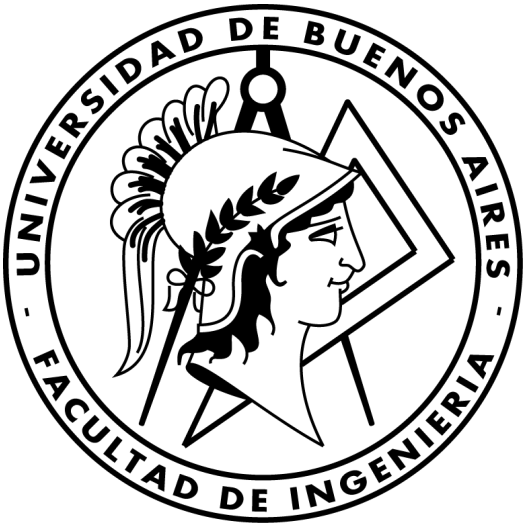
Facultad de Ingeniería
Universidad de Buenos Aires



Trabajo Práctico N°2
Organización de Datos 75.06

Grupo 41 “Datonga”
Compuesto por:

Bravo, Facundo - 100151
Kler, Alejandro - 100596

GitHub: https://github.com/facuub/Datos-TP2

Índice:

1. Introducción

1.1 Objetivo

El objetivo de este trabajo es construir un algoritmo de Machine Learning que permita determinar la probabilidad de que un usuario de Trocafone realice una conversión en un periodo de tiempo a partir de su actividad en el sitio en un periodo de tiempo anterior.

1.2 Incidencia del trabajo práctico anterior

A partir del trabajo anterior pudimos ver que el tráfico del sitio aumenta significativamente en los últimos 15 días de Mayo, lo que nos hizo centrarnos en este periodo para obtener la Mayoría de los “features” predictores.

1.3 Competencia

Se ha desarrollado una competencia entre los grupos de la materia, donde cada grupo puede hacer submits y registrar su score. Aquellos que tengan mejor score, estarán mejor posicionados. Durante este informe, se hace referencia a esta competencia mencionando a Keagle, que es la plataforma utilizada.

2. Algoritmos utilizados

2.1 Algoritmo de predicción

Desde el comienzo el algoritmo utilizado para el modelo fue XGBoost, ya que es uno de los algoritmos más efectivos en competencias de machine learning y dio buenos resultados en este problema desde el comienzo.

2.2 Optimización de hiperparametros

Utilizamos Cross-Validation para optimizar el número de árboles estimadores en XGBoost. El resto de los parámetros fueron optimizados a mano en cada cambio de features.

2.3 Función Booster

Utilizamos tanto la función “dart” como “gbtree” de XGBoost. En un principio la función “dart” dio mejores resultados pero finalmente el mayor score de kaggle fue obtenido con la función “gbtree”.

3. Features

3.1 Features que aportaron

3.1.1 Checkouts de los últimos 15 días

El primer feature que utilizamos para el modelo fue la cantidad de checkouts por usuario en los últimos 15 días de Mayo. Este feature resultó ser un buen predictor inicial ya que, siendo el único feature, la predicción del modelo nos arrojó un score de 0.81843 en kaggle.

3.1.2 Checkouts y conversiones

Después de ver el éxito del primer feature, se agregaron además los checkouts y las conversiones de los últimos 30 días, 5 días y último día de Mayo. Esto mejoró el score de kaggle logrando un 0.84728.

3.1.3 Canal de origen

Posteriormente agregamos el número de veces que entro cada usuario por cada canal en los últimos 15 días, 5 días y último día. Esto mejoró el score de kaggle hasta un 0.85415.

3.1.4 Viewed product y visited site

Análogamente, se agregaron la cantidad de visited site y viewed product de los últimos 15 días, 5 días y último día de Mayo. Esto junto con una mejora en los hiperparametros nos dejó un score de 0.86557 en Kaggle.

3.1.5 Cantidad de búsquedas y checkouts por marca

Agregamos por último la cantidad de búsquedas por usuario de los últimos 15 días de Mayo y el total de checkouts de teléfonos de teléfonos iPhone y Samsung. Dichas marcas fueron elegidas debido al alto volumen que tienen frente a las demás. Esto nos dejó un score de 0.86734 en Kaggle, que luego fue mejorado a 0.86943 usando la función “gbtree” de XGBoost.

3.2 Features que no aportaron

De las features anteriormente mencionadas, fueron eliminadas algunas que empeoraban la precisión del algoritmo, entre las cuales se encuentran:

- Checkouts y conversiones de los últimos 60 dias.
- Los ingresos por canal email de los últimos 15, 5 y último día de Mayo.
- Los ingresos por canal social de los últimos 15, 5 y último día de Mayo.
- Los ingresos por canal pago de los últimos 15, 5 y último día de Mayo.

