی بین تیم‌های تجاری و فنی ارتباط است. پیدا کردن زبان مشترک جهت همکاری سخت است. بیشتر اوقات این شکاف دلیل شکست پروژه‌های بزرگ است.

## ارزیابی ریسک

ارزیابی ریسک نیز یکی از چالش‌های مهم است، زیرا مدل‌ها اغلب تمایل دارند از آنچه در ابتدا نتایج می‌دادند دور شوند. ارزیابی هزینه/خطر چنین شکست‌هایی یک کار ضروری و چالش برانگیز است.

## بازتاب تغییر اهداف کسب و کار در مدل

وابستگی‌های زیادی با داده‌ها وجود دارد که به طور مداوم عملکرد را حفظ می‌کند، استانداردهای مدل را تغییر می‌دهد و از حاکمیت هوش مصنوعی اطمینان می‌دهد. همگام شدن با اهداف در حال تحول کسب و کار و آموزش مدل مستمر چالش برانگیز است.

# 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: Kubeflow** یک پلت فرم بسیار مقیاس پذیر برای استقرار گردش های کاری ML در Kubernetes فراهم می کند. این 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: Kubeflow** برای ارائه یک پلت فرم امن برای گردش کار ML طراحی شده است. 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: Kubeflow** 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: Kubeflow** به گونه ای طراحی شده است که بسیار قابل حمل باشد، به این معنی که 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.
* **قابلیت همکاری: Kubeflow** به گونه ای طراحی شده است که با ابزارها و فناوری های دیگر که معمولاً در اکوسیستم ML استفاده می شود، قابل همکاری باشد. این امر ادغام با منابع داده موجود، ابزارهای تجزیه و تحلیل و سایر چارچوب‌های ML را آسان می‌کند، که به ساده‌سازی توسعه و استقرار گردش‌های کاری ML کمک می‌کند.
* **تحمل خطا:** Kubeflow 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:** Kubeflow 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.

# فرضیات و محدودیت‌ها

MLOps جهت استقرار مؤثر مدل ML در دنیای واقعی ضروری است، اما هزینه آن به دلیل پیچیدگی و منابع موردنیاز می‌تواند بسیار گران باشد. عامل اصلی تعیین‌کننده در استفاده یا عدم استفاده از MLOps ها، بررسی کامل بازگشت سرمایه با بگارگیری یادگیری ماشین است.

* پیاده‌سازی MLOpsها ممکن است گران باشد.
* توانایی ایجاد و حفظ یک چرخه حیات MLOps به طور مؤثر به منابع کافی (افراد، زیرساخت‌ها و ابزار) نیاز دارد.
* MLOps اگر به درستی انجام نشود، به ابزارها و مهارت‌های فراوانی نیاز دارد که بر پیچیدگی آن افزوده می‌شود.
* 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.

## محدودیت MLOPs در فضای داده های حجیم

داده های بزرگ به مجموعه داده های بزرگ و پیچیده ای اشاره دارد که پردازش و تجزیه و تحلیل با استفاده از ابزارهای سنتی پردازش داده دشوار است. در حالی که داده‌های بزرگ می‌توانند بینش‌های ارزشمندی ارائه دهند و به بهبود دقت مدل‌های یادگیری ماشینی کمک کنند، اما در مورد MLO ها نیز چالش‌هایی را به همراه دارد:

* **پیش پردازش داده و مهندسی ویژگی:** داده های بزرگ می توانند نویز، ناقص و ناهمگن باشند، که استخراج ویژگی های معنادار برای مدل های یادگیری ماشین را دشوار می کند. تیم های زیرساخت باید تلاش قابل توجهی را در پیش پردازش داده ها و مهندسی ویژگی سرمایه گذاری کنند تا اطمینان حاصل شود که مدل ها داده های ورودی با کیفیت بالا را دریافت می کنند.
* **آموزش مدل:** داده های بزرگ همچنین می تواند روند آموزش مدل را کند کند، زیرا به منابع محاسباتی و زمان بیشتری نیاز دارد. بنابراین، باید فرآیند آموزش مدل را بهینه کرده تا زمان و هزینه مورد نیاز برای آموزش مدل ها بر روی داده های بزرگ را به حداقل برسانند.
* **استقرار مدل:** استقرار مدل‌های یادگیری ماشین آموزش‌دیده بر روی داده‌های بزرگ به دلیل پیچیدگی مدل‌ها و اندازه داده‌ها می‌تواند چالش برانگیز باشد. بنابراین، مدل‌ها مقیاس‌پذیر، کارآمد باشند که بتوانند حجم زیادی از داده‌ها را در زمان واقعی مدیریت کنند.
* **نظارت و نگهداری:** در نهایت، داده‌های بزرگ نیز می‌تواند نظارت و نگهداری مدل‌های یادگیری ماشین را در تولید دشوار کند. بنابراین، باید فرآیندهای نظارت و نگهداری قوی را برای شناسایی اجرا کرد.

# تست و ارزیابی MLOPs

پس از ایجاد تست‌ها، می‌توانیم هر زمان که تغییری در سیستم ایجاد شد، تست‌ها را به‌طور خودکار اجرا شود و در طول زمان به بهبود آنها ادامه دهیم. جهت جلوگیری از افزایش هزینه‌های پایین دستی و از دست دادن زمان، پاداش دادن به اجرای تست‌ها و شناسایی منابع اشتباه در سریع‌ترین زمان ممکن در چرخه توسعه، مناسب خواهد بود.

مشکل مقیاس پذیری استقرار مدل

با استقبال گسترده‌تر رویکردهای یادگیری ماشینی در سازمان‌ها، تمایل به استقرار تعداد زیادی مدل وجود دارد. به عنوان مثال، یک سرویس طبقه بندی اخبار ممکن است مدل های سفارشی را برای هر دسته خبری آموزش دهد. یکی دیگر از دلایل مهمی که سازمان‌ها تمایل به آموزش مدل‌های زیادی دارند، محافظت از حریم خصوصی داده‌ها است، زیرا جدا کردن داده‌های هر کاربر و آموزش مدل‌ها به طور جداگانه امن‌تر است. 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 0.5G 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.

## ارزیابی Kubeflow Pipelines

## اسکریپت‌های ارزیابی Kubeflow Pipelines بار کاری معمولی را شبیه‌سازی می‌کنند و معیارهای عملکردی مانند تأخیرهای سرور و مدت زمان اجرای خط لوله را ثبت می‌کنند.

## برای شبیه‌سازی یک بار کاری معمولی، اسکریپت معیار یک فایل مانیفست خط لوله را در نمونه‌ای از Kubeflow Pipelines به عنوان خط لوله یا نسخه خط لوله آپلود می‌کند و چندین اجرا را به طور همزمان ایجاد می‌کند. می توان مانیفست خطوط لوله مورد استفاده در اسکریپت معیار را مشخص کرد، یا می توان از یکی از خطوط لوله نمونه از پیش بارگذاری شده در Kubeflow استفاده کرد.

## علاوه بر این، استفاده از یک مانیفست خط لوله که نماینده مورد استفاده خاص شما باشد، نیز تمرین خوبی است. به عنوان مثال، اگر خوشه خطوط لوله Kubeflow شما عمدتاً برای خطوط لوله وظایف تشخیص تصویر استفاده می شود، پس بهتر است از خط لوله تشخیص تصویر در اسکریپت های معیار استفاده شود.

## همانطور که قبلا ذکر شد، پس از انتخاب خط لوله مناسب، اسکریپت های Benchmark آن را چندین بار به طور همزمان اجرا می کنند. در میان تمام عملیاتی که خطوط لوله Kubeflow می توانند انجام دهند، اجرای یک خط لوله بدون شک غیرقابل پیش بینی ترین و پرهزینه ترین است.

