Curso 1 - Introducción al aprendizaje automático en la producción

En el primer curso de la especialización en ingeniería de aprendizaje automático para la producción, identificará los distintos componentes y diseñará un sistema de producción de ML de principio a fin: alcance del proyecto, necesidades de datos, estrategias de modelado y limitaciones y requisitos de despliegue; y aprenderá a establecer una línea de base del modelo, a abordar la deriva del concepto y a crear un prototipo del proceso para desarrollar, desplegar y mejorar continuamente una aplicación de ML en producción.

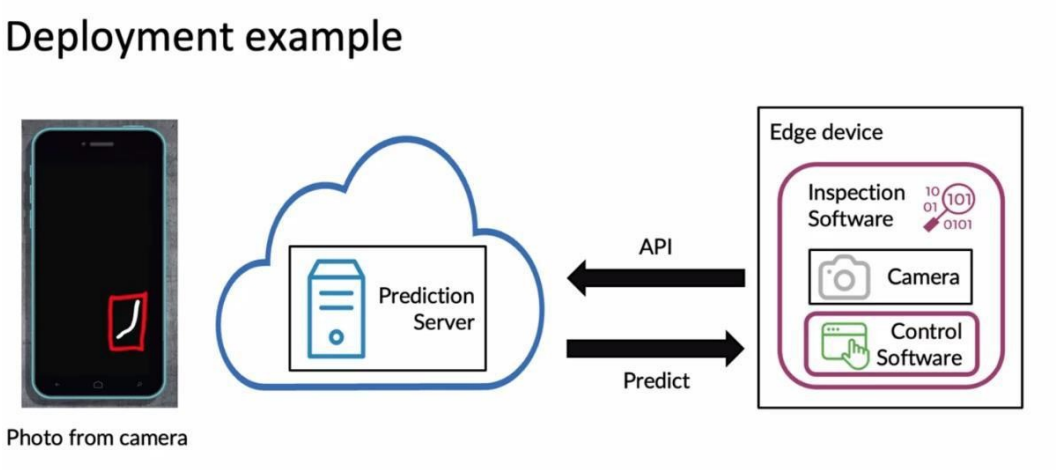
Entender los conceptos de aprendizaje automático y aprendizaje profundo es esencial, pero si quieres construir una carrera efectiva en el campo de la IA, también necesitas capacidades de ingeniería de producción. La ingeniería de aprendizaje automático para la producción combina los conceptos fundamentales del aprendizaje automático con la experiencia funcional de los roles modernos de desarrollo de software e ingeniería para ayudarle a desarrollar habilidades listas para la producción. Semana 1: Visión general del ciclo de vida de ML y su implementación Semana 2: Selección y entrenamiento de un modelo Semana 3: Definición de datos y línea de base

Semana 1: Visión general del ciclo de vida del LD y su despliegue

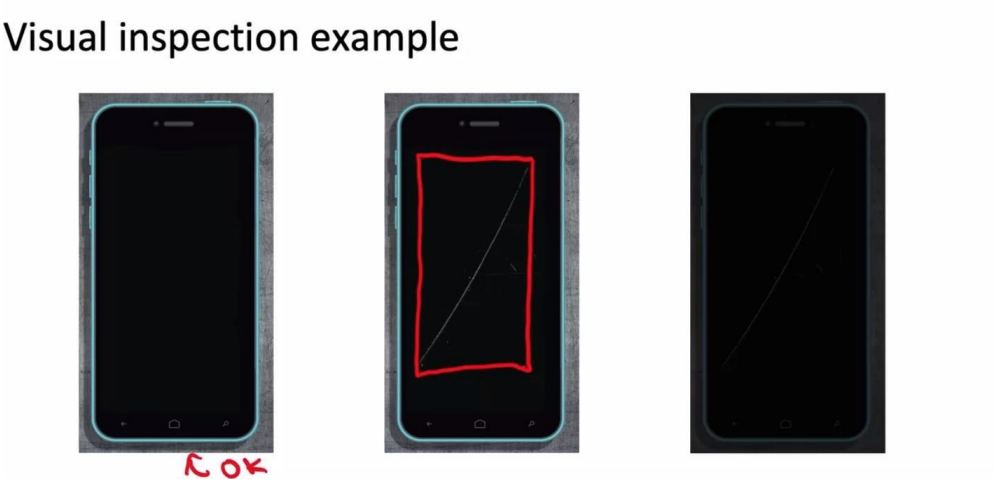
Contenido

Introducción

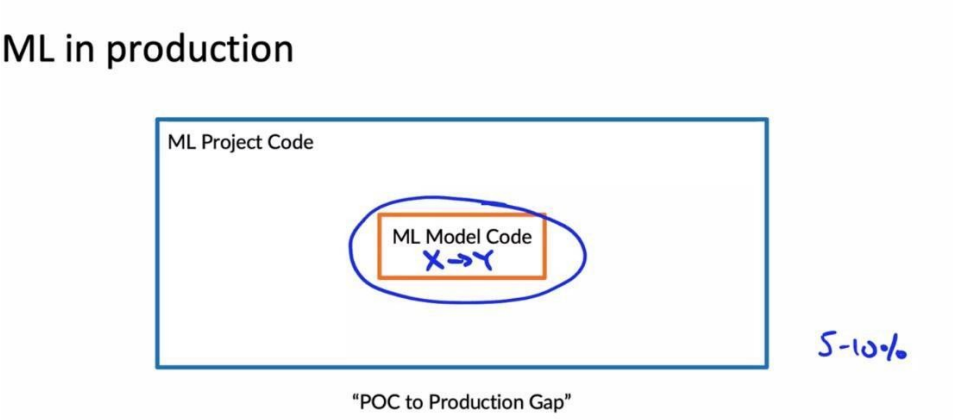
- Hay retos de aprendizaje automático para la producción que no se limitan al proceso de desarrollo de modelos en un cuaderno Jupyter.
- Se puede considerar el aprendizaje automático de producción como el aprendizaje automático en sí mismo, junto con los conocimientos y habilidades necesarios para el desarrollo de software moderno.
- Si trabajas en un equipo de aprendizaje automático, realmente necesitas conocimientos tanto de aprendizaje automático como de software para tener éxito. Esto se debe a que tu equipo no se limitará a producir un único resultado.
- Desarrollará un producto o servicio que funcionará continuamente como parte de una misión crítica para el trabajo de su empresa
- A menudo, los aspectos más difíciles de la creación de sistemas de aprendizaje automático resultan ser los que menos se esperan, como el despliegue.
- Está muy bien ser capaz de construir un modelo, pero ponerlo en manos de la gente y ver cómo lo usan puede ser muy revelador
- Los modelos de aprendizaje automático son geniales, pero a menos que se sepa cómo ponerlos en producción, es difícil conseguir que creen la máxima cantidad de valor.



- En realidad, esto se hace habitualmente en las fábricas, y se denomina inspección visual de defectos automatizada.
- Lo que hace el software de inspección es que controlará una cámara que toma fotos de los smartphones mientras salen de la línea de fabricación.
- Luego hace una llamada a la API para pasar la imagen a un **servidor de predicción**. Y el trabajo del servidor de predicción es aceptar estas llamadas a la API, recibir una imagen, tomar una decisión sobre si este teléfono es defectuoso o no, y luego devolver esta predicción.
- El software de control de la inspección puede entonces tomar la decisión de control adecuada para permitir su avance en la línea de fabricación.
- Tienes que tomar este modelo de aprendizaje automático, ponerlo en un servidor de producción, configurar las interfaces API y escribir todo el resto del software con el fin de desplegar este algoritmo de aprendizaje en la producción.
- Este servidor de predicción está a veces en la nube y a veces también en el borde.
- De hecho, en la industria manufacturera, utilizamos mucho los despliegues de **borde,** porque no se puede dejar caer la fábrica cada vez que se cae el acceso a Internet.

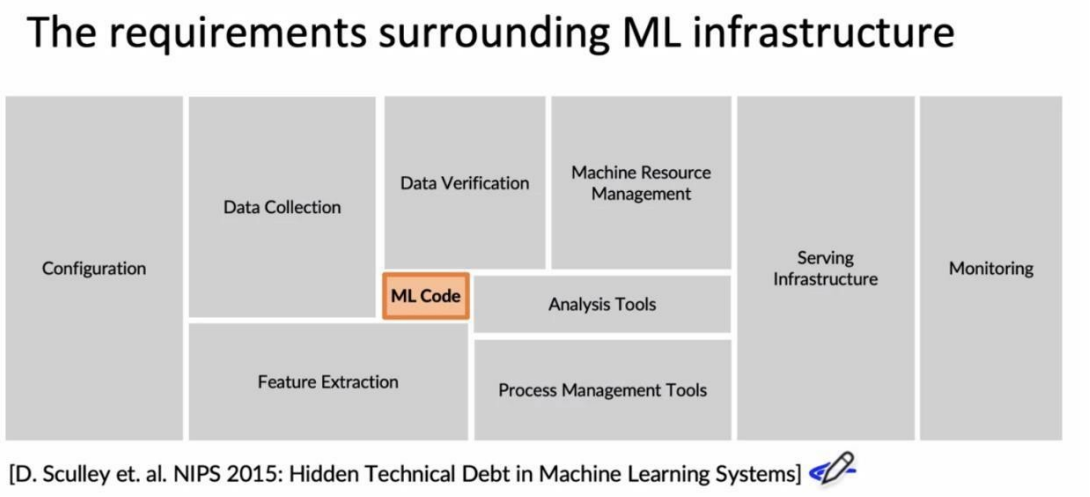


- En este escenario, hay un teléfono bueno a la izquierda, y el del medio tiene un gran arañazo
- Ha entrenado a su algoritmo de aprendizaje para que reconozca que las imágenes como la de la izquierda están bien, mientras que dibuja cuadros delimitadores alrededor de los arañazos u otros defectos para las imágenes que se parecen a la del medio
- Cuando lo despliegue en la fábrica, puede encontrar que el despliegue de producción en la vida real le da imágenes que pueden ser mucho más oscuras (imagen de la derecha) porque las condiciones de iluminación en la fábrica han cambiado por alguna razón, en comparación con el momento en que se recogió el conjunto de entrenamiento.
- Este problema se denomina a veces **deriva de concepto** o **deriva de datos**.
- Este es un ejemplo de los muchos problemas prácticos que nosotros, como ingenieros de aprendizaje automático, deberíamos resolver si queremos asegurarnos de que no sólo lo hacemos bien en el conjunto de pruebas de retención
- Queremos que nuestros sistemas creen realmente valor en un entorno práctico de despliegue de producción.
- A veces veo muchos proyectos construidos con éxito en el entorno de desarrollo, pero aún así se necesitan otros seis meses de trabajo para el despliegue práctico.
- Esta es sólo una de las muchas cosas prácticas que un equipo de aprendizaje automático tiene que vigilar y manejar para desplegar estos sistemas.



