Inteligência Computacional — COC 361
2021/2

Trabalho Computacional

Gabriel de Oliveira da Fonseca, Gustavo Pires Machado

March 6, 2022

1 Introdução

- Descrição do problema.
- Pesquisa bibliográfica (opcional).

Para o presente trabalho, o conjunto de dados escolhido reúne dados coletados entre 1º de Julho de 2015 e 31 de Agosto de 2017 por uma rede hoteleira que incluem diversos atributos relacionados às reservas efetuadas por seus clientes. Utilizando-se das diversas metodologias de Inteligência Computacional discutidas ao longo do curso, o trabalho tem por objetivo a construção de um modelo de classifcação para a previsão de reservas canceladas. A partir deste modelo, espera-se que a rede hoteleira possa se beneficiar de uma maior previsibilidade das reservas que serão efetivamente concretizadas, aumentando por fim sua margem de lucro.

2 Dataset e Tecnologia

- Descrição dos Dados.
- Apresentação da tecnologia.

O dataset escolhido possui 36 colunas, foi retirado da plataforma Kaggle [?], e conta com 119390 entradas. De forma geral, está razoavelmente organizado e possui um baixo número de dados faltantes (nulos). Os dados presentes compreendem diversos aspectos relacionados às reservas, como número de hóspedes em diferentes faixas etárias, dados pessoais dos clientes (como nome, e-mail e telefone), além de nuances mais promissoras, como indicadores de que um determinado cliente já cancelou reservas anteriormente.

Um outro importante aspecto a ser analisado diz respeito à qual classe pertence o conjunto de dados: balanceada ou desbalanceada. Isso pode ser visto através da predominância das classes alvo, que nesse estudo são as reservas canceladas ou não. Contando cada uma das classes, chegamos à proporção de 39% das reservas canceladas para 61% não canceladas. Dessa forma, podemos concluir que o dataset é desbalanceado.

Além disso, conforme disposto na tabela abaixo, 20 de suas colunas são numéricas, e portanto 16 categóricas.

#	Coluna	# entradas não-nulas	Tipo
0	hotel	119390	object
1	is_canceled	119390	int64
2	lead_time	119390	int64
3	arrival_date_year	119390	int64
4	arrival_date_month	119390	object
5	arrival_date_week_number	119390	int64
6	arrival_date_day_of_month	119390	int64
7	stays_in_weekend_nights	119390	int64
8	stays_in_week_nights	119390	int64
9	adults	119390	int64
10	children	119386	float64
11	babies	119390	int64
12	meal	119390	object
13	country	118902	object
14	market_segment	119390	object
15	distribution_channel	119390	object
16	is_repeated_guest	119390	int64
17	previous_cancellations	119390	int64
18	previous_bookings_not_canceled	119390	int64
19	reserved_room_type	119390	object
20	assigned_room_type	119390	object
21	booking_changes	119390	int64
22	deposit_type	119390	object
23	agent	103050	float64
24	company	6797	float64
25	days_in_waiting_list	119390	int64
26	customer_type	119390	object
27	adr	119390	float64
28	required_car_parking_spaces	119390	int64
29	total_of_special_requests	119390	int64
30	reservation_status	119390	object
31	reservation_status_date	119390	object
32	name	119390	object
33	email	119390	object
34	phone-number	119390	object
35	credit_card	119390	object

Table 1: Atributos do dataset.

Já em relação às tecnologias utilizadas para a implementação da solução deste trabalho, foram adotadas essencialmente a plataforma Google Collaboratory [?], onde o trabalho foi desenvolvido no formato de um notebook Jupyter, e diversas bibiliotecas em Python. Dentre as bibliotecas utilizadas, vale citar a Pandas, que implementa abstrações para facilitar o manuseio de data frames, bem como Scikit-Learn, que disponibiliza uma vasta gama de ferramentas para

visualização de dados e treinamento de modelos. Especificamente, também foi utilizada a API Keras do TensorFlow, que foi utilizada no treinamento dos modelos de redes neurais.

3 Metodologia

- Apresentação da solução do problema proposto.
- Descrição teórica (matemática) dos modelos utilizados.

