

Hola Data Conf

Modelos en producción: Lo bueno, lo malo y lo feo

12 de noviembre de 2022, DigitalHouse



HOLA

Bienvenidxs

- Soy Founder & Cloud Architect en deployr.ai
- Estudié Data Science y Deep Learning en esta casa.
- Trabajé como Data Scientist en un par de startups aplicando Procesamiento del Lenguaje Natural y Análisis de Series Temporales.



Ramiro Savoie

 /ramirosavoie

ramiro@deployr.ai

Cuando regresé a DigitalHouse para dar una clase mi primera slide fue ésta

¿Que viene después de esto?

DigitalHouse >
Coding School

- Modelo con 95% de accuracy en test.

```
# test_loss, test_acc  
results = model.evaluate(X_test, y_test)  
results
```

```
5473/5473 [=====] - 1s 92us/step  
[0.20057440025566178, 0.9521286314635483]
```

- ¿Cómo llega a nuestros usuarios y clientes?

La situación actual

- Resolver el problema correctamente y crear un modelo funcional, aunque sigue siendo crucial, **no es suficiente.**
- Más y más compañías necesitan lanzar sus modelos a producción para mostrar **valor real al negocio.**
- Managers o "managers de managers" empiezan a preguntar por el **ROI de la inversión en IA.**



Lo feo...



Los quiero escuchar

- ¿Quienes trabajan en Ciencia de Datos?
- ¿Cómo es la situación actual en tu empresa u organización con tu equipo de datos?
- ¿Tienen equipos de Ciencia de Datos / Analítica Avanzada que no tienen **ni un solo modelo productivo?**



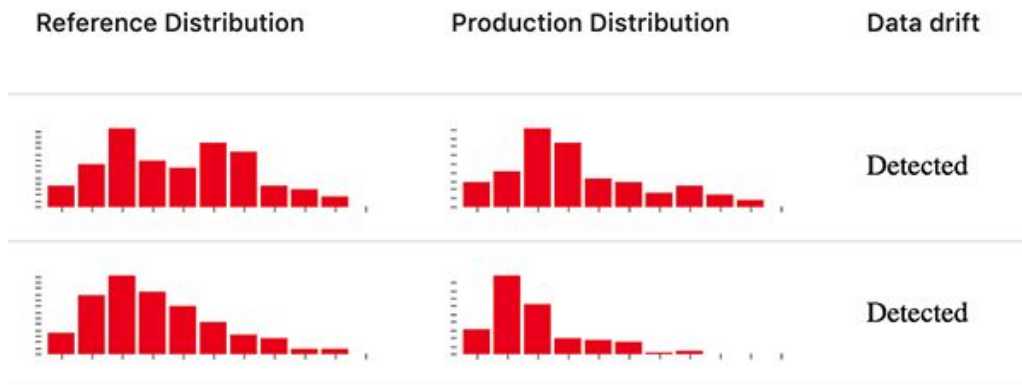
Qué dicen esos equipos respecto a esto:

- *“Cuesta y lleva mucho tiempo poner modelos en producción, el roadmap es largo, es algo muy diferente a lo que estamos acostumbrados a hacer.”*
- *“No hay feedback del negocio hasta que llegamos a producción.”*
- *“El área de Producto no nos entiende, no integra nuestros desarrollos con el resto de los features.”*
- *“No nos ponemos de acuerdo en cuál es la métrica a impactar.”*



Para los que sí lograron poner sus modelos en producción:

- ¿Generan **impacto** en sus métricas del negocio?
- ¿Puede ese modelo procesar y **responder** a pedidos en una fracción pequeña de tiempo?
- ¿Saben cómo se va a comportar si la **distribución** de los datos de entrada cambia con el tiempo?
- ¿Como testearían en su proyecto una versión completamente **nueva** del modelo de una forma segura?



Hacia dónde van los caminos

- En algún punto cada proyecto de ML se va a encontrar con esas preguntas y para responderlas se va a requerir un **conjunto diferente de skills** y conceptos que los de la fase de research.
- Esto es **independiente del dominio**, ya sea predecir el comportamiento de los clientes o las ventas, detectar o contar objetos en imágenes o complejos análisis de textos.
- Cada uno de esos proyectos deberá finalmente ser **productizado y mantenido** en algún punto y se va a encontrar con los contratiempos antes mencionados.



Áreas grises

- La parte de research de los proyectos de DS ya está bastante **bien explorada**, tenemos librerías estándar, herramientas y conceptos (Jupyter, Pandas, herramientas de trackeo de experimentos, etc.)
- Sin embargo y al mismo tiempo, la parte de **ingeniería o producción** sigue siendo un misterio para la mayoría de los practicantes de ML, hay muchas **áreas grises** y **no hay estándares claros** y por mucho tiempo no hubo un camino de guía para tener un proyecto de ML bien mantenido y diseñado.