- Un segundo reto de la implantación de modelos de aprendizaje automático y producción es que se necesita mucho más que código de aprendizaje automático.
- En la última década se ha prestado mucha atención a los modelos de aprendizaje automático (por ejemplo, redes neuronales u otros algoritmos que aprenden una función que va de una entrada a una salida), y se han producido avances sorprendentes en los modelos de aprendizaje automático
- Pero resulta que si miras un sistema de aprendizaje automático en producción, este pequeño rectángulo naranja representa el código de aprendizaje automático, y todo el rectángulo azul representa todos los códigos que necesitas para todo el proyecto de aprendizaje automático.
- En muchos proyectos de aprendizaje automático, quizá sólo el 5-10% sea realmente código de aprendizaje automático

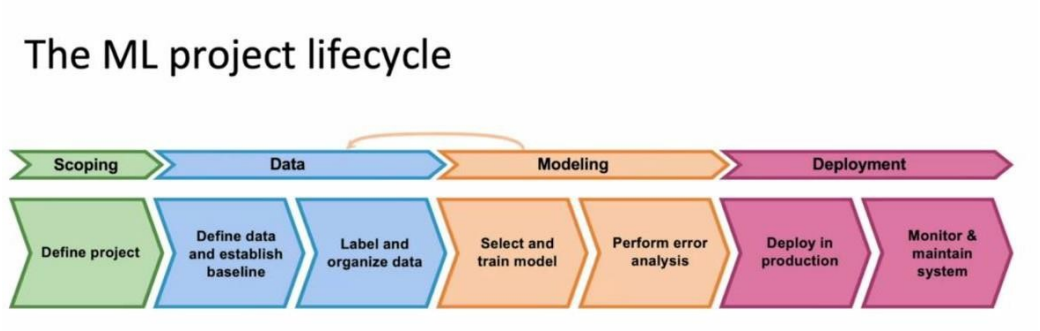
- Por eso, cuando se tiene un modelo de prueba de concepto que funciona en el cuaderno Jupyter, todavía puede costar mucho trabajo pasar de esa prueba de concepto inicial al despliegue de producción. A veces la gente se refiere a esto como la brecha de la prueba de concepto (POC) a la producción.
- Y gran parte de esa brecha se debe a veces a la gran cantidad de trabajo que se necesita para escribir todo este código más allá del código inicial del modelo de aprendizaje automático.



- Además de los códigos de aprendizaje automático, también hay muchos otros componentes para la gestión de los datos, como la recopilación de datos, la verificación de datos y la extracción de características.
- Y después de servirlo, también hay que considerar cómo supervisar y analizar el sistema. A menudo hay muchos otros componentes que deben construirse para permitir un despliegue de producción que funcione.
- Así que en este curso aprenderá cuáles son todas estas otras piezas de software necesarias para un valioso despliegue de producción

Pasos de un proyecto de ML

- Cuando estoy construyendo un sistema de aprendizaje automático, pensar en el ciclo de vida del proyecto de aprendizaje automático es una forma eficaz de planificar todos los pasos que necesito para trabajar.
- Cuando estés trabajando en un sistema de aprendizaje automático, creo que también encontrarás que este marco te permite planificar todas las cosas importantes que necesitas hacer para que el sistema funcione, y también para minimizar las sorpresas.

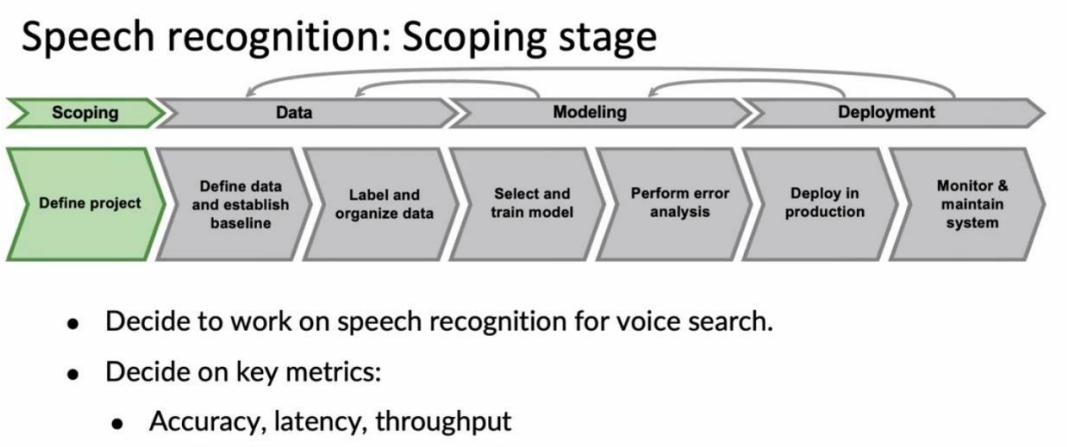


- La primera es **el alcance**, en el que hay que definir el proyecto o decidir en qué se va a trabajar.
- A qué quiere aplicar exactamente el aprendizaje automático, y qué es X y qué es Y.
- Tras elegir el proyecto, hay que adquirir los datos que se necesitan para el algoritmo. Esto incluye la definición de los datos y el establecimiento de una línea de base, y luego el etiquetado y la organización de los datos.

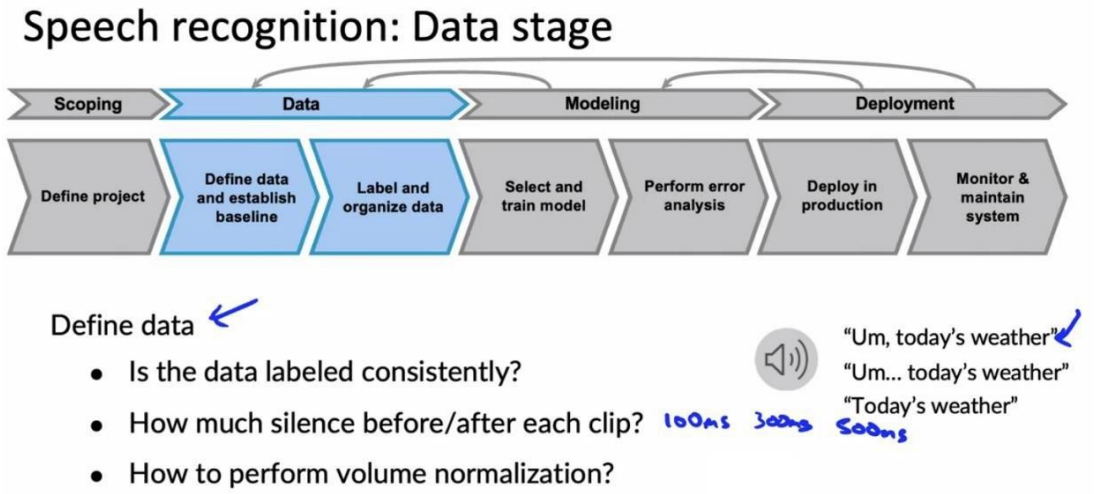
- Después de tener los datos, hay que entrenar el modelo. Durante la fase del modelo, hay que seleccionar y entrenar el modelo y también realizar el análisis de errores.
- Es posible que sepas que el aprendizaje automático suele ser una tarea muy iterativa. Durante el proceso de análisis de errores, puedes volver atrás y actualizar el modelo, o también puedes volver a la fase anterior y decidir si también necesitas recoger más datos.
- Como parte del análisis de errores antes de llevar un sistema al despliegue, a menudo también realizo una comprobación/auditoría final, para asegurarme de que el rendimiento del sistema es lo suficientemente bueno y de que es lo suficientemente fiable para la aplicación.
- A veces, un ingeniero piensa que cuando despliega un sistema, ha terminado.
- Ahora le digo a la mayoría de la gente que **cuando se despliega un sistema por primera vez, sólo se está a mitad de camino de la línea de meta**, porque a menudo sólo después de activar el tráfico en vivo se aprende la segunda mitad de las lecciones importantes necesarias para que el sistema funcione bien.
- Para llevar a cabo el paso de despliegue, hay que desplegarlo en producción, escribir el software necesario para ponerlo en producción, y luego también supervisar el sistema, hacer un seguimiento de los datos que siguen llegando y mantener el sistema.
- Por ejemplo, si la distribución de los datos cambia, puede ser necesario actualizar el modelo.
- Tras el despliegue inicial, el mantenimiento suele consistir en volver a realizar más análisis de errores y quizás volver a entrenar el modelo, o puede significar tomar los datos que se obtienen.
- Ahora que el sistema está desplegado y funciona con datos en vivo, hay que volver a alimentar ese resultado en el conjunto de datos para, a continuación, actualizar potencialmente los datos, volver a entrenar el modelo, y así sucesivamente hasta que se pueda poner un modelo actualizado en el despliegue.
- No dudes en hacer una captura de pantalla de esta imagen y utilizarla con tus amigos o por ti mismo para planificar también tu proyecto de aprendizaje automático.

Caso práctico - Reconocimiento de voz

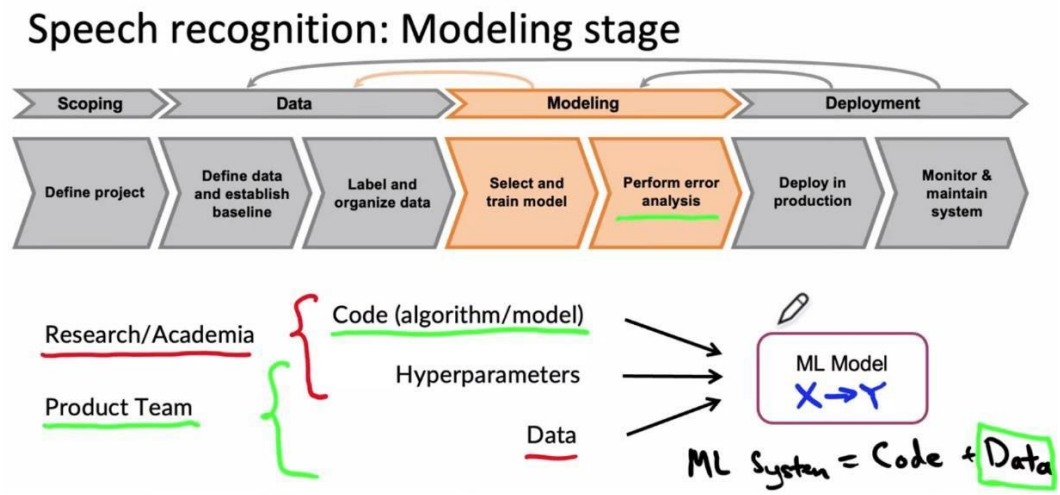
- Utilicemos el ciclo de vida del proyecto de aprendizaje automático para poner un ejemplo de reconocimiento de voz, de modo que pueda comprender todos los pasos necesarios para construir e implantar un sistema de este tipo



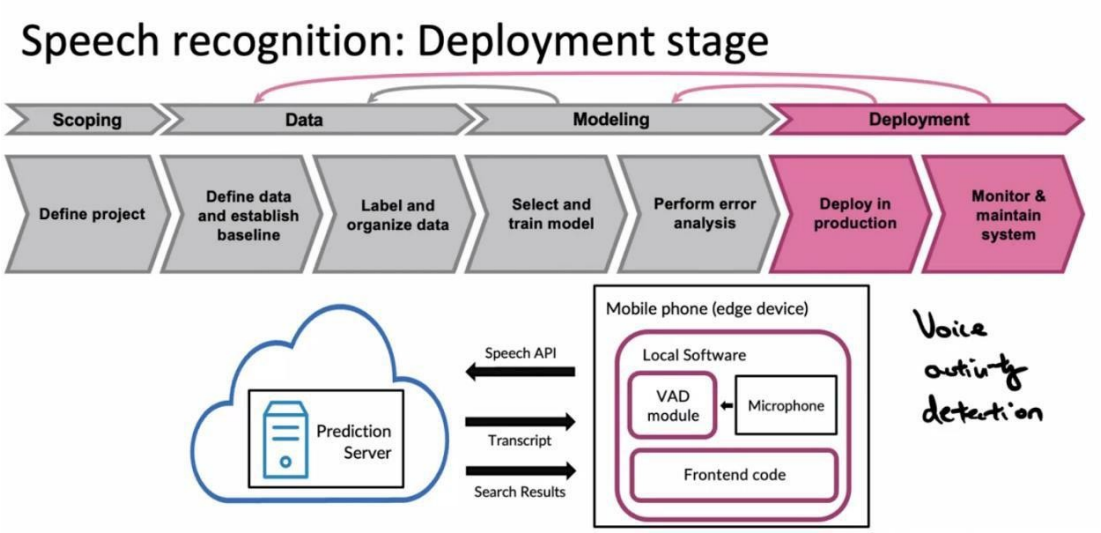
- Para definir el alcance, primero tenemos que definir el proyecto y simplemente tomar la decisión de trabajar en el reconocimiento de voz, digamos para la búsqueda por voz como parte de la definición del proyecto.
- Eso también te anima a intentar estimar las métricas clave. Esto dependerá mucho del problema. Casi todas las aplicaciones tendrán su propio conjunto de objetivos y métricas.
- Pero en el caso del reconocimiento de voz, algunas cosas que me importaban eran la precisión del sistema de voz, la latencia, el tiempo que tarda el sistema en transcribir el habla, el rendimiento, el número de consultas por segundo que manejamos.
- Y luego, si es posible, también se puede tratar de estimar los recursos necesarios. Así que cuánto tiempo, cuánto calcular el presupuesto, así como el calendario. ¿Cuánto tiempo se necesitará para llevar a cabo este proyecto?



- El siguiente paso es la etapa de los datos, en la que hay que definirlos y establecer una línea de base, así como etiquetar y organizar los datos. ¿Qué es lo difícil de esto?
- Uno de los retos de los sistemas prácticos de reconocimiento del habla es etiquetar los datos de forma coherente
- Aquí tienes un clip de audio de una grabación bastante típica que podrías obtener si estás trabajando en el reconocimiento de voz para la búsqueda por voz. Y la pregunta es, dado este clip de audio que acabas de escuchar, ¿cómo lo transcribirías? 1) "Um, el tiempo de hoy", 2) "Um... el tiempo de hoy 3) "El tiempo de hoy".
- Resulta que cualquiera de estas tres formas de transcribir el audio está bien y es razonable.
- Probablemente preferiría la primera o la segunda, no la tercera. Pero lo que perjudica el rendimiento de su algoritmo de aprendizaje es que un tercio de la transcripción utilice la primera, un tercio la segunda y un tercio la tercera forma de transcribir.
- Porque entonces tus datos son inconsistentes y confusos para el algoritmo de aprendizaje.
- ¿Cómo se supone que el algoritmo de aprendizaje adivina cuál de estas transcripciones específicas debe utilizar para un clip de audio?
- Tal vez podamos pedir a todo el mundo que se estandarice en la primera convención. Esto tendrá un impacto significativo en el rendimiento de su algoritmo de aprendizaje.
- Otros ejemplos de preguntas de definición de datos para un clip de audio, como cuánto silencio quiere antes y después de cada clip cuando un orador ha dejado de hablar.
- ¿Quieres incluir otros 100 milisegundos de silencio después de eso? ¿O 300 milisegundos o 500 milisegundos, medio segundo? ¿O cómo se realiza la normalización del volumen?
- Algunos altavoces hablan alto, otros menos alto y luego hay un caso complicado de un solo clip de audio con un volumen muy alto y otro muy bajo, todo dentro del clip de audio.
- Por lo tanto, cómo se realiza la normalización del volumen de preguntas como todas estas son **preguntas de definición de datos**.
- Si está trabajando en un sistema de producción, no tiene que mantener el conjunto de datos fijo.
- A menudo edito el conjunto de entrenamiento o incluso el conjunto de prueba si es necesario para mejorar la calidad de los datos y conseguir que un sistema de producción funcione mejor.



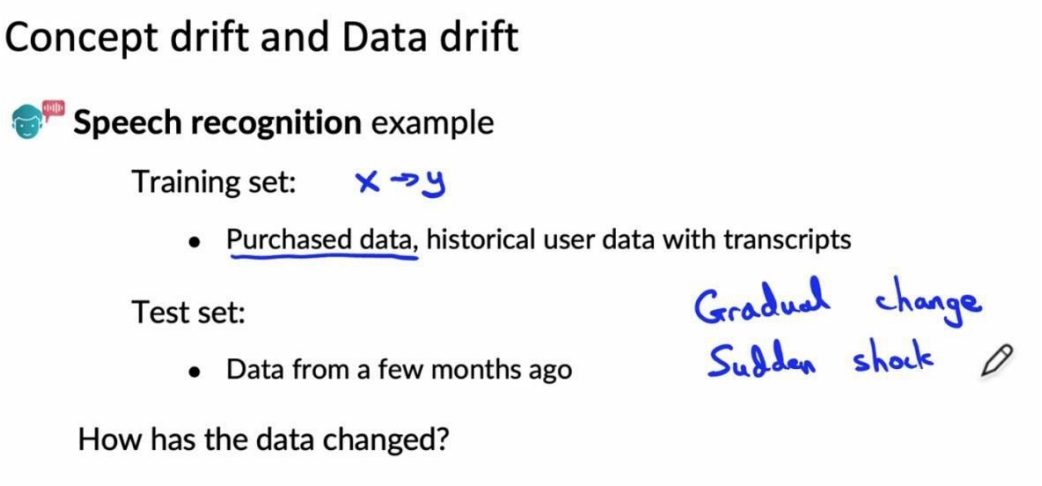
- Los tres insumos clave que intervienen en el entrenamiento de un modelo de aprendizaje automático son el código que constituye el algoritmo o la arquitectura del modelo de red neuronal que se elija.
- También tienes que elegir los hiperparámetros y luego están los datos y ejecutar el código con tus hiperparámetros en tus datos.
- Me he dado cuenta de que en muchos trabajos de investigación o académicos se tiende a mantener los datos fijos y a variar el código y quizás los hiperparámetros para intentar obtener un buen rendimiento.
- Por el contrario, si tu objetivo principal es simplemente construir y desplegar un sistema de aprendizaje automático que funcione, he descubierto que puede ser aún más eficaz **mantener el código fijo** y centrarse en cambio en la optimización de los datos y tal vez de los hiperparámetros con el fin de obtener un modelo de alto rendimiento,
- Un sistema de aprendizaje automático incluye tanto los códigos como los datos, y también los hiperparámetros, que quizá sean más fáciles de optimizar que el código o los datos.
- Y descubrí que en lugar de adoptar una visión centrada en el modelo de tratar de optimizar el código a su conjunto de datos fijos para muchos problemas, puede utilizar una implementación de código abierto de algo que se descarga de GitHub y en su lugar sólo **se centran en la optimización de los datos**.
- El análisis de errores puede indicarle dónde se queda corto su modelo, y cómo mejorar sistemáticamente sus datos (y quizás también el código)
- Si el análisis de errores puede indicarle cómo mejorar sistemáticamente los datos, puede ser una forma muy eficaz de llegar a un modelo de alta precisión.
- Parte del truco consiste en no sentir que hay que recopilar más datos todo el tiempo, porque siempre podemos utilizar más datos.
- En lugar de limitarse a recopilar más y más y más datos, lo cual es útil pero puede resultar **caro,** si el análisis de errores puede ayudarle a ser más específico en cuanto a los datos que debe recopilar, puede ayudarle a ser mucho más eficiente en la construcción de un modelo preciso.



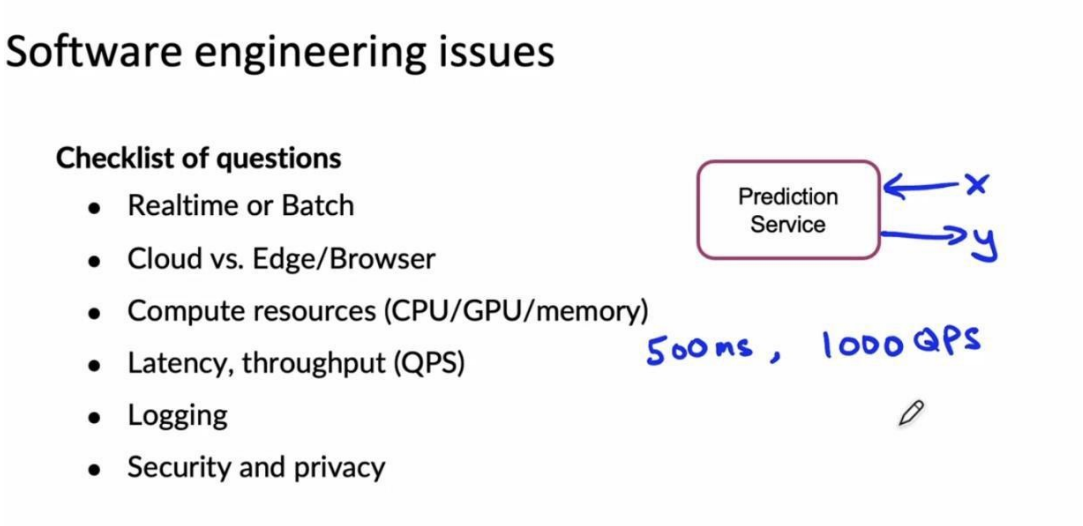
- Finalmente, cuando haya entrenado el modelo y cuando el análisis de errores parezca sugerir que está funcionando lo suficientemente bien, entonces estará listo para pasar a la implementación
- Así es como se puede desplegar un sistema de voz. Tienes un teléfono móvil. Se trataría de un dispositivo de borde con un software que se ejecuta localmente en el teléfono. Ese software interviene en el micrófono para grabar lo que alguien está diciendo. Para la búsqueda de voz y en una implementación típica de reconocimiento de voz, se utilizaría un módulo VAD. VAD son las siglas en inglés de "voice activity detection" (detección de actividad vocal).
- El trabajo del VAD permite al teléfono inteligente seleccionar sólo el audio que contiene, con suerte, a alguien hablando, para que pueda enviar sólo ese clip de audio a su servidor de predicción.
- Y en este caso puede que el servidor de predicción viva en la nube. Este sería un patrón de despliegue común. El servidor de predicción devuelve entonces tanto la transcripción al usuario para que pueda ver lo que el sistema cree que ha dicho. Y también devuelve a los resultados de la búsqueda.
- Si estás haciendo una búsqueda por voz y la transcripción y los resultados de la búsqueda se muestran en el código del frontend que se ejecuta en tu teléfono móvil.
- Por lo tanto, la implementación de este tipo de sistema sería el trabajo necesario para desplegar un modelo de voz en producción, incluso después de que esté en funcionamiento, aunque todavía hay que supervisar y mantener el sistema.
- Así que esto es algo que me pasó una vez que mi equipo había construido un sistema de reconocimiento de voz y fue entrenado principalmente en las voces de los adultos. Lo pusimos en producción, en producción aleatoria, y nos dimos cuenta de que, con el tiempo, cada vez más jóvenes, adolescentes, a veces incluso más jóvenes, utilizaban nuestro sistema de reconocimiento de voz y las voces de los jóvenes son diferentes. Así que el rendimiento de mi sistema de voz comenzó a degradarse. Simplemente no éramos tan buenos en el reconocimiento del habla de las voces más jóvenes. Y así tuvimos que volver y encontrar una manera de recoger más datos son las cosas con el fin de arreglarlo.
- Uno de los principales retos a la hora de la implantación es la deriva del concepto o de los datos, que es lo que ocurre cuando la distribución de los datos cambia, por ejemplo, cuando hay más voces jóvenes que llegan al sistema de reconocimiento del habla.
- Saber cómo poner en marcha los monitores adecuados para detectar estos problemas y luego también cómo solucionarlos a tiempo es una habilidad clave necesaria para asegurarse de que su despliegue de producción crea un valor.