A solução construída para o problema consistiu na implementação de um ferramental de modelos de classificação baseados em diversos algoritmos apresentados ao longo da disciplina de Inteligência Computacional. Como tarefa preliminar, conforme apresentado posteriormente na seção **??**, foi realizada também uma série de ajustes de pré-processamento no dataset a fim de otimizar os modelos treinados. Nas subseções a seguir, são apresentados detalhadamente os modelos adotados para construção da solução.

3.1 Regressão Logística

A Regressão Logística é um modelo de classificação binária, cuja função discriminante é definida pela probablidade *a posteriori* como [**?**]:

$$\begin{aligned} P(v(t)) = 1|\mathbf{x}(t), \boldsymbol{\theta}) = g(\mathbf{x}(t), \boldsymbol{\theta}) \\ P(v(t)) = 0|\mathbf{x}(t), \boldsymbol{\theta}) = 1 - g(\mathbf{x}(t), \boldsymbol{\theta}) \end{aligned} \tag{3.1}$$

Tal função de probabilidade em representar os valores observados é maximizada pelo modelo através da seguinte função objetivo:

$$l(\boldsymbol{\theta}) = -\sum_{t=1}^{N} v(t) log(\hat{v}(t)) + (1 - v(t)) log(1 - \hat{v}(t)) \tag{3.2}$$

Onde,

$$\hat{v}(t) = \frac{1}{1 + e^{-\theta Tx}} \tag{3.3}$$

Pela equação (3.2), podemos observar que quando $v(t) = 0$, apenas o segundo termo do somatório prevalece e o custo tende a infinito ($log(0)$) no caso em que $\hat{v}(t) = 1$. Situação similar ocorre quando $v(t) = 1$ e $\hat{v}(t) = 0$. Sabendo que esta função objetivo determina o erro entre a classe predita e a classe observada, podemos minimizá-la utilizando métodos de PNL como o do gradiente. Dessa forma, construímos uma superfície de separação entre as classes, discriminada por uma curva sigmóide, como dado pela equação (3.3).

3.2 Classificação Bayesiana

O teorema de Bayes relaciona o conhecimento prévio do problema na forma da seguinte equação:

$$P(C_i|\mathbf{x}) = \frac{p(\mathbf{x}|C_i)P(C_i)}{p(\mathbf{x})} \tag{3.4}$$

Onde $P(C_i|\mathbf{x})$ é a probabilidade de observar a classe C_i dado que os valores das variáveis $\mathbf{x}$ são conhecidos, chamada **probabilidade *a posteriori***; $p(\mathbf{x}|C_i)$ representa a distribuição

de probabilidades das variáveis **x** quando a classe observada é C_i, chamada de distribuição de **probabilidade condicional**; e $P(C_i)$ é a probabilidade de ocorrência da classe C_i na ausência de qualquer observação, chamada **probabilidade *a priori***. Por fim, o denominador é apenas a probabilidade de observar valores das variáveis **x**, e funciona como um fator de padronização [**?**].

O modelo de Classificação Bayesiana faz uso do teorema de Bayes para decidir qual classe tem a maior probabilidade condicional, e portanto será escolhida. Para isso, as probabilidades condicionais são calculadas diretamenta pela definição quando se tratando de atributos nominais, e através da PDF do atributo aplicada em x para variáveis numéricas. Por fim, a probabilidade condicional geral para cada classe será dada pelo produtório das probabilidades condicionais de cada atributo.

3.3 Árvores de Decisão

Árvores de Decisão são modelos de aprendizado supervisionado que podem ser utilizados tanto para classificação quanto para regressão. De forma ampla, buscam definir um modelo de predição através do aprendizado de regras inferidas pelos atributos do dataset. São chamadas de árvore pois o modelo é construído como um diagrama de decisão, cuja representação é similar a uma árvore, conforme ilustrado abaixo:

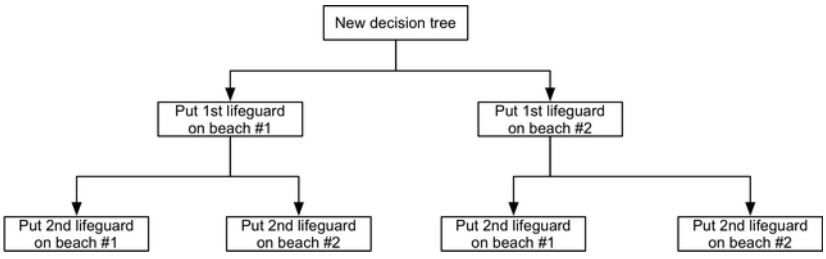

Figure 1: Exemplo de Árvore de Decisão [**?**]

A predição de uma classe é baseada essencialmente em percorrer o diagrama utilizando os valores dos atributos conhecidos como entrada para as regras inferidas pelo modelo. A acurácia do modelo está, portanto, diretamente correlacionada à inferência das regras e também ao tamanho que a árvore pode adotar, o que permite uma maior especificidade das regras inferidas.

3.4 Random Forest

Modelos do tipo Random Forest realizam a agregação de diversas Árvores de Decisão (DT, do inglês *decision tree*), já explicadas na subseção **??**, para composição de classificadores ou regressores. Para o problema específico de classificação, a saída de uma Random Forest é a classe selecionada pela maioria das DTs. Já para regressões, são retornadas as médias dos valores retornados por cada DT [**?**].

O objetivo central na construção do modelo é diminuir a variância entre as saídas das DTs e ao mesmo tempo evitar *overfitting* - que ocorre quando o modelo é específico demais para o conjunto de treinamento e apresenta baixa acurácia para conjuntos de teste. Para isso, o número de DTs utilizadas é configurável através de um hiperparâmetro, o que permite realizar um ajuste que propicie a menor variância possível para análise.

3.5 Gradient Boosting

O método de Gradient Boosting preconiza a construção de um modelo preditivo forte a partir de modelos intermediários fracos - ideia geral dos algoritmos de Boosting-, normalmente DTs. É similar, portanto, ao de Random Forest no que tange a utilização de DTs como elementos de construção do modelo preditivo final. Diferem, entretanto, na forma como os diferentes modelos intermediários (DTs) são agregados, já que no Gradient Boosting eles são sequencialmente adicionados, enquanto em Random Forest são utilizados em paralelo.

O algoritmo de Gradient Boosting consiste em realizar o treinamento de um modelo fraco inicial e aplicar à ele próprio uma função de correção a fim de melhorar a acurácia de um novo modelo a ser criado para o mesmo conjunto de dados. Nesse caso, como o nome do método sugere, a função de correção é derivada do gradiente da função em cada etapa da interação, conforme a seguinte equação:

$$F_{m+1}(x) = F_m(x) + h_m(x) \tag{3.5}$$

Onde $h_m(x)$ é a função de correção, que será o gradiente da função de avaliação utilizada para o problema. Além disso, o método possui 3 hiperparâmetros: tamanho da árvore, taxa de aprendizado e subamostra. Os dois últimos servem, respectivamente, para controlar a proporção em que a função de correção é adicionada e o particionamento da amostra original durante a construção do modelo.

3.6 SVM

O método de Support Vector Machine (SVM) consiste na utilização das chamadas funções de núcleo para manipular o espaço de características de um dado problema de forma indireta a fim de obter uma medida de similaridade entre diferentes entradas do conjunto de dados. A partir da aplicação da função de núcleo, é construída então uma matriz de núcleo, que contém uma representação do conjunto inicial com suas respectivas similaridades em relação a todos os outros registros. Esse matriz é, posteriormente, utilizada para construção de um hiperplano que separa as diferentes classes alvo.

A fim de minimizar o erro de classificação produzido pelo método, o conceito de margem de separação é adotado no SVM, e consiste em definir um limiar definido a partir do hiperplano para o qual o modelo é capaz de realizar classificações corretas. Com isso, ao maximizarmos a margem de separação, chegamos ao modelo com a maior capacidade de classficar corretamente os registros.

O modelo disponibiliza 2 hiperparâmetros, C e γ. O primeiro, C, é um parâmetro de regularização da SVM, e define a margem utilizada pelo modelo para determinar o quanto de erro é aceitável. Isso permite controlar a troca entre a fronteira de decisão e o termo de classificação errônea. Quando C estiver alto, o modelo classificará um número maior de pontos, o que também pode ocasionar perda de acurácia e overfitting. Já o hiperparâmetro γ define o quanto a distância entre os pontos influencia o cálculo dos hiperplanos. Quanto maior for Gamma, mais influência possuem pontos próximos e maior a probabilidade de overfitting. Por outro lado, quanto menor for Gamma, maior a influência de pontos mais distantes e mais generalista o modelo.

3.7 Redes Neurais

4 Resultados

4.1 Visualização e Caracterização dos dados

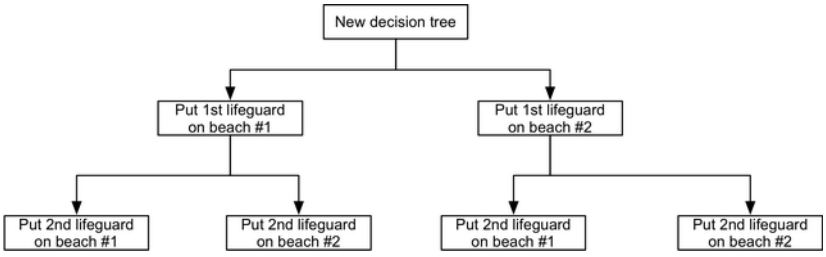

Figure 2: Exemplo de Árvore de Decisão [?]

Conforme apresentado na tabela **??**, algumas colunas como *agent* e *company* possuem entradas com valores nulos, cuja manutenção afetaria as análises efetuadas. Como a variável que define a empresa associada à reserva é nula em 94% das entradas, optou-se por removê-la. Aproveitou-se também para remover variáveis intuitivamente irrelevantes para o modelo, como nome, e-mail e telefone. Após isso, foi possível remover todas as entradas contendo valores nulos, já que representavam uma pequena parte do dataset utilizado (≈ 10%).

Como uma etapa de engenharia de atributos, optou-se por tratar algumas informações pouco relevantes para o modelo em sua forma original. Para isso, um novo atributo de localização (*guest_location*) foi criado para substituir o atributo de país, mapeando Portugal em "Local" e outros países em "Internacional", já que o número de reservas provenientes de Portugal era inicialmente muito maior que as de outros países individualmente. Além disso, as variáveis *children* e *babies* foram somadas em uma nova variável *kids*. Por final, dois novos atributos *total_guests* e *total_stays* foram criados a fim de representar, respectivamente, as somas de hóspedes (adultos, crianças e bebês) e diárias (diárias em finais de semana e dias de semana).

Após as modificações citadas acima, realizou-se o procedimento de conversão de variáveis categóricas para variáveis indicadoras. Por exemplo, a variável *meal* foi dividida em 4 novas variáveis: *meal_FB*, *meal_HB*, *meal_SC* e *meal_Undefined*, que representam as diferentes possibilidades de refeições a serem escolhidas pelos clientes.

- Visualização e Caracterização dos dados (distribuições, correlações, etc.)
- Descrição do procedimento de validação (validação cruzada)
- Resultados dos modelos lineares:
    1. Regressão Linear / Regressão Logística
    2. Classificação Bayesiana (problemas de classificação)
- Resultados dos modelos não lineares
    1. Árvores de decisão
    2. Random Forest (testar pelo menos 2 opções de hiperparâmetros)

3. Gradient Boosting (testar pelo menos 2 opções de hiperparâmetros)
4. SVM (testar pelo menos 2 opções de hiperparâmetros)
5. Redes Neurais (testar pelo menos 3 topologias/hiperparâmetros)

- Discussão e comparação dos resultados

5 Conclusões

- Discussão sobre as características do problema
- Discussão dos resultados obtidos em função das características do problema
- Recomendação sobre o melhor modelo para a aplicação
- Trabalhos futuros (opcional)

References

[1] https://www.kaggle.com/mojtaba142/hotel-booking. Acessado em 20/02/2022.

[2] https://colab.research.google.com/.

[3] Evsukoff, A. Ensinando Máquinas, 2017.

[4] M. Wagner, H. Principles of Operations Research, 1975.

[5] https://en.wikipedia.org/wiki/Random_forest.