## سایر عملیات، مانند ایجاد خط لوله (نسخه) یا ایجاد یک آزمایش، معمولاً هزینه قابل پیش بینی و متوسطی را القا می کند. به عنوان مثال، ایجاد یک نسخه خط لوله، یک ردیف جدید در جدول نسخه های خط لوله و یک فایل جدید در سرور MinIO معرفی می کند. اندازه فایل جدید به مانیفست نسخه خط لوله بستگی دارد. اگر مورد نادر یک مانیفست بسیار بزرگ را کنار بگذاریم و یک مانیفست با اندازه متوسط را برای هر نسخه خط لوله ایجاد شده فرض کنیم، هزینه کل ایجاد یک نسخه خط لوله به صورت خطی با تعداد نسخه های خط لوله افزایش می یابد. با این حال، از سوی دیگر، هزینه اجرای یک خط لوله یا نسخه خط لوله شامل عدم اطمینان بسیار بیشتر و گاهی اوقات هزینه بسیار بالایی است.

## یک خط لوله یا یک نسخه خط لوله می تواند دارای اجزای دلخواه باشد و از این رو اجرای یک خط لوله یا نسخه خط لوله می تواند پیچیدگی های زمانی و مکانی دلخواه را به همراه داشته باشد. به عنوان مثال، یک مرحله در یک خط لوله می تواند از یک تصویر ظرف سفارشی استفاده کند که یک کار آموزشی بسیار گران قیمت را انجام می دهد. علاوه بر این، اجراها در نمونه Kubeflow Pipelines نیز فضای DB بیشتری را نسبت به خطوط لوله، نسخه‌های خط لوله، آزمایش‌ها و غیره مصرف می‌کنند. بنابراین، به منظور درک عملکرد و مقیاس پذیری نقاط درد در نمونه Kubeflow Pipelines، تمرکز بر عملیات اجرا در بارهای کاری موثرتر است.

## ارزیابی عملیات kserve

با دنبال کردن این مراحل، می‌توان به طور موثر عملیات kserve را محک زد و اطمینان حاصل کرد که می‌تواند حجم کاری مورد انتظار را مدیریت کند.

**محیط آزمایش:** محیط آزمایشی را طوری تنظیم کرد که با محیط تولید تا حد امکان مشابه باشد. این شامل استقرار مدل(های مشابه)، استفاده از مشخصات سخت افزاری و نرم افزاری یکسان و شبیه سازی الگوهای ترافیکی مورد انتظار است.

**ابزار ارزیابی**: چندین ابزار ارزیابی در دسترس هستند، مانند Apache Benchmark ، Siege و Vegeta. ابزار ارزیابی که متناسب با نیازها بوده و بتواند الگوهای ترافیکی را ایجاد کند که انتظار میرود در تولید مشاهده کنیم.

**معیارهای** **ارزیابی**: معیارهای اندازه گیری مانند زمان پاسخگویی، توان عملیاتی و میزان خطا را تعریف کنیم.

**پیکربندی kserve:** پیکربندی kserve برای استفاده از تعداد مورد نظر از replica ها و منابع.

**ابزار سنجش**: ابزار سنجش را با پارامترهای دلخواه اجرا کرده و معیارها را ثبت کنید.

**تجزیه و تحلیل نتایج:** نتایج را برای شناسایی هر گونه تنگنا یا مشکلات عملکرد تجزیه و تحلیل استفاده میشود. در صورت لزوم، پیکربندی kserve یا ابزار سنجش را تنظیم شده و این روند را تکرار می‌شود.

**بهینه سازی عملکرد**: هر گونه تغییر لازم را برای بهینه سازی عملکرد kserve ، مانند تنظیم تخصیص منابع یا تنظیم پیکربندی مدل انجام میشود.

## تست مدل

در این مرحله، عملکرد مدل آموزش‌دیده را بر روی مجموعه‌ای از نقاط داده مجزا به نام داده‌های آزمایشی (که در مرحله دریافت داده تقسیم و نسخه‌سازی شد) ارزیابی می‌شود. استنباط مدل آموزش دیده بر اساس معیارهای انتخاب شده بر اساس مورد استفاده ارزیابی می‌شود. خروجی این مرحله گزارشی از عملکرد مدل آموزش دیده است. در ماژول Deploy، مدل‌های آموزش دیده را به ترتیب در محیط‌های توسعه‌دهنده، تست و تولید مستقر می‌شود. ابتدا با تست اپلیکیشن (که در محیط‌های توسعه دهنده و تست انجام می‌شود) شروع می‌شود.