Despliegue - Principales retos

- Existen dos grandes categorías de retos a la hora de implantar un modelo de aprendizaje automático.
- En primer lugar, están los problemas de aprendizaje automático o de estadística, y en segundo lugar, los problemas del motor de software.

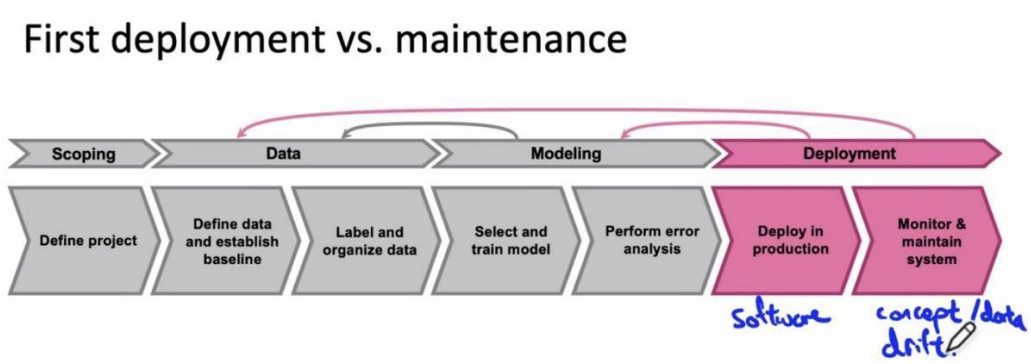


- Uno de los retos de muchas implantaciones es la deriva del concepto y de los datos. En términos generales, esto significa que si sus datos cambian después de que su sistema ya ha sido desplegado
- He entrenado algunos sistemas de reconocimiento de voz, y cuando construía sistemas de voz, a menudo tenía algunos datos comprados. Se trata de datos comprados o con licencia, que incluyen tanto la entrada x, el audio, como la transcripción y, que es la salida del sistema de voz.
- Yo recopilaría un conjunto de desarrollo o un conjunto de validación, así como un conjunto de prueba, con datos de los **últimos meses**. Puedes probarlo con datos bastante recientes para asegurarte de que tu sistema funciona, incluso con datos relativamente recientes.
- Después de poner el sistema en marcha, la pregunta es si los datos cambiarán o si, después de haberlo puesto en marcha durante unas semanas o unos meses, los datos han vuelto a cambiar.
- Los datos pueden haber cambiado, como por ejemplo el idioma, o tal vez la gente está usando un nuevo modelo de smartphone que tiene un micrófono diferente, por lo que el audio suena diferente. Esto hace que el rendimiento de un sistema de reconocimiento de voz se vea afectado.
- Es importante que reconozca cómo han cambiado los datos y si necesita actualizar su algoritmo de aprendizaje como resultado.
- Cuando los datos cambian, a veces se trata de un **cambio gradual**, como la lengua inglesa, que sí cambia, pero lo hace muy lentamente con la introducción de nuevo vocabulario a un ritmo relativamente lento.
- A veces, los datos cambian de forma muy repentina cuando se produce un **choque repentino** en el sistema.
- Por ejemplo, cuando COVID-19 llegó la pandemia, muchos sistemas de fraude con tarjetas de crédito empezaron a no funcionar porque los patrones de compra de los individuos cambiaron de repente.
- Muchas personas que hacían relativamente pocas compras en línea empezaron de repente a utilizar mucho más las compras en línea. El uso de las tarjetas de crédito cambió repentinamente, lo que puso en peligro muchos sistemas antifraude.
- Este cambio tan repentino en la distribución de los datos hizo que muchos equipos de aprendizaje automático tuvieran que pelearse un poco al principio de COVID para recoger nuevos datos y volver a entrenar los sistemas para que se adaptaran a esta nueva distribución de datos.
- Otro ejemplo de **deriva conceptual**, digamos que x es el tamaño de una casa, e y es el precio de una casa, porque se está tratando de estimar los precios de la vivienda. Si debido a la inflación o a los cambios en el mercado, las casas pueden encarecerse con el tiempo. La casa del mismo tamaño, acabará teniendo un precio más alto.
- Eso sería la **deriva del concepto.** Tal vez el tamaño de las casas no haya cambiado, pero sí el precio de una casa determinada.
- Mientras que la **deriva de los datos** se produciría si, por ejemplo, la gente empieza a construir casas más grandes, o empieza a construir casas más pequeñas y, por tanto, la distribución de entrada de los tamaños de las casas cambia realmente con el tiempo.
- Cuando se implanta un sistema de aprendizaje automático, una de las tareas más importantes suele ser asegurarse de que se puede detectar y gestionar cualquier cambio.
- Esto incluye tanto la deriva conceptual, que es cuando la definición de lo que es y dado x, cambia.
- Así como la **Deriva de datos**, que se produce si la distribución de x cambia, aunque el mapeo de x o y no cambie.



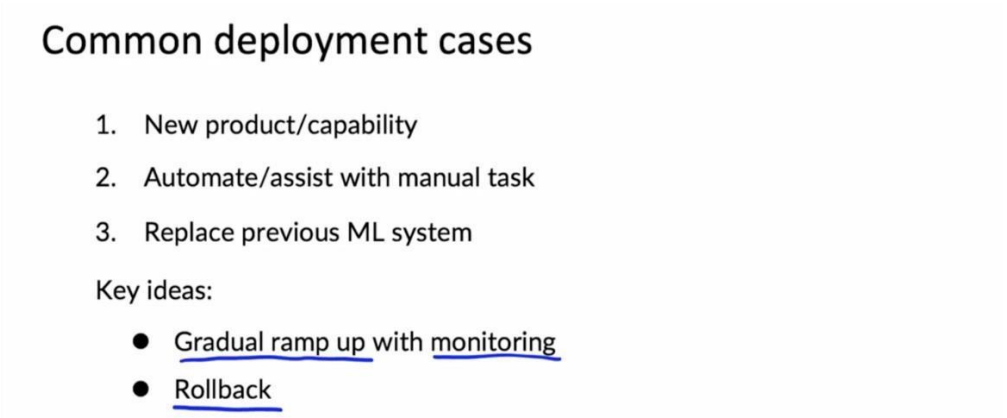
- Otra serie de problemas son los de ingeniería de software
- Supongamos que está implementando un servicio de predicción cuyo trabajo es tomar las consultas X y la predicción de salida Y.
- He aquí una lista de preguntas (arriba) que puede ayudarle a tomar las decisiones adecuadas para gestionar los problemas de ingeniería de software.
- Una decisión que tiene que tomar para su aplicación es: ¿necesita predicciones en tiempo real o predicciones por lotes?
- Por ejemplo, si se está construyendo un sistema de reconocimiento de voz, en el que el usuario habla y se necesita obtener una respuesta en medio segundo, es evidente que se necesitan predicciones en tiempo real.
- Por el contrario, también he construido sistemas para hospitales que toman los registros de los pacientes. Por ejemplo, las historias clínicas electrónicas: Ejecutamos un proceso por lotes durante la noche (una vez por noche) para ver si hay algo asociado a los pacientes
- Tanto si necesitas escribir un software en tiempo real, en el que puedan responder en cientos de milisegundos, como si puedes escribir un software que simplemente haga muchos cálculos de la noche a la mañana, esto afectará a la forma de implementar tu software.
- La segunda pregunta que debes hacerte es: ¿tu servicio de predicción se ejecuta en la nube o se ejecuta en el borde o incluso en un navegador web?
- Hoy en día hay muchos sistemas de reconocimiento del habla que se ejecutan en la nube, ya que disponer de los recursos informáticos de la nube permite un reconocimiento del habla más preciso.
- También hay algunos sistemas de voz, por ejemplo, muchos sistemas de voz dentro de los coches que realmente funcionan en el borde. También hay algunos sistemas de reconocimiento de voz en el móvil que funcionan, incluso si el Wi-Fi está apagado.
- Cuando despliego sistemas de inspección visual en fábricas, casi siempre lo hago también en el borde. Porque a veces, inevitablemente, la conexión a Internet entre la fábrica y el resto de Internet puede caerse. No puedes permitirte el lujo de cerrar la fábrica
- Con el auge de los navegadores web modernos, existen mejores herramientas para desplegar algoritmos de aprendizaje allí mismo, dentro de un navegador web también.
- A la hora de crear un servicio de predicción, también es útil tener en cuenta los recursos informáticos de los que se dispone.
- Ha habido bastantes ocasiones en las que he entrenado una red neuronal en una GPU muy potente, sólo para darme cuenta de que no podía permitirme un conjunto de GPUs igual de potentes para las implantaciones. Tuve que hacer algo más para comprimir o reducir la complejidad del modelo.
- Si sabes cuántos recursos de CPU o GPU y quizás también cuántos recursos de memoria tienes para tu servicio de predicción, eso podría ayudarte a elegir la arquitectura de software adecuada.
- Dependiendo de su aplicación, especialmente si se trata de una aplicación en tiempo real, la latencia y los rendimientos medidos en términos de QPS (consultas por segundo) serán otras métricas de ingeniería de software que puede necesitar.

- En el reconocimiento de voz, no es raro querer obtener una respuesta para el usuario en 500 milisegundos. De este presupuesto de 500 milisegundos, es posible que sólo pueda asignar, por ejemplo, 300 milisegundos al reconocimiento de voz. Esto supone un requisito de latencia para su sistema.
- El rendimiento se refiere a la cantidad de consultas por segundo que necesita manejar dados los recursos de computación,
- Por ejemplo, si estás construyendo un sistema que necesita manejar 1000 consultas por segundo, sería útil asegurarte de revisar tu sistema para que tengas suficientes recursos computacionales, para alcanzar el requerimiento de QPS.
- El siguiente paso es el registro, ya que al construir el sistema puede ser útil registrar la mayor cantidad de datos posible para su análisis y revisión, así como para proporcionar más datos para el reentrenamiento del algoritmo de aprendizaje en el futuro.
- Por último, en lo que respecta a la seguridad y la privacidad, creo que los niveles de seguridad y privacidad requeridos para las distintas aplicaciones pueden ser muy diferentes. Por ejemplo, cuando trabajaba en historias clínicas electrónicas, registros de pacientes, los requisitos de seguridad y privacidad eran claramente muy altos porque los registros de pacientes son información muy sensible.
- Si guarda esta lista en algún lugar, repasarla cuando diseñe su software podría ayudarle a tomar las decisiones adecuadas sobre el motor de software cuando implemente su servicio de predicción.