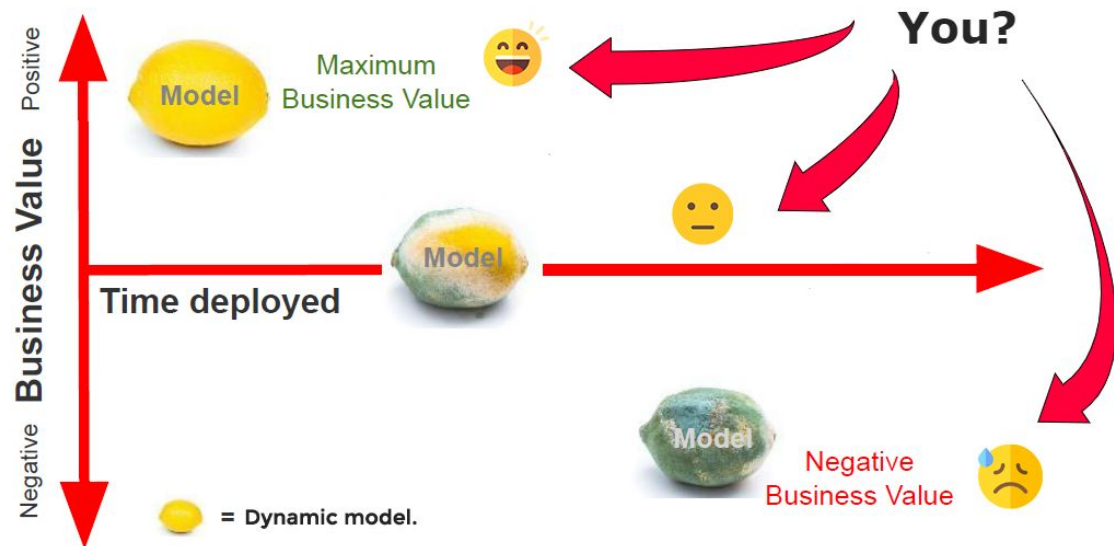


¿Qué pasa con nuestros modelos?

- Lo que está productivo es un MVM y se lo "estira" demasiado, **no se itera** e incluso puede ser un modelo de AutoML.
- Le faltan datos, faltan features, falta tiempo de entrenamiento, falta probar con otros modelos, falta el feedback loop.

¿Qué pasa con nuestros modelos?

- El problema no se termina cuando el modelo está deployado, tenemos **model decay**.
- Lo que es peor, no hay visibilidad del daño que puede hacer un modelo **estancado**.



¿Qué pasa con nuestros modelos?

- Esto sucede porque estás usando ML, lo que no te contaron sobre eso.
- No están solos, esto le pasa a **toda la industria.**



Fuente: modelop.com

Key findings

100%

of respondents now have dedicated budgets for model operationalization (ModelOps), up from 51% in 2021

86%

say their C-suites are demanding answers about the ROI of their AI investments, but **48%** struggle to provide them

80%

say a lack of staff with the right skill sets is creating challenges for their ModelOps practices, up from 70% last year

Just 5%

say they have full visibility into all models in production across their enterprises

56%

say they lack effective processes for maintaining an inventory of models that are deployed within their enterprises

98%

think it's at least 'fairly important' to have a ModelOps platform that improves collaboration across data, analytics and AI teams, with 52% describing this as 'crucial'

Lo bueno...



¿Qué tenemos que lograr?

- Tener un proceso **ordenado** para la generación de modelos, un funnel que termine en producción.
- Dar **visibilidad** del mismo en cada paso.
- Saber cómo está la **salud** del modelo en todo momento.
- Que la **degradación** del modelo no impacte (tanto) al negocio. Tener control sobre la misma.
- **Automatizar** todo este proceso para mantener frescos estos modelos
- Tener visibilidad de **cuanto se tarda** en llevar a producción el modelo, previsibilidad y su **impacto** en KPIs.

¿Qué tenemos que lograr?

- Que el Data Scientist puede **enfocarse** en hacer Ciencia de Datos, que haya mejor **.fit()** con el puesto para el que fue contratado.
- Los referentes de la industria ya **tomaron nota** de esto en 2021



DeepLearning.AI

EXPERT PANEL

Machine Learning Engineering for Production (MLOps)

Wed, JUNE 30
10 to 11 am PT
RSVP: <https://mlep.eventbrite.com>

ANDREW NG
Founder,
DeepLearning.AI

ROBERT CROWE
TensorFlow Developer Engineer,
Google

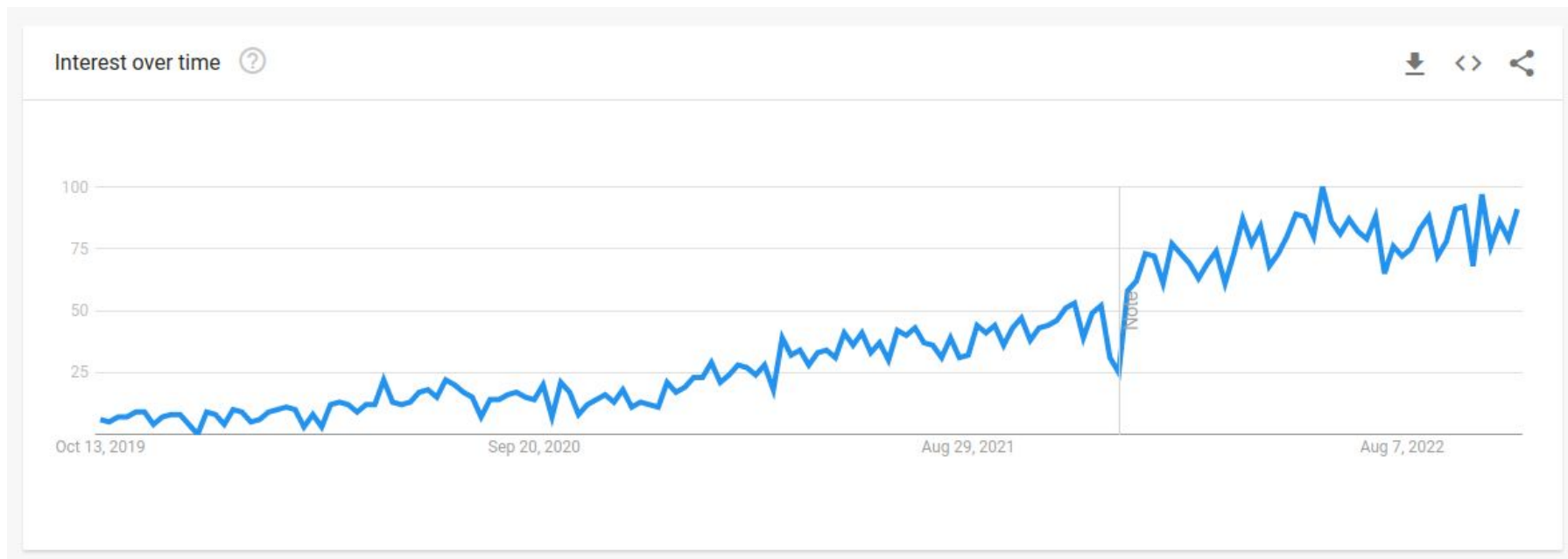
LAURENCE MORONEY
AI Advocate,
Google

RAJAT MONGA
Co-founder, Stealth Startup;
Former lead of TensorFlow,
Google

CHIP HUYEN
Adjunct Lecturer,
Stanford University

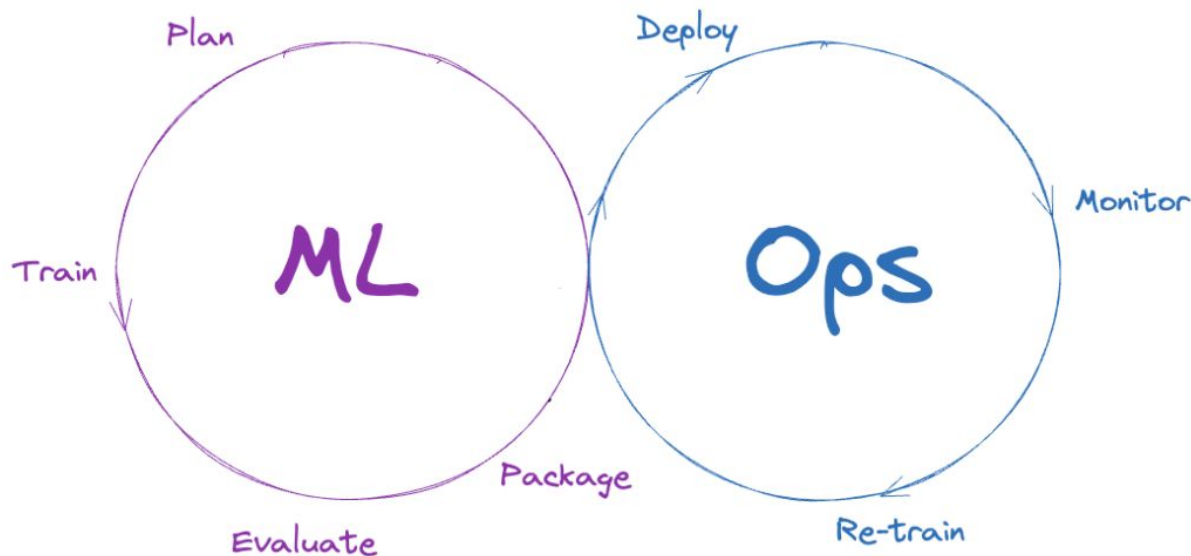
¿Qué tenemos que lograr?

