Análise de Dados de E-Commerce para previsão de entregas de pedidos

Lucas Auada Braga[1], Vitor Kenzo Koga Onoue[1]

[1]Faculdade de Computação e Informática - Universidade Presbiteriana Mackenzie (UPM) - São Paulo – SP – Brasil

10403286@mackenzista.com.br, 10402362@mackenzista.com.br

Abstract. *This work proposes the development of a machine learning model to predict delivery delays in e-commerce orders. Using historical order data, the model can estimate whether a delivery will be on time or delayed. The proposal aims to help retailers anticipate logistical problems and improve customer experience, directly impacting on the company's reputation.*

Resumo. *Este trabalho propõe o desenvolvimento de um modelo de aprendizado de máquina para prever atrasos em entregas de pedidos de e-commerce. Utilizando dados históricos de pedidos, o modelo é capaz de estimar se uma entrega será realizada no prazo ou sofrerá atrasos. Serão utilizados os algoritmos Regressão Logística e Random Forest para treinar os dados, e métricas como acurácia, precisão, recall e F1-score. A proposta visa auxiliar lojistas a anteciparem problemas logísticos e melhorarem a experiência do cliente, com impacto direto na reputação da empresa.*

1. Introdução

A entrega pontual de produtos é um aspecto essencial para a satisfação de consumidores em plataformas de e-commerce. Com o aumento das compras online, prever possíveis falhas logísticas se tornou estratégico para empresas que desejam manter a competitividade no mercado.

A predição de atrasos pode ajudar empresas a tomarem decisões proativas, como ajustar o método de envio ou informar o cliente com antecedência. Isso reduz impactos negativos na reputação e melhora o relacionamento com o consumidor.

O objetivo deste projeto é desenvolver um modelo preditivo, utilizando técnicas de aprendizado de máquina, capaz de prever se um pedido será entregue dentro do prazo ou sofrerá atrasos com base em dados históricos públicos.

O projeto selecionado pertence à área de Inteligência Artificial e tem como foco o uso de modelos supervisionados para resolver um problema de predição no contexto de logística em e-commerce.

2. Descrição do problema

O desafio consiste em prever, no momento da realização de um pedido, se a entrega ocorrerá no prazo ou não. Essa previsão deve considerar variáveis como localização do

cliente, tipo de produto, prazo estimado de entrega, entre outros. O modelo deve aprender com padrões anteriores para oferecer previsões precisas.

3. Aspectos éticos e responsabilidade

- Os dados são públicos e anonimizados (Kaggle - Olist Dataset).
- Transparência e interpretabilidade foram priorizadas com o uso de modelos explicáveis.
- Class balance foi aplicado para mitigar viés.
- O modelo não é usado para decisões críticas sem validação humana.

4. Descrição do conjunto de dados

O conjunto de dados utilizado neste projeto foi obtido na plataforma Kaggle, uma comunidade voltada à ciência de dados que disponibiliza diversos datasets públicos. O dataset escolhido é composto por informações anonimizadas de pedidos realizados na loja Olist, uma empresa brasileira que atua no setor de e-commerce.

Os dados estão organizados em múltiplos arquivos no formato .csv, interligados por identificadores únicos, como order_id e customer_id. Para o escopo deste projeto, foram utilizados apenas alguns arquivos e campos relevantes, incluindo:

- Data de compra (order_purchase_timestamp)
- Data estimada de entrega (order_estimated_delivery_date)
- Data real de entrega (order_delivered_customer_date)
- Estado do cliente (customer_state)
- Valor total do pedido (order_price)
- Valor total do frete (freight_value)

A preparação dos dados e a análise exploratória estão documentadas no repositório do projeto.

Etapas de preparação dos dados

4.1. Arquivo orders

- Foram removidas as linhas com valor nulo na coluna order_delivered_customer_date, pois representam pedidos não entregues, o que inviabiliza a análise de atraso.
- As colunas de data foram convertidas do tipo string para datetime.

- Foi criada a coluna estimated_delivery_days, calculando a diferença entre a data estimada de entrega e a data da compra.
- A coluna purchase_month foi criada para representar o mês da compra, com a hipótese de que sazonalidade pode influenciar o prazo de entrega.
- A variável alvo delay foi criada a partir da comparação entre a data real e a estimada de entrega: True se houve atraso, False caso contrário.

4.2. Arquivo customers

- Foram mantidas apenas as linhas com o campo customer_state preenchido, pois esta é uma das variáveis utilizadas como entrada no modelo.

4.3. Arquivo order_items

- Linhas com valores ausentes em freight_value foram removidas.
- Como cada linha representa um item individual de um pedido, foi necessário agregar os dados por order_id, somando os valores de frete e de produtos, e contando o número de itens no pedido.