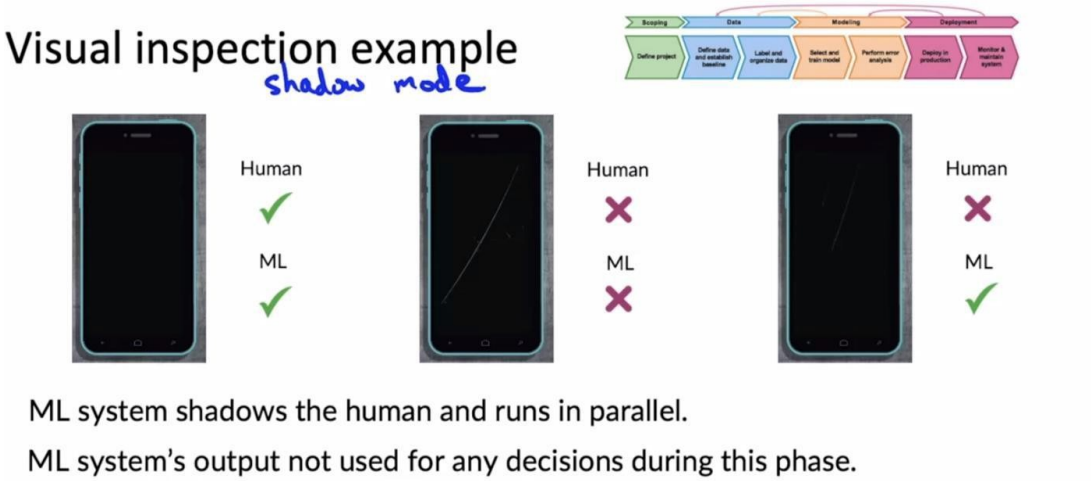


- Para resumir, el despliegue de un sistema requiere dos grandes conjuntos de tareas: está la escritura del software que permite desplegar el sistema en producción. Por otro lado, hay que supervisar el rendimiento del sistema y seguir manteniéndolo, sobre todo ante la deriva de conceptos y de datos.
- Una de las cosas que se ven cuando se construyen sistemas de aprendizaje automático es que las prácticas para los primeros despliegues serán bastante diferentes en comparación con cuando se está actualizando o manteniendo un sistema que ya se ha desplegado previamente.
- Sé que para algunos ingenieros el despliegue del modelo de aprendizaje automático es como llegar a la línea de meta. Desgraciadamente, creo que el primer despliegue significa que puedes estar sólo a mitad de camino, y que la segunda mitad de tu trabajo está empezando sólo después de tu primer despliegue. Esto se debe a que incluso después de haber desplegado, hay un montón de trabajo para alimentar los datos de nuevo y tal vez para actualizar el modelo, para seguir manteniendo el modelo, incluso en la cara de los cambios en los datos.

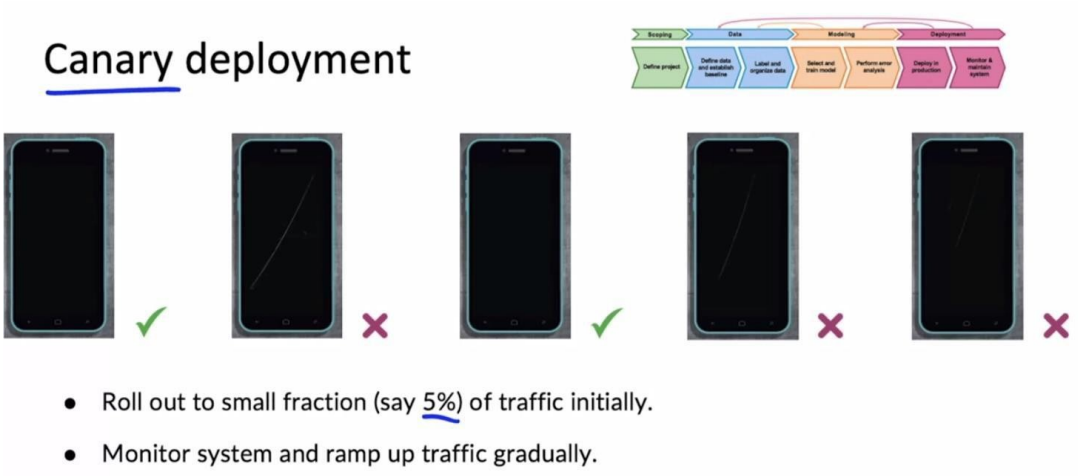
Patrones de despliegue



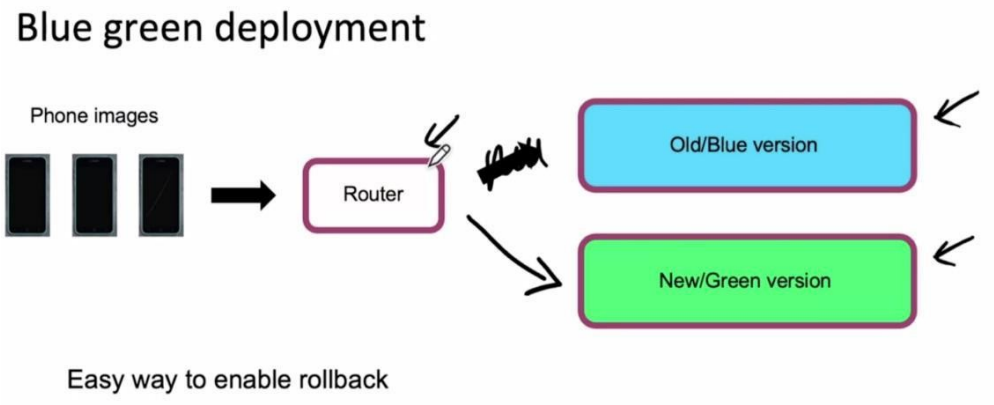
- Un tipo de despliegue es el que se produce cuando se ofrece un nuevo producto o capacidad que no se había ofrecido anteriormente.
- Por ejemplo, si está ofreciendo un servicio de reconocimiento de voz que no ha ofrecido antes, en este caso, un patrón de diseño común es comenzar con una pequeña cantidad de tráfico y luego aumentar gradualmente.
- Un segundo caso de uso común es si hay algo que ya está haciendo una persona, pero ahora nos gustaría utilizar un algoritmo de aprendizaje para automatizar o ayudar con esa tarea.
- Por ejemplo, si tiene personas en la fábrica que inspeccionan los arañazos de los smartphones, pero ahora le gustaría utilizar un algoritmo de aprendizaje para ayudar o automatizar esa tarea.
- El hecho de que la gente estuviera haciendo esto previamente te da unas cuantas opciones más para el despliegue. Y ves que el despliegue en modo sombra se aprovecha de esto.
- Y, por último, un tercer caso de implementación común es si ya has estado haciendo esta tarea con una implementación anterior de un sistema de aprendizaje automático, pero ahora quieres reemplazarlo por uno mejor.
- En estos casos, **dos temas recurrentes** que se ven son que a menudo se quiere un aumento gradual con la supervisión. En otras palabras, en lugar de enviar toneladas de tráfico a un algoritmo de aprendizaje que quizá no esté totalmente probado, se puede enviar sólo una pequeña cantidad de tráfico y monitorizarlo y luego aumentar el porcentaje o la cantidad de tráfico.
- Y la segunda idea que se ve algunas veces es la reversión. Significa que si por alguna razón el algoritmo no está funcionando, es bueno si usted puede volver al sistema anterior



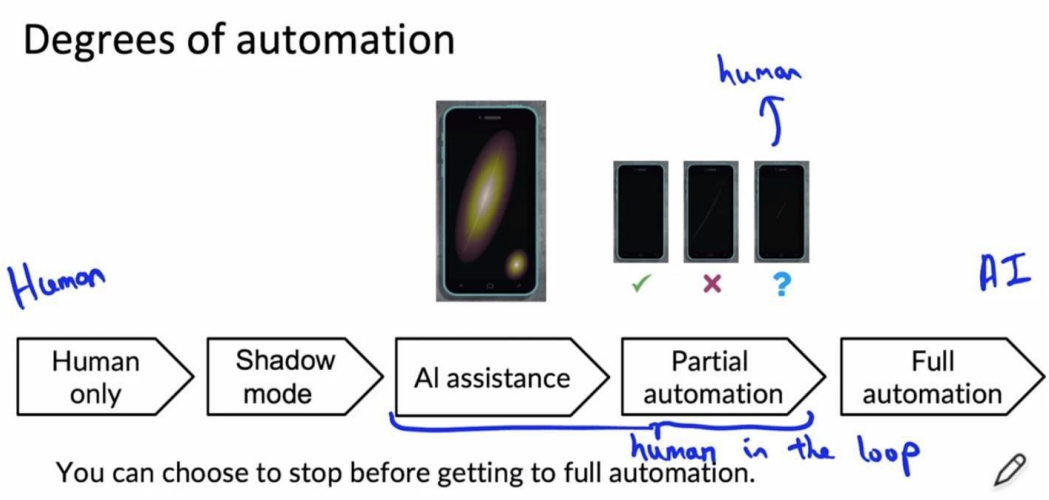
- Cuando hay personas realizando inicialmente una tarea, un patrón de despliegue común es utilizar **el despliegue en modo sombra.** Esto significa que un algoritmo de aprendizaje automático seguirá de cerca al inspector humano y lo ejecutará en paralelo.
- Durante esta fase inicial, la salida de los algoritmos de aprendizaje no se utiliza para ninguna decisión en la fábrica. Así que, diga lo que diga el algoritmo de aprendizaje, vamos a seguir el criterio humano por ahora.
- Así que digamos que para este smartphone (imagen de la izquierda), el humano dice que está bien, que no tiene ningún defecto. El algoritmo de aprendizaje dice que está bien. Tal vez para este ejemplo (imagen central) de un gran tramo hacia abajo la persona del medio dice que no está bien y el algoritmo de aprendizaje está de acuerdo.
- Y para este ejemplo (imagen de la derecha) con un tramo más pequeño, tal vez la persona dice que esto no está bien, pero el algoritmo de aprendizaje comete un error y realmente piensa que esto está bien.
- El propósito de un despliegue en modo sombra es que le permite recoger datos de **cómo el algoritmo de aprendizaje está funcionando y cómo se compara con el juicio humano**.
- Esto permite verificar si las predicciones del algoritmo de aprendizaje son precisas y, por lo tanto, utilizarlo para decidir si permitir o no que el algoritmo de aprendizaje tome algunas decisiones reales en el futuro.
- El uso de un despliegue en modo sombra puede ser una forma muy eficaz de permitirle verificar el rendimiento de un algoritmo de aprendizaje antes de permitirle tomar decisiones reales.