Interés en “modelos en producción” desde 2019 hasta la fecha en [Google Trends](#)

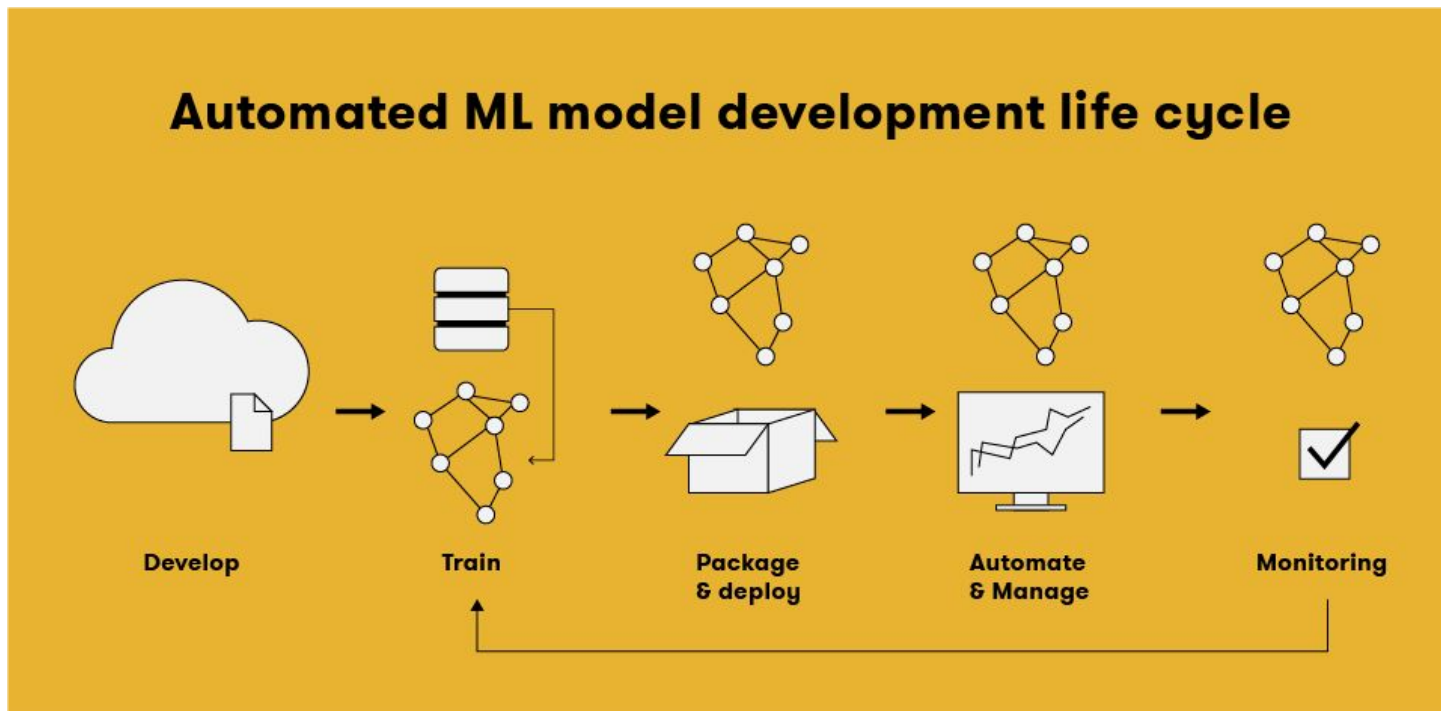


¿Qué tenemos que lograr?

- Ya hay metodologías, herramientas y empresas que están avanzando en este tema.
- Este es el problema que se propone resolver MLOps



El ciclo de vida del modelo



Los principales pilares de una estrategia de MLOps

¿Cómo implementar una solución robusta y madura con herramientas y servicios disponibles?