Integração dos dados

Após o pré-processamento individual de cada arquivo, os dados foram integrados por meio dos campos de identificação únicos. O dataset final utilizado no treinamento do modelo contém as seguintes colunas:

- order_purchase_timestamp
- order_estimated_delivery_date
- order_delivered_customer_date
- customer_state
- order_price
- freight_value
- estimated_delivery_days
- purchase_month
- delay (variável alvo)

Essa estrutura permitiu a construção de um dataset consolidado, limpo e adequado para o treinamento dos modelos preditivos.

PRINTS ANALISE EXPLORATORIA:

Starting Exploratory Data Analysis

```
import matplotlib.pyplot as plt
import seaborn as sns

dataset = pd.read_csv('csvs/clean_dataset.csv')
```

✓ 3.2s Python

```
# First five rows
dataset.head()
```

✓ 0.0s Python

	order_purchase_timestamp	order_estimated_delivery_date	order_delivered_customer_date	customer_state	order_price	freight_value	estimated_delivery_days	purchase_month	delay
0	2017-10-02 10:56:33	2017-10-18	2017-10-10 21:25:13	SP	29.99	8.72	15	10	False
1	2018-07-24 20:41:37	2018-08-13	2018-08-07 15:27:45	BA	118.70	22.76	19	7	False
2	2018-08-08 08:38:49	2018-09-04	2018-08-17 18:06:29	GO	159.90	19.22	26	8	False
3	2017-11-18 19:28:06	2017-12-15	2017-12-02 00:28:42	RN	45.00	27.20	26	11	False
4	2018-02-13 21:18:39	2018-02-26	2018-02-16 18:17:02	SP	19.90	8.72	12	2	False

```
# General info
print(dataset.info())
```

✓ 0.0s Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 96476 entries, 0 to 96475
Data columns (total 9 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   order_purchase_timestamp       96476 non-null  object
 1   order_estimated_delivery_date  96476 non-null  object
 2   order_delivered_customer_date  96476 non-null  object
 3   customer_state                 96476 non-null  object
 4   order_price                    96476 non-null  float64
 5   freight_value                  96476 non-null  float64
 6   estimated_delivery_days        96476 non-null  int64
 7   purchase_month                 96476 non-null  int64
 8   delay                          96476 non-null  bool
dtypes: bool(1), float64(2), int64(2), object(4)
memory usage: 6.0+ MB
None
```

```
# Checking null values - as they were removed previously, expected to be 0
print(dataset.isnull().sum())
```

✓ 0.0s Python

```
order_purchase_timestamp         0
order_estimated_delivery_date    0
order_delivered_customer_date    0
customer_state                   0
order_price                      0
freight_value                    0
estimated_delivery_days          0
purchase_month                   0
delay                            0
dtype: int64
```

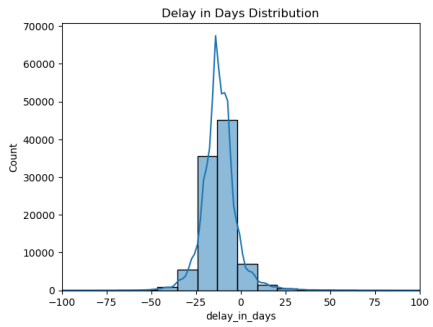
```
# Calculating delay
dataset['order_purchase_timestamp'] = pd.to_datetime(dataset['order_purchase_timestamp'])
dataset['order_estimated_delivery_date'] = pd.to_datetime(dataset['order_estimated_delivery_date'])
dataset['order_delivered_customer_date'] = pd.to_datetime(dataset['order_delivered_customer_date'])

dataset['delay_in_days'] = (dataset['order_delivered_customer_date'] - dataset['order_estimated_delivery_date']).dt.days
dataset['delay_in_days'].sample(5)
```

✓ 0.0s Python

```
82685   -19
95727   -17
4562    -25
46505   -10
650       1
Name: delay_in_days, dtype: int64
```

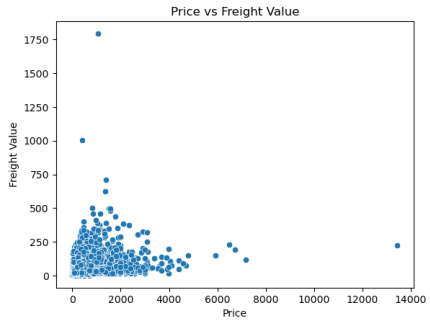
```
# Delay histogram
sns.histplot(dataset['delay_in_days'].dropna(), bins=30, kde=True)
plt.title('Delay in Days Distribution')
plt.xlim(-100, 100)
plt.show()
```

✓ 1.0s Python



