Jornadas de Data Mining y Business Intelligence

Casuística de Implementación II

Maestría en Explotación de Datos y Gestión del Conocimiento

Franco Lianza, Alexis Walker

26 diciembre 2021

Índice

1 DataOps y MLOps: Cómo escalar el delivery de productos de datos sin morir en el intento

Con el ya instituido entendimiento en distintos estratos y roles dentro de las compañías sobre la necesidad de trabajar con datos, y su carácter de activo que conlleva a reconocer el valor estratégico de su explotación; es importante ocuparse del ciclo de vida de los mismos. Este concepto se identifica en la industria como: *DataOps* y *MLOps*.

Muchos intentos de incorporar esta capacidad dentro del departamento de IT terminan en proyectos inconclusos debido a la falta de gestión de la deuda técnica y metodológica vinculada al gobierno y ciclo de vida del entregable de un científico de datos.

Se recorre un poco de historia para reconocer el estado del arte actual que permite sortear el inconveniente antes planteado:

2008 - se introduce el concepto de DevOps que pretende agilizar el proceso de desarrollo de software focalizando en la entrega continua (construcción, implementación y mantenimiento).

2014 - se empieza a discutir sobre *DataOps*, el universo ya no solo se centra en el código sino que se suman los datos.

2015 - Google, en su paper *Hidden Technical Debt in Machine Learning Systems*, visibiliza la carencia metodológica y la deuda técnica que existía en los equipos de ciencias de datos. Lo que introduce la idea de *MLOps*, donde ahora son tres los conceptos: código, datos y modelos; y su gestión para escalar la entrega de este tipo de productos.

1.1 Racionales

Hoy, las soluciones originadas a partir de los datos, se tornaron críticas; esto debido a estar cada vez más embebidas en los sistemas *cores* de una empresa. Por otro lado, la evidencia concreta del valor que son capaces de otorgar, ha significado una importante alfabetización de las áreas usuarias, implicando una mayor y más compleja demanda. Lo que se traduce en la exposición de aquello profesionales encargados de desarrollarlas.

Los problemas comunes que surgen en este contexto suelen ser: el constante flujo de nuevas necesidades producto de los casos de éxito, la baja calidad de los datos y la posibilidad de evolucionar del entregable (siempre existe algo para mejorar).

Estos inconvenientes se suelen abordar, erróneamente, sobreexigiendo al equipo de desarrollo (lo que no escala debido al flujo constante de requerimientos), siendo optimistas en el resultado obtenido, o bien, sobrestimando la capacidad necesaria para amortiguar cualquier desfajase de calendario.

Al panorama descrito se suma una baja credibilidad en las chances de éxito que este tipo de proyecto tiene, sustentada por estudios realizados por las principales consultoras globales.

La realidad detallada, es el *driver*, la argumentación que tracciona la necesidad de implementar nuevas herramientas para aumentar y mejorar el procesamiento. Concretamente el primer concepto que delimita este camino es *DataOps*, y posteriormente *MLOps*.

Para acortar los ciclos de desarrollo manteniendo la calidad, se combina: **Agile + DevOps + SPC = DataOps**. Donde, el agilismo incorpora al usuario, debido a la naturaleza cambiante del entorno, para poder hacer entregas rápidas que agreguen valor. Dentro de estas pretensiones, la infraestructura es provista como servicio, evitando así, demoras en su configuración. Y, por último, se automatiza el control estadístico de procesos (SPC), es decir, en las distintas etapas de enriquecimiento de los datos se realizan validaciones para garantizar la calidad del resultado final.

1.2 Conceptos heredados de la industria del software

El cambio conceptual planteado y argumentado hasta el momento esta muy bien instanciado en algunas herramientas tecnológicas que permiten materializar las ideas subyacentes.

1.2.1 DataOps

El concepto cumbre e inicial de esta evolución es el **versionado de código** debido a que el equipo de datos trabajan sobre un código base. E inmediatamente se vincula a éste su **ciclo de vida** (figura 1).

Fig. 1: Integración/Entrega/Despliegue Continuo

Integración continua: los desarrollos que pueden hacer distintos miembros el equipo deben ser validados antes de integrarse al código fuente base.

Entrega continua: se agrega valor en cada iteración.

Despliegue continuo: implementación con baja fricción.

Por último, las pruebas no solo verifican la calidad de código, sino también, en este esquema, la de los datos, incorporando entonces el **control estadístico de procesos**.

1.2.2 MLOps

Cuando, al *workflow* de trabajo donde se presentan los conceptos enumerado anteriormente (garantizando la calidad de los datos), se suma la convivencia con los modelos, hablamos de *MLOps*.

Esta evolución responde a una problemática real originada por: la necesidad de monitoreo, etiquetado de los datos, comprensión de los modelos, optimización de inferencia, dispositivos *edge* (por ejemplo: celulares) y la privacidad de datos sensibles.

1.3 Ejemplo

MLflow es una tecnología *open source* que convive facilmente con otras y ayuda a resolver las principales etapas enumeradas en el flujo de trabajo (figura 2).

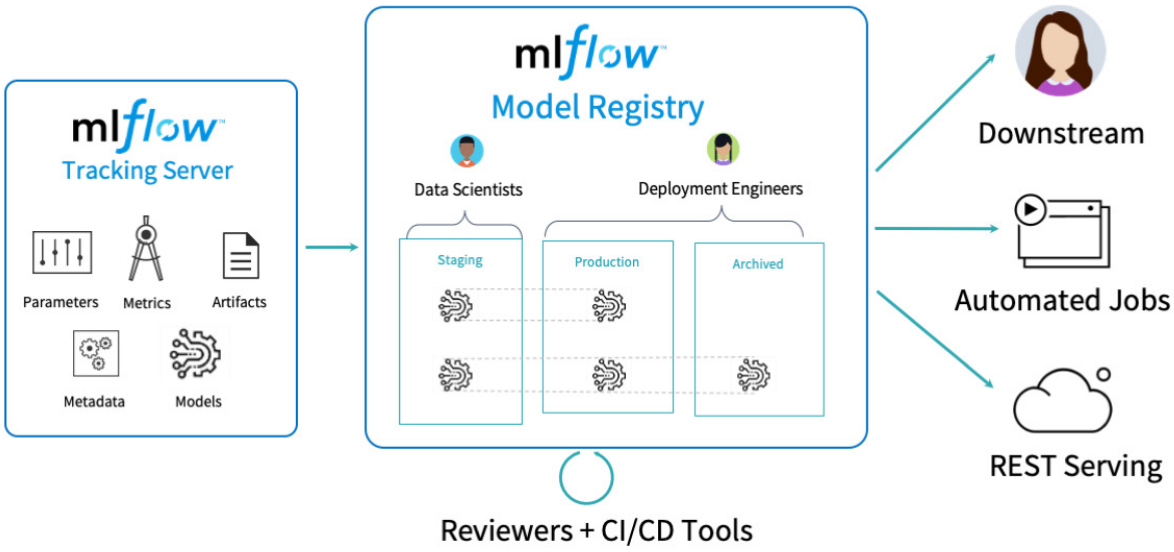

Fig. 2: mlflow

Concretamente permite administrar el ciclo de vida de *ML*, incluida la experimentación, la reproducibilidad, la implementación y un registro de modelos central.

Es posible el monitoreo de datos (controlando también su dominio), de parámetros, de métricas de resultados y, además, control del entorno donde se entrenó el modelo y la serialización del mismo.

Con esto bajo gobierno, es sencillo promover un experimento como resultado definitivo en el registro.

1.4 Conclusiones

Lo expresado no se trata de implementar un software que ayude al equipo de ciencia de datos a desarrollar mejor, no se describe un problema tecnológico. Por el contrario, se propone un cambio en la dinámica de trabajo para allornarlos a la criticidad que ha cobrado hoy en día el producto que construyen.

Es por todo esto que el proceso de ejecutar *ML* debe ser tan sencillo como el trabajo de ejecutar sistemas de TI.

2 Que hotel va primero para vos? Un ejemplo del desafio de usar Machine Learning para odernar hoteles en Despegar.com

Despegar.com se encontraba ante la incógnita de si era posible ordenar los hoteles que se mostraban a los usuarios de manera basado en la probabilidad de compra en tiempo real de manera de optimizar la conversión. La idea era generar un sistema que ante una búsqueda del usuario clasifique y ordene los resultados obteniendo algún tipo de score que pueda ser interpretado como probabilidad de compra.

2.1 El problema del Sorting

El sorting es un tipo de problema que consiste en la aplicación de algoritmos de machine learning cuyo objetivo es optimizar la ubicación de los items de los mas relevantes a los menos relevantes. Para esto, existen 3 enfoques distintos:

- ***Pointwise***: el modelo se aplica sobre cada ítem de manera individual, es decir se utilizan las características propias del ítem sin mirar a las del resto. El ordenamiento se da por el score individual de cada ítem.
- ***Pairwise***: el modelo se aplica sobre cada par de ítem, de forma que se comparan las características de cada uno a fin de obtener el mejor de ambos. El ordenamiento final es relativo a los scores de cada par.
- ***Listwise***: el modelo se aplica sobre la lista entera de items y busca optimizar alguna métrica de ordenamiento como Ganancia *Acumulada Descontada Normalizada (NDCG)* o la *Precisión Media Promedio (MAP)*.

Solo el ordenamiento *pointwise* provee scores que pueden ser interpretables como probabilidades, por lo que el proyecto utilizo este enfoque por sobre los otros.

Otro de los problemas encontrados durante la investigación sobre el sorting es el *sesgo de la posición*. En los análisis exploratorios realizados se encontró que la probabilidad de saber si un ítem es relevante para un usuario-contexto disminuye con la posición. Este sesgo complica el trabajo de entender cuanto influye las características del hotel y cuanto influye la posición en la que se muestra.

2.2 Diseno

El diseno del producto fue separado en 4 etapas: estimación del sesgo de la posición de los datos actuales, entrenamiento del modelo (feature engineering y modelado), post procesamiento de scores y evaluación offline.

2.2.1 Estimacion del sesgo

La posición de un ítem influye en la probabilidad de compra, es decir, es mas probable comprar items que aparezcan en los primeros resultados. Este sesgo debe ser medido de alguna forma para poder incorporarlo al modelo. El equipo de *Despegar.com* utilizaron el *algoritmo Expectation-Maximization* obteniendo un **ponderador** para utilizar en el modelo. Este ponderador puede ser interpretado como *la probabilidad de que el usuario haya visto el ítem.*

2.2.2 Entrenamiento del modelo

Los datos que se utilizaron para el armado del dataset consistían en características propias del contexto, del hotel, del contexto y hotel y finalmente, el target binario de si la persona compraba el hotel o no. La figura 3 muestra algunos ejemplos utilizados.

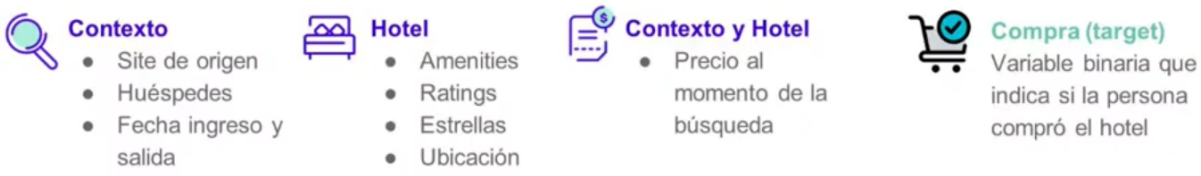

Fig. 3: Features utilizados

Algunas de estas features pueden ser obtenidas en tiempo real y otras tuvieron que pre-computarlas y dejarlas en un *feature store* para que se pudiesen consumir de manera online.

Para el modelado, se utilizo un *XGBoost por cada destino* por varias razones: por su potencia predictiva y por su portabilidad. El sistema transaccional de Despegar.com se encuentra programado en Scala, por lo que la elección de XGBoost les permitió programarlos en Python y luego ponerlos productivos en Scala.

La evaluación del modelado consitio en dos momentos: offline y online. La evaluación offline fue medida con métricas de clasificación (F-score) y métricas de probabilidad (Negative Log Loss). La online, mediante 3 KPIs definidos por el negocio: *tasa de conversión*, *margen por usuario* e *ingreso*. A su vez, el modelo fue deployado utilizando un esquema A/B para algunos de los destinos seleccionados.

2.2.3 Post procesamiento de scores

Luego del entrenamiento, los scores obtenidos fueron post procesados en función del enfoque.

En el *enfoque de clasificación*, se definió un umbral por el cual los hoteles eran clasificados en "relevantes" y "no relevantes". De esta manera se optimizaba de manera eficiente el f-score.

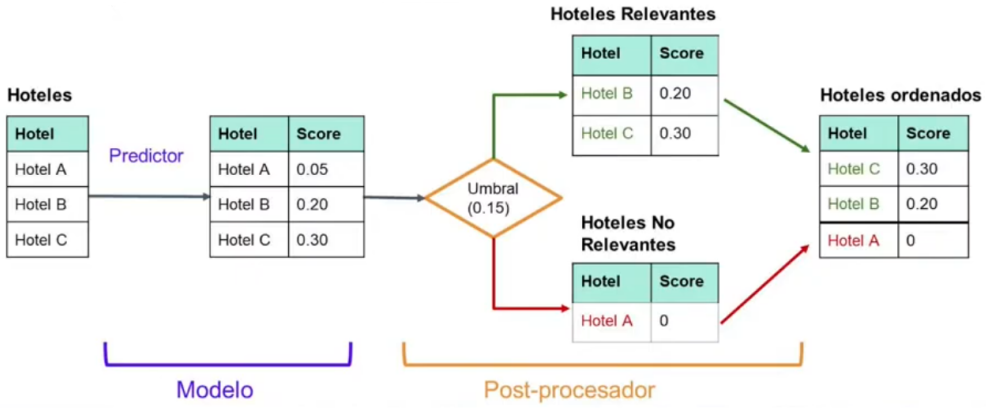

Fig. 4: Enfoque de Clasificacion

En el *enfoque de probabilidades*, se optimiza la métrica negative log loss (NLL) para medir diferencias entre el score (z) y el target de compra o no compra (y).

$$NLL = \sum_{i=-1}^{+N} y_i + \log z_i + (1 - y_i) \log(1 - z_i) \tag{1}$$

A su vez, se utilizaron *gráficos de calibración*. Estos gráficos permiten observar la relación de los scores medidos y la tasa de conversión. La idea general de esta herramienta es agrupar hoteles de manera similar y graficar las medidas. Resulto de gran importancia durante el desarrollo ya que se puede compartir con

otras áreas de Despegar.com ya que son simples de interpretar: siempre y cuando se muestre una tendencia positiva, el modelo funciona correctamente.

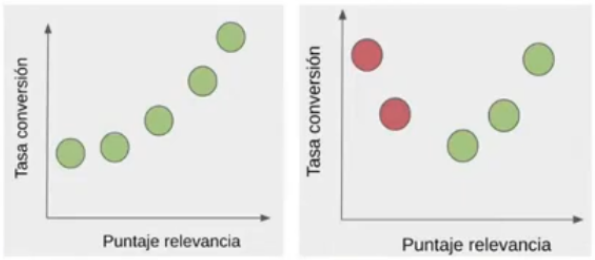

Fig. 5: Grafico de Calibración.

2.2.4 Evaluación

El equipo decidió realizar una POC con el enfoque de *clasificación* obteniendo resultados mixtos en los KPIs. En la siguiente iteración, se cambio al enfoque de *probabilidades*.

El enfoque de *probabilidades* resultó ser el mejor. La Figura 6 muestra los gráficos de calibración obtenidos por enfoque.

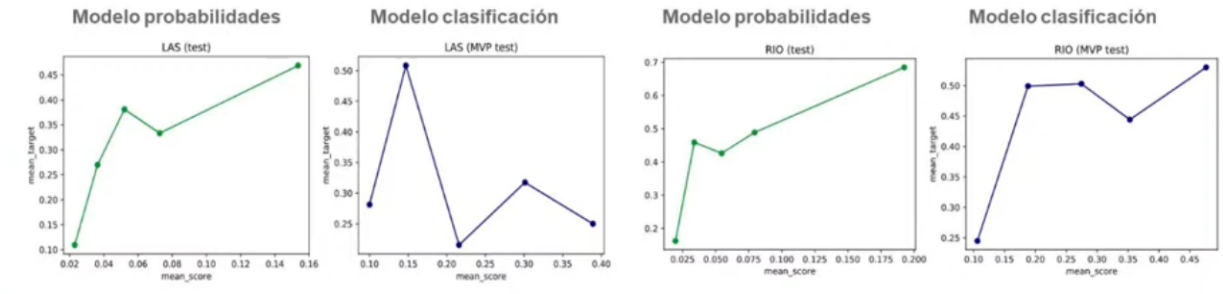

Fig. 6: Graficos de calibración obtenidos por enfoque.

En la Figura 7 se detalla el comportamiento de cada KPI en función de su enfoque.

Modelo\KPIs	Δ Mg/User	Δ GB	Δ CVR
POC	-	=	+
V1	+	+	+

Fig. 7: Resultados obtenidos por enfoque.

2.3 Conclusiones

Es interesante mencionar que la ciencia de datos puede estar en cualquier lado, incluso en ordenamiento de listas. Durante toda la charla, los chicos de Despegar.com mencionaron la importancia de analizar el estado

del arte y de hacer pruebas de concepto (POCs) para poder involucrar al negocio lo mas temprano posible.

Otro punto interesante a recalcar es que, si bien la métrica mas lógica a optimizar es el f-score (al tratarse de un problema de clasificación), no siempre es la que optimiza mejor los KPIs. No hay que perder el foco de las métricas de negocio a optimizar. Las métricas "clásicas" deben ser de soporte, pero las métricas de negocio son las que nos tienen que definir los modelos o enfoques a seguir.