- Los ingresos por canal directo de los últimos 15 días de Mayo.
- Los ingresos por canal orgánico de los últimos 15 y último día de Mayo.
- La cantidad de checkouts de celulares de marcas que no son ni iPhone ni Samsung.
- La cantidad de búsquedas de los últimos 30 y 5 días de Mayo.
- La cantidad de búsquedas totales.

Además de estas, ideamos una serie de features más complejas, la Mayoría de las cuales terminaron empeorando el score final del algoritmo, por lo cual no fueron incluidas en el modelo final. A continuación se enumeran dichos features:

- Cantidad de viewed product por marca.
- Origen (región) predominante de cada usuario, agregado como One Hot Encoding.
- Dummy variable si el usuario es predominantemente de Brasil

4. Conclusiones

4.1 Medición de importancia de features

Utilizamos random forests para medir la importancia de los features y de esta manera poder eliminar features innecesarios, pero notamos que la precisión del algoritmo empeoraba si quitábamos features, aunque la importancia de éstas fuera baja.

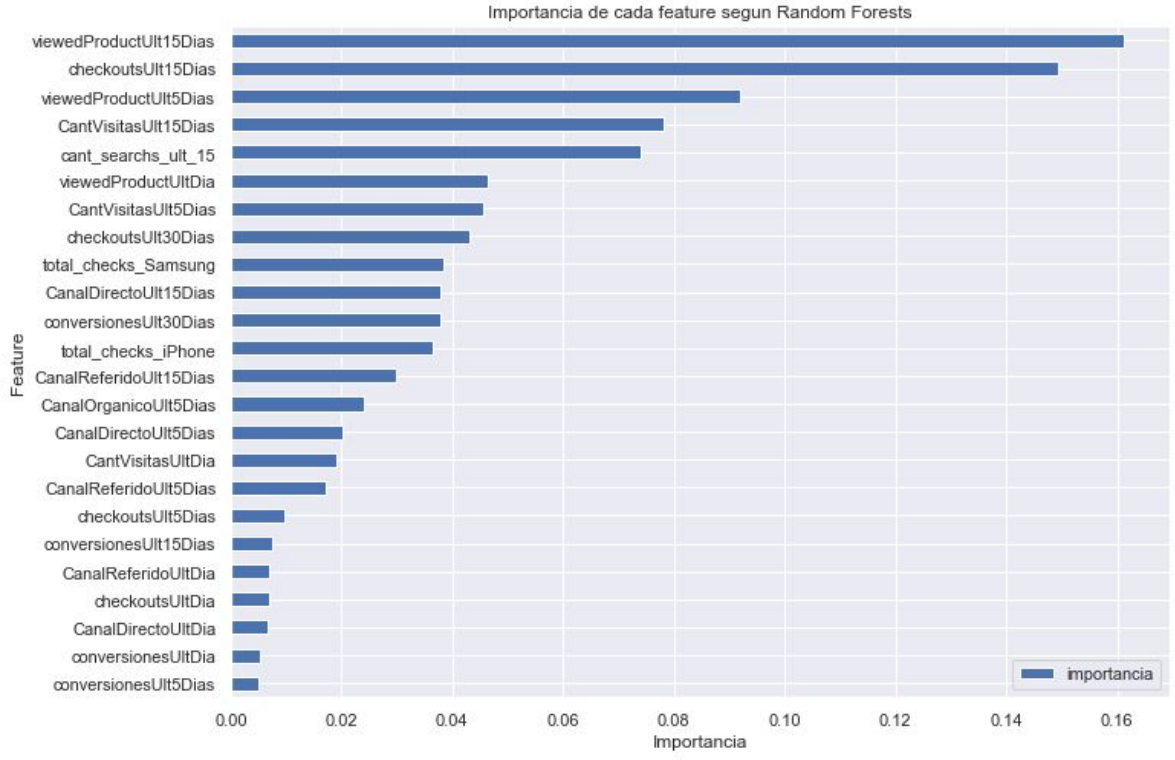


Figura 1: Importancia de cada feature según Random Forest

4.2 La importancia del tiempo

Siendo nuestro escenario un ecommerce exclusivo de celulares, podemos concluir que la compra de celulares no es algo tan repetitivo, por lo que los features basados en la actividad de los últimos días son mucho más relevantes. En nuestro caso, este fenómeno es acentuado por el hot sale producido 15 días antes.

4.3 Precisión del modelo

Notamos que se puede construir un modelo de Machine Learning de buena precisión con features relativamente simples y con los datos más recientes de actividad de cada usuario, llegando a un score de 0.86734 cuando el score de la empresa ronda los 0.88. Notamos que para aumentar la precisión más allá de este punto, hacen falta features más complejas u otro conjunto de features simples que no logramos obtener. Pasado un punto, lograr mejoras pequeñas se hace cada vez más difícil y hay que elegir cuidadosamente las nuevas features a integrar al modelo ya que la mayoría empeora la precisión del algoritmo.