Introdução

Esse artigo é a parte 2 do último estudo feito sobre a empresa alimentícia. O link para a parte 1, que contém a análise completa dos dados feita no Power BI, pode ser encontrada clicando aqui.

Nessa continuação vamos fazer um modelo de machine learning para prever se o cliente vai responder positivamente ou não em relação campanha de marketing feita pela empresa. Para isso, vamos implementar, em Python, o modelo XGBoost de machine learning.

Objetivo

O grande objetivo desse modelo é identificar com precisão os clientes que não foram atingidos pelas campanhas de marketing para que a empresa possa mudar as suas estratégias e conseguir converter esse público não atingido.

Montagem do Modelo

Figura 1 – Parte dos dados

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProdu
0	1826	1970	Graduation	Divorced	$84,835.00	0	0	6/16/14	0	189	104	379	
1	1	1961	Graduation	Single	$57,091.00	0	0	6/15/14	0	464	5	64	
2	10476	1958	Graduation	Married	$67,267.00	0	1	5/13/14	0	134	11	59	
3	1386	1967	Graduation	Together	$32,474.00	1	1	5/11/14	0	10	0	1	
4	5371	1989	Graduation	Single	$21,474.00	1	0	4/8/14	0	6	16	24	

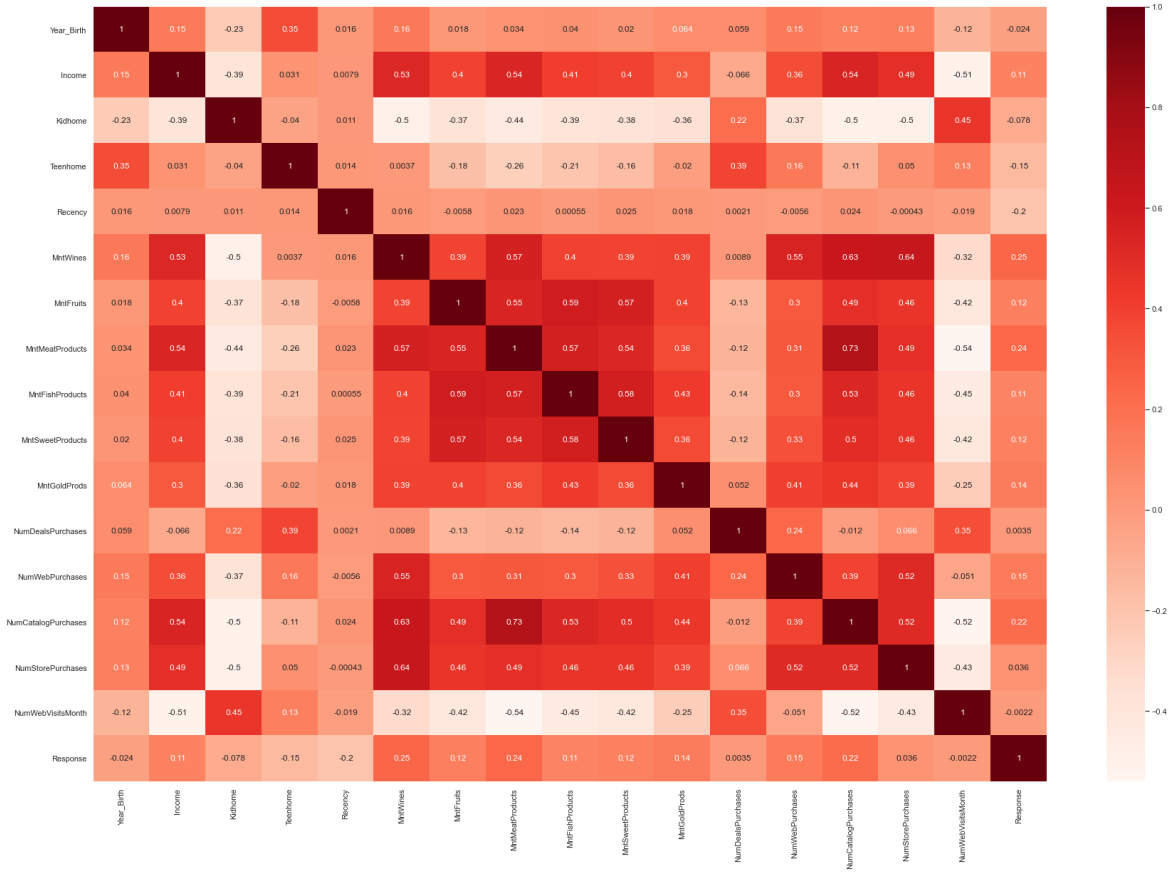
Para começar, vamos relembrar os dados, vistos na figura 1. Nela vemos que precisamos fazer uma limpeza nos dados, por exemplo, retirar algumas colunas como ‘ID’, reformatar a coluna ‘Income’ que, além de ter o símbolo ‘$’, tem o seu data type como ‘object’ e não um número inteiro ou float e, por fim, utilizar a técnica de ‘getdummies’ para transformarmos as colunas ‘Education’ e ‘Marital_Status’ que contém dados categóricos.