- Cuando esté listo para dejar que un algoritmo de aprendizaje comience a tomar decisiones reales, un patrón de despliegue común es utilizar un despliegue canario.
- En un despliegue canario se desplegaría a una pequeña fracción, tal vez el 5% (o incluso menos del tráfico inicialmente) y se empezaría a dejar que el algoritmo tomara decisiones reales.
- Pero al ejecutar esto sólo en un pequeño porcentaje del tráfico, si el algoritmo comete algún error sólo afectará a una pequeña fracción del tráfico.
- Y esto le da más posibilidades de controlar el sistema y aumentar el porcentaje de tráfico que recibe sólo gradualmente y sólo cuando tenga más confianza en este rendimiento.
- La expresión "despliegue de canarios" es una referencia al idioma inglés, que alude a cómo los mineros del carbón solían utilizar canarios para detectar si había una fuga de gas.
- Con el despliegue del canario, es de esperar que esto le permita detectar los problemas desde el principio antes de que haya consecuencias demasiado grandes para una fábrica u otro contexto en el que esté desplegando su algoritmo de aprendizaje.



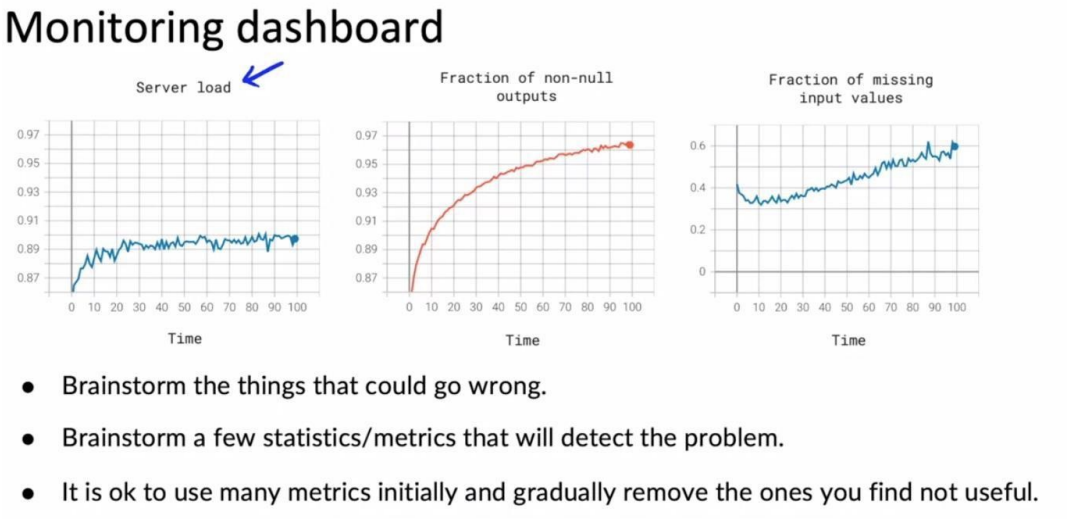
- Otro patrón de despliegue que se utiliza a veces es el despliegue azul-verde.
- Digamos que tienes un sistema, un software de cámara para recoger imágenes de teléfono en tu fábrica. Estas imágenes telefónicas se envían a un software que las toma y las envía a un sistema de inspección visual.
- En la terminología de un despliegue azul-verde, la **versión antigua de su software se llama versión azul** y la nueva versión, el algoritmo de aprendizaje que acaba de implementar, se llama versión verde.
- En un despliegue azul-verde, lo que se hace es que el router envíe imágenes a la versión antigua (azul) y que ésta tome las decisiones. Y luego, cuando quieras cambiar a la nueva versión, lo que harías es que el router dejara de enviar imágenes a la antigua y cambiara de repente a la nueva versión.
- Así que la forma en que se implementa el despliegue azul-verde es que tendrías un servicio de predicción antiguo que podría estar funcionando en algún tipo de servicio. A continuación, se pondrá en marcha un nuevo servicio de predicción, la versión verde, y se hará que el router cambie de repente el tráfico del antiguo al nuevo.
- La ventaja de un despliegue azul-verde es que hay una manera fácil de habilitar la reversión. Si algo va mal, puedes simplemente hacer que el router vuelva a reconfigurar su router para enviar el tráfico de vuelta a la versión antigua o azul, asumiendo que mantuviste tu versión azul del servicio de predicción en funcionamiento.
- En una implementación típica de un despliegue azul-verde, la gente piensa en cambiar el tráfico al 100% al mismo tiempo. Pero, por supuesto, también se puede utilizar una versión más gradual en la que se envía lentamente el tráfico.



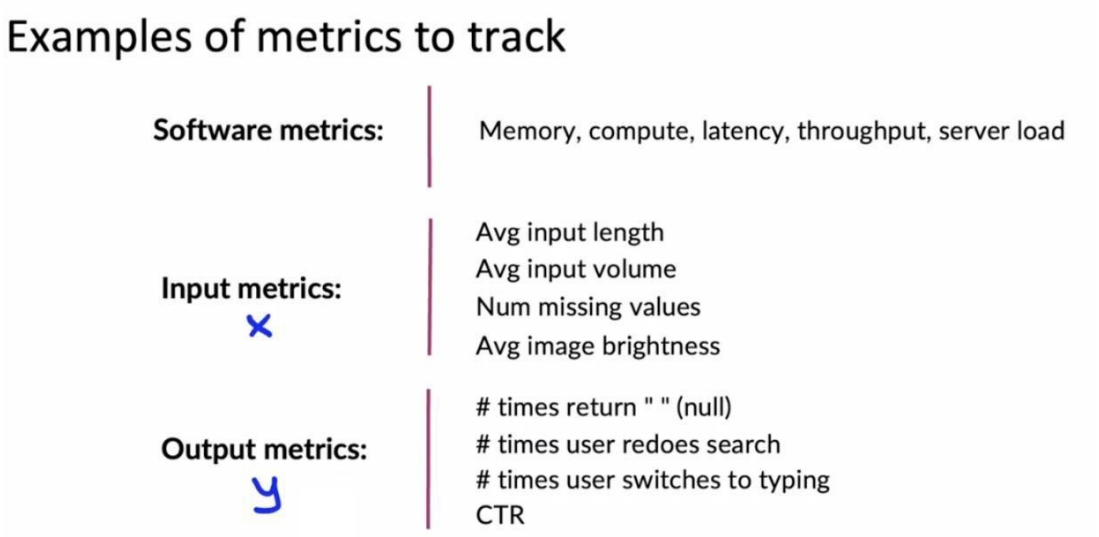
- Uno de los marcos más útiles que he encontrado para pensar en cómo desplegar un sistema es pensar en el despliegue no como un 0 / 1 (es decir, desplegar o no desplegar), sino diseñar un sistema pensando en cuál es el **grado apropiado de automatización**.

- Por ejemplo, en la inspección visual de smartphones, un extremo sería si no hay automatización, (sistema sólo humano). Un modo ligeramente automatizado sería si su sistema está ejecutando un modo de sombra, donde los algoritmos de aprendizaje están dando predicciones pero no se utilizan realmente en la fábrica.
- Un grado ligeramente mayor de automatización sería la asistencia mediante IA, en la que un inspector humano toma las decisiones, pero un sistema de IA proporciona una interfaz de usuario para resaltar las regiones en las que hay un rasguño para ayudar a llamar la atención de la persona hacia donde puede ser más útil que mire. El diseño de la interfaz de usuario o UI es fundamental para la asistencia humana.
- Un grado aún mayor de automatización es la automatización parcial. Si el algoritmo de aprendizaje está seguro de que el smartphone está bien o es defectuoso, entonces esa es la decisión final. Pero si el algoritmo de aprendizaje no está seguro, es decir, si la predicción del algoritmo de aprendizaje no es demasiado segura, entonces enviamos esto a un humano para que tome la decisión.
- Así que esto sería una automatización parcial, donde si el algoritmo de aprendizaje está seguro de su predicción, vamos el algoritmo de aprendizaje. Pero para el pequeño subconjunto de imágenes en las que el algoritmo no está seguro, lo enviamos a un humano para que lo juzgue.
- Y el juicio humano también puede ser un dato muy valioso para retroalimentar para seguir entrenando y mejorando el algoritmo.
- Creo que esta automatización parcial es a veces un punto de diseño muy bueno para aplicaciones en las que el **rendimiento de** los algoritmos de aprendizaje **no es lo suficientemente bueno para la automatización completa.**
- Y luego, por supuesto, más allá de la automatización parcial, está la **automatización total**, en la que podemos hacer que el algoritmo de aprendizaje tome todas las decisiones.
- Por lo tanto, existe un espectro que va desde el uso de las decisiones humanas a la izquierda, hasta el uso de las decisiones del sistema de IA a la derecha. Y muchas aplicaciones de despliegue empezarán por la izquierda y se desplazarán gradualmente hacia la derecha.
- No es necesario llegar a la automatización total. Puedes optar por dejar de utilizar la asistencia de la IA o la automatización parcial o puedes optar por llegar a la automatización total en función del rendimiento de tu sistema y de las necesidades de la aplicación.
- En este espectro, tanto la asistencia de la IA como la automatización parcial son ejemplos de **despliegue de seres humanos en el bucle.**
- Muchas empresas de software de consumo en Internet tienen que utilizar la automatización completa porque simplemente no es factible que alguien en el back end haga algún trabajo cada vez que alguien hace una búsqueda en la web o hace la búsqueda del producto.
- Pero fuera del software de consumo de Internet, por ejemplo, la inspección de las cosas y las fábricas, en realidad son muchas las aplicaciones en las que el mejor punto de diseño puede ser un despliegue humano en el bucle en lugar de un despliegue de automatización completa.

Monitorización

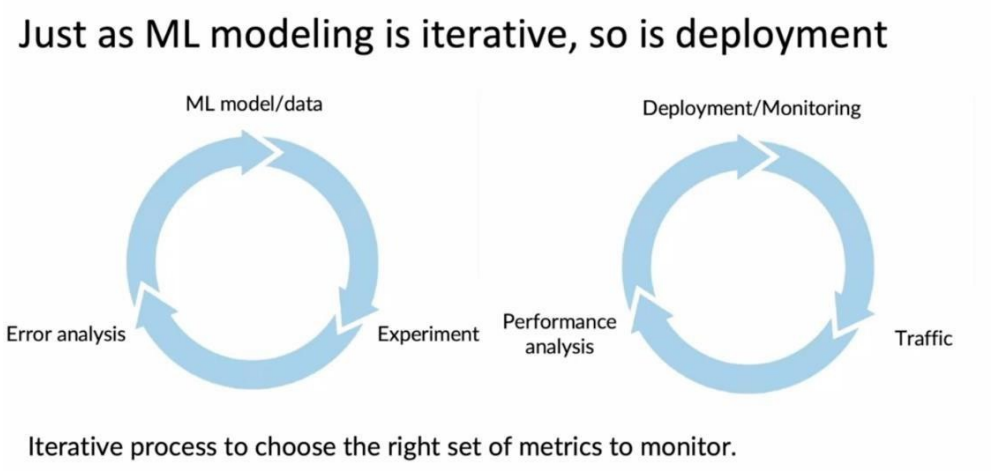


- ¿Cómo puede supervisar un sistema de aprendizaje automático para asegurarse de que cumple sus expectativas de rendimiento? La forma más habitual de supervisar un sistema de aprendizaje automático es utilizar un panel de control para hacer un seguimiento de su rendimiento a lo largo del tiempo.
- Dependiendo de su aplicación, sus cuadros de mando pueden monitorizar diferentes métricas. Por ejemplo, usted puede tener un tablero para monitorear la carga del servidor, o un tablero diferente para monitorear la fracción de salidas no nulas. A veces, la salida de un sistema de reconocimiento de voz es nula cuando las cosas que los usuarios no dijeron nada.
- Si esto cambia drásticamente con el tiempo, puede ser una indicación de que algo está mal
- Uno de los aspectos más comunes que he visto en muchas tareas de datos estructurados es el control de la fracción de entradas que faltan
- Cuando intente decidir qué supervisar, mi recomendación es que se siente con su equipo y haga una lluvia de ideas sobre todas las cosas que podrían ir mal. Luego querrás saber si algo va mal. Para todas las cosas que podrían ir mal, haz una lluvia de ideas sobre algunas estadísticas o métricas que detecten ese problema.
- Por ejemplo, si le preocupa que el tráfico de usuarios aumente, causando que el servicio se sobrecargue, entonces las cargas del servidor podrían ser una métrica que podría rastrear y así sucesivamente para los otros ejemplos

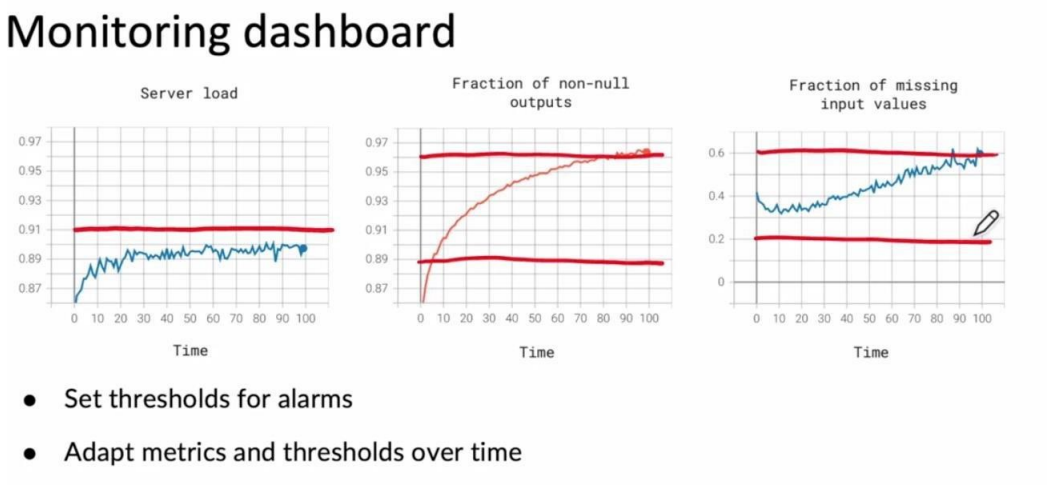


- He aquí algunos ejemplos de métricas que nosotros o yo hemos visto que otros utilizan en una variedad de proyectos. En primer lugar están las métricas de software, como la memoria, el cálculo, la latencia, el rendimiento, la carga del servidor, es decir, cosas que le ayudan a supervisar la salud de su implementación de software del servicio de predicción u otras piezas de software alrededor de su algoritmo de aprendizaje.
- Muchas herramientas de MLOps ya hacen un seguimiento de estas métricas de software.
- Además de las métricas del software, a menudo elegiría otras métricas que ayuden a controlar la salud estadística o el rendimiento del algoritmo de aprendizaje. A grandes rasgos, hay dos tipos de métricas en torno a las cuales podría hacer una lluvia de ideas.
- Una es la métrica de entrada, que mide si su distribución de entrada **X** ha cambiado
- Por ejemplo, si estás construyendo un sistema de reconocimiento de voz, podrías controlar la duración media de la entrada en segundos del clip de audio alimentado a tu sistema.
- Si estos cambian por alguna razón, que podría ser algo que usted querrá echar un vistazo sólo para asegurarse de que no ha perjudicado el rendimiento de su algoritmo.
- El número (o porcentaje) de valores perdidos es una métrica muy común (por ejemplo, cuando se utilizan datos estructurados, algunos de los cuales pueden tener valores perdidos)
- Para el ejemplo de la inspección visual de la fabricación, podría supervisar el brillo medio de la imagen si cree que las condiciones de iluminación podrían cambiar, y quiere asegurarse de saber si lo hacen, para poder pensar en diferentes métricas para ver si su distribución de entrada x podría haber cambiado.
- Un segundo conjunto de métricas que ayudan a entender si el algoritmo está funcionando bien son las **métricas de salida**

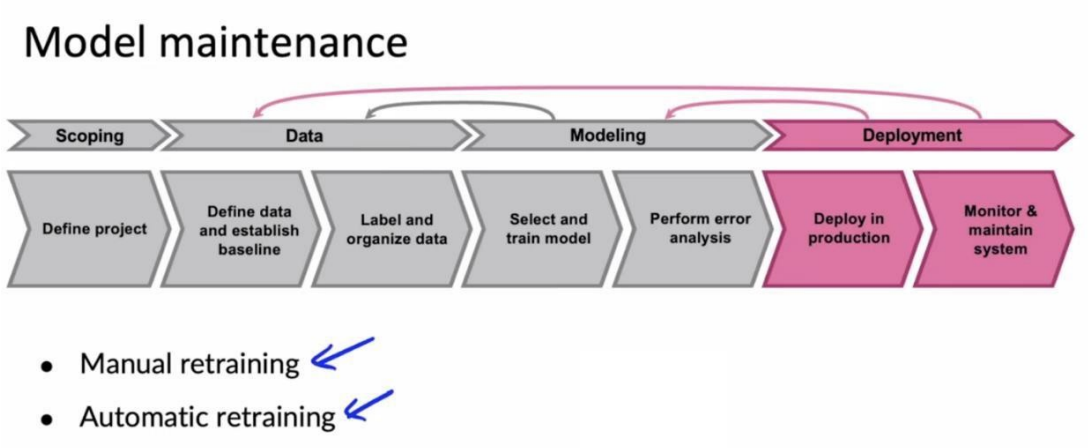
- Por ejemplo, cuántas veces su sistema de reconocimiento de voz devuelve null, la cadena vacía, porque las cosas que el usuario no dice nada, o si usted ha construido un sistema de reconocimiento de voz para la búsqueda en la web utilizando la voz, usted podría decidir para ver con qué frecuencia el usuario hace dos búsquedas muy rápidas en una fila con sustancialmente la misma entrada.
- Eso podría ser una señal de que reconoces mal su consulta la primera vez. Es una señal imperfecta, pero podrías probar esta métrica y ver si te ayuda.
- También se puede controlar el número de veces que el usuario intenta utilizar el sistema de voz por primera vez y luego pasa a teclear, lo que podría ser una señal de que el usuario se ha frustrado o se ha rendido con el sistema de voz y podría indicar una degradación del rendimiento.
- Dado que las métricas de entrada y salida son específicas de la aplicación, la mayoría de las herramientas de MLOps necesitarán ser configuradas específicamente para rastrear las métricas de entrada y salida de su aplicación.



- Es posible que ya sepa que el modelado de aprendizaje automático es un proceso altamente iterativo, por lo que como despliegue.
- Le animo a que piense en los despliegues como un proceso iterativo también. Cuando tengas tus primeros despliegues en marcha y pongas en marcha un conjunto de paneles de control.
- Pero eso es sólo el comienzo de este proceso iterativo. Un sistema en funcionamiento te permite obtener datos de usuarios reales o tráfico real. Ver cómo se comporta el algoritmo de aprendizaje con datos reales sobre tráfico real es lo que te permite hacer un análisis del rendimiento, y esto, a su vez, te ayuda a actualizar tu despliegue y a seguir monitorizando tu sistema.
- Por lo general, se necesitan unos cuantos intentos para converger en el conjunto correcto de métricas a monitorizar. No es raro que despliegues un sistema de aprendizaje automático con un conjunto inicial de métricas, sólo para ejecutar el sistema durante unas semanas y darte cuenta de que algo podría ir mal que no habías pensado antes, y tendrás que elegir una nueva métrica para monitorear.
- O para que tengas alguna métrica que supervises durante unas semanas y luego decidas que apenas cambian, entonces podemos deshacernos de esa métrica en favor de centrar la atención en otra cosa.



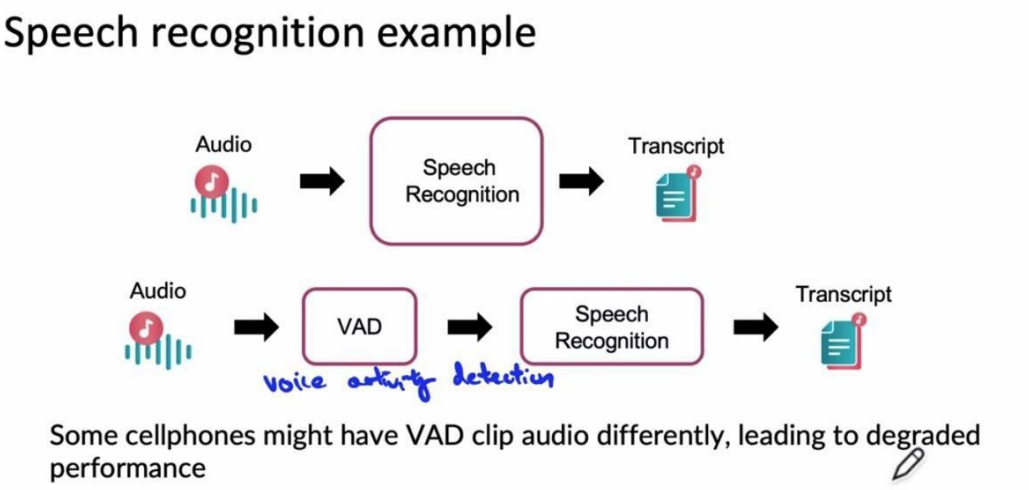
- Después de haber elegido un conjunto de métricas para monitorear, una práctica común sería establecer umbrales para las alarmas. Usted puede decidir sobre la base de este conjunto, si la carga del servidor nunca va por encima de 0,91, puede activar una alarma o una notificación para que usted o el equipo para ver si hay un problema y tal vez girar algunos servidores más.
- Está bien si adaptas las métricas y los umbrales a lo largo del tiempo para asegurarte de que te están señalando los casos más relevantes de preocupación. Si algo va mal con tu algoritmo de aprendizaje, si es un problema de software como que la carga del servidor es demasiado alta, entonces eso puede requerir cambiar la implementación del software,
- Si se trata de un problema de rendimiento o de precisión asociado a la exactitud del algoritmo de aprendizaje, es posible que tenga que actualizar su modelo.
- Por eso, muchos modelos de aprendizaje automático necesitan un poco de mantenimiento o reentrenamiento a lo largo del tiempo.