Pilar: Reproducibilidad y versionado

- Una de las principales características de los proyectos de ML maduros es la capacidad de **reproducir resultados**.
- Al principio es algo que no nos interesa mucho porque es bueno **probar** varios caminos.
- Pero al final, nuestros experimentos deben ser **reproducibles**.
- Encontrar un buen learning rate por ejemplo puede ser una meta, pero si se corre el experimento con los mismos valores ¿Obtenemos el mismo **resultado**?
- Cuando se trabaja en equipo, alguien tiene que continuar este trabajo y debería obtener los **mismos resultados** que nosotros con el mismo experimento.

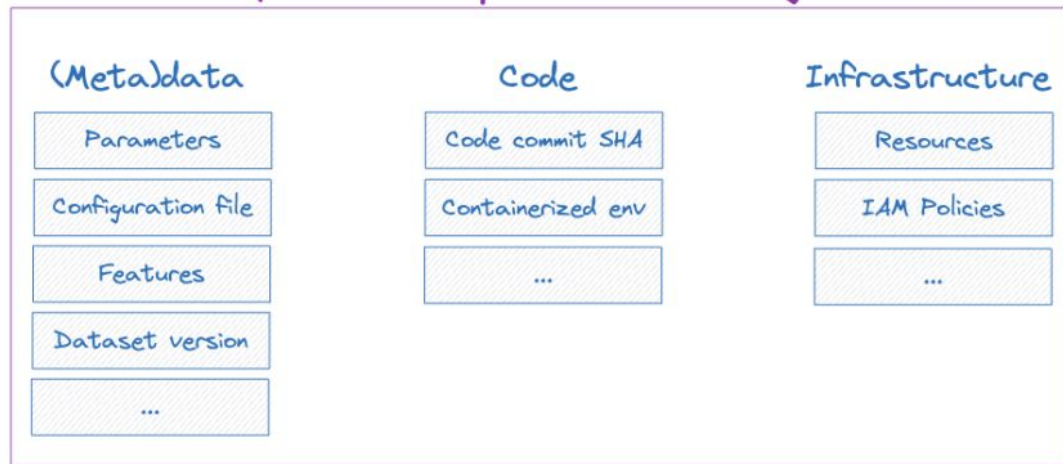
Pilar: Reproducibilidad y versionado

- Deberíamos ser capaces de ejecutar el **mismo código** con la misma configuración de parámetros.
- Como el código de las Jupyter Notebooks o de los scripts de Python cambia docenas de veces por día, cambiando no solo los valores numéricos de los parámetros sino también la lógica del conjunto deberíamos **versionarlo**.

¿Qué más se puede versionar para hacer el proyecto más reproducible?

- Básicamente código de **EDA**, de procesamiento de datos (transformaciones), de entrenamiento, etc.
 - Archivos de **configuración** usados por esta base de código.
 - La **infraestructura**: servicios, recursos y configuración del Cloud, desde un simple storage o base de datos, políticas de IAM, pipelines de componentes.
- Nos sirve para poder replicar en otra cuenta o en otro ambiente.

Reproducibility and Versioning



¿Cómo implementarlo?

- Para el **código**: `git`, aplicable también a archivos de configuración, pequeños datasets o documentación.
 - No está tan bueno para archivos grandes o binarios.
 - Con Jupyter Notebooks era difícil también hasta hace un tiempo que llegó ReviewNB.
- Para la **infraestructura**: CloudFormation, CDK o Terraform
- Para la **data**: DVC o Neptune.ai, que hacen más sencillo guardar y hacer seguimiento cualquier tipo de metadata relacionada al proyecto o el modelo.

Pilar: Monitoreo

- Considerado "la frutilla del postre", el paso final de MLOps o de los sistemas de ML.
- Es todo lo opuesto, debe ser implementado ASAP, incluso **antes** de que el modelo esté productivo.
- No solo debe ser **observado** el proceso de inferencia sino también cada experimento de entrenamiento.

¿Que deberíamos trackear en una sesión de experimentación?

- Histórico de las métricas de entrenamiento como accuracy, F1, función de pérdida en training y validación.
- CPU, GPU, RAM y Disco **consumidos** por el script de entrenamiento.
- Performance de predicción en el set de holdout, luego del training.
- Pesos iniciales y finales del modelo.



¿Que deberíamos trackear del entrenamiento e inferencia?

- **Monitoreo a nivel de servicio:**

Del Servicio Web en Flask, del Microservicio en Kubernetes, de la AWS Lambda.

- Es importante conocer **cuánto** toma procesar un request del usuario, cual es el tamaño promedio del payload, cuanto recursos de CPU, GPU y RAM usa el servicio.

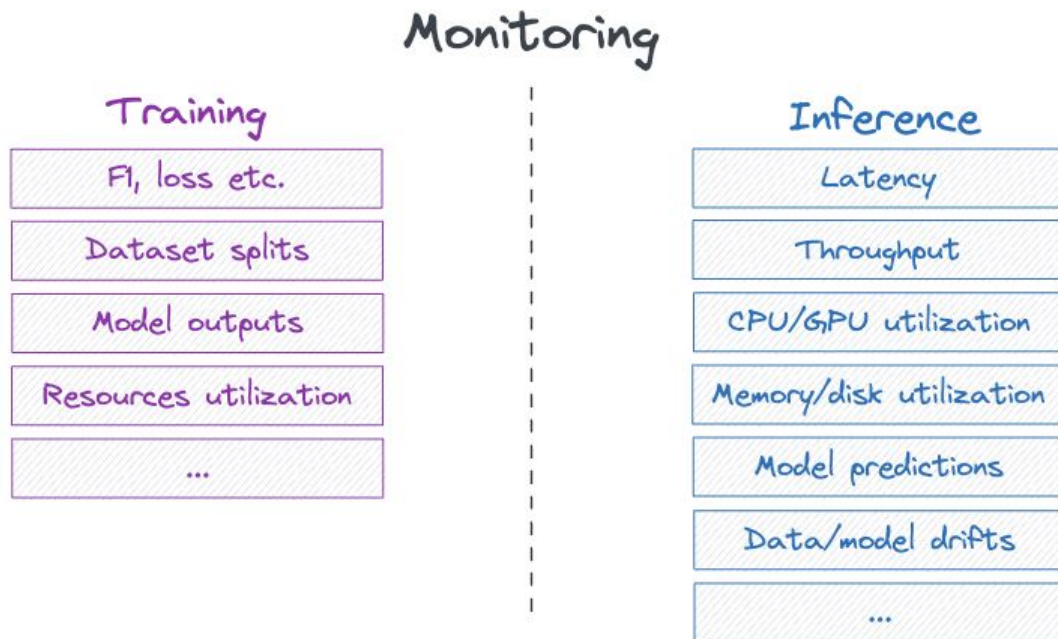
- **Monitoreo a nivel del modelo:**

- Datos de entrada que el mismo recibe.
- Distribución de la variable objetivo a lo largo del tiempo.



¿Que deberíamos trackear del entrenamiento e inferencia?

- Por ejemplo en modelos financieros se puede considerar al salario como uno de los features de entrada y su distribución puede ir cambiando a lo largo del tiempo debido a salarios más altos.
- Si no se hace esto puede generar que el modelo se **estanque** y debe ser **reentrenado**.



¿Cómo implementarlo?

- Para el **entrenamiento** hay varias herramientas de seguimiento de experimentos como Weights & Biases, Neptune.ai y MLFlow.
- La mayoría de ellas pueden ser fácilmente integradas en el código (instaladas con `pip`) y permiten **loggear** y **visualizar** métricas en real time durante el entrenamiento o el procesamiento de datos.



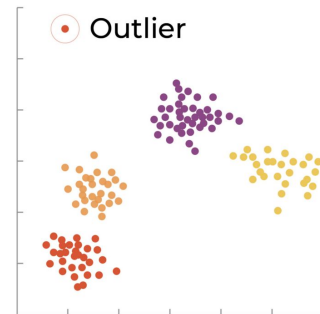
¿Cómo implementarlo?

- Para inferencia o deployment de modelos, depende del servicio o modelo que se use.
- Si es con AWS Lambda/EC2, se puede soportar el logging con CloudWatch.
- Dentro de Kubernetes el stack más popular es Prometheus para exportar métricas y Grafana para crear dashboards personalizados y visualizar métricas y data en real time.



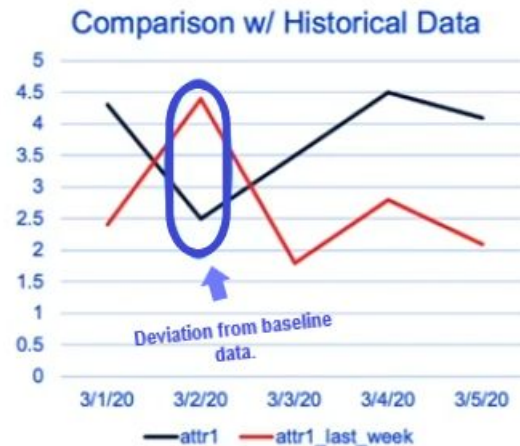
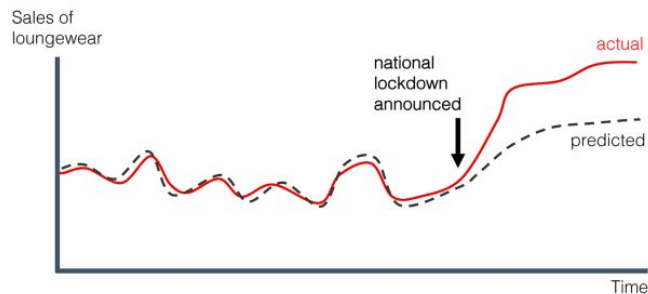
Pilar: Testeo

- En los equipos de ML **poco** se dice sobre testing o escribir tests.
- Es más común escribir test unitarios, de integración o e2e en los proyectos de ingeniería de software tradicional.
- ¿Cómo luce un test de ML? Hay varias cosas que deberían mantenerse siempre **validadas**:
 - Cantidad y calidad de los datos de entrada.
 - Esquema de los features de la data de entrada (rango esperado de los valores).
 - Datos producidos por los jobs de procesamiento (transformaciones) como así también los jobs mismos.
 - Compliance de los features y los pipelines de datos (GDPR).



Pilar: Testeo

- Esto va a mantener los pipelines de ML **robustos y resilientes**.
- Tener estos tests permite detectar **cambios inesperados** en los datos o en la infraestructura tan pronto como aparezcan, dando más tiempo a **reaccionar** acorde.



¿Cómo implementarlo?

- Para **validación de datos** con frameworks como Great Expectations y DeepChecks.
- Por supuesto dependiendo del caso de uso y la voluntad de usar herramientas externas se pueden implementar algunos chequeos básicos por **cuenta propia**.
- Una de las más simples ideas puede ser computar **estadísticos** desde los datos de entrenamiento y usarlos como esperanza de otro dataset como los de testeo en producción.
- Cualquier tipo de **pipeline, transformación** e incluso los más simples **scripts** pueden ser testeados usualmente de la misma forma que se testearía el código tradicional.

¿Cómo implementarlo?

- Si se utiliza un job de ETL para hacer procesamiento y transformar nuevos datos de entrada a un ritmo regular, seguramente se va a querer **chequear que funciona** y produce **resultados válidos** antes de pushear esos datos hacia adelante en el script de entrenamiento.
- En lo que se refiere a **infraestructura**, se debe preferir siempre IaaS para dar de alta cualquier recurso y aunque no es muy común, IaaS puede ser testeado de forma unitaria también.
- En cuanto a test de **compliance**, depende de la compañía y proyecto y debe ser validado por áreas de **data governance**.

Pilar: Automatización

- El último pero no menos importante, está **relacionado con todo** lo que hemos discutido hasta ahora.
- Como y que debe ser automatizado es una de las preguntas claves para los ML engineers.

¿Cómo luce un workflow ideal?

- **Nueva data** llega al almacén de datos crudos.
- La data es **limpiada** y **procesada**, se crean los **features** a partir de ella.
- La data también se testea contra su **esquema**, concordancia con GDPR, en un problema de CV puede incluir chequeos de calidad de imagen y blureo de caras.
- Si es aplicable, los features procesados se guardan en el **Feature Store** para uso posterior.
- Cuando la data está lista, el script de **entrenamiento** se dispara automáticamente.

¿Cómo luce un workflow ideal?

- Toda la historia de entrenamiento y **métricas** son guardadas y visualizadas.
- El modelo está listo y resulta que es muy **prometedor**, dado que fue automáticamente **evaluado** así que un **deploy automático** es disparado.
- Y así continúa el proceso...
- Se puede continuar con gestión de fallas, alertas, labelado automático, chequeo de performance decay o data drift del modelo y disparado de retraining automático.

¿Cómo luce un workflow ideal?

- En teoría es una descripción de un sistema casi sin **intervención manual**.



¿Cómo implementarlo?

- Lo (no tan) malo

Lo (no tan) malo...

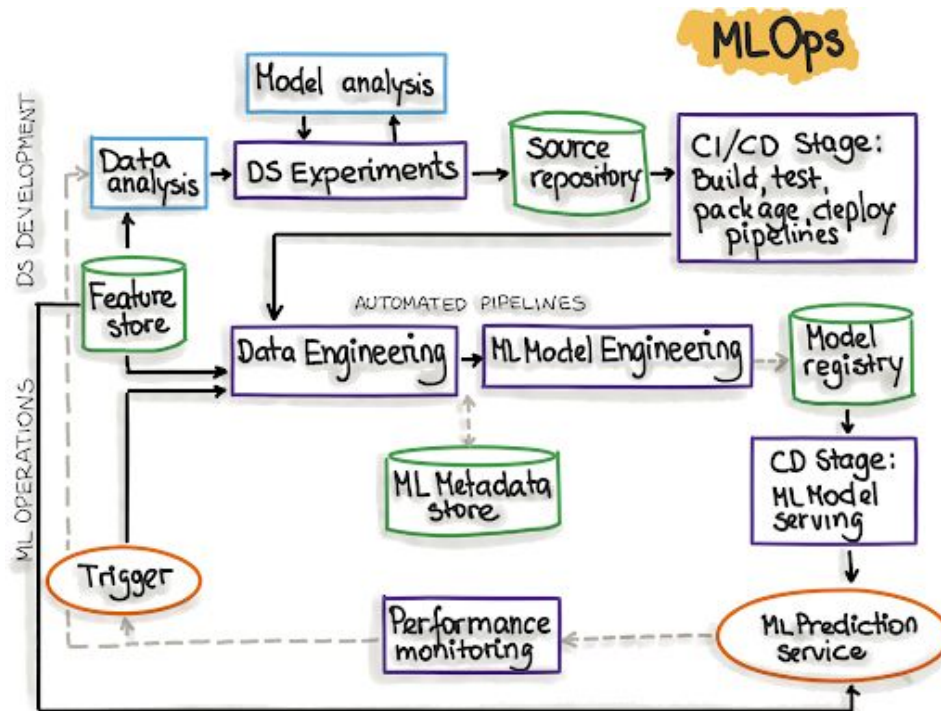


¿Cómo implementarlo?

- Este es el pilar **más borroso** ahora por varios motivos
- No hay una **receta única** que sirva para todos los casos.
- No existe algo así como la "cantidad correcta de automatización"
- El camino hacia la automatización es largo
- Lo que quiere decir es que **depende del equipo** y las metas de proyecto como así también la estructura del equipo.
- Faltan más expertos que hagan esto.
- Falta formalizar aún más la disciplina de MLOps.

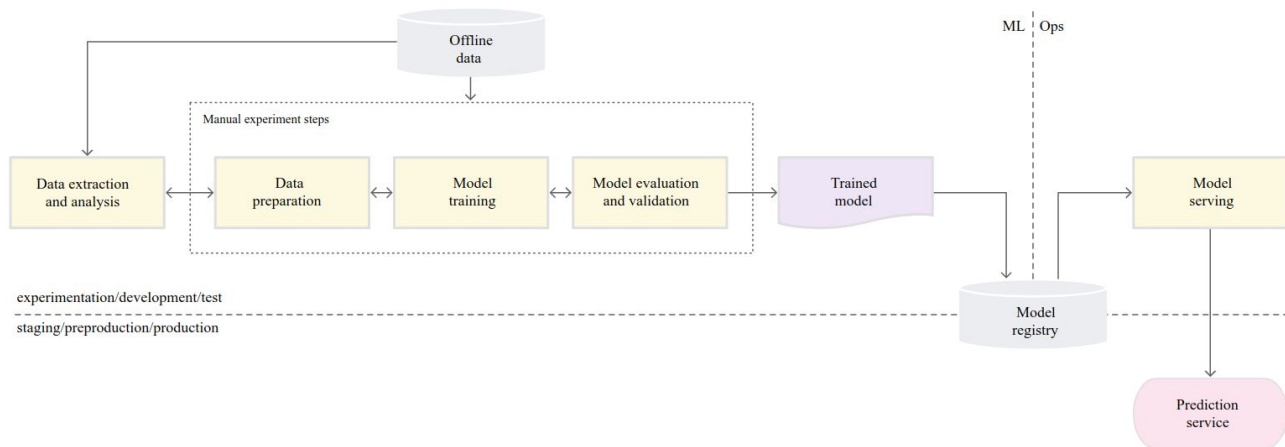
¿Cómo implementarlo?

- Sin embargo **hay algunas guías** o arquitecturas de ejemplo que pueden dar una pista para responder a la pregunta: "¿Cuánto se debería automatizar?"



¿Cómo implementarlo?

- **Toma tiempo** implementarlo si queremos que sea aplicable a más de un modelo o caso de uso.
- Google nos propone un modelo de madurez de 3 estadios desde procesos completamente manuales hasta la automatización total.



¿Cómo implementarlo?

- Si ya conocemos qué partes del sistema se van a automatizar ¿Qué tools se van a usar?
- Hay docenas de tools para cada componente en el sistema de MLOps pero hay sitios que nos muestran las tools más usadas.



Algunos aprendizajes

- Si lograste llegar a producción con tus modelos, no te duermas!
- Volvé a **iterar**, probá otro pipeline de datos y otros modelos.
- Si aplicaste MLOps, ahora va a ser más fácil porque ya tenés una **plataforma** para experimentar.
- No es el final del camino, hay que seguir trabajando y **monitoreando**.
- Si los puedo ayudar con esta tarea, me escriben!

¡Muchas gracias!