Figura 2 – Parte dos dados 2

:hases	NumStorePurchases	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5	AcceptedCmp1	AcceptedCmp2	Response	Complain	Country
4	6	1	0	0	0	0	0	1	0	SP
3	7	5	0	0	0	0	1	1	0	CA
2	5	2	0	0	0	0	0	0	0	US
0	2	7	0	0	0	0	0	0	0	AUS
1	2	7	1	0	0	0	0	1	0	SP

Na figura 2, podemos observar a coluna ‘Response’, que é o target do nosso modelo. Ela contém a informação se a pessoa aceitou (1) ou não (0) a campanha de marketing.

Figura 3 – Heatmap dos dados



No heatmap, figura 3, temos quanto correlacionadas as features estão do target. Vemos que ‘MntWines’, ‘MntMeatProducts’ e ‘NumCatalogPurchases’ são as mais correlacionadas positivamente a ‘Response’ (já sabíamos disso da análise da parte 1), ou seja, contribuem positivamente para o cliente para o resultado ‘1’ da ‘Response’.

Figura 4 – Dados limpos

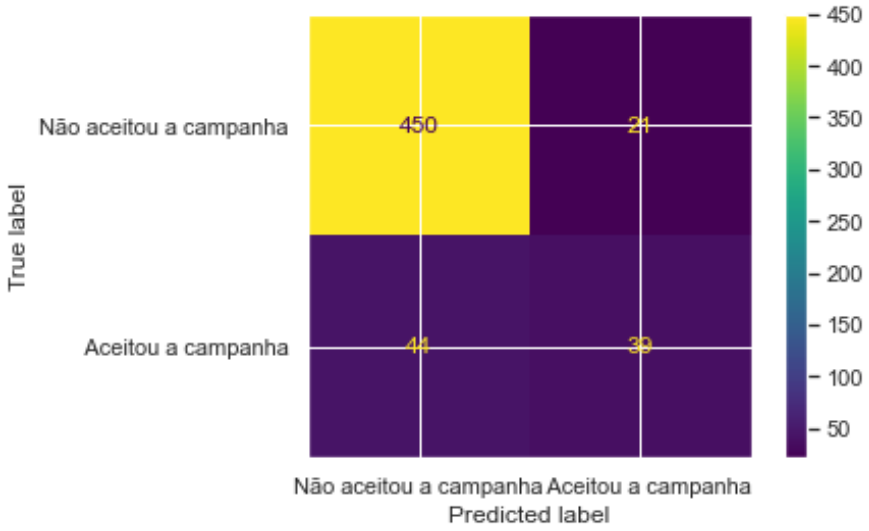
	Year_Birth	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProds	NumDealsPu
0	52	84835	0	0	0	189	104	379	111	189	218	
1	61	57091	0	0	0	464	5	64	7	0	37	
2	64	67267	0	1	0	134	11	59	15	2	30	
3	55	32474	1	1	0	10	0	1	0	0	0	
4	33	21474	1	0	0	6	16	24	11	0	34	

Com os dados limpos, visto na figura 4, podemos proceder para montar o modelo. Antes de monta-lo, vamos verificar se o dataset está balanceado. Temos que a coluna ‘Response’ possui apenas 15 % de de ‘1’, ou seja, não estamos trabalhando com um dataset balanceado e isso pode comprometer os resultados do modelo.