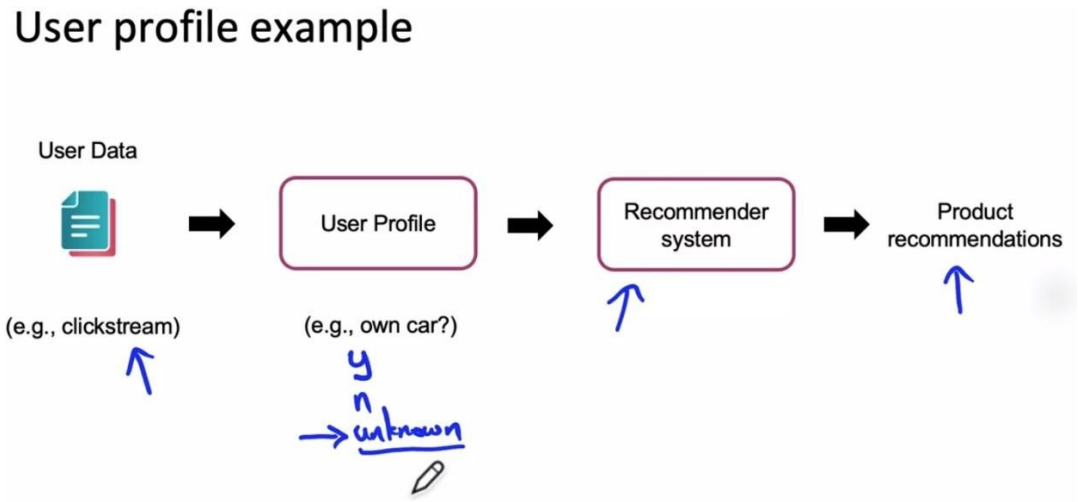
- Cuando hay que actualizar un modelo, se puede reentrenar manualmente (junto con los ingenieros), o también se puede poner en marcha un sistema en el que haya un reentrenamiento automático.
- Hoy en día, el reentrenamiento manual es mucho más común que el entrenamiento automático. Para muchas aplicaciones, los desarrolladores son reacios a que el algoritmo de aprendizaje sea totalmente automático en cuanto a la decisión de reentrenar y empujar el nuevo modelo a la producción, pero hay algunas aplicaciones, especialmente en el software de consumo de Internet, en las que el entrenamiento automático sí se produce.
- Pero lo más importante es que sólo mediante la monitorización del sistema se puede detectar si hay algún problema que pueda hacer que se vuelva a realizar un análisis de errores más profundo, o que se vuelva a obtener más datos con los que se pueda actualizar el modelo para mantener o mejorar el rendimiento del sistema.

Supervisión de oleoductos y gasoductos

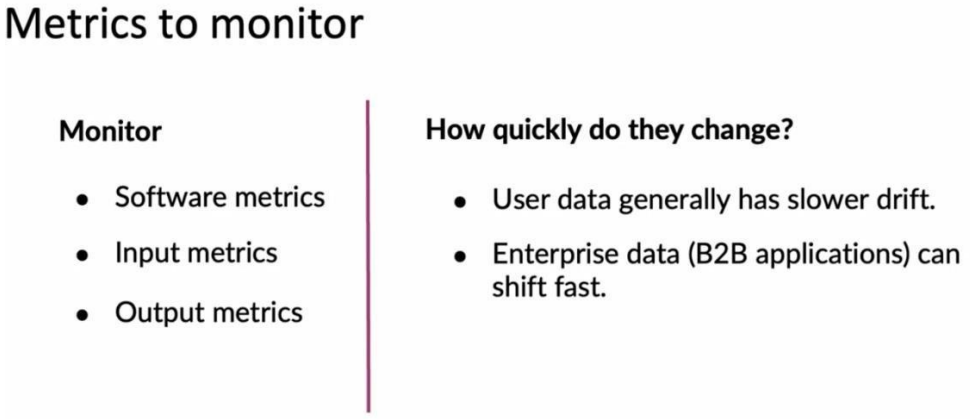
- Muchos sistemas de IA no se limitan a un único modelo de aprendizaje automático que ejecuta un servicio de predicción, sino que implican una cadena de múltiples pasos.



- Sigamos con nuestro ejemplo de reconocimiento de voz, ya has visto como un sistema de reconocimiento de voz puede tomar como entrada el audio y voy a poner una transcripción.
- La forma en que se implementa el reconocimiento de voz es un proceso un poco más complejo, en el que el audio se envía a un módulo llamado VAD (detección de actividad de voz), cuyo trabajo es ver si alguien está hablando. Y sólo si el módulo VAD cree que alguien está hablando, se molesta en pasar el audio a un sistema de reconocimiento de voz cuyo trabajo es generar la transcripción.
- Utilizamos un módulo VAD porque si tu sistema de reconocimiento de voz se ejecuta en la nube, no querrás transmitir más ancho de banda del necesario a tu servidor en la nube.
- Así, el módulo VAD examina el largo flujo de audio del teléfono móvil y recorta o acorta el audio sólo en la parte en la que alguien está hablando y transmite sólo eso al servidor en la nube para realizar el reconocimiento de voz.
- Este es un ejemplo de un proceso de aprendizaje automático en el que un algoritmo de aprendizaje realiza un paso para decidir si alguien está hablando o no, y luego el segundo paso, también realizado por un algoritmo de aprendizaje, para generar la transcripción del texto.
- Cuando hay dos módulos de este tipo trabajando juntos, los cambios en el primer módulo pueden afectar también al rendimiento del segundo.
- Por ejemplo, digamos que debido a la forma en que funciona el micrófono de un nuevo teléfono móvil, el módulo VAD acaba recortando el audio de forma diferente. Quizá deja más silencio al principio o al final o menos silencio al principio o al final.
- Esto hará que la entrada de los sistemas de reconocimiento de voz cambie, y eso podría causar un rendimiento degradado del sistema de reconocimiento de voz.



- Veamos un ejemplo relacionado con los perfiles de los usuarios. He utilizado los datos como los datos de flujo de clics que muestran lo que los usuarios están haciendo clic en. Y esto se puede utilizar para construir un perfil de usuario que trata de capturar los atributos clave o características clave de un usuario.
- Por ejemplo, una vez construí perfiles de usuario que trataban de esperar muchos atributos de los usuarios, incluyendo si el usuario parecía tener un coche o no. Porque esto nos ayudaría a decidir si merecía la pena intentar ofrecer una oficina de seguros de coches a ese usuario.
- Y así, si el usuario es propietario de un coche puede ser sí o no o desconocido. La forma típica en que se construye el perfil del usuario es con un algoritmo de aprendizaje para tratar de predecir si este usuario del coche.
- Este tipo de perfil de usuario, que puede tener una lista muy larga de atributos previstos, puede servir para recomendar un sistema.
- Otro algoritmo de aprendizaje que luego toma esta comprensión del usuario para tratar de generar recomendaciones de productos.
- Ahora bien, si algo de los datos del flujo de clics cambia, tal vez la distribución de la entrada cambie, entonces con el tiempo perdemos nuestra capacidad de averiguar si un usuario es dueño de un coche, donde el porcentaje de la etiqueta desconocida aquí puede subir.
- Y como los perfiles de los usuarios cambian, la información que recibe el sistema de recomendación también cambia, lo que puede afectar a la calidad de las recomendaciones de productos.



- Así que cuando se construyen estos complejos pipelines de aprendizaje automático, que pueden tener componentes basados en el aprendizaje ML o componentes no basados en el ML a lo largo del pipeline.
- Me parece útil hacer una lluvia de ideas sobre las métricas que hay que supervisar para detectar los cambios, incluyendo la deriva del concepto o los datos, o ambos, y las múltiples etapas del proceso.

- Las métricas que se deben supervisar incluyen métricas de software para cada uno de los componentes del canal, o quizás para el canal en su conjunto, así como métricas de entrada y potencialmente de salida para cada uno de los componentes del canal.
- Y también mediante una lluvia de ideas sobre las métricas asociadas a los componentes individuales de la tubería.
- El principio que viste en el último video de hacer una lluvia de ideas sobre todas las cosas que podrían ir mal, incluyendo las cosas que podrían ir mal con los componentes individuales de la tubería y el diseño de métricas para rastrearlas, todavía se aplica. Sólo que ahora estás viendo múltiples componentes en la tubería.
- Por último, ¿con qué rapidez cambian los datos? La velocidad a la que cambian los datos depende mucho del problema.
- Por ejemplo, supongamos que construimos un sistema de reconocimiento facial. La velocidad a la que cambia la apariencia de las personas no suele ser tan rápida. Los peinados y la ropa de la gente sí cambian con los cambios de la moda. Y a medida que las cámaras mejoran, vamos obteniendo imágenes de las personas con una resolución cada vez mayor. Pero, en general, la apariencia de las personas no cambia tanto.
- A veces, las cosas también pueden cambiar muy rápidamente, por ejemplo, si una fábrica recibe un nuevo lote de material para la fabricación de teléfonos móviles, por lo que todos los teléfonos móviles cambian de aspecto. Así que algunas aplicaciones tendrán datos que cambian en una escala de tiempo de meses o incluso años.
- Algunas aplicaciones con datos que podrían cambiar repentinamente en cuestión de minutos. Hablando en términos muy generales, creo que, por término medio, los datos **de los usuarios suelen cambiar con relativa lentitud**. Si diriges un negocio orientado al consumidor con un gran número de usuarios, es bastante raro que millones de usuarios cambien repentinamente su comportamiento al mismo tiempo.
- Hay algunas excepciones, por supuesto, siendo COVID-19 una de ellas, en la que una conmoción en la sociedad provoca que el comportamiento de muchas personas cambie al mismo tiempo.
- Así que hay excepciones, pero en promedio, si tienes un grupo muy grande de usuarios, sólo hay unas pocas fuerzas que pueden cambiar simultáneamente el comportamiento de muchas personas al mismo tiempo.
- En cambio, si se trabaja en una aplicación B2B o de empresa a empresa, me parece que los **datos de la empresa o los datos comerciales pueden cambiar con bastante rapidez**. Porque la fábrica de teléfonos móviles puede decidir de repente utilizar un nuevo revestimiento para los teléfonos móviles, y de repente todo el conjunto de datos cambia porque los teléfonos móviles de repente tienen un aspecto diferente. O, a veces, si el director general de esa empresa decide cambiar la forma en que opera el negocio, todos esos datos pueden cambiar muy rápidamente.
- Sé que estas dos viñetas hablan en general y que hay ciertas excepciones a ambas. Pero tal vez esto te dé una forma de pensar sobre la rapidez con la que tus datos pueden cambiar o no cambiar.

Laboratorios

- https://github.com/https-deeplearning-ai/MLEP-public/tree/main/course1/week1-ungraded-lab

Referencias de lectura

- https://towardsdatascience.com/machine-learning-in-production-why-you-should-care-about-data-and-concepto-drift-d96d0bc907fb
- https://christophergs.com/machine%20learning/2020/03/14/how-to-monitor-machine-learning-models/ http://arxiv.org/abs/2011.09926
- https://papers.nips.cc/paper/2015/file/86df7dcfd896fcaf2674f757a2463eba-Paper.pdf
- http://arxiv.org/abs/2010.02013