## تست برنامه

قبل از استقرار یک مدل یادگیری ماشینی جهت تولید، تست استحکام، مقیاس پذیری و امنیت مدل بسیار مهم است. از این رو، ما مرحله "تست برنامه" را داریم که در آن تمام مدل‌های آموزش دیده و برنامه کاربردی را در یک محیط تولید مانند به نام محیط تست یا مرحله بندی به طور دقیق تست می‌شود. در این مرحله، ما ممکن است تست‌هایی مانند تست‌های A/B، تست‌های یکپارچه سازی، تست‌های پذیرش کاربر (UAT) ، یا تست بار را انجام دهیم.

# آموزش و تحویل

* برگزاری جلسات آموزش مفاهیم کلی و نصب و مدیریت بستر Mlops
* تحویل تمامی مستندات مربوط به فاز توسعه
* تاییدیه نماینده فنی سحاب در خصوص تایید صحت مستندات تولید شده
* تاییدیه نماینده فنی سحاب در خصوص تکمیل فاز آموزش
* روش تحویل مستندات?
* روش تحویل بستر Mlops?

# نصب و راه اندازی MLOPs

* 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

# منابع موردنیاز

# زمان‌بندی و شکست کارها

پیاده‌سازی MLOps معمولاً شامل مراحل زیر است:

* توسعه مدل: با استفاده از ابزارهایی مانند Jupyter Notebook، TensorFlow، PyTorch و غیره، مدل‌های یادگیری ماشین را توسعه و آموزش دهید.
* کنترل نسخه مدل: نسخه‌های مدل را با استفاده از سیستم‌های کنترل نسخه مانند Git ذخیره و مدیریت کنید.
* تست مدل: مدل‌ها را با استفاده از تست‌های خودکار آزمایش کنید تا مطمئن شوید که الزامات عملکرد و دقت را برآورده می‌کنند.
* استقرار مدل: مدل‌ها را با استفاده از زیرساخت‌ها به عنوان ابزارهای کد مانند Terraform یا CloudFormation در تولید مستقر کنید.
* مانیتورینگ مدل: مدل‌های در حال تولید را با استفاده از ابزارهایی مانند TensorFlow Serving، Seldon یا KFServing جهت شناسایی و پاسخگویی به مسائل نظارت کنید.
* تعمیر و نگهداری مدل: به طور مرتب مدل‌ها را به روز رسانی و نگهداری کنید تا عملکرد آنها را بهبود بخشید و مشکلاتی را که پیش می‌آید برطرف کنید.
* همکاری: تشویق به همکاری بین دانشمندان داده، توسعه دهندگان و تیم‌های عملیاتی جهت اطمینان از MLOهای روان و مؤثر.
* 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.
* قابلیت توضیح مدل: اطمینان حاصل شود که مدل‌ها شفاف و قابل توضیح هستند، به طوری که تصمیمات آنها قابل درک و اعتماد توسط ذینفعان باشد.

**مراحل اجرای Kubeflow معمولاً شامل موارد زیر است:**

* آماده سازی: جمع‌آوری نیازمندی‌ها، ارزیابی زیرساخت‌ها و تعیین محدوده پروژه اجرای Kubeflow.
* 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: مدل‌های قطار 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.
* همکاری: تشویق به همکاری بین دانشمندان داده، توسعه دهندگان و تیم‌های عملیاتی جهت اطمینان از MLOهای روان و مؤثر.
* 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.
* بررسی و بهینه‌سازی: به طور مداوم پیاده‌سازی Kubeflow از جمله عملکرد، مقیاس‌پذیری و ادغام با ابزارها و خدمات دیگر را بررسی و بهینه کنید.

# تحلیل هزینه

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