Resultados e Discussão

Com o modelo feito, a métrica de avaliação foi ‘aucpr’ (area under the curve precision and recall) e, depois de 46 iterações, o resultado foi de 0.61964. Todos os hyperparameters nessa primeira rodada foram deixados com os valores default. De posse dos resultados podemos plotar a confusion matrix para avaliarmos a precision e o recall.

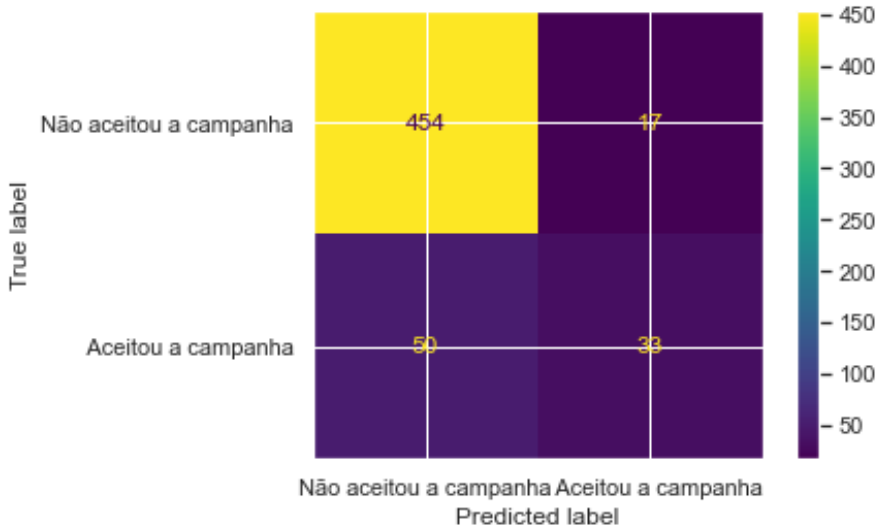
Figura 5 – Confusion matrix com hyperparameters default



Na figura 5, temos que 95,5 % dos clientes que não aceitaram a campanha (I) foram classificados corretamente, no entanto, apenas 53 % dos que aceitaram a campanha (II) foram classificados de maneira correta.

Para tentarmos melhorarmos os resultados, foi feito uma mudança nos hyperparameters, por exemplo, ‘learning_rate’, ‘max_depth’, ‘reg_lambda’ e etc.

Figura 6 – Confusion matrix com mudança nos hyperparameters



Com esses parâmetros alterados, temos o resultado do 'aucpr' de 0.59301 que, no primeiro momento, parece ser inferior, mas analisando a segunda confusion matrix, figura 6, temos um ganho na identificação dos clientes que não foram atingidos pela campanha, 96,4 %, e uma diminuição no caso dos que aceitaram a campanha, 35,9 %. Portanto, houve um pequeno ganho na precision de (I) detrimento de (II).

Na minha visão, o segundo modelo não se mostrou melhor em relação ao primeiro, pois esse ganho não foi significativo o suficiente para justificar a perda de precisão das pessoas que foram atingidas pela campanha.

Conclusões

O objetivo do modelo era identificar com precisão as pessoas que não foram atingidas pela campanha e ele o atingiu com uma precisão de 95,5 % dos casos. Esse resultado representa que futuramente será mais fácil identificar quem não será atingido pelas campanhas, avaliar os motivos e trabalhar em cima deles para poder converter o cliente.

Todo o código pode ser encontrado no meu github clicando aqui.