```python
# Price vs Freight value scatter
sns.scatterplot(data=dataset, x='order_price', y='freight_value')
plt.title('Price vs Freight Value')
plt.xlabel('Price')
plt.ylabel('Freight Value')
plt.show()
```

0.3s Python



Exemplo de Registros no Dataset

A seguir, apresentamos dois exemplos de registros presentes no dataset final, já com os dados tratados e prontos para treinamento do modelo:

order_purchase_timestamp	order_estimated_delivery_date	order_delivered_customer_date	customer_state	order_price	freight_value	estimated_delivery_days	purchase_month	delay
2017-10-02 10:56:33	2017-10-18	2017-10-10 21:25:13	SP	29.99	8.72	15	10	False
2017-09-18 14:31:30	2017-09-28	2017-10-09 22:23:46	SP	109.90	8.96	9	9	True

5. Metodologia e resultados esperados

A abordagem deste projeto consiste na aplicação de técnicas de aprendizado de máquina para resolver um problema de classificação binária: prever se um pedido será entregue dentro do prazo ou com atraso. A predição é realizada com base em informações extraídas do pedido (data de compra e entrega), dados do cliente (estado), valores monetários (pedido e frete), além de variáveis derivadas como dias estimados para entrega e mês da compra.

O processo de desenvolvimento foi dividido em quatro etapas principais:

5.1 Coleta e Seleção de Dados

O conjunto de dados foi obtido na plataforma Kaggle e é composto por múltiplos arquivos .csv. Os dados foram integrados em um único dataset consolidado, contendo apenas as colunas relevantes para o problema proposto. Essa integração envolveu junções com base em identificadores únicos e seleção criteriosa de atributos preditivos.

5.2 Limpeza e Preparação dos Dados

Durante essa etapa, foram removidos registros com valores nulos em campos essenciais, como datas de entrega e estado do cliente. Também foram criadas novas variáveis, incluindo:

- estimated_delivery_days: número de dias previstos para entrega do pedido;
- purchase_month: mês em que o pedido foi realizado;
- delay: variável alvo binária indicando se houve atraso na entrega (True) ou não (False).

Essas variáveis foram tratadas e normalizadas, e variáveis categóricas como o estado do cliente foram codificadas com one-hot encoding.

5.3 Treinamento dos Modelos

Foram utilizados dois algoritmos de classificação supervisionada: **Regressão Logística** e **Random Forest**, implementados com o framework scikit-learn em Python. Ambos os modelos foram treinados com os dados preparados e balanceados, utilizando o parâmetro class_weight='balanced' para lidar com o desbalanceamento entre as classes (já que a maioria dos pedidos não apresenta atraso).

5.4 Avaliação dos Resultados

Após o treinamento, os modelos foram avaliados por meio das seguintes métricas:

- **Acurácia**: proporção de acertos globais;
- **Precisão**: proporção de previsões positivas corretas;
- **Recall**: capacidade de identificar corretamente os atrasos;
- **F1-score**: média harmônica entre precisão e recall.

A expectativa é que o modelo desenvolvido seja capaz de prever com eficácia se um pedido será entregue no prazo, além de fornecer insights sobre as variáveis que mais influenciam nos atrasos logísticos.

6. Resultados

Modelo	Acurácia	Precisão	Recall	F1-score
Regressão Logística	0.64	Sem atraso: 0.95 Com atraso: 0.13	Sem atraso: 0.64 Com atraso: 0.63	Sem atraso: 0.77 Com atraso: 0.22
Random Forest	0.91	Sem atraso: 0.92 Com atraso: 0.32	Sem atraso: 0.98 Com atraso: 0.10	Sem atraso: 0.95 Com atraso: 0.15

Matrizes de confusão:

Regressão Logística

VN: 17039	FP: 9538
FN: 878	VP: 1488

Random Forest

VN: 26068	FP: 509
FN: 2127	VP: 239

Modelo com melhor desempenho: Regressão Logística

7. Conclusão

Neste projeto, desenvolvemos e avaliamos modelos de aprendizado de máquina com o objetivo de prever se pedidos de e-commerce serão entregues dentro do prazo ou com atraso. Utilizando um conjunto de dados reais da plataforma Olist, realizamos o pré-processamento dos dados, extração de variáveis relevantes e treinamento de dois algoritmos principais: Regressão Logística e Random Forest.

Os resultados demonstraram o impacto do desbalanceamento das classes (pouco menos de 10% de atrasos), o que exigiu a aplicação de técnicas de balanceamento (class_weight='balanced') para que os modelos pudessem aprender a identificar corretamente os atrasos. A Regressão Logística obteve um recall de 63% para o atraso, sendo capaz de identificar uma quantidade razoável dos pedidos problemáticos, ainda que com uma precisão baixa (13%). Já o modelo de Random Forest teve um desempenho

ruim. Apesar da acurácia alta, o modelo foi capaz de prever muitas entregas no prazo, mas encontrou poucos atrasos. Tendo em vista que o problema é encontrar os pedidos que demoraram mais do que devia, o modelo não teve boa performance.

8. Endereços Github e YouTube

https://github.com/VitorOnoue/ai_college

https://www.youtube.com/watch?v=ZXl7D-cRYqE

9. Referências

Chawla, N. V., et al. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly.

Kaggle - Olist E-commerce Dataset. Disponível em: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce.