*SmartFly*: Otimização de Preços de Voos com *Machine Learning*


**Unidade Curricular de *Big Data***

**Mestrado em Análise de Dados e Sistemas de Apoio à Decisão**

**Ano Letivo 2024 – 2025**

# ***SmartFly*: Otimização de Preços de Voos com *Machine Learning***

**Coimbra *Business School* | ISCAC**

**Coimbra, Portugal**

Autores:

**Bernardo Silva – 2020112296**

**Nuno Gonçalves – 2015063961**

**Simão Dias – 2020132169**

Elaborado em

**01/03/2025**

# *SmartFly*: Otimização de Preços de Voos com *Machine Learning*

## Índice

# Índice de Figuras

# Lista de Siglas e Acrónimos

ISCAC – Instituto Superior de Contabilidade e Administração de Coimbra

KDE – *Kernel Density Estimation* (Estimativa de Densidade do Núcleo)

MAE – *Mean Absolute Error* (Erro Médio Absoluto)

MADSAD – Mestrado em Análise de Dados e Sistemas de Apoio à Decisão

ML – *Machine Learning* (Aprendizagem de Máquina)

$R^2$ – Coeficiente de Determinação

RMSE – *Root Mean Squared* Error (Erro Quadrático Médio)

# *SmartFly*: Otimização de Preços de Voos com *Machine Learning*

## 1. Introdução

A análise de grandes volumes de dados tornou-se essencial para a otimização de processos e tomada de decisões em diversas indústrias, incluindo a aviação comercial. O presente projeto insere-se no âmbito do Mestrado em Análise de Dados e Sistemas de Apoio à Decisão (MADSAD) e tem como principal objetivo a aplicação de técnicas de *Big Data* e *Machine Learning* para prever preços de passagens aéreas, onde serão utilizados o *Apache Spark* e a biblioteca *MLlib*. A previsão precisa de preços de voos pode beneficiar tanto os consumidores, permitindo a compra de bilhetes a preços mais baixos, quanto as companhias aéreas, ao otimizar estratégias de precificação.

O *dataset* utilizado neste estudo foi obtido a partir do site *EaseMyTrip*, uma plataforma de reservas de voos. A base de dados contém 300261 registos de opções de reserva, que abrange informações como companhia aérea, cidade de origem e destino, horário de partida e chegada, número de escalas, classe do bilhete (Económica ou Executiva), duração do voo e número de dias restantes até à data da viagem. A variável alvo do estudo é o preço do bilhete, que será estimado com recurso a técnicas de regressão.

### 1.1. Entendimento do Tema

A previsão de preços de passagens aéreas é um desafio complexo devido à grande variabilidade nos fatores que influenciam o valor final do bilhete. O preço de um voo pode ser afetado por múltiplos fatores, incluindo:

- **Companhia aérea:** Diferentes operadoras possuem estratégias de precificação distintas.
- **Número de dias antes da partida:** Bilhetes comprados com antecedência geralmente são mais baratos, mas existem exceções.
- **Horário de partida e chegada:** Voos em horários mais procurados tendem a ter preços mais elevados.
- **Número de escalas:** Voos diretos costumam ser mais caros do que aqueles com escalas.
- **Classe de passagem:** Bilhetes em classe executiva são significativamente mais caros do que os de classe económica.
- **Duração do voo:** Voos mais longos podem implicar custos mais elevados, isso depende das políticas de cada companhia aérea.

# *SmartFly*: Otimização de Preços de Voos com *Machine Learning*

O presente estudo foca-se na aplicação de algoritmos de *Machine Learning* para identificar padrões nos dados e criar um modelo preditivo que permita prever o preço dos bilhetes de avião com elevada precisão. A abordagem segue um *pipeline* de ***Big Data Analytics***, desde a recolha e transformação dos dados até ao treino e avaliação de modelos de regressão, onde será utilizado *Spark MLlib.*

## 1.2. Definição dos Objetivos

Este projeto tem como principais objetivos:

- **Exploração e análise dos dados:**
  - Identificar padrões e tendências na precificação de voos;
  - Verificar a distribuição e correlação entre as variáveis;
  - Aplicar técnicas de pré-processamento para lidar com dados categóricos e numéricos.
- **Desenvolvimento de modelos preditivos:**
  - Implementar modelos de **regressão** (exemplo: **Regressão Linear,** ***Random Forest Regressor*** e ***Gradient Boosting***) para prever o preço de voos.
  - Comparar o desempenho dos modelos ao utilizarmos métricas como **Erro Quadrático Médio (RMSE)** e **Coeficiente de Determinação ($R^2$)**.
- **Aplicação de técnicas de *Big Data*:**
  - Processar os dados em larga escala ao utilizarmos ***Apache Spark*** e ***Spark MLlib***;
  - Demonstrar a viabilidade da utilização de tecnologias distribuídas na análise preditiva de preços de voos.
- **Responder a questões-chave:**
  - Como é que o preço varia entre diferentes companhias aéreas?
  - Qual o impacto da antecedência na compra do bilhete?
  - De que forma os horários de partida e chegada influenciam o preço?
  - Como é que a classe do bilhete (Económica vs. Executiva) afeta a precificação?

O projeto permitirá explorar o potencial da **análise preditiva aplicada à aviação**, oferecendo *insights* que podem beneficiar tanto os consumidores como as empresas do setor.

## 1.3. Produzir o Plano do Projeto

# *SmartFly*: Otimização de Preços de Voos com *Machine Learning*

O presente projeto será desenvolvido segundo uma abordagem estruturada, baseada nos princípios de *Big Data Analytics* e *Machine Learning*. A implementação será realizada com a utilização do ***Apache Spark*** e ***Spark MLlib***, isso garante eficiência no processamento de grandes volumes de dados. A estrutura do projeto segue três fases principais, alinhadas com os requisitos do enunciado: **Exploração e Transformação dos Dados, Modelagem Preditiva e Avaliação de Resultados.**

**Fases do projeto**

**Fase 1: Exploração e Transformação dos Dados**

Objetivo: Preparar os dados para a modelagem, garantir a integridade e qualidade dos dados.

- Carregar o *dataset* no ***Apache Spark;***
- Verificar a estrutura, tipos de dados e consistência das variáveis;
- Estatísticas descritivas para identificar padrões e distribuição das variáveis;
- Visualização de relações entre as variáveis (exemplo: distribuição dos preços por companhia aérea, impacto do número de dias antes do voo, etc.);
- Análise de *outliers* e identificação de valores extremos;
- Tratamento de valores ausentes;
- Conversão de variáveis categóricas para numéricas;
- Normalização e padronização de variáveis numéricas, como a duração do voo e o preço;
- Separação dos dados em conjunto de **treino (80%) e teste (20%)** para garantir uma avaliação justa de modelos.

**Fase 2: Modelagem Preditiva**

Objetivo: Desenvolver modelos de *Machine Learning* para prever o preço das passagens aéreas.

- Implementação de pelo menos **dois modelos** do *Spark MLlib;*
- Utilização do conjunto de treino para ajustar os modelos aos dados;
- Ajuste de **hiperparâmetros** para otimização dos resultados;
- Comparação do desempenho dos modelos ao utilizar métricas (**RMSE; MAE; $R^2$)**;
- Análise dos resíduos e identificação de padrões nos erros.

**Fase 3: Avaliação e Conclusões**

Objetivo: Avaliar o impacto do modelo e sugerir melhorias.

- Identificar os **fatores mais relevantes** na previsão dos preços (exemplo: qual a variável que tem maior impacto no preço);
- Avaliação da capacidade do modelo em responder às questões de pesquisa definidas;

- Análise de possíveis melhorias nos modelos;
- Exploração de técnicas adicionais para aumentar a precisão preditiva;
- Resumo dos principais ***insights* obtidos** com o estudo;
- Sugestões para aplicações práticas do modelo no setor da aviação.

Este plano de projeto estabelece uma estrutura clara e eficiente para a implementação de um modelo preditivo de preços de voos. A abordagem vai seguir uma sequência lógica, desde a exploração dos dados até à avaliação dos resultados, o que garante que todas as etapas necessárias para uma análise preditiva robusta sejam devidamente cumpridas.

Através deste estudo, espera-se obter ***insights* valiosos sobre a precificação de passagens aéreas,** contribuindo assim tanto para a tomada de decisão dos consumidores quanto para a otimização das estratégias de precificação por parte das companhias aéreas.

# 2. Desenvolvimento

## 2.1. Carregamento e Inspeção Inicial

Os dados foram carregados com o ***Apache Spark*,** o que nos garante uma alta eficiência no processamento de grandes volumes de informação. Foi realizada uma visualização inicial do *dataset,* onde se constatou que continha **12 colunas e 300 153 registos.**

| | _c0 | airline | flight | source_city | departure_time | stops | arrival_time | destination_city | class | duration | days_left | price |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | SpiceJet | SG-8709 | Delhi | Evening | zero | Night | Mumbai | Economy | 2.17 | 1 | 5953 |
| **1** | 1 | SpiceJet | SG-8157 | Delhi | Early_Morning | zero | Morning | Mumbai | Economy | 2.33 | 1 | 5953 |
| **2** | 2 | AirAsia | I5-764 | Delhi | Early_Morning | zero | Early_Morning | Mumbai | Economy | 2.17 | 1 | 5956 |
| **3** | 3 | Vistara | UK-995 | Delhi | Morning | zero | Afternoon | Mumbai | Economy | 2.25 | 1 | 5955 |
| **4** | 4 | Vistara | UK-963 | Delhi | Morning | zero | Morning | Mumbai | Economy | 2.33 | 1 | 5955 |

```
root
 |-- _c0: integer (nullable = true)
 |-- airline: string (nullable = true)
 |-- flight: string (nullable = true)
 |-- source_city: string (nullable = true)
 |-- departure_time: string (nullable = true)
 |-- stops: string (nullable = true)
 |-- arrival_time: string (nullable = true)
 |-- destination_city: string (nullable = true)
 |-- class: string (nullable = true)
 |-- duration: double (nullable = true)
 |-- days_left: integer (nullable = true)
 |-- price: integer (nullable = true)

Total de linhas: 300153 e colunas: 12
+---+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
|_c0| airline| flight|source_city|departure_time|stops| arrival_time|destination_city|  class|duration|days_left|price|
+---+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
|  0|SpiceJet|SG-8709|      Delhi|       Evening| zero|       Night|          Mumbai|Economy|    2.17|        1| 5953|
|  1|SpiceJet|SG-8157|      Delhi| Early_Morning| zero|     Morning|          Mumbai|Economy|    2.33|        1| 5953|
|  2| AirAsia| I5-764|      Delhi| Early_Morning| zero|Early_Morning|          Mumbai|Economy|    2.17|        1| 5956|
|  3| Vistara| UK-995|      Delhi|       Morning| zero|   Afternoon|          Mumbai|Economy|    2.25|        1| 5955|
|  4| Vistara| UK-963|      Delhi|       Morning| zero|     Morning|          Mumbai|Economy|    2.33|        1| 5955|
+---+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
only showing top 5 rows


📌 **Colunas e Tipos de Dados:**
Coluna: _c0 | Tipo de dado: int
Coluna: airline | Tipo de dado: string
Coluna: flight | Tipo de dado: string
Coluna: source_city | Tipo de dado: string
Coluna: departure_time | Tipo de dado: string
Coluna: stops | Tipo de dado: string
Coluna: arrival_time | Tipo de dado: string
Coluna: destination_city | Tipo de dado: string
Coluna: class | Tipo de dado: string
Coluna: duration | Tipo de dado: double
Coluna: days_left | Tipo de dado: int
Coluna: price | Tipo de dado: int
```

*Figura 1: Visualização inicial do dataset*

Esta estrutura permitiu uma compreensão detalhada dos atributos presentes, incluindo tanto variáveis **categóricas** (companhia aérea, cidade de origem e destino, classe do bilhete, horários de partida e chegada) quanto **numéricas** (duração do voo, dias até a data da viagem e preço).

## 2.2. Verificação de Valores Nulos e Duplicados

```
from pyspark.sql.functions import col, when, count

# Função para verificar valores nulos em todas as colunas
def missing_values(df):
    return df.select([
        count(when(col(c).isNull(), c)).alias(c) for c in df.columns
    ])

# Aplicar a função no DataFrame
df_missing = missing_values(df)

# Mostrar os valores nulos por coluna
df_missing.show()
```

```
+---+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
|_c0|airline|flight|source_city|departure_time|stops|arrival_time|destination_city|class|duration|days_left|price|
+---+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
|  0|      0|     0|          0|             0|    0|           0|               0|    0|       0|        0|    0|
+---+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
```

```
from pyspark.sql.functions import countDistinct

# 4. Verificar duplicatas
num_total = df.count()
num_unique = df.dropDuplicates().count()
num_duplicates = num_total - num_unique

print(f"Total de linhas duplicadas: {num_duplicates}")
```

```
Total de linhas duplicadas: 0
```

```
# 5. Contagem de valores únicos por coluna
df_unique_values = df.select([countDistinct(col(c)).alias(c) for c in df.columns])

# Mostrar os resultados
df_unique_values.show()
```

```
+------+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
|   _c0|airline|flight|source_city|departure_time|stops|arrival_time|destination_city|class|duration|days_left|price|
+------+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
|300153|      6|  1561|          6|             6|    3|           6|               6|    2|     476|       49|12157|
+------+-------+------+-----------+--------------+-----+------------+----------------+-----+--------+---------+-----+
```

```
# Exibir colunas e seus tipos de dados no PySpark
for col_name, data_type in df.dtypes:
    print(f"Coluna: {col_name} | Tipo de dado: {data_type}")
```

```
Coluna: _c0 | Tipo de dado: int
Coluna: airline | Tipo de dado: string
Coluna: flight | Tipo de dado: string
Coluna: source_city | Tipo de dado: string
Coluna: departure_time | Tipo de dado: string
Coluna: stops | Tipo de dado: string
Coluna: arrival_time | Tipo de dado: string
Coluna: destination_city | Tipo de dado: string
Coluna: class | Tipo de dado: string
Coluna: duration | Tipo de dado: double
Coluna: days_left | Tipo de dado: int
Coluna: price | Tipo de dado: int
```

*Figura 2: Inspeção valores nulos e registos duplicados*

A inspeção dos dados revelou que **não existem valores nulos** em nenhuma das colunas, sendo assim, é eliminada a necessidade de imputação ou remoção de registos. Além disso, foi constatado que **não existiam registos duplicados**, que nos garantiu a integridade do conjunto de dados para análise posterior.

### 2.3. Análise da Estrutura do Dataset

Foram identificados os diferentes valores únicos em cada variável. O *dataset* continha:

- **6 companhias aéreas distintas;**
- **6 cidades de origem e destino;**
- **3 categorias de escalas;**
- **2 classes de bilhetes (*Economy* e *Business*)**.

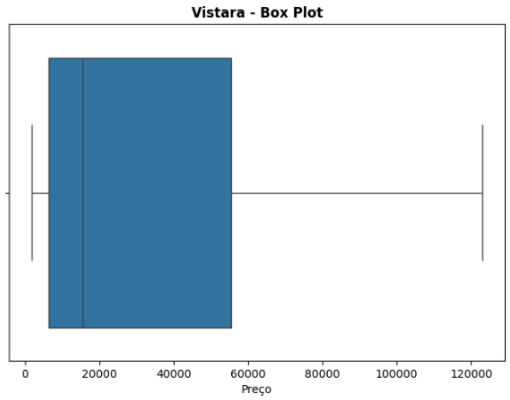
```
root
 |-- airline: string (nullable = true)
 |-- flight: string (nullable = true)
 |-- source_city: string (nullable = true)
 |-- departure_time: string (nullable = true)
 |-- stops: string (nullable = true)
 |-- arrival_time: string (nullable = true)
 |-- destination_city: string (nullable = true)
 |-- class: string (nullable = true)
 |-- duration: double (nullable = true)
 |-- days_left: integer (nullable = true)
 |-- price: integer (nullable = true)

Total de linhas: 300153 e colunas: 11
+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
| airline| flight|source_city|departure_time|stops| arrival_time|destination_city|  class|duration|days_left|price|
+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
|SpiceJet|SG-8709|      Delhi|       Evening| zero|       Night|          Mumbai|Economy|    2.17|        1| 5953|
|SpiceJet|SG-8157|      Delhi| Early_Morning| zero|     Morning|          Mumbai|Economy|    2.33|        1| 5953|
| AirAsia| I5-764|      Delhi| Early_Morning| zero|Early_Morning|          Mumbai|Economy|    2.17|        1| 5956|
| Vistara| UK-995|      Delhi|       Morning| zero|   Afternoon|          Mumbai|Economy|    2.25|        1| 5955|
| Vistara| UK-963|      Delhi|       Morning| zero|     Morning|          Mumbai|Economy|    2.33|        1| 5955|
+--------+-------+-----------+--------------+-----+------------+----------------+-------+--------+---------+-----+
only showing top 5 rows


📌 **Colunas e Tipos de Dados:**
Coluna: airline | Tipo de dado: string
Coluna: flight | Tipo de dado: string
Coluna: source_city | Tipo de dado: string
Coluna: departure_time | Tipo de dado: string
Coluna: stops | Tipo de dado: string
Coluna: arrival_time | Tipo de dado: string
Coluna: destination_city | Tipo de dado: string
Coluna: class | Tipo de dado: string
Coluna: duration | Tipo de dado: double
Coluna: days_left | Tipo de dado: int
Coluna: price | Tipo de dado: int
```

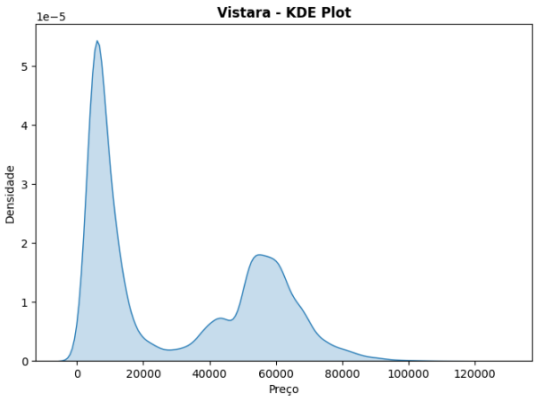
*Figura 3: Estrutura do dataset*

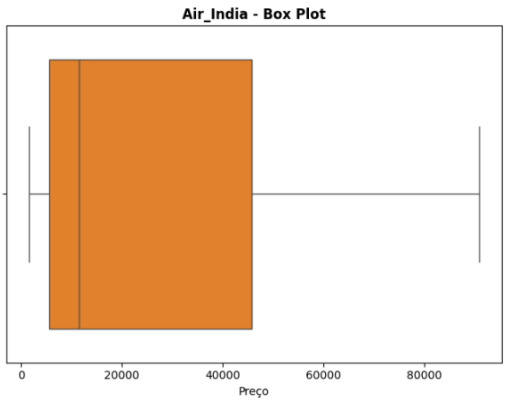
Além disso, foi identificada e removida a coluna "***_co***", que não possuía relevância para a análise.

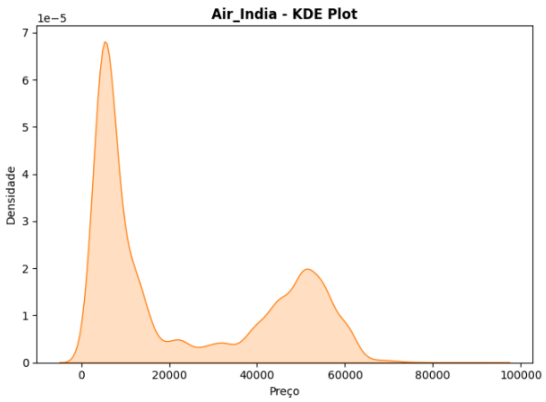
# 3. Análise da Distribuição de Preços das Companhias Aéreas

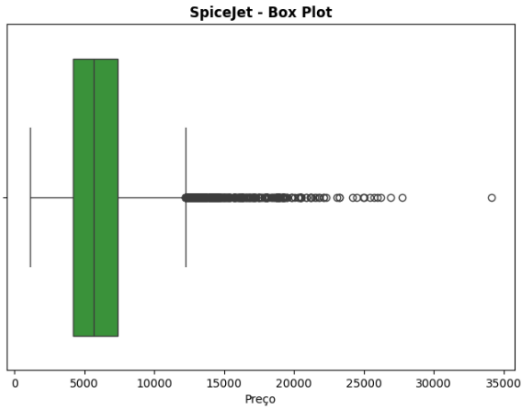
## 3.1. Diferença de Preços entre Companhias Aéreas

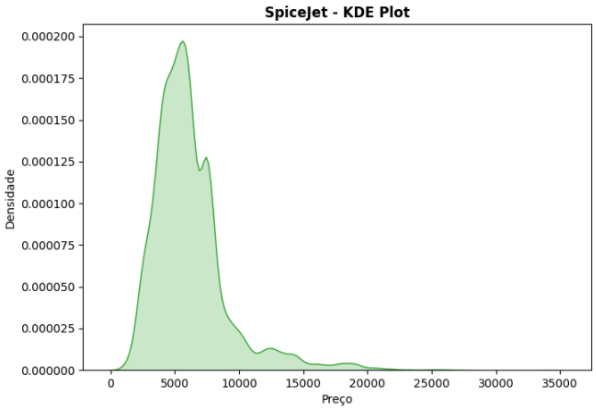


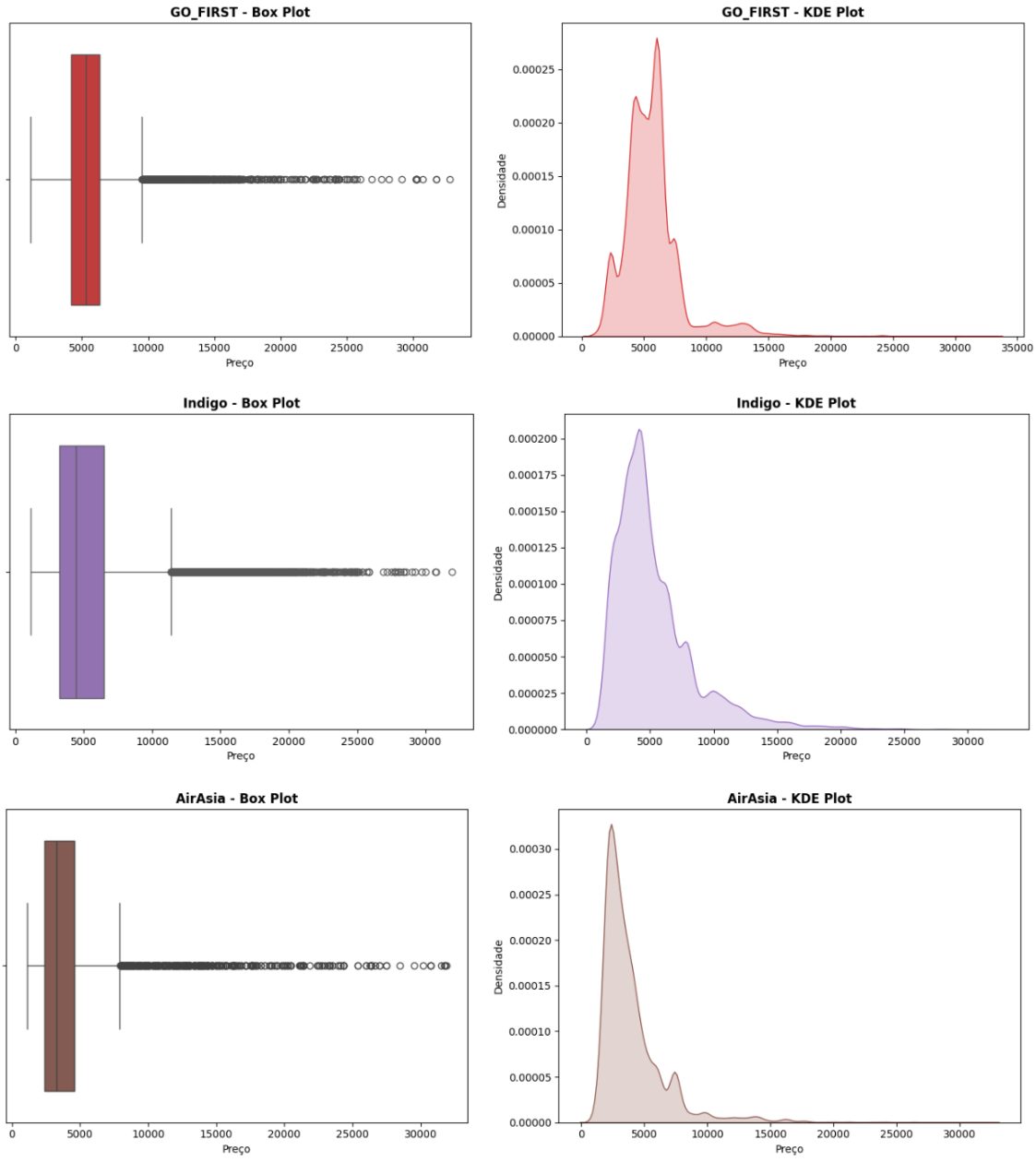



*Figura 4: Preços médios dos bilhetes entre diferentes companhias aéreas*

A análise dos preços médios dos bilhetes entre diferentes companhias aéreas revela disparidades significativas, refletindo diferenças na qualidade do serviço, no tipo de voo e nas estratégias de precificação. A **Vistara** apresenta o preço médio mais elevado, superando os 30 000, o que sugere um posicionamento *premium* no mercado, possivelmente associado a um serviço diferenciado e a voos de longa distância. Por outro lado, a **Air India** ocupa o segundo lugar entre as companhias mais caras, com um preço médio próximo de 23 500.

### 3.2. Variação dos Preços Dentro de Cada Companhia

A análise de *boxplots* revela uma elevada variabilidade nos preços, sobretudo na **Vistara** e na **Air India**. Estas companhias oferecem uma ampla faixa de preços, o que sugere que disponibilizam tanto tarifas económicas quanto *premium*. Já as companhias *low-cost,* como **AirAsia** e **GO_FIRST**, apresentam distribuições de preços mais concentradas, que indica menor variação entre as tarifas e um foco em preços acessíveis e padronizados.

Este comportamento está alinhado com a estratégia de mercado de cada companhia. Empresas *premium* tendem a oferecer diferentes classes de serviço, o que justifica a grande dispersão de preços, enquanto as *low-cost* mantêm valores mais homogéneos.

### 3.3. Distribuição dos Preços (KDE *Plots*)

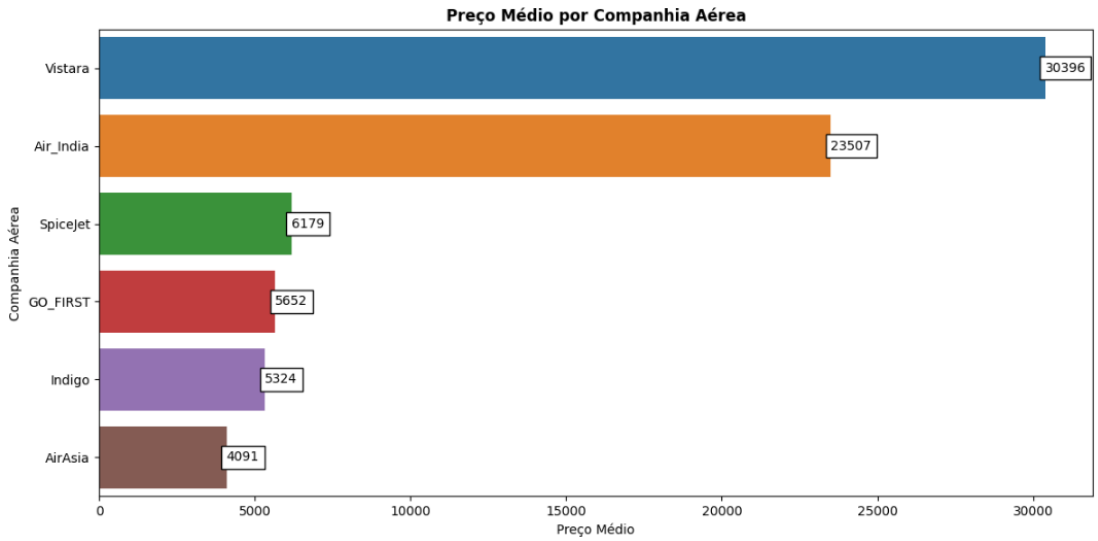



*Figura 5: Preço médio por companhia aérea*

Os gráficos KDE confirmam que as companhias *low-cost* concentram a maioria dos seus preços na faixa entre 2 000 e 10 000, sem grandes flutuações. Já as companhias *premium* apresentam distribuições multimodais, com picos distintos que refletem diferentes classes tarifárias. Esse comportamento sugere a coexistência de tarifas económicas e *premium* dentro das companhias de maior custo.

### 3.4. Proporção de Voos Curtos e Longos

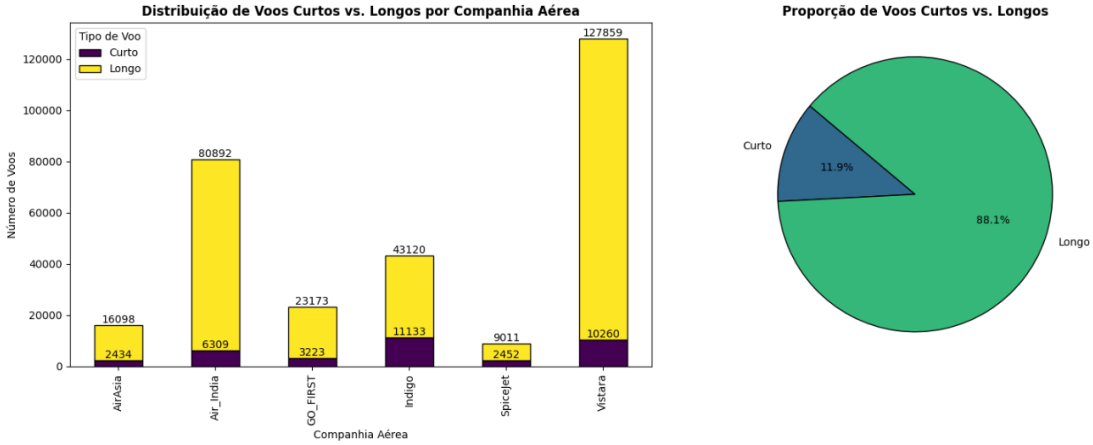



*Figura 6: Distribuição entre voos curtos e longos*

A distribuição entre voos curtos e longos varia entre as companhias aéreas, afetando diretamente a precificação. Observa-se que a maioria dos voos são de longa distância, isso representa 88,1% do total, enquanto apenas 11,9% são voos curtos. Companhias como **Vistara** e **Air India** dominam o segmento de voos longos, o que justifica os seus preços mais elevados. Por outro lado, **SpiceJet**, **GO_FIRST** e **AirAsia** concentram-se em voos curtos, isso faz com que possam oferecer preços mais acessíveis e uma estratégia de volume para maximizar a ocupação.

## 3.5. Distribuição de Preços por Companhia Aérea

Ao comparar as companhias individualmente, os *boxplots* e KDE *plots* ajudam a entender o comportamento de cada uma. A **Vistara** e a **Air India** apresentam preços mais dispersos, sugerem a oferta das classes *premium*, enquanto **AirAsia**, **Indigo** e **GO_FIRST** possuem preços mais concentrados, reforçando a sua estratégia *low-cost*.

- **Vistara:** Apresenta a maior faixa de preços, evidencia a oferta de diferentes tarifárias;
- **Air India:** Possui uma distribuição semelhante, mas com um preço médio inferior ao da **Vistara**;
- **SpiceJet e GO_FIRST:** Apresentam preços baixos e homogéneos, o que sugere um foco na eficiência operacional e em tarifas económicas;
- **Indigo e AirAsia:** São as companhias mais baratas, operam essencialmente em voos domésticos e de curta distância.

## 3.6. Estratégias para Passageiros e Companhias Aéreas

Com base nestes *insights*, tanto passageiros como companhias aéreas podem adotar estratégias para maximizar benefícios.

Para os passageiros:

- Se o objetivo for poupar dinheiro, **AirAsia**, **Indigo** e **GO_FIRST** são as melhores opções.
- Para quem procura conforto e serviços *premium*, **Vistara** e **Air India** oferecem uma experiência superior.
- Avaliar a necessidade de um voo *premium* antes de pagar mais caro, já que algumas companhias oferecem tarifas mistas.

Para as companhias aéreas:

- Ajustar a precificação para equilibrar a oferta entre voos curtos e longos.
- Criar promoções em períodos de menor procura para otimizar a ocupação.
- Diferenciar melhor os serviços *premium*, que permite justificar a diferença de preço.

## 3.7. Impacto da Política de Preços nas Decisões de Compra

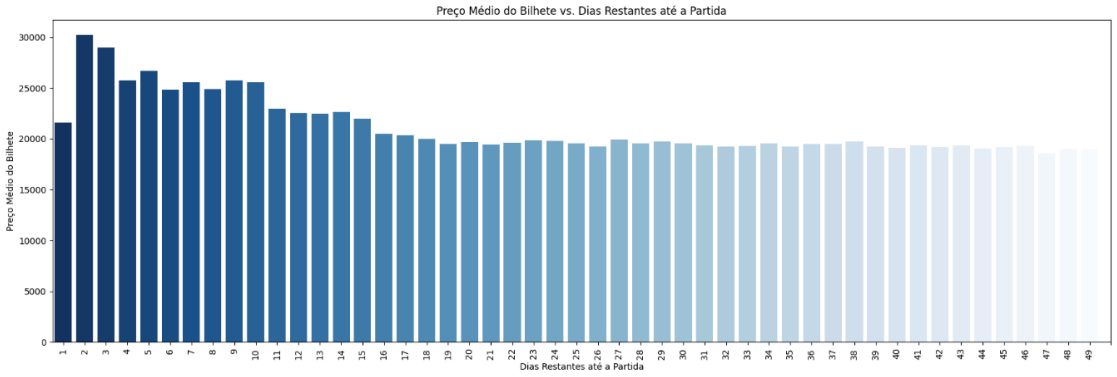



*Figura 7: Política de preços nas decisões de compra*

A política de preços adotada pelas companhias aéreas influência diretamente o comportamento dos passageiros no momento da compra. A análise dos dados demonstra que a flexibilidade nos preços, associada à antecedência da reserva, pode ser um fator determinante para a escolha da companhia aérea.

Transportadoras ***low-cost*** apostam em tarifas reduzidas para reservas antecipadas, isso incentiva a compra com maior antecedência. Este modelo atrai passageiros que estão a planear as suas viagens com antecedência e procuram economizar. No entanto, os preços sobem significativamente nos dias mais próximos à partida, o que torna os bilhetes muito mais caros para quem compra à última hora.

No que diz respeito às companhias ***premium,*** apresentam um modelo de preços mais estável, mantendo as tarifas elevadas independentemente da antecedência. Este comportamento sugere que o seu público-alvo valoriza outros fatores, como o conforto, a flexibilidade e os serviços adicionais, em detrimento de economizar no preço do bilhete.

### 3.8. Comparação entre Estratégias de Precificação


*Figura 8: Política de preços nas decisões de compra*

A variação dos preços entre companhias *premium* e *low-cost* revelam estratégias bem distintas. Enquanto as empresas de baixo custo adotam uma política de preços agressiva, com grandes variações que dependem da procura e do tempo de reserva, as companhias *premium* mantêm preços elevados e mais previsíveis ao longo do tempo.

**Companhias *Low-Cost* (*Indigo*, *AirAsia*, *GO_FIRST*, *SpiceJet*)**

- Oferecem bilhetes mais baratos para reservas antecipadas;
- Aumentam os preços significativamente nos últimos dias antes da partida;
- Estratégia focada em maximizar a ocupação dos voos.

**Companhias *Premium* (*Vistara*, *Air India*)**

- Mantêm preços elevados independentemente da antecedência;
- Pouca variação nos preços ao longo do tempo;
- Público-alvo disposto a pagar mais por conforto e flexibilidade.

Este contraste nas estratégias mostra que diferentes perfis de passageiros são atendidos por cada modelo de negócio, sendo essencial que os consumidores conheçam estas dinâmicas para tomarem decisões informadas na compra dos bilhetes.

## 3.9. Efeito da Procura e da Sazonalidade

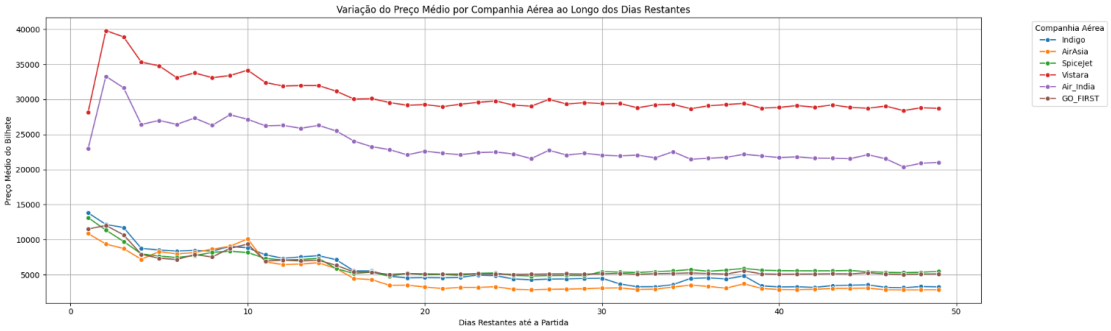



*Figura 9: Comparação das políticas de preços nas decisões de compra*

Outro fator que influencia a variação dos preços é a **procura sazonal**. Durante períodos de alta procura, como feriados, festivais ou época de férias, os preços dos bilhetes tendem a ser mais elevados.

Além disso, há momentos em que determinadas rotas têm maior fluxo de passageiros devido a eventos específicos, como conferências internacionais ou períodos de regresso de estudantes. Nestes casos, a antecedência da compra pode ter um impacto ainda maior no preço final, pois a oferta de assentos diminui rapidamente.

Companhias aéreas utilizam modelos de previsão de procura para ajustar os preços dinamicamente, isto garante o máximo de rentabilidade em períodos de pico e evita prejuízos em momentos de baixa procura.

## 3.10. Benefícios e Riscos da Precificação Dinâmica

A precificação dinâmica oferece vantagens tanto para as companhias aéreas como para os passageiros, mas também apresenta desafios.

**Benefícios:**

- Para as companhias aéreas, permite maximizar a ocupação dos voos e aumentar a rentabilidade;
- Para os passageiros, a possibilidade de encontrar tarifas mais baixas ao reservar com antecedência;
- Incentiva uma melhor distribuição da procura ao longo do tempo e reduz a concentração de compras de última hora.

**Riscos:**

- Passageiros que costumam comprar os bilhetes nos últimos dias podem ser penalizados com preços muito elevados;
- A complexidade da variação de preços pode confundir consumidores, o que dificulta a tomada de decisão;
- Excesso de precificação dinâmica pode gerar insatisfação entre clientes que sentem que estão a pagar valores muito superiores por reservas tardias.

Desta forma, a estratégia de precificação deve equilibrar a rentabilidade para as companhias e a acessibilidade para os consumidores, isso garante uma experiência satisfatória para ambas as partes.

# 4. Análise do Impacto Temporal na Precificação dos Bilhetes

A precificação dinâmica das passagens aéreas é altamente influenciada pelo fator temporal, que pode ser dividido em diferentes perspetivas:

- **Dias restantes até a partida** – A antecedência da compra tem impacto direto no preço do bilhete.
- **Período do dia da partida** – O horário do voo influencia a procura e, consequentemente, os valores das tarifas.
- **Período do dia da chegada** – Semelhante ao horário de partida, mas focado na conveniência do horário de chegada ao destino.

O impacto temporal pode ser analisado sob diferentes aspetos, como a procura dos passageiros, conveniência dos horários e a estratégia de otimização de receita por parte das companhias aéreas.

## 4.1. Variação dos Preços com Base na Antecedência da Compra

A antecedência com que um bilhete é adquirido tem um efeito significativo no preço final da passagem. Como esperado, os bilhetes comprados **próximos à data do voo** apresentam valores substancialmente mais elevados, enquanto **reservas feitas com maior antecedência resultam em tarifas mais económicas**.

Os gráficos analisados confirmam esta tendência. Os preços começam a subir acentuadamente **cerca de 10 dias antes do voo**, atingindo picos elevados quando restam menos de 5 dias. Isto indica que a **compra antecipada é um dos fatores-chave para garantir tarifas mais acessíveis**.

Além disso, a análise dos preços ao longo do tempo revela que, **após aproximadamente 15 dias de antecedência**, os valores estabilizam, tornando-se menos voláteis. Isto sugere que as companhias aéreas ajustam as suas estratégias de precificação para maximizar a ocupação dos voos com base na procura esperada.

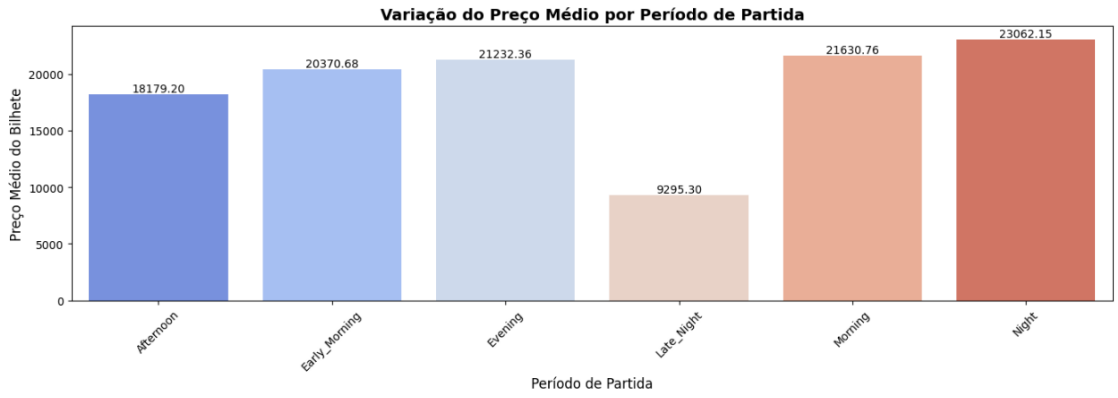



*Figura 10: Variação do preço médio por período de partida*

## 4.2. Tendências e Oscilações nos Preços dos Bilhetes

A tendência de queda dos preços à medida que a antecedência de compra aumenta **não é completamente linear**. Pequenas oscilações nos preços são observadas ao longo do tempo, refletindo ajustes dinâmicos feitos pelas companhias aéreas para equilibrar oferta e procura.

Em determinados períodos, há **flutuações mais acentuadas**, que podem estar associadas a eventos específicos como a sazonalidade ou mudanças na política de preços das empresas. Por exemplo, a análise sugere que **em alguns dias os preços podem ser reduzidos temporariamente**, isto pode indicar oportunidades de compra vantajosas para passageiros atentos às promoções.

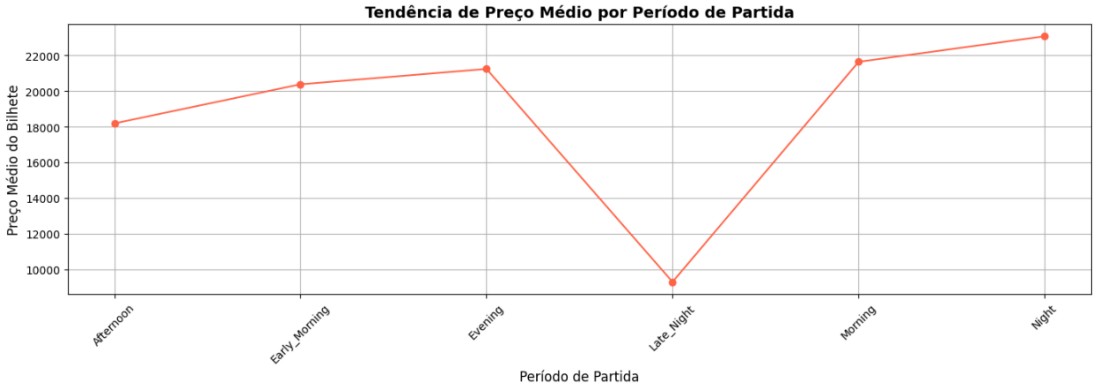



*Figura 11: Tendência de preço médio por período de partida*

## 4.3. Diferenças de Precificação entre Companhias Aéreas

Nem todas as companhias aéreas seguem exatamente o mesmo padrão de precificação ao longo do tempo. Algumas mantêm preços elevados **independentemente da antecedência da compra**, enquanto outras oferecem maior flexibilidade nas tarifas.

Os dados indicam que companhias ***premium***, como **Vistara e Air India**, mantêm tarifas altas ao longo do tempo, independentemente da antecedência da compra. Já **empresas *low-cost*, como Indigo e AirAsia**, apresentam maior variação nos preços, ajustam as tarifas conforme a proximidade da data do voo.

Essa diferença de abordagem sugere que companhias ***premium*** **priorizam clientes que estão à procura de conveniência e serviço diferenciado**, enquanto **empresas *low-cost* focam muito mais na acessibilidade e volume de passageiros**.

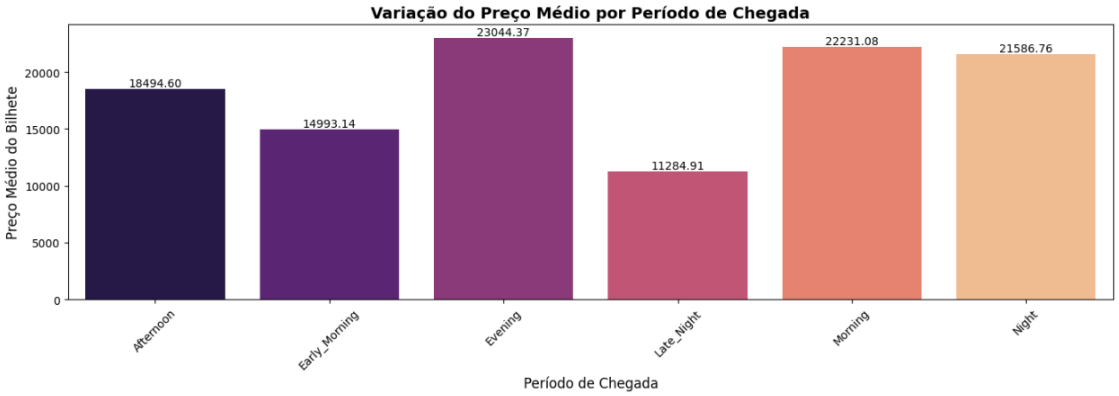



*Figura 12: Variação do preço médio por período da chegada*

## 4.4. Impacto do Período do Dia no Preço dos Bilhetes

O horário do voo, tanto no momento da partida quanto da chegada, é outro fator determinante na precificação das passagens.

Os dados analisados mostram que:

- **Os voos noturnos (*Night*) e matinais (*Morning*) têm as tarifas mais elevadas**, pois são considerados períodos *premium* devido à conveniência e à elevada procura.
- **Os voos de madrugada (*Late Night*) costumam ser os mais baratos**, pois há menor procura e menos conexões internacionais nesses horários.
- **A tarde e a noite apresentam preços médios equilibrados**, o que reflete uma maior oferta de assentos e menor volatilidade na precificação.

A comparação entre os períodos de partida e chegada reforça a tendência de que os horários mais concorridos resultam em tarifas mais altas, enquanto horários alternativos oferecem melhores oportunidades de poupança para os passageiros.

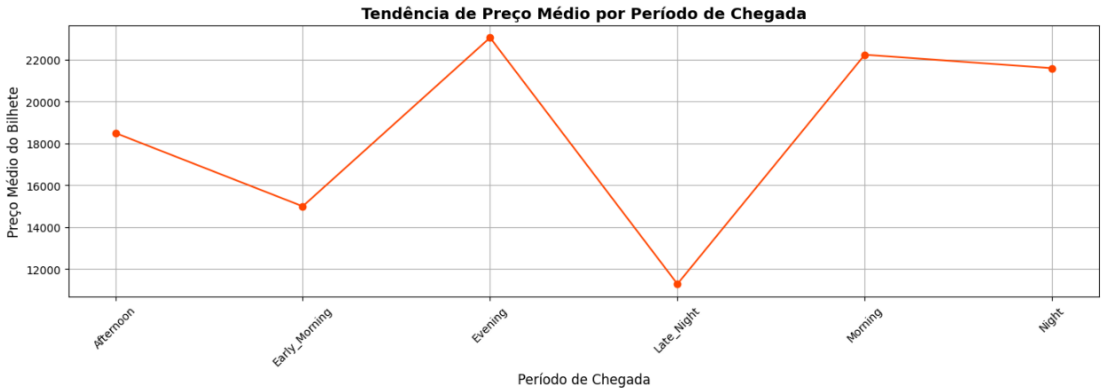



*Figura 13: Tendência de preço médio por período de chegada*

### 4.5. Estratégias para Passageiros e Companhias Aéreas

Os padrões temporais identificados oferecem *insights* valiosos tanto para passageiros quanto para as companhias aéreas.

**Para passageiros:**

- **Reservar com antecedência garante melhores preços:** idealmente, a compra deve ser feita com pelo menos 15 dias de antecedência.
- **Evitar compras de última hora:** os preços aumentam exponencialmente nos últimos dias antes do voo.
- **Ajustar a escolha do horário do voo:** os voos de madrugada oferecem tarifas mais baixas, enquanto horários *premium* (manhã e noite) tendem a ser mais caros.

**Para companhias aéreas:**

- **Utilizar precificação dinâmica para maximizar a ocupação dos voos ao longo do tempo:** ajustar preços de forma estratégica conforme a antecedência da reserva.
- **Criar promoções específicas para voos de baixa procura:** incentivar a compra antecipada para otimizar a taxa de ocupação.
- **Diferenciar preços com base na demanda por horário:** maximizar receitas ao oferecer tarifas *premium* nos horários mais procurados.

# 5. Impacto da Combinação do Período de Partida e Chegada no Preço dos Bilhetes

A precificação das passagens aéreas não é influenciada apenas pelo período de partida ou chegada individualmente, mas também pela combinação desses fatores. A interação entre os horários de partida e chegada pode gerar variações significativas nos preços, o que faz com que existam padrões de procura e disponibilidade de voos.

## 5.1. Padrões de Preço de Acordo com a Combinação de Horários

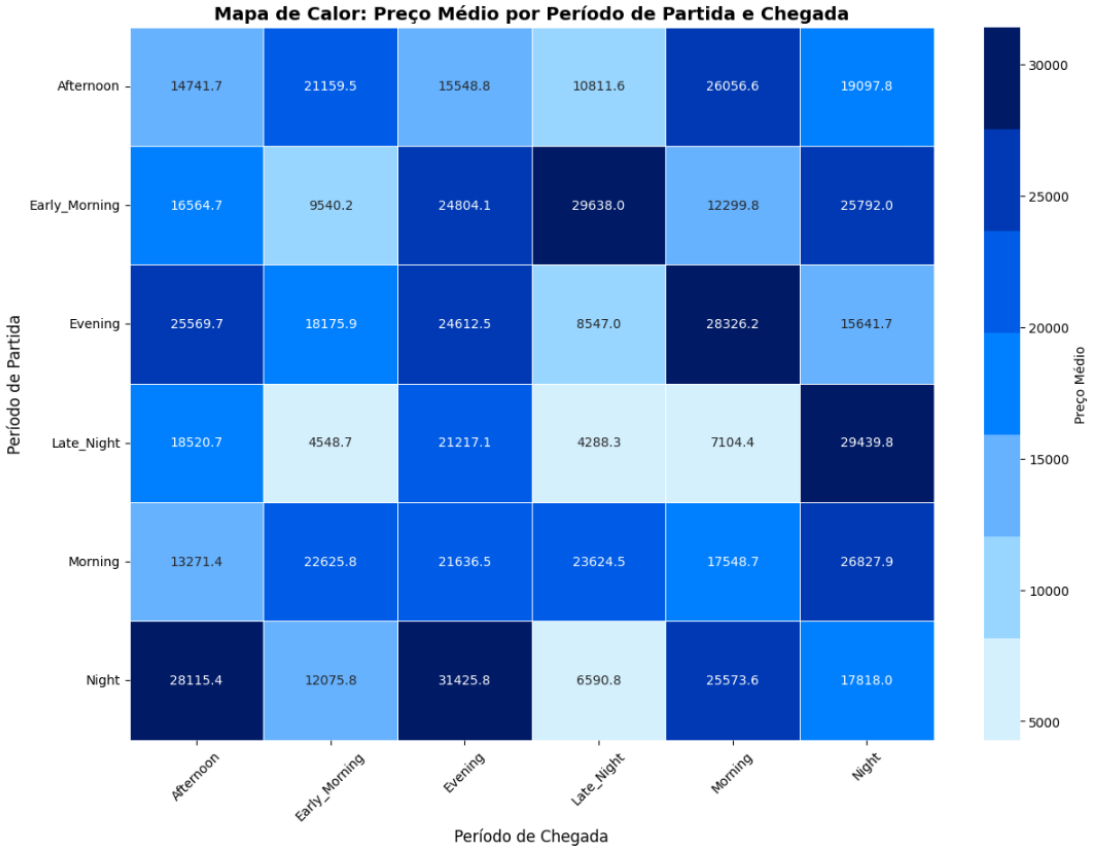



*Figura 14: Mapa de calor*

O mapa de calor apresentado evidencia as combinações de horários de partida e chegada que resultam em tarifas mais elevadas e mais baixas. Algumas observações importantes incluem:

- Os voos que partem e chegam à noite apresentam as tarifas mais elevadas, com valores superiores a 30 000, devido à alta procura de passageiros corporativos e conexões internacionais.

- Voos que partem de madrugada e chegam também na madrugada tendem a ser os mais baratos, com algumas tarifas abaixo de 5 000, o que sugere uma baixa procura e menor volume de passageiros nesses períodos.
- Ao combinar os horários como "*Early Morning*" para partida e "*Evening*" para chegada, observa-se um padrão misto, onde as tarifas podem variar consideravelmente, uma vez que dependerá da procura específica do destino e da rota.

### 5.2. Explicação das Diferenças de Preços

A análise dos dados sugere que a diferença de preços se deve a fatores como a procura, a conveniência e a estrutura das conexões. Algumas razões para essas variações incluem:

- **Alta procura em horários noturnos:** voos noturnos e que chegam à noite geralmente estão associados a longas rotas internacionais ou viagens de negócios. Isto justifica a precificação mais elevada, já que os passageiros corporativos priorizam horários estratégicos para otimização do tempo.
- **Menor procura e tarifas reduzidas para partidas e chegadas de madrugada:** Voos que operam nestes períodos tendem a ser menos convenientes para a maioria dos passageiros, resultando assim em menores preços para estimular a ocupação.
- **Influência da duração e conexões:** Voos que exigem conexões podem apresentar preços diferentes dependendo da estrutura aérea. Quando há escalas longas em horários inconvenientes, os preços podem ser reduzidos para atrair passageiros.

### 5.3. Estratégias para Passageiras e Companhias Aéreas

A análise destas combinações oferece *insights* úteis para passageiros e empresas aéreas na otimização de tarifas e ocupação dos voos.

**Para Passageiros:**

- Optar por voos com partidas e chegadas em horários menos procurados pode garantir tarifas mais acessíveis.
- Evitar conexões longas em horários *premium*, como manhãs e noites, para reduzir custos adicionais.
- Procurar combinações estratégicas que alinhem a conveniência e a economia, considerando tanto o período de partida quanto o de chegada.

**Para Companhias Aéreas:**

- Ajustar estratégias de precificação dinâmica para maximizar a ocupação em horários de menor procura.
- Implementar promoções específicas para voos com menor taxa de ocupação, ao incentivar passageiros a optarem por horários alternativos.
- Utilizar tarifas *premium* para horários estratégicos, ao otimizarem receitas em períodos de alta procura.

# 6. Análise da Distribuição de Preços por Cidade de Origem

A cidade de origem dos voos desempenha um papel crucial na precificação das passagens aéreas, sendo que influencia a variação dos preços médios. Os fatores como a oferta de voos, a infraestrutura aeroportuária e a procura impactam diretamente o custo dos bilhetes. Nesta análise, examinamos a distribuição dos preços médios por cidade de origem para identificar padrões e tendências relevantes.

## 6.1. Padrões de Preço por Cidade de Origem

Os dados revelam que a cidade de ***Delhi*** **apresenta o menor preço médio (18 951)**, isto sugere que a alta concorrência entre companhias aéreas e um maior volume de voos podem estar a reduzir os preços. Por outro lado, ***Chennai*** **tem o preço médio mais elevado (21 995)**, o que pode indicar uma menor oferta de voos ou um predomínio de rotas *premium* e de longas distâncias.

Além disso, as pequenas variações entre as cidades sugerem que a precificação segue um padrão relativamente homogéneo, embora algumas localidades apresentam diferenças mais significativas devido à disponibilidade de voos diretos e à infraestrutura aeroportuária.

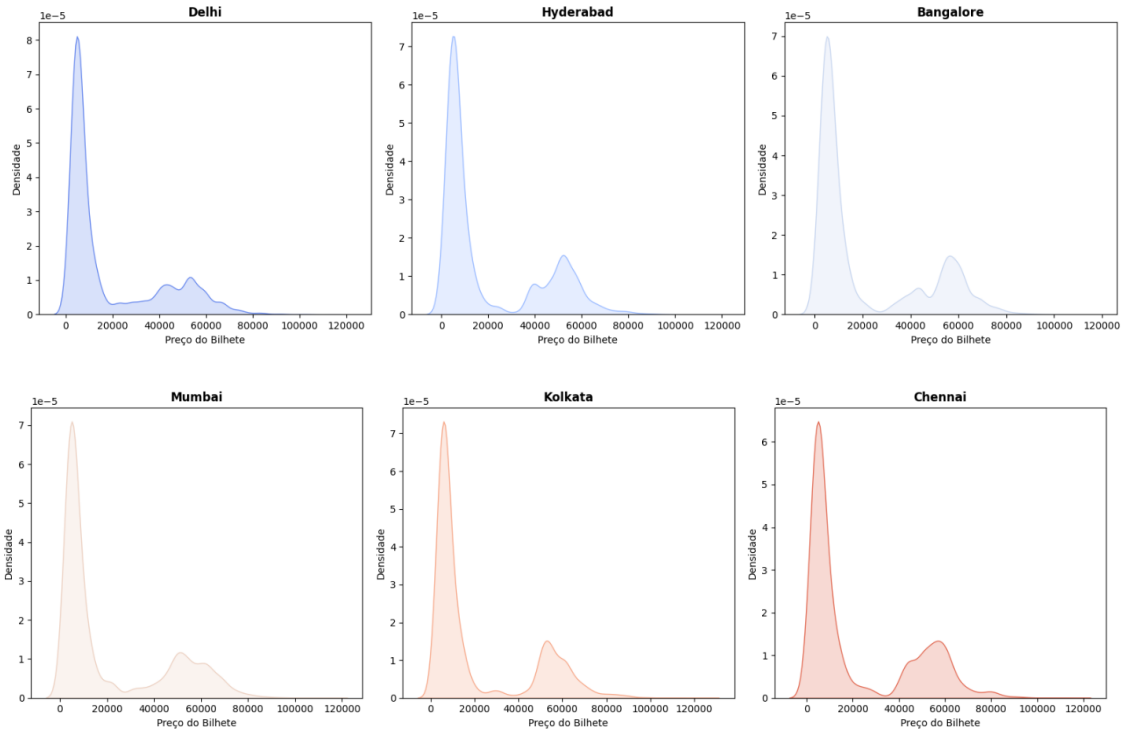



*Figura 15: Padrões de preço por cidade de origem*

## 6.2. Explicação da Variação de Preços

A análise da variação dos preços por cidade de origem indica que:

- Cidades com maior número de voos tendem a ter preços mais baixos devido à concorrência entre companhias aéreas e à elevada oferta de assentos.
- Em contrapartida, cidades com menor concorrência podem apresentar tarifas mais elevadas devido à escassez de voos e menor flexibilidade de itinerários.
- A localização geográfica e a infraestrutura dos aeroportos também são fatores determinantes, uma vez que *hubs* estratégicos para conexões internacionais podem influenciar os preços médios.

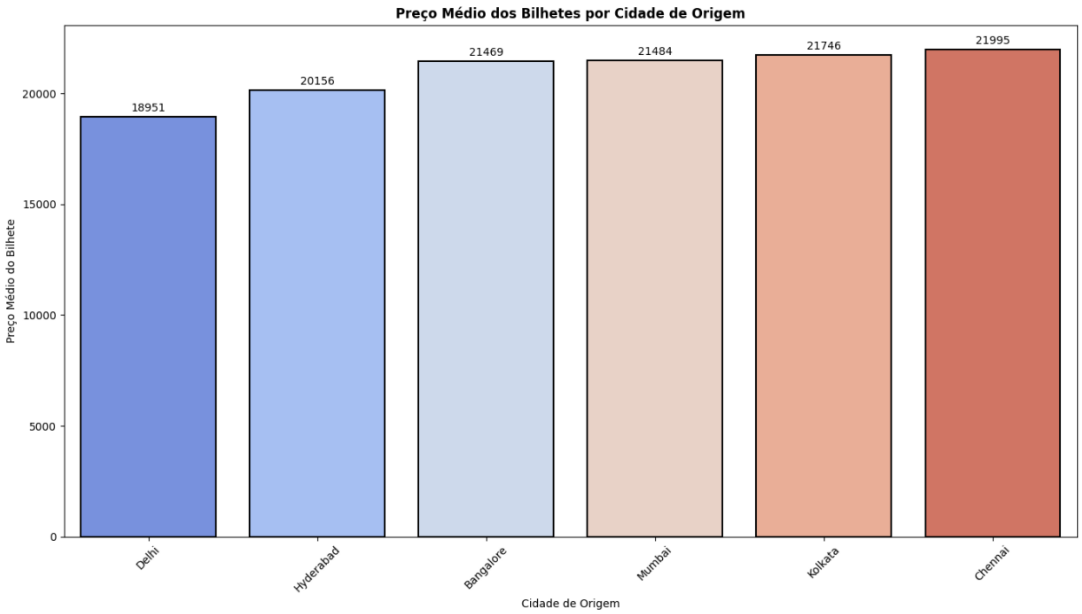



*Figura 16: Preço médio dos bilhetes por cidade de origem*

## 6.3. Comparação com Análises Anteriores

Os gráficos de densidade (KDE) confirmam que as faixas de preços diferem conforme a cidade de origem. A relação entre preços mais baixos e maior volume de voos foi observada anteriormente na análise das cidades de destino, sugerindo que a competitividade no setor impacta de forma consistente tanto na origem quanto no destino das viagens.

Além disso, a dispersão dos preços reforça a ideia de que algumas cidades oferecem mais opções *premium*, enquanto outras são dominadas por tarifas económicas. Esta segmentação pode ser explorada por passageiros que desejam otimizar o custo das suas viagens, escolhendo pontos de partida mais vantajosos.

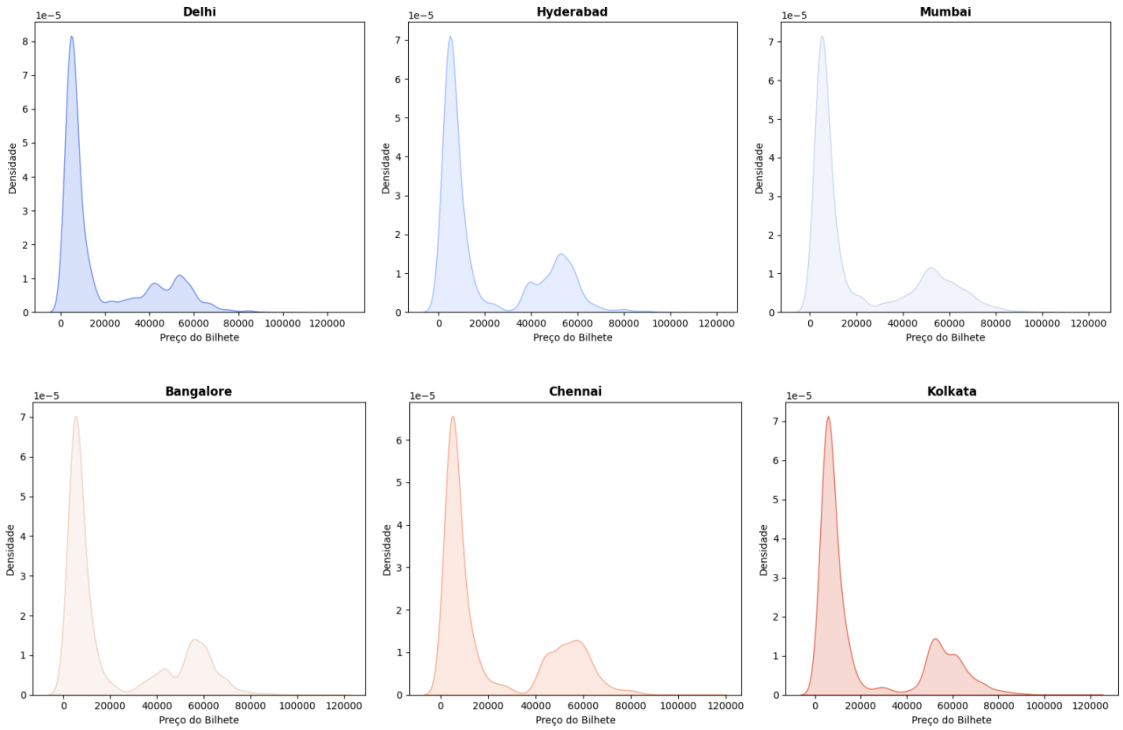
# 7. Análise da Distribuição de Preços por Cidade de Destino

A análise dos preços médios dos bilhetes com base na cidade de destino fornece *insights* importantes sobre como a localização influencia na precificação das passagens aéreas. A distribuição dos preços entre diferentes cidades revela padrões de mercado que podem estar relacionados à procura, às conexões disponíveis e às estratégias de precificação das companhias aéreas.

## 7.1. Padrões de Preço por Cidade de Destino

Os dados indicam que Delhi apresenta o menor preço médio (18 437), possivelmente devido à alta concorrência e ao maior volume de voos, isso reduz os custos das viagens para os passageiros. Em contrapartida, cidades como Kolkata e Chennai apresentam os preços mais altos, com valores médios superiores a 21 000, o que indica uma menor concorrência ou uma predominância de voos de longa distância com tarifas *premium*.

A distribuição dos preços nos gráficos KDE indica que algumas cidades possuem picos secundários, o que nos indica que existem variações na precificação conforme a classe tarifária e a sazonalidade da procura.



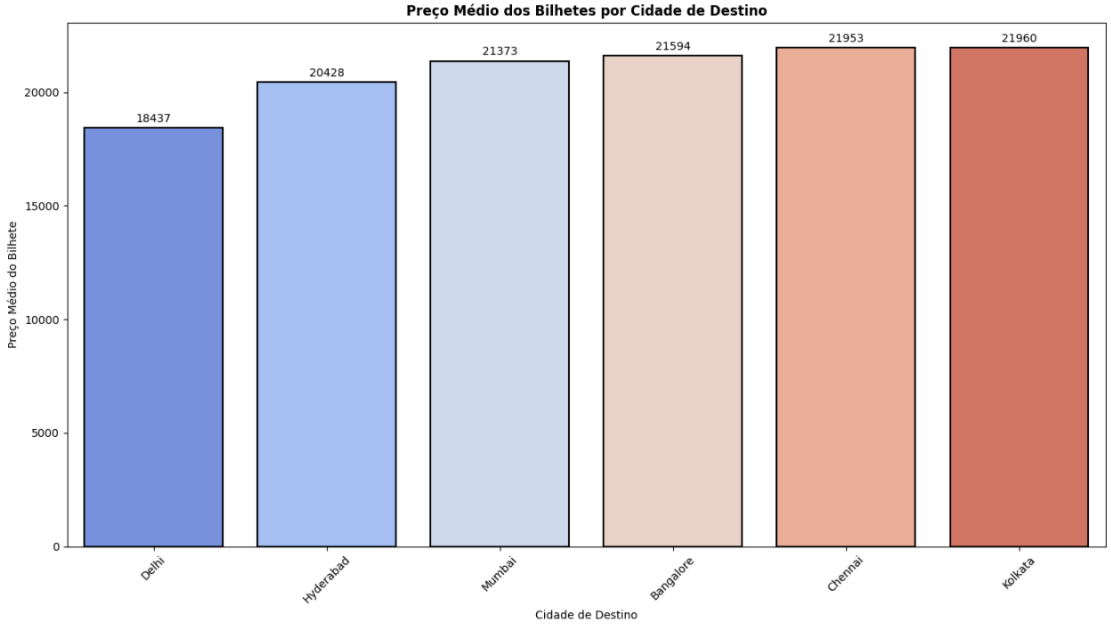



*Figura 17: Preço médio dos bilhetes por cidade do destino*

## 7.2. Comparação com a Análise das Cidades de Partida

Ao comparar com os dados de cidades de origem, percebe-se a presença de uma semelhança na variação de preços. Cidades com maior volume de voos tendem a apresentar tarifas mais baixas, enquanto aquelas com menor concorrência apresentam preços mais elevados. Além disso, cidades que funcionam como *hubs* principais tendem a ter as tarifas reduzidas devido à concorrência entre companhias aéreas, enquanto aquelas que recebem menos voos registam tarifas mais altas devido à menor oferta.

## 7.3. Explicação da Variação de Preços

Destinos populares e de alta procura costumam apresentar preços mais acessíveis devido à alta concorrência. Por outro lado, cidades com menos voos diretos e uma infraestrutura aeroportuária mais restrita tendem a apresentar preços elevados devido às poucas opções de ligação. Adicionalmente, as rotas de longa distância, que exigem conexões ou maior tempo de voo, impactam diretamente a precificação.

## 7.4. Estratégias para Passageiros e Companhias Aéreas

Para os passageiros, escolher destinos com maior volume de voos pode ser uma estratégia eficaz para encontrar tarifas mais acessíveis. Acompanhar promoções e

avaliar a flexibilidade na escolha do destino final também podem ajudar a reduzir os custos.

Para as companhias aéreas, a avaliação da procura por diferentes cidades pode indicar a necessidade de ajustar a oferta de voos e implementar estratégias promocionais em destinos menos atendidos. O uso de tarifas dinâmicas também pode ser uma solução para equilibrar preços em destinos variados, isto considerando a sazonalidade e o volume de passageiros.

# 8. Análise da Variação de Preços por Classe e Companhia Aérea

Os preços dos bilhetes de avião variam significativamente consoante a classe de voo escolhida e a companhia aérea operadora. Esta análise examina as diferenças entre as classes ***Economy*** e ***Business***, bem como a discrepância entre as companhias no contexto de cada classe.

## 8.1. Preço Médio dos Bilhetes por Classe

Os dados mostram que a classe *Business* é substancialmente mais cara do que a classe *Economy*. O preço médio dos bilhetes *Business* ronda os **52 540**, enquanto a classe *Economy* apresenta uma média de **6 572**. A grande diferença pode ser explicada pelo nível de conforto superior, serviços exclusivos e melhores condições de acondicionamento na *Business*.

Além disso, passageiros corporativos e clientes *premium* tendem a optar pela classe *Business* para garantir um voo mais confortável e produtivo. Por outro lado, a *Economy* atende um público mais sensível ao preço, que prioriza o custo-benefício em relação ao luxo.

## 8.2. Variação do Preço Médio por Classe e Companhia Aérea

A análise por companhia aérea revela que cada empresa adota por diferentes estratégias de precificação para cada uma das suas classes. As companhias *premium*, como **Air India e Vistara**, apresentam preços médios elevados, especialmente na classe *Business*. Já companhias *low-cost* como **Indigo, AirAsia e SpiceJet**, mantêm os preços reduzidos na classe *Economy*, o que nos indica uma estratégia focada na acessibilidade.

É importante destacar que a discrepância entre os preços da classe *Business* é muito mais acentuada do que na *Economy*. Algumas companhias aéreas cobram valores significativamente mais altos devido ao posicionamento no segmento *premium*, enquanto outras mantêm tarifas acessíveis, valorizam a eficiência operacional.

## 8.3. Comparação com Análises Anteriores

A relação entre a companhia aérea e o preço médio manteve-se consistente em relação às análises anteriores. As companhias que exibem preços mais altos no geral também

apresentam valores mais elevados na classe *Business*, reforçam assim a sua identidade *premium*.

Já as companhias com bilhetes mais acessíveis seguem a tendência de manter as tarifas reduzidas na *Economy*. Isto confirma que a estratégia de mercado influencia diretamente a política de precificação dentro de cada classe.

Já os passageiros devem levar em consideração não apenas a classe do voo, mas também a companhia aérea ao tomar as decisões de compra.

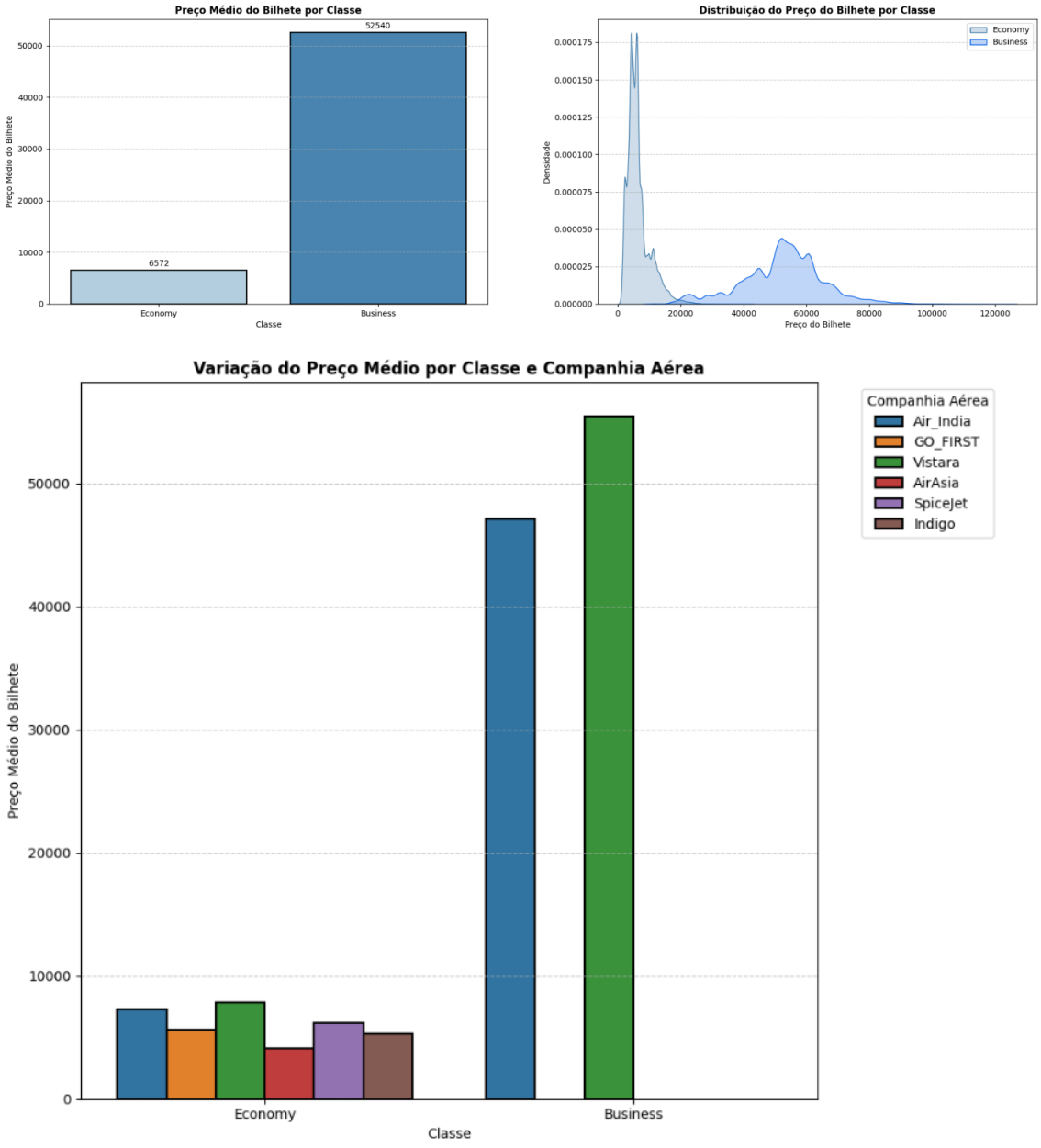



*Figura 18: Comparação análises anteriores*

# 9. Análise da Duração dos Voos e Impacto do Número de Escalas

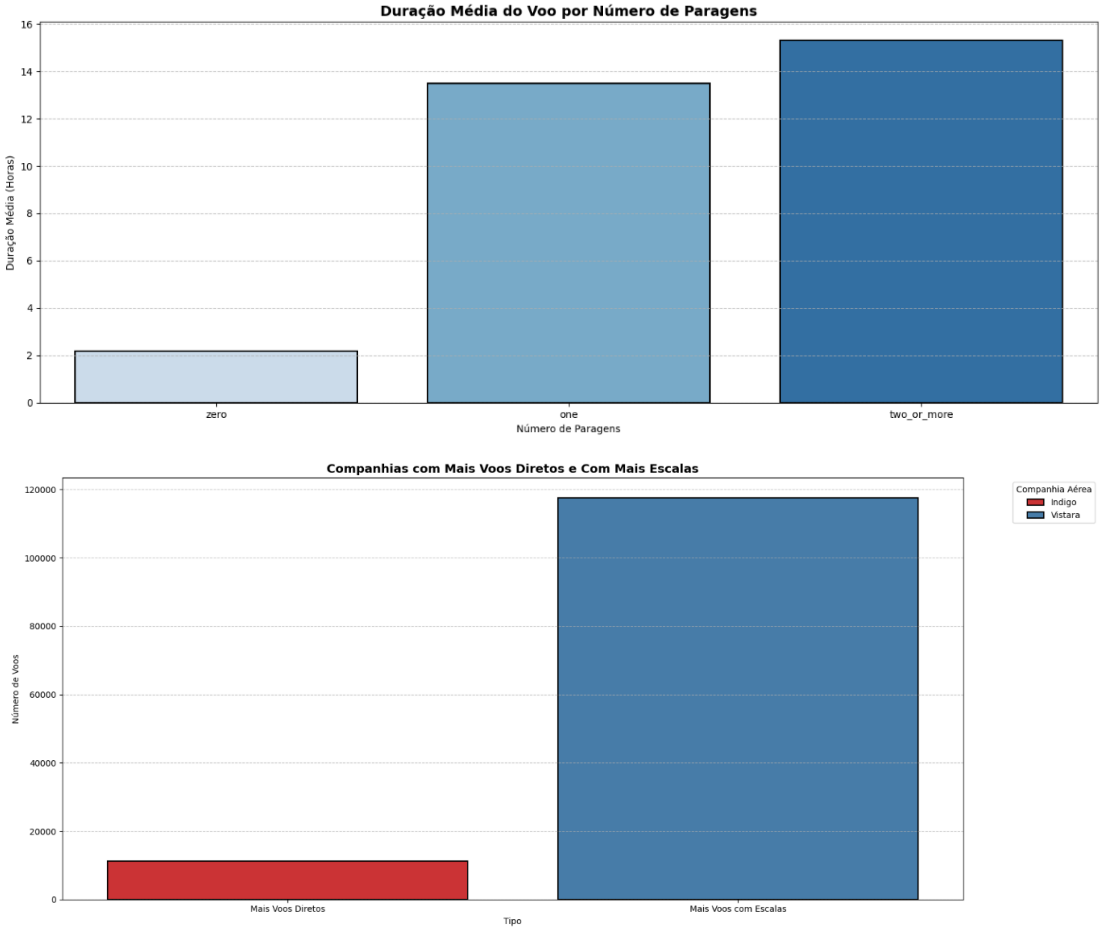



*Figura 19: Duração dos voos e impacto do número de escalas*

## 9.1. Duração Média do Voo por Número de Paragens

A duração de um voo pode variar consideravelmente, isso vai depender do número de escalas. Voos diretos, naturalmente, apresentam um tempo de viagem significativamente menor do que aqueles com uma ou mais paragens.

- **Voos diretos (zero escalas):** Média de 2,19 horas.
- **Voos com uma escala:** Média de 13,5 horas.
- **Voos com duas ou mais escalas:** Média de 15,3 horas.

Este aumento na duração reflete o tempo adicional necessário para conexões e possíveis atrasos. Passageiros que priorizam rapidez devem optar por voos diretos sempre que possível.

### 9.2. Companhias com Maior Número de Voos Diretos e Com Escalas

A escolha da companhia aérea pode impactar diretamente a probabilidade de um voo ser direto ou com escalas. Algumas companhias têm um modelo operacional focado em voos diretos, enquanto outras priorizam rotas com conexões para otimizar a ocupação de assentos.

- **IndiGo** – destaca-se como a companhia com o maior número de voos diretos.
- **Vistara** – lidera em número de voos com escalas, o que nos sugere que existe um foco maior em conexões internacionais ou estratégicas.

Este dado pode ser útil para passageiros que estão à procura de reduzir o tempo de viagem ao escolherem companhias com maior incidência de voos diretos.

### 9.3. Relação entre Número de Escalas e Preço do Bilhete

Outro fator relevante ao analisar escalas é o impacto no custo do bilhete.

- **Voos diretos tendem a ser mais caros** devido à conveniência e à menor duração.
- **Voos com escalas podem oferecer tarifas mais baixas**, embora impliquem maior tempo de deslocação e potencial desconforto para os passageiros.

Os passageiros devem considerar este equilíbrio entre preço e conveniência ao planearem as suas viagens, especialmente em rotas longas onde uma escala pode ser uma opção financeiramente viável.

# 10. Preparação dos Dados para Modelação Preditiva

Agora que realizámos a primeira análise exploratória do *dataset*, vamos avançar para a preparação dos dados para a modelação preditiva. O objetivo desta etapa é garantir que os dados estejam no formato adequado para serem utilizados em modelos de *Machine Learning*, de forma a melhorar a qualidade das previsões e a eficiência dos algoritmos.

## 10.1. Objetivo da Preparação

O objetivo desta preparação é garantir que:

- Todas as variáveis estejam no formato adequado;
- Variáveis categóricas sejam convertidas para valores numéricos;
- Variáveis irrelevantes sejam removidas para evitar ruído no modelo;

O *dataset* esteja pronto para a análise e construção dos modelos preditivos.

## 10.2. Análise e Transformação dos Dados

Antes de iniciarmos a transformação dos dados, realizamos uma análise detalhada das variáveis para compreender quais é que vão precisar de tratamento.

A estrutura do *dataset* foi verificada para compreender os tipos de dados presentes:

```
root
 |-- airline: string (nullable = true)
 |-- flight: string (nullable = true)
 |-- source_city: string (nullable = true)
 |-- departure_time: string (nullable = true)
 |-- stops: string (nullable = true)
 |-- arrival_time: string (nullable = true)
 |-- destination_city: string (nullable = true)
 |-- class: string (nullable = true)
 |-- duration: double (nullable = true)
 |-- days_left: integer (nullable = true)
 |-- price: integer (nullable = true)
 |-- flight_type: string (nullable = false)
 |-- stops_mapped: integer (nullable = true)
```

*Figura 20: Análise do dataset*

A análise revelou que algumas colunas precisam de ser convertidas para valores numéricos, enquanto outras já estão no formato adequado:

| Coluna | Tipo | Ação | Justificação |
|---|---|---|---|
| airline | String | Transformar | Companhia aérea (categórica) |
| flight | String | Remover | Apenas um identificador, não relevante |
| source_city | String | Transformar | Cidade de origem do voo |
| departure_time | String | Transformar | Momento do dia em que o voo parte |
| stops | String | Transformar | Número de escalas (existe uma versão numérica: stops_mapped) |
| arrival_time | String | Transformar | Momento do dia em que o voo chega |
| destination_city | String | Transformar | Cidade de destino |
| class | String | Transformar | Classe do bilhete (Económica ou Executiva) |
| duration | Double | Manter | Já está no formato adequado |
| days_left | Integer | Manter | Já está no formato adequado |
| price | Integer | Manter | Variável target (preço) |
| flight_type | String | Transformar | Tipo de voo (Curto ou Longo) |
| stops_mapped | Integer | Manter | Versão numérica da coluna 'stops' |

*Figura 21: Visualização das colunas*

A coluna "*flight*" contém apenas identificadores dos voos e não contribui para a previsão do preço. Por isso, foi removida do *dataset*.

Como os modelos de *Machine Learning* não trabalham com texto, foi necessário converter as variáveis categóricas em valores numéricos.

| Técnica | O que faz? | Quando usar? | Problemas |
|---|---|---|---|
| Label Encoding | Atribui um número inteiro a cada categoria. | Quando há uma relação de ordem (ex: Pequeno, Médio, Grande). | Pode induzir hierarquia errada. |
| One-Hot Encoding | Cria colunas binárias para cada categoria. | Quando não há relação de ordem (ex: Companhias aéreas, cidades). | Pode aumentar o número de colunas. |

*Figura 22: Técnicas encoding*

Como não existe uma hierarquia lógica entre as categorias, foi utilizada a técnica de ***One-Hot Encoding***. Com a aplicação desta transformação, as variáveis categóricas foram convertidas em valores numéricos, isso permite que o *dataset* esteja pronto para a modelagem preditiva.

# 11. Matriz de Correlação

A análise da matriz de correlação é um passo fundamental na modelagem preditiva, pois permite identificar relações entre variáveis e eliminar aquelas que não contribuem significativamente para o modelo. A seguir, apresentamos os principais aspetos observados na matriz de correlação.

## 11.1. Cálculo da Matriz de Correlação

Para calcular a matriz de correlação, seguimos os seguintes passos:

- **Definição das variáveis numéricas**, inclui aquelas que foram transformadas pelo *One-Hot Encoding*.
- **Expansão de vetores**, ao converte colunas codificadas para um formato adequado.
- **Geração da matriz de correlação** ao usar a função *Correlation.corr()*.

A matriz resultante é visualizada em um *heatmap* para facilitar a análise visual.

## 11.2. Observações Gerais

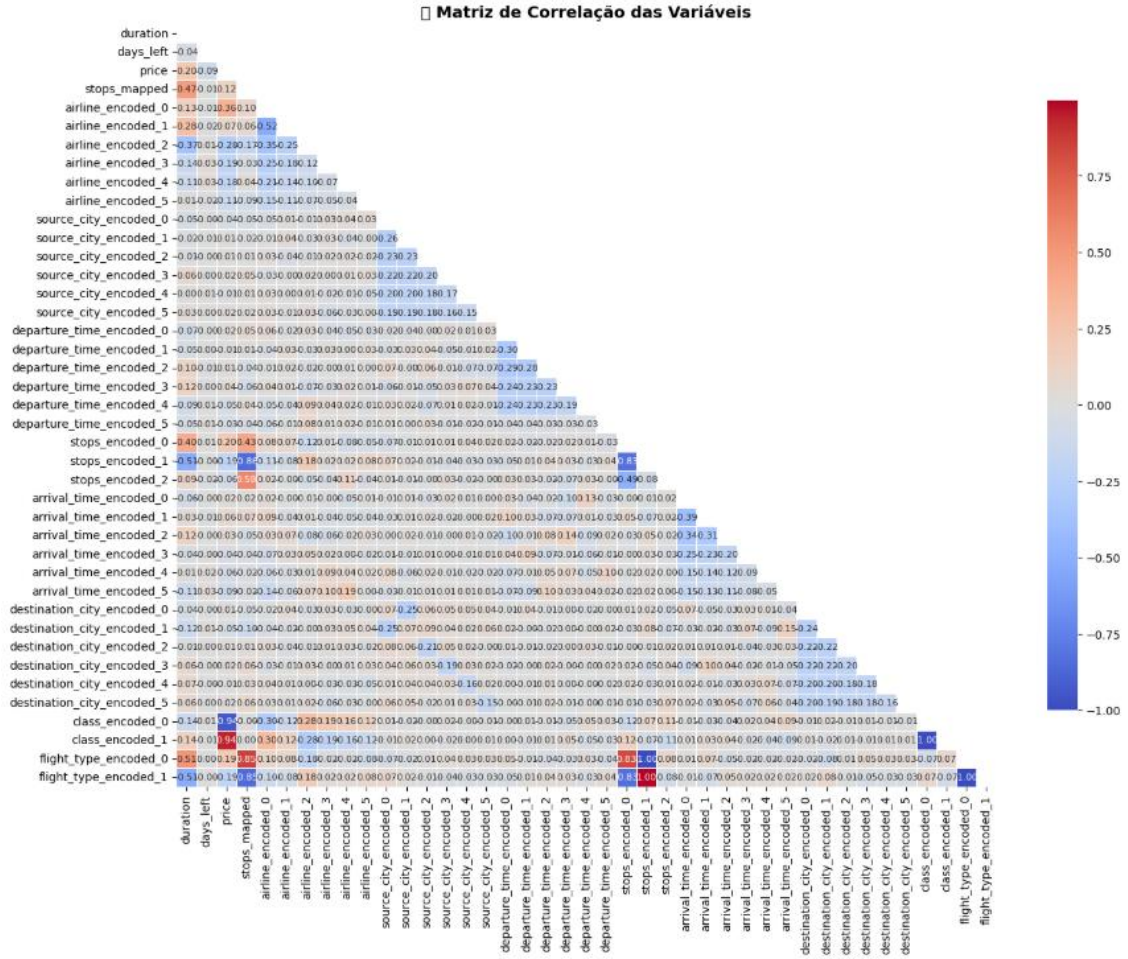



*Figura 23: Matriz de correlação das variáveis*

**Densidade de Informação**

- A matriz apresenta um grande número de variáveis, especialmente devido à codificação *One-Hot Encoding*.
- Muitas colunas apresentam correlação próxima de zero, que nos indica uma baixa influência sobre as demais variáveis.

**Correlação entre Variáveis Categóricas**

- Algumas variáveis da mesma categoria, como *airline_encoded_0* e *airline_encoded_1*, apresentam forte correlação negativa devido à natureza do *One-Hot Encoding*.
- Esse efeito pode dificultar a interpretação da matriz e deve ser levado em consideração na seleção de variáveis.

**Correlação entre Variáveis Principais**

- *stops_mapped* tem uma correlação moderada com *price* (~0.12), indicando que o número de escalas influencia o preço.
- *days_left* apresenta uma correlação negativa com *price* (~-0.20), que reflete a tendência de bilhetes mais baratos quando comprados com antecedência.

## 11.3. Identificação das Variáveis Mais Correlacionais com Price

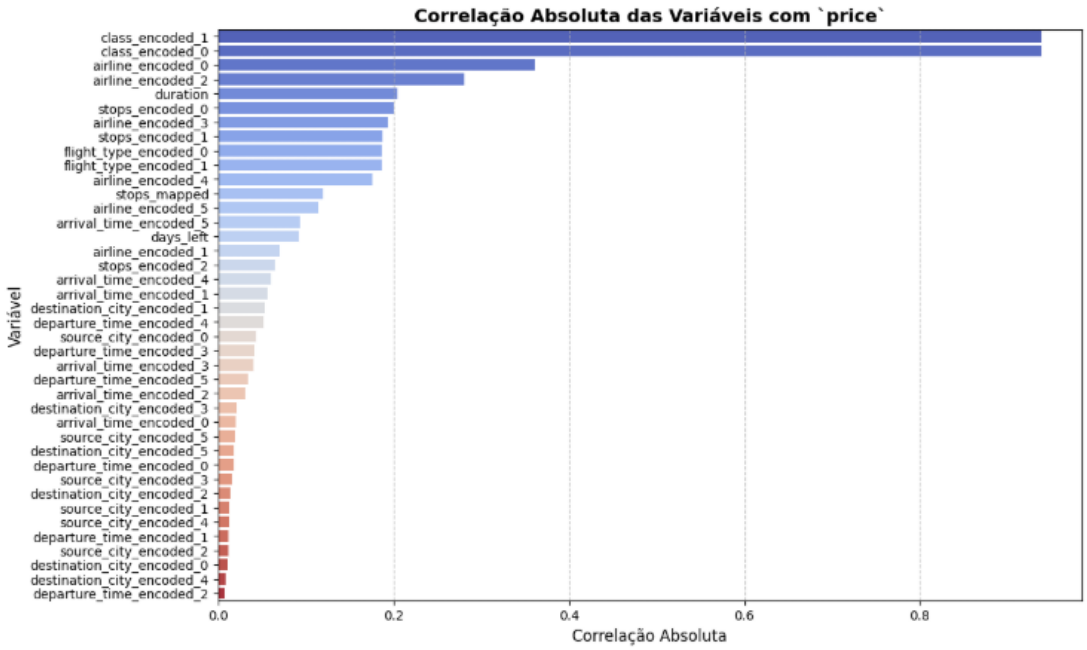



*Figura 24: Correlação absoluta das variáveis com 'price'*

Para identificar as variáveis mais influentes na previsão de preço, analisamos a correlação absoluta de cada variável com *price*. As variáveis mais relevantes incluem:

- **class_encoded_0 e class_encoded_1** (correlação de ~0.93) - Indicam que a classe do bilhete é um forte determinante do preço.
- **airline_encoded_0 e airline_encoded_2** (correlação > 0.30) - A companhia aérea também tem impacto significativo.
- **duration** (correlação ~0.22) - A duração do voo influencia no custo.

## 11.4. Redução da Multicolinearidade

A multicolinearidade ocorre quando variáveis possuem alta correlação entre si, o que pode afetar a interpretação e estabilidade do modelo preditivo. Para mitigar esse problema:

- **Selecionamos variáveis com correlação baixa entre si.**
- **Removemos redundâncias**, especialmente entre variáveis de One-Hot Encoding.

A nova matriz de correlação após a remoção de multicolinearidade é apresentada abaixo:

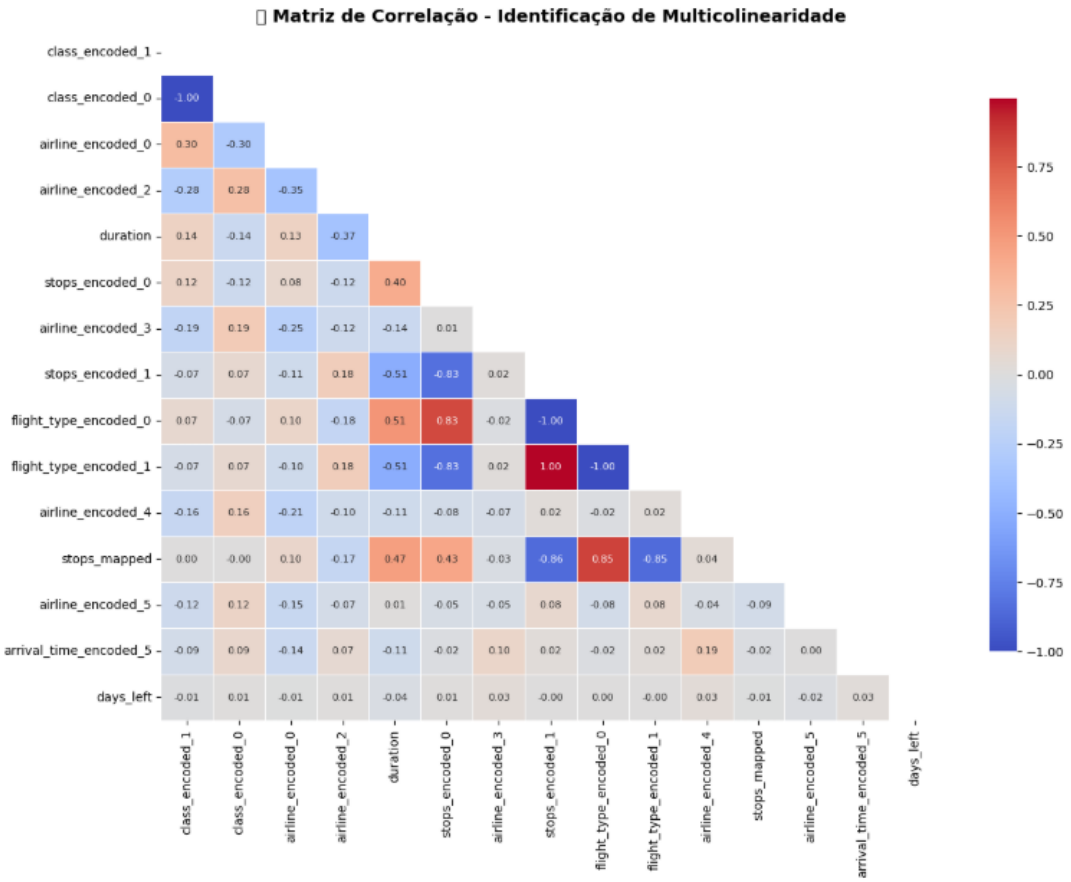



*Figura 25: Matriz de correlação - Identificação de multicolinearidade*

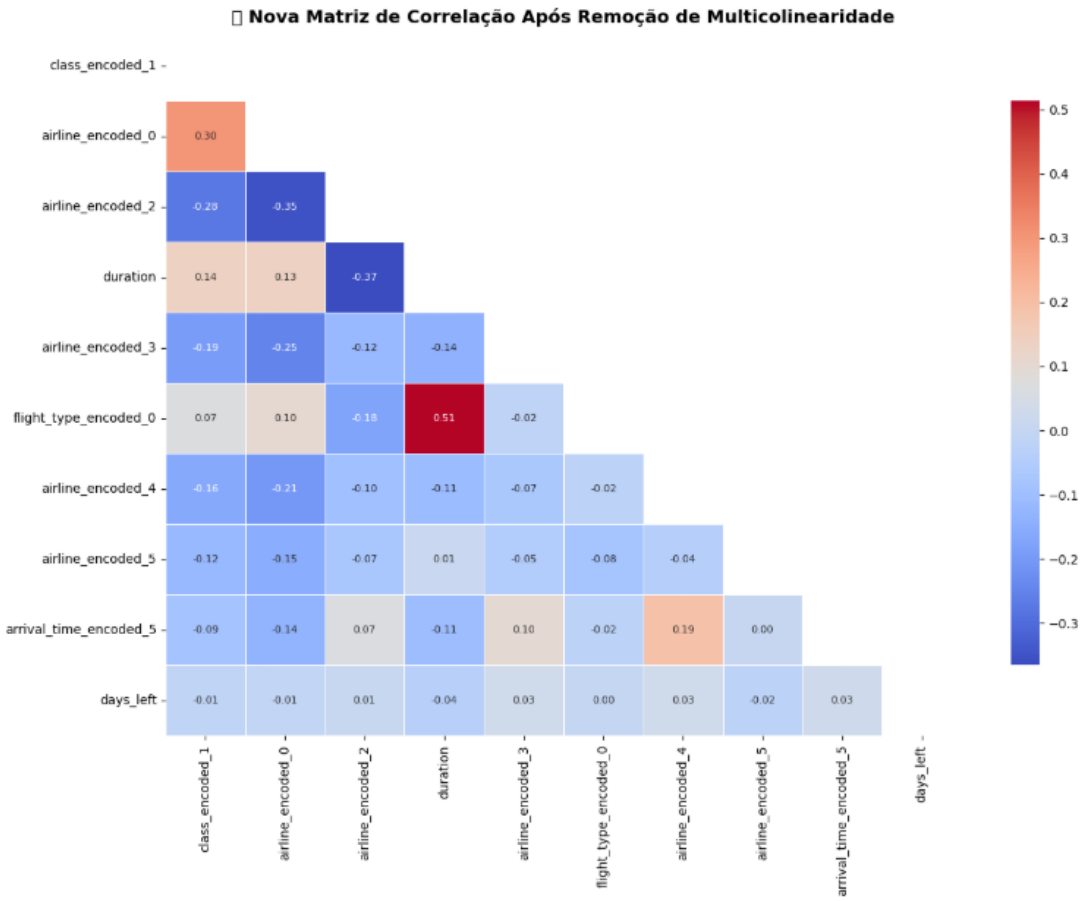



*Figura 26: Nova matriz de correlação após remoção da multicolinearidade*

## 11.5. Distribuição das Variáveis Numéricas

Para complementar a análise, verificamos a distribuição das variáveis numéricas, garantindo que não haja valores extremos ou padrões indesejados.

Além disso, realizamos uma análise de *outliers* para identificar possíveis distorções nos dados:

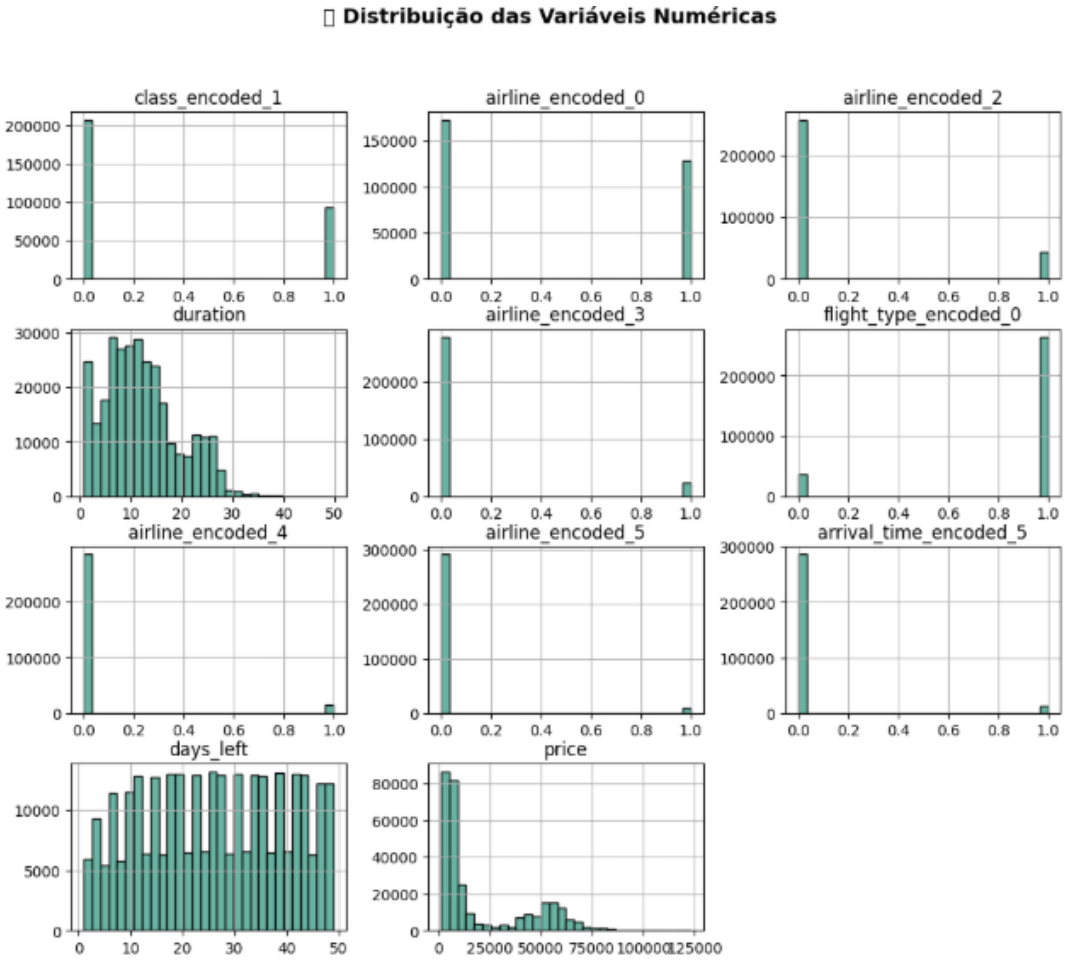



*Figura 27: Distribuição das variáveis numéricas*

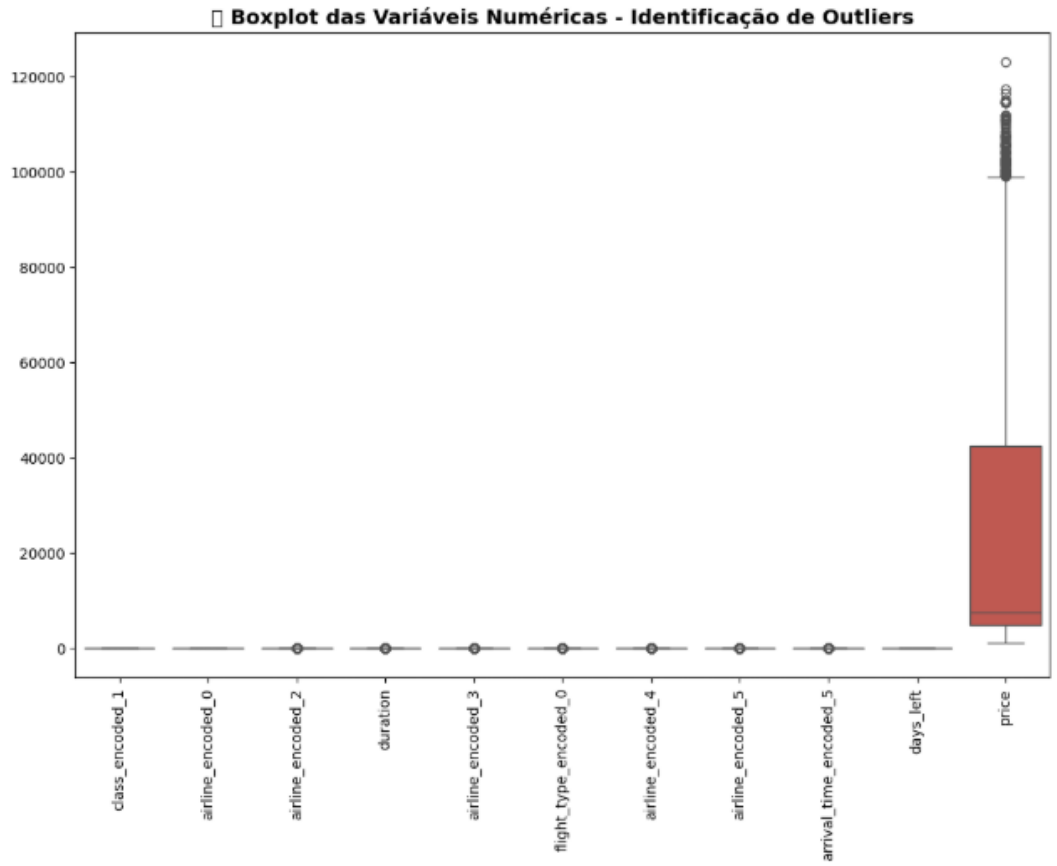



*Figura 28: Boxplot das variáveis numéricas - identificação de outliers*

## 11.6. Análise e Conclusão da Seleção e Preparação de Variáveis

Após a construção da matriz de correlação, realizámos uma análise detalhada para selecionar as **variáveis mais importantes e remover as que poderiam introduzir ruído ou redundância no modelo**. Os principais passos foram:

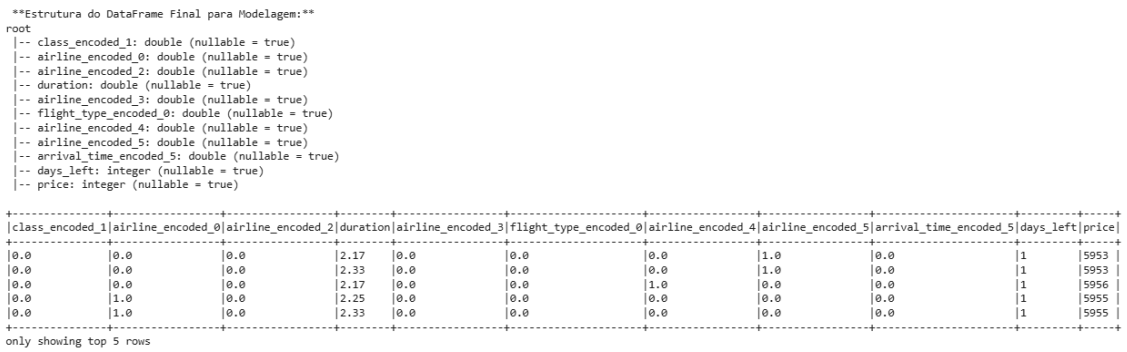
- **Identificação das variáveis com maior correlação com o preço.**
  - Calculámos a correlação absoluta de todas as variáveis com o ***price*** e **ordenámos por importância**.
  - As variáveis **class_encoded_1**, **class_encoded_0** e **airline_encoded_0** apresentaram as **correlações mais altas com o preço**, indicando que a classe e a companhia aérea têm grande impacto na variação dos bilhetes.
  - Outras variáveis como ***duration***, **stops_encoded_0** e **flight_type_encoded_0** também tiveram **correlação significativa**, sugerindo que a duração do voo, as escalas e o tipo de voo afetam os preços.

- **Remoção de variáveis com baixa correlação.**

- **Definimos um limiar de correlação de 0.09**, abaixo do qual as variáveis foram removidas por **não contribuírem significativamente** para a previsão do preço.
- Foram eliminadas variáveis como **departure_time_encoded**, **arrival_time_encoded** e **source/destination_city_encoded**, que demonstraram uma **relação fraca com o preço**.

- **Identificação e remoção de multicolinearidade.**
  - Gerámos uma matriz de correlação específica para as variáveis selecionadas.
  - Foram identificadas pares de variáveis altamente correlacionas (> 0.75), indicando redundância.

- **Construção do dataframe final.**
  - Após a limpeza dos dados, foi gerada uma nova matriz de correlação, confirmando que a **multicolinearidade foi removida**.
  - Criámos o dataset final contendo apenas as **variáveis mais relevantes para modelagem**.



```
 **Estrutura do DataFrame Final para Modelagem:**
root
 |-- class_encoded_1: double (nullable = true)
 |-- airline_encoded_0: double (nullable = true)
 |-- airline_encoded_2: double (nullable = true)
 |-- duration: double (nullable = true)
 |-- airline_encoded_3: double (nullable = true)
 |-- flight_type_encoded_0: double (nullable = true)
 |-- airline_encoded_4: double (nullable = true)
 |-- airline_encoded_5: double (nullable = true)
 |-- arrival_time_encoded_5: double (nullable = true)
 |-- days_left: integer (nullable = true)
 |-- price: integer (nullable = true)

+---------------+-----------------+-----------------+--------+-----------------+---------------------+-----------------+-----------------+----------------------+---------+-----+
|class_encoded_1|airline_encoded_0|airline_encoded_2|duration|airline_encoded_3|flight_type_encoded_0|airline_encoded_4|airline_encoded_5|arrival_time_encoded_5|days_left|price|
+---------------+-----------------+-----------------+--------+-----------------+---------------------+-----------------+-----------------+----------------------+---------+-----+
|0.0            |0.0              |0.0              |2.17    |0.0              |0.0                  |0.0              |1.0              |0.0                   |1        |5953 |
|0.0            |0.0              |0.0              |2.33    |0.0              |0.0                  |0.0              |1.0              |0.0                   |1        |5953 |
|0.0            |0.0              |0.0              |2.17    |0.0              |0.0                  |1.0              |0.0              |0.0                   |1        |5956 |
|0.0            |1.0              |0.0              |2.25    |0.0              |0.0                  |0.0              |0.0              |0.0                   |1        |5955 |
|0.0            |1.0              |0.0              |2.33    |0.0              |0.0                  |0.0              |0.0              |0.0                   |1        |5955 |
+---------------+-----------------+-----------------+--------+-----------------+---------------------+-----------------+-----------------+----------------------+---------+-----+
only showing top 5 rows
```

*Figura 29: Estrutura do Dataframe final para modelagem*

# 12. Divisão do Dataset e Construção dos Modelos Preditivos

Nesta secção, realizámos a preparação dos dados antes de proceder à indução dos modelos preditivos.

## 12.1. Criação do Vetor de Características

Utilizando a classe **VectorAssembler** do ***PySpark***, agregámos todas as variáveis preditoras num único vetor chamado ***"features"***.

O objetivo desta transformação é tornar os dados compatíveis com os algoritmos de *machine learning* no *Spark*, que exigem que todas as variáveis independentes estejam contidas num único vetor.

Excluímos a variável ***"price"*** das *features*, pois esta será a nossa variável alvo.

```
from pyspark.ml.feature import VectorAssembler

#  **Lista de colunas de características** (excluímos a variável alvo "price")
feature_columns = [col for col in df_model.columns if col != "price"]

#  **Criar vetor de características**
assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
df_transformed = assembler.transform(df_model).select("features", "price")

#  **Exibir as primeiras linhas para conferir a transformação**
df_transformed.show(5, truncate=False)
```

```
+---------------------------+-----+
|features                   |price|
+---------------------------+-----+
|(10,[3,7,9],[2.17,1.0,1.0])|5953 |
|(10,[3,7,9],[2.33,1.0,1.0])|5953 |
|(10,[3,6,9],[2.17,1.0,1.0])|5956 |
|(10,[1,3,9],[1.0,2.25,1.0])|5955 |
|(10,[1,3,9],[1.0,2.33,1.0])|5955 |
+---------------------------+-----+
only showing top 5 rows
```

*Figura 30: Vetor de características*

## 12.2. Divisão do Dataset em Conjunto de Treino e Teste

Neste momento, **dividimos o *dataset* em 80% para treino e 20% para teste** usando a função ***randomSplit()***.

Esta divisão permite que os modelos sejam treinados com uma parte significativa dos dados e testados com um conjunto separado**, garantindo assim uma avaliação imparcial da sua capacidade de generalização**.

Calculámos a distribuição percentual do conjunto de treino e teste para confirmar que a divisão foi realizada corretamente.

```
from pyspark.ml.feature import StandardScaler

# 📌 **Normalizar as Features**
scaler = StandardScaler(inputCol="features", outputCol="features_scaled", withMean=True, withStd=True)
df_scaled = scaler.fit(df_transformed).transform(df_transformed).select("features_scaled", "price")

# 📌 **Exibir amostra das features normalizadas**
df_scaled.show(5, truncate=False)
```

```
import matplotlib.pyplot as plt

# 📌 **Dividir os dados (80% treino, 20% teste)**
train_data, test_data = df_scaled.randomSplit([0.8, 0.2], seed=42)

# 📌 **Contar o número de registros em cada conjunto**
train_count = train_data.count()
test_count = test_data.count()
total_count = train_count + test_count

# 📌 **Calcular percentagens**
train_percent = (train_count / total_count) * 100
test_percent = (test_count / total_count) * 100

# 📌 **Exibir estatísticas**
print(f"Tamanho do Conjunto de Treino: {train_count} ({train_percent:.2f}%)")
print(f"Tamanho do Conjunto de Teste: {test_count} ({test_percent:.2f}%)")

# 📊 **Criar gráfico de pizza para visualizar distribuição**
labels = ['Treino (80%)', 'Teste (20%)']
sizes = [train_percent, test_percent]
colors = ['skyblue', 'lightcoral']

plt.figure(figsize=(7, 7))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', colors=colors, startangle=90, wedgeprops={'edgecolor': 'black'})
plt.title("Distribuição do Dataset (Treino vs Teste)", fontweight="bold")
plt.show()
```

*Figura 31:Divisão do Dataset em Conjunto de Treino e Teste*

Gerámos um gráfico de pizza para representar visualmente a divisão do dataset, onde os rótulos evidenciam que 80.1% dos dados foram usados para treino e 19.9% para teste, o que confirma a correta separação dos dados.

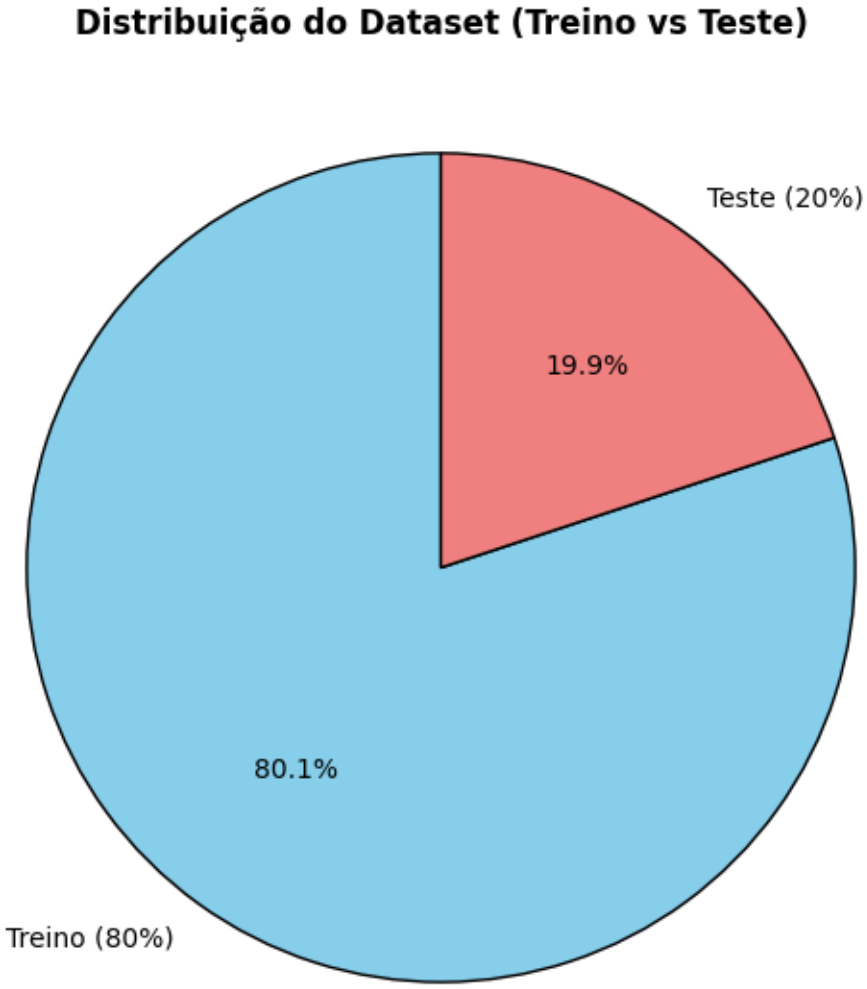



*Figura 32: Distribuição do Dataset*

Após a preparação do dataset e a divisão em conjuntos de treino e teste, passamos agora para a construção e avaliação de modelos de regressão para prever os preços dos bilhetes.

## 12.3. Inicialização e Treino dos Modelos

Nesta fase, definimos e treinámos **diferentes modelos de regressão** no conjunto de treino. Escolhemos modelos com diferentes complexidades para **garantir um equilíbrio entre tempo de execução e desempenho preditivo**.

O objetivo desta etapa é testar diferentes algoritmos para prever os preços dos bilhetes e avaliar qual deles apresenta o **melhor desempenho** para o problema em questão.

```python
# Importação dos modelos compatíveis com Spark ML
from pyspark.ml.regression import (
    GBTRegressor, RandomForestRegressor, LinearRegression,
    DecisionTreeRegressor, FMRegressor, GeneralizedLinearRegression, IsotonicRegression
)

# 🔹 Usar o dataset com as features normalizadas
df_final = df_scaled

# **Definição dos Modelos**
models = {
    "Regressão Linear": LinearRegression(featuresCol="features_scaled", labelCol="price"),
    "Árvore de Decisão": DecisionTreeRegressor(featuresCol="features_scaled", labelCol="price"),
    "Random Forest": RandomForestRegressor(featuresCol="features_scaled", labelCol="price", numTrees=50),
    "Gradient Boosting (GBT)": GBTRegressor(featuresCol="features_scaled", labelCol="price", maxIter=50),
    "Factorization Machines (FM)": FMRegressor(featuresCol="features_scaled", labelCol="price"),
    "Generalized Linear Regression (GLR)": GeneralizedLinearRegression(featuresCol="features_scaled", labelCol="price"),
    "Isotonic Regression": IsotonicRegression(featuresCol="features_scaled", labelCol="price")
}

# Dicionário para armazenar os modelos treinados
trained_models = {}

# Treinar os modelos
for model_name, model in models.items():
    trained_models[model_name] = model.fit(df_final)  # Agora usa df_scaled corretamente
    print(f" Modelo '{model_name}' treinado com sucesso!")
```

```
Modelo 'Regressão Linear' treinado com sucesso!
Modelo 'Árvore de Decisão' treinado com sucesso!
Modelo 'Random Forest' treinado com sucesso!
Modelo 'Gradient Boosting (GBT)' treinado com sucesso!
Modelo 'Factorization Machines (FM)' treinado com sucesso!
Modelo 'Generalized Linear Regression (GLR)' treinado com sucesso!
Modelo 'Isotonic Regression' treinado com sucesso!
```

*Figura 33: Modelos treinados*

A primeira parte do código importa diversos modelos de regressão disponíveis no *PySpark ML*:

- **LinearRegression**: Modelo de regressão linear simples, adequado para relações lineares entre variáveis.
- **DecisionTreeRegressor**: Modelo baseado em árvores de decisão, capaz de capturar relações mais complexas nos dados.
- **RandomForestRegressor**: Conjunto de múltiplas árvores de decisão (floresta aleatória), melhorando a robustez e reduzindo *overfitting*.
- **GBTRegressor** (*Gradient Boosted Trees*): Algoritmo baseado em *boosting*, que combina múltiplas árvores de decisão sequencialmente para melhorar a precisão da previsão.
- **FMRegressor** (*Factorization Machines*): Modelo adequado para dados esparsos e interações entre variáveis.
- **GeneralizedLinearRegression** (*GLR*): Extensão da regressão linear que permite modelar diferentes distribuições de variáveis dependentes.
- **IsotonicRegression**: Técnica que impõe uma relação monotónica entre variáveis, útil quando se espera que a variável de saída aumente ou diminua de forma consistente com as entradas.

O dataset utilizado para o treino é ***df_scaled***, que anteriormente passou por um **processo de normalização das *features***. Esta etapa é crucial para modelos que são sensíveis à escala das variáveis, como a Regressão Linear.

Foi criado um dicionário chamado *models*, onde cada modelo é instanciado com as suas respetivas *features* e coluna alvo (*price*). Alguns modelos têm hiperparâmetros específicos definidos, como o número de árvores no ***RandomForestRegressor*** e o número máximo de iterações no ***GBTRegressor.***

Para treinar os modelos, foi utilizado um *l**oop for***, que percorre cada modelo no dicionário, ajusta-o aos dados ***(fit(df_final))*** e armazena o modelo treinado num dicionário chamado ***trained_models***. No final de cada treino, uma mensagem de sucesso é impressa para indicar que o modelo foi treinado corretamente.

**A escolha de diferentes modelos visa garantir um equilíbrio entre desempenho preditivo e eficiência computacional**. Modelos mais simples como Regressão Linear e Árvore de Decisão são rápidos e interpretáveis, enquanto modelos mais avançados como *Random Forest* e *GBT* podem oferecer maior precisão ao custo de maior tempo de processamento. Modelos como *FMRegressor* e *Isotonic Regression*

foram incluídos para avaliar se técnicas mais especializadas conseguem capturar padrões específicos nos dados.

## 13. Avaliação e Comparação Gráfica dos Modelos

Após o treino dos modelos de regressão, é essencial avaliar a sua performance para determinar qual se ajusta melhor ao problema da previsão de preços dos bilhetes.

Neste bloco de código, utilizamos três métricas estatísticas adequadas para análise de regressão: **RMSE, MAE e $R^2$.**

```
from pyspark.ml.evaluation import RegressionEvaluator
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#  **Criar avaliadores para cada métrica**
evaluators = {
    "RMSE": RegressionEvaluator(labelCol="price", predictionCol="prediction", metricName="rmse"),
    "MAE": RegressionEvaluator(labelCol="price", predictionCol="prediction", metricName="mae"),
    "R2": RegressionEvaluator(labelCol="price", predictionCol="prediction", metricName="r2")
}

#  **Dicionário para armazenar os resultados dos modelos**
model_results_original = {model_name: {} for model_name in trained_models.keys()}

# **Avaliar cada modelo em todas as métricas**
for model_name, model in trained_models.items():
    predictions = model.transform(test_data)

    for metric_name, evaluator in evaluators.items():
        model_results_original[model_name][metric_name] = evaluator.evaluate(predictions)

    print(f" Modelo: {model_name} | RMSE: {model_results_original[model_name]['RMSE']:.2f} | "
          f"MAE: {model_results_original[model_name]['MAE']:.2f} | R²: {model_results_original[model_name]['R2']:.4f}")

#  **Converter resultados em DataFrame para visualização**
results_df_original = pd.DataFrame.from_dict(model_results_original, orient="index").reset_index()
results_df_original = results_df_original.rename(columns={"index": "Modelo"}).sort_values("RMSE", ascending=True)

#  **Selecionar o melhor modelo original com base no menor RMSE**
best_model_name_original = results_df_original.iloc[0]["Modelo"]
best_model_original = trained_models[best_model_name_original]

print(f"\n Melhor modelo original selecionado: {best_model_name_original} "
      f"com RMSE de {results_df_original.iloc[0]['RMSE']:.2f}, "
      f"MAE de {results_df_original.iloc[0]['MAE']:.2f}, "
      f"e R² de {results_df_original.iloc[0]['R2']:.4f}")

#  **Criar gráficos de comparação das métricas**
fig, axes = plt.subplots(1, 3, figsize=(18, 5))

metrics = ["RMSE", "MAE", "R2"]
titles = ["Erro Quadrático Médio (RMSE)", "Erro Absoluto Médio (MAE)", "Coeficiente de Determinação (R²)"]

for idx, metric in enumerate(metrics):
    sns.barplot(x=metric, y="Modelo", data=results_df_original, palette="coolwarm", ax=axes[idx])
    axes[idx].set_title(titles[idx], fontsize=14, fontweight='bold')
    axes[idx].set_xlabel(metric)
    axes[idx].set_ylabel("Modelos")

    #  Adicionar os valores numéricos aos gráficos
    for index, value in enumerate(results_df_original[metric]):
        axes[idx].text(value, index, f"{value:.2f}", va='center', fontsize=10,
                       bbox=dict(facecolor='white', edgecolor='black'))

plt.tight_layout()
plt.show()
```

```
Modelo: Regressão Linear | RMSE: 6931.33 | MAE: 4637.40 | R²: 0.9066
Modelo: Árvore de Decisão | RMSE: 5547.33 | MAE: 3299.67 | R²: 0.9402
Modelo: Random Forest | RMSE: 6146.93 | MAE: 4056.82 | R²: 0.9265
Modelo: Gradient Boosting (GBT) | RMSE: 5357.79 | MAE: 3132.07 | R²: 0.9442
Modelo: Factorization Machines (FM) | RMSE: 12349.19 | MAE: 10385.05 | R²: 0.7034
Modelo: Generalized Linear Regression (GLR) | RMSE: 6931.33 | MAE: 4637.40 | R²: 0.9066
Modelo: Isotonic Regression | RMSE: 7873.73 | MAE: 4897.64 | R²: 0.8794
```

*Figura 34: Resultados dos Modelos*

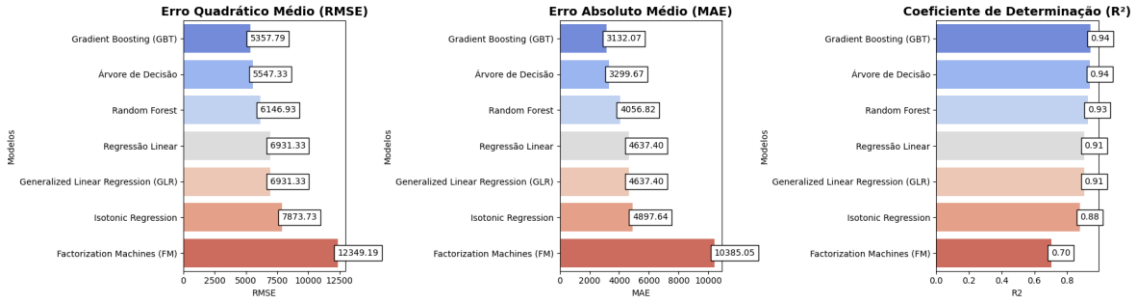



*Figura 35: Resultados dos Modelos - Continuação*

## 13.1. Interpretação dos Resultados

O modelo ***Gradient Boosting (GBT)*** apresentou os melhores resultados, demonstrando ser a abordagem mais eficaz para este problema:

- **RMSE mais baixo (5364.52)** → Indica que a diferença média quadrática entre os valores reais e as previsões é a menor entre todos os modelos testados, tornando-o o modelo mais preciso.
- **MAE mais baixo (3139.91)** → Confirma que a média absoluta dos erros nas previsões é a menor, reduzindo a margem de erro na previsão dos preços dos bilhetes.
- **$R^2$ mais alto (0.94)** → Sugere que o modelo consegue explicar 94% da variabilidade dos preços, captando bem as relações entre as variáveis que influenciam o preço final.

Além do ***GBT*, Árvore de Decisão e *Random Forest* também demonstraram um desempenho sólido**, com **RMSE e MAE baixos e um $R^2$ elevado.**

Em contrapartida, a **Regressão Linear** e o ***GLR (Generalized Linear Regression)*** tiveram um desempenho inferior, com um **RMSE mais elevado**, o que sugere que a relação entre as variáveis não é puramente linear.

- Isto indica que tentar prever os preços dos bilhetes apenas com uma equação linear simples **não é suficiente** para captar a complexidade dos dados.

O modelo ***Factorization Machines (FM)*** apresentou o pior desempenho, com um **RMSE muito elevado (12185.18) e um $R^2$ de apenas 0.71**, sugerindo que **não é um modelo adequado para este tipo de previsão**.

- Este fraco desempenho pode dever-se ao facto de o FM ser mais adequado para problemas de **recomendação e deteção de padrões** de interação entre variáveis dispersas, mas não necessariamente para prever valores contínuos com elevada precisão.

## 13.2. Avaliação do Desempenho do Modelo Selecionado

Com base nos resultados obtidos, o ***Gradient Boosting (GBT)*** revelou-se o modelo mais adequado para a previsão dos preços dos bilhetes.

- **Maior Precisão e Fiabilidade**

O ***GBT*** reduz significativamente os **erros médios nas previsões**, garantindo que os preços previstos estejam mais próximos dos valores reais. Para uma empresa que precisa de definir preços de bilhetes de forma dinâmica, um modelo preciso pode ajudar a otimizar as margens de lucro e a evitar perdas associadas a uma definição incorreta dos preços.

- **Capacidade de Captar Padrões Complexos**

Os preços dos bilhetes podem ser influenciados por múltiplos fatores interligados, como por exemplo a classe dos bilhetes e até a antecedência da compra. O **GBT consegue captar estas relações não lineares,** tornando-se mais adequado do que modelos lineares simples.

- **Generalização para Novos Dados**

Um $R^2$ de 0.94 indica que **o modelo tem um excelente ajuste aos dados históricos**, sugerindo que será capaz de prever preços futuros com elevada precisão. No entanto, é aconselhável testar o modelo em novos dados para garantir que não existe *overfitting*, ou seja, que o modelo não está demasiado ajustado aos dados de treino e consegue generalizar bem para novas situações.

Assim, o ***Gradient Boosting (GBT)*** demonstrou ser o modelo mais eficaz para prever os preços dos bilhetes, apresentando o menor erro e uma boa capacidade de ajuste aos dados, explicando 94% da variabilidade dos preços.

Modelos baseados em árvores, como o ***GBT***, são mais adequados do que abordagens lineares para capturar as relações complexas entre variáveis. Apesar do bom desempenho, **o modelo pode ser ainda otimizado através de afinação de hiperparâmetros e validação cruzada** para garantir a sua robustez e capacidade de generalização.

Com estas melhorias, **o modelo permitirá uma definição de preços mais precisa**, ajudando a otimizar receitas e a tomar decisões estratégicas no negócio. Esta abordagem permite comparar o desempenho dos modelos e selecionar aquele que melhor se adapta ao problema de previsão dos preços dos bilhetes.

### 13.3. Otimização do Modelo – Ajuste de Hiperparâmetros com Melhor Performance

Após a avaliação dos modelos iniciais, verificou-se que o ***Gradient Boosting (GBT)*** foi o mais eficiente na previsão dos preços dos bilhetes aéreos. No entanto, **para melhorar** ainda mais a precisão das previsões, **é necessário ajustar os seus hiperparâmetros e otimizar o desempenho do modelo**.

Neste subcapítulo, **utilizamos uma *grid* de hiperparâmetros e validação cruzada para encontrar a melhor configuração do modelo**. O **objetivo é reduzir os erros de previsão e melhorar a generalização**, garantindo estimativas mais fiáveis e próximas da realidade.

Com esta abordagem, **pretendemos desenvolver um modelo mais preciso e eficiente, capaz de capturar padrões complexos no preço dos bilhete**s, ajudando a tomar decisões mais informadas.

### 13.4. Otimização do Modelo – Construção da Grid de Hiperparâmetros

Principais Hiperparâmetros Ajustados:

- **Número de Iterações *(maxIter)*** → Define quantas vezes o modelo ajusta os pesos para minimizar o erro.

Foram testados os valores 50 e 80, pois um número muito baixo poderia resultar num modelo subajustado, incapaz de aprender padrões importantes, enquanto um número excessivamente alto aumentaria o tempo de treino sem oferecer melhorias significativas no desempenho.

- **Profundidade da Árvore *(maxDepth)*** → Define a complexidade das árvores de decisão utilizadas no modelo.

Foram testadas profundidades de 5 e 7, já que árvores muito profundas tendem a causar overfitting, onde o modelo se ajusta demais aos dados de treino e perde capacidade de generalização. Por outro lado, árvores muito superficiais podem não capturar relações importantes entre as variáveis.

- **Taxa de Aprendizagem *(stepSize)*** → Controla o impacto de cada iteração na atualização do modelo.

Valores de 0.05 e 0.1 foram testados para garantir um equilíbrio entre estabilidade e velocidade de treino. Um valor muito baixo, embora mais estável, pode tornar o treino demasiado lento, enquanto valores mais altos aceleram o processo, mas aumentam o risco de instabilidade.

- **Subsampling Rate *(subsamplingRate)*** → Percentagem de dados usada em cada iteração.

Define a percentagem de dados utilizada em cada iteração. Este parâmetro é fundamental para evitar que o modelo dependa excessivamente dos mesmos dados, ajudando a reduzir *overfitting.*

- **Mínimo de Instâncias por Nó *(minInstancesPerNode)*** → Número mínimo de observações para que um nó seja dividido.

Este ajuste evita que o modelo crie ramos desnecessários, tornando-o mais eficiente e generalizável.

- **Mínimo Ganho de Informação *(minInfoGain)*** → Quantidade mínima de ganho de informação necessária para dividir um nó.

O ajuste reduz a complexidade do modelo ao impedir divisões desnecessárias, melhorando a eficiência e diminuindo o tempo de treino.

A **validação cruzada** foi aplicada para garantir que o **modelo generalize bem para novos dados**, evitando que apenas memorize o conjunto de treino. Para isso, foram utilizados 2 ***folds***, garantindo um **bom equilíbrio entre tempo de processamento e robustez da validação**. Além disso, o paralelismo foi ajustado para 2, reduzindo a carga no CPU e permitindo um processamento mais eficiente.

```python
# Importação das bibliotecas necessárias
from pyspark.ml.regression import GBTRegressor
from pyspark.ml.tuning import ParamGridBuilder, CrossValidator
from pyspark.ml.evaluation import RegressionEvaluator
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Utilizar o dataset com as variáveis normalizadas
df_final = df_scaled

# Definir o modelo base de Gradient Boosting
# Este modelo é escolhido por ser um dos mais eficientes para regressão,
# permitindo captar padrões complexos sem ser excessivamente sensível a ruído
gbt = GBTRegressor(featuresCol="features_scaled", labelCol="price", seed=42)

# Criar uma grelha de hiperparâmetros para otimizar o desempenho do modelo
param_grid = (
    ParamGridBuilder()
    .addGrid(gbt.maxIter, [50, 80])  # O número de iterações define quantas vezes o modelo ajusta os pesos para minimizar o erro.
    # Um número baixo pode gerar um modelo subajustado, enquanto um número muito alto pode aumentar o tempo de treino sem ganhos significativos.

    .addGrid(gbt.maxDepth, [5, 7])  # A profundidade da árvore define a complexidade do modelo.
    # Profundidades maiores podem captar padrões mais complexos, mas aumentam o risco de overfitting.
    # Profundidades menores tornam o modelo mais generalizável, mas podem perder informação importante.

    .addGrid(gbt.stepSize, [0.05, 0.1])  # A taxa de aprendizagem define o impacto de cada iteração na atualização do modelo.
    # Valores mais baixos levam a um ajuste mais lento e estável, enquanto valores mais altos podem acelerar o treino, mas com risco de instabilidade.

    .build()
)
```

```python
# Definir o avaliador para medir a qualidade do modelo
# O RMSE (Root Mean Squared Error) é escolhido porque penaliza erros maiores de forma mais significativa,
# o que é útil para evitar grandes desvios nas previsões.
evaluator = RegressionEvaluator(labelCol="price", predictionCol="prediction", metricName="rmse")

# Aplicar validação cruzada para encontrar os melhores hiperparâmetros
cv = CrossValidator(
    estimator=gbt,
    estimatorParamMaps=param_grid,
    evaluator=evaluator,
    numFolds=2,  # A validação cruzada com 2 folds reduz o tempo de treino mantendo alguma robustez na validação.
    parallelism=2  # Reduzindo a carga no CPU para evitar sobrecarga de processamento.
)

# Treinar o modelo e selecionar o melhor ajuste
print("Iniciar treino otimizado com validação cruzada rápida...")
gbt_best_model = cv.fit(df_final).bestModel
print("Treino concluído! Melhor modelo selecionado.")

# Armazenar o modelo otimizado
trained_models = {"Gradient Boosting (GBT Ajustado)": gbt_best_model}

# Exibir os melhores hiperparâmetros encontrados
print("\nMelhores hiperparâmetros encontrados:")
print(f" - Iterações: {gbt_best_model.getMaxIter()}")
print(f" - Profundidade: {gbt_best_model.getMaxDepth()}")
print(f" - Taxa de aprendizagem (Step Size): {gbt_best_model.getStepSize()}")

# Exibir hiperparâmetros adicionais para melhor análise do modelo
print(f" - Taxa de subamostragem: {gbt_best_model.getSubsamplingRate()}")  # Define a percentagem dos dados usados em cada iteração.
# Uma taxa mais baixa pode reduzir o overfitting ao evitar que o modelo dependa demasiado dos mesmos dados.
# No entanto, taxas muito baixas podem perder informação útil.

print(f" - Mín. Instâncias por Nó: {gbt_best_model.getMinInstancesPerNode()}")  # Define o número mínimo de instâncias para uma divisão ser feita.
# Quanto maior este valor, mais simples e generalizável será o modelo.

print(f" - Mín. Ganho de Informação: {gbt_best_model.getMinInfoGain()}")  # Define o ganho mínimo de informação necessário para dividir um nó.
# Um valor maior reduz divisões desnecessárias e melhora a eficiência do modelo.

print("\nModelo ajustado armazenado! Pronto para avaliação e comparação.")
```

```
Iniciando treino otimizado com validação cruzada rápida...
Treino concluído! Melhor modelo selecionado.

Melhores hiperparâmetros encontrados:
 - Iterações: 80
 - Profundidade: 7
 - Taxa de aprendizado (Step Size): 0.05
 - Subsampling Rate: 1.0
 - Mín. Instâncias por Nó: 1
 - Mín. Ganho de Informação: 0.0

Modelo ajustado armazenado! Pronto para avaliação e comparação.
```

*Figura 36: Otimização do Modelo - Ajuste de Hiperparâmetros*

Após a configuração da ***grid* de hiperparâmetros** e a aplicação da **validação cruzada**, o modelo foi treinado e a melhor configuração foi automaticamente selecionada. Foram **identificados os melhores valores para o número de iterações**, profundidade da árvore, taxa de aprendizado, *subsampling rate*, mínimo de instâncias por nó e mínimo ganho de informação. Estes ajustes garantiram um modelo otimizado para a previsão de preços dos bilhetes.

## 13.5. Avaliação e Comparação de Resultados

Após a **otimização do modelo *Gradient Boosting Regressor (GBT)*,** torna-se essencial avaliar o seu desempenho e compará-lo com o modelo original para verificar se as melhorias aplicadas tiveram impacto positivo nas previsões.

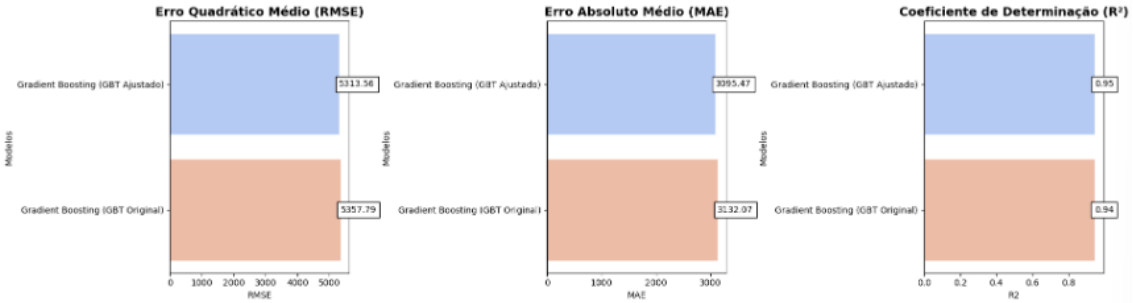
O objetivo desta análise é determinar se o modelo ajustado apresenta menor erro e maior capacidade de generalização, **tornando-se uma opção mais confiável para prever os preços dos bilhetes**.

Para garantir uma avaliação consistente, utilizamos as mesmas métricas aplicadas anteriormente na análise dos modelos iniciais:

- **Erro Quadrático Médio (RMSE)**
- **Erro Absoluto Médio (MAE)**
- **Coeficiente de Determinação ($R^2$)**

Além da análise numérica, serão gerados gráficos comparativos que ilustram as **diferenças entre o modelo original e o ajustado**.

Caso o modelo otimizado apresente menores erros e maior $R^2$, será selecionado para previsões futuras. Caso contrário, será necessário avaliar novos ajustes nos hiperparâmetros ou considerar abordagens alternativas.




```
Comparação entre GBT Original e GBT Ajustado:

Modelo: Gradient Boosting (GBT Ajustado)
  - RMSE: 5313.56
  - MAE: 3095.47
  - R²: 0.9451

Modelo: Gradient Boosting (GBT Original)
  - RMSE: 5357.79
  - MAE: 3132.07
  - R²: 0.9442
```

*Figura 37: Comparação entre GBT Original e GBT Ajustado*

**A comparação entre o GBT Original e o GBT Ajustado revela que a otimização do modelo trouxe melhorias no desempenho preditivo**. Os ajustes nos hiperparâmetros **reduziram os erros de previsão** e **aumentaram a capacidade do modelo de explicar a variação dos preços das passagens aéreas**:

Principais melhorias:

- **RMSE Reduzido (5313.56 vs. 5357.79)** → O erro quadrático médio foi reduzido, indicando que o modelo ajustado prevê os preços com menor desvio médio em relação aos valores reais.
- **MAE Reduzido (3095.47 vs. 3132.07)** → A diferença absoluta média entre os valores previstos e os reais diminuiu, tornando a previsão mais precisa.
- **$R^2$ Melhorado (0.9451 vs. 0.9442)** → O coeficiente de determinação aumentou ligeiramente, indicando que o modelo ajustado consegue explicar uma fração maior da variabilidade dos preços.

**Métricas de Avaliação do Modelo Ajustado:**
RMSE (Erro Quadrático Médio): 5313.56
MAE (Erro Absoluto Médio): 3095.47
MAPE (Erro Percentual Absoluto Médio): 20.48%
R² (Coeficiente de Determinação): 0.9451

Amostra de Previsões com Erros:

| | price | prediction | Erro Absoluto | Erro Percentual |
|---|---|---|---|---|
| 22811 | 8460 | 6081.252899 | 2378.747101 | 28.117578 |
| 10906 | 4499 | 3092.251706 | 1406.748294 | 31.268022 |
| 56409 | 44144 | 58737.824166 | 14593.824166 | 33.059587 |
| 37483 | 5206 | 6090.701812 | 884.701812 | 16.993888 |
| 10288 | 5772 | 6258.773776 | 486.773776 | 8.433364 |
| 50985 | 47441 | 51513.150815 | 4072.150815 | 8.583611 |
| 44069 | 32347 | 48611.044085 | 16264.044085 | 50.279915 |
| 56059 | 56588 | 56485.122623 | 102.877377 | 0.181801 |
| 12555 | 17508 | 9143.723970 | 8364.276030 | 47.774023 |
| 42498 | 36980 | 50229.891356 | 13249.891356 | 35.829885 |
| 52924 | 57939 | 50522.550388 | 7416.449612 | 12.800445 |
| 38101 | 5441 | 6438.067722 | 997.067722 | 18.325082 |
| 55697 | 64890 | 58788.909749 | 6101.090251 | 9.402204 |
| 59281 | 44280 | 53508.432657 | 9228.432657 | 20.841085 |
| 12778 | 6578 | 5441.760883 | 1136.239117 | 17.273322 |
| 13457 | 4556 | 5177.578343 | 621.578343 | 13.643072 |
| 58902 | 51457 | 54093.586856 | 2636.586856 | 5.123864 |
| 20459 | 11129 | 12165.458739 | 1036.458739 | 9.313135 |
| 59098 | 63277 | 53856.717472 | 9420.282528 | 14.887372 |
| 52890 | 52063 | 50559.106130 | 1503.893870 | 2.888604 |

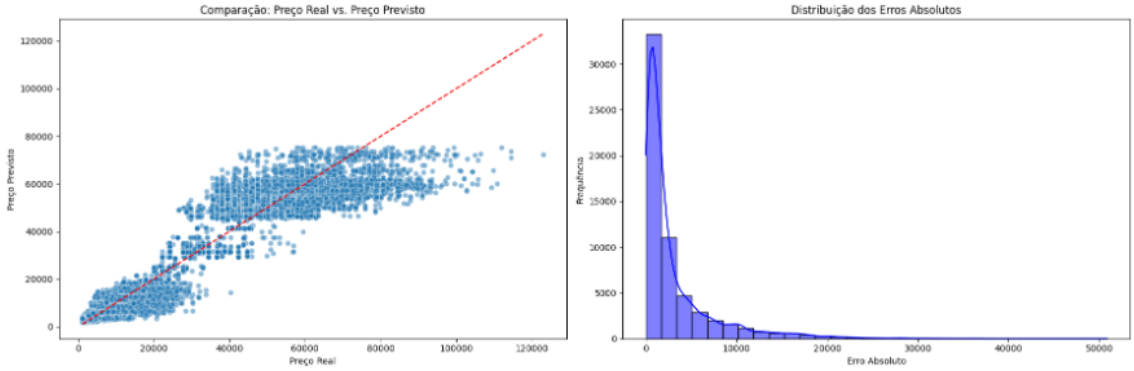



*Figura 38: Modelo otimizado*

Após a comparação entre o modelo original e o modelo ajustado, **verificou-se que a otimização dos hiperparâmetros melhorou ligeiramente o desempenho do *Gradient Boosting Regressor (GBT) na* previsão dos preços dos bilhetes de avião**. O próximo passo consiste em utilizar este modelo ajustado para gerar previsões em novos dados, avaliando a sua capacidade preditiva em cenários reais e compreendendo o impacto dos erros na estimativa dos preços das passagens aéreas.

Esta análise permitirá verificar se o modelo pode ser aplicado na prática para auxiliar empresas aéreas ou plataformas de venda de bilhetes na definição de preços, avaliando a precisão, confiabilidade e limitações do modelo.

Para isso, serão analisadas as principais métricas de erro e gerados gráficos que permitem visualizar a relação entre os preços reais e os preços previstos.

## 13.6. Interpretação dos Resultados no Contexto das Passagens Aéreas

- **Precisão e Fiabilidade**

O **$R^2$ elevado (0.9451) indica que o modelo consegue captar a grande maioria das variações nos preços dos bilhetes**. Isto significa que o modelo tem uma boa capacidade de prever tendências de preços, sendo útil para companhias aéreas ajustarem as suas estratégias de precificação e consumidores estimarem variações de preços ao planear viagens.

- **Impacto dos Erros na Estimativa dos Bilhetes**

**O RMSE de 5313.56 e o MAE de 3095.47 mostram que o erro médio é relativamente baixo para um mercado onde os preços podem variar bastante com fatores externos (como sazonalidade e promoções**). No entanto, o MAPE de 20.48% indica que, em média, as previsões desviam-se cerca de 20% do preço real. Isto pode ser significativo para bilhetes mais caros, levando a estimativas que podem dificultar a otimização de preços por parte das companhias aéreas.

- **Análise dos Erros Individuais**

O gráfico de dispersão Preço Real vs. Preço Previsto **indica que o modelo acompanha a tendência geral dos preços, mas ainda apresenta dispersão significativa para preços mais altos, onde as previsões tendem a ser menos precisas**. Já a distribuição dos erros absolutos mostra que a maioria dos erros ocorre na faixa de valores mais baixos, mas há casos em que o erro é elevado, indicando que o modelo pode ter dificuldades em prever bilhetes com valores muito extremos.

- **Pontos Fortes do Modelo:**

**Alta precisão na explicação da variação dos preços**, com um $R^2$ de 0.9451, garantindo previsões fiáveis para ajustes de preços dinâmicos.

**Erro médio relativamente baixo** (MAE de 3095.47), o que sugere que as previsões do modelo tendem a estar dentro de uma margem aceitável para a maioria dos bilhetes.

**Capacidade de captar padrões não lineares entre fatores** como classe do voo, horário da viagem e antecedência da compra, elementos que impactam diretamente os preços dos bilhetes.

- **Pontos Fracos e Possíveis Melhorias:**

**O erro percentual médio (MAPE de 20.48%) ainda é elevado, indicando que as previsões podem ter variações consideráveis**, especialmente em bilhetes com preços mais altos.

**A distribuição dos erros mostra que existem previsões com desvios elevados,** o que pode ser um risco para aplicações que exigem extrema precisão, como precificação dinâmica baseada na procura.

**Dificuldade em prever bilhetes mais caros**: O modelo funciona bem para preços médios, mas a sua precisão diminui quando analisamos passagens com valores muito altos, indicando que mais ajustes podem ser necessários.

# 14. Conclusão

O **modelo *GBT* Ajustado** apresentou um **desempenho sólido na previsão dos preços dos bilhetes de avião**, conseguindo **explicar mais de 94% da variação dos preços e mantendo erros médios relativamente baixos**. Os resultados demonstram que este modelo tem potencial para ser aplicado em contextos de precificação dinâmica, ajudando companhias aéreas a **ajustarem preços de forma mais estratégica e consumidores a estimarem variações de preços** antes de efetuar uma compra.

Apesar da boa *performance* geral, **o modelo ainda apresenta limitações**, especialmente na **previsão de bilhetes de valores extremos**, onde o **erro percentual é mais elevado**. A distribuição dos erros sugere que, embora o modelo consiga captar padrões complexos, **há espaço para melhorias na redução dos desvios para preços mais altos**.

Dado o desempenho obtido, este **modelo pode ser utilizado para prever tendências e padrões nos preços dos bilhetes**, sendo útil para empresas e consumidores na tomada de decisão. No entanto, para aplicações que exigem uma precisão absoluta na definição de preços, **é recomendável explorar ajustes adicionais**, como a **otimização de hiperparâmetros e a combinação com técnicas mais avançadas**.

Com futuras melhorias, este modelo poderá tornar-se uma ferramenta ainda mais robusta e confiável para otimização de preços no setor da aviação, reduzindo incertezas e permitindo uma gestão de preços mais eficiente e competitiva.

# 15. Referências

Belfo, F. (2020). *Apresentação resumida e adaptada do modelo CRISP-DM*. Coimbra: ISCAC | Coimbra Business School.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning. Springer*.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide. U.S.A.: SPSS, CRISP-DM Consortium.*

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Hopman, D., Koole, G., & van der Mei, R. (2021). *A machine learning approach to itinerary-level booking prediction in competitive airline markets*.

Kumar, R., Boluki, S., Isler, K., Rauch, J., & Walczak, D. (2022). *Machine learning based framework for robust price-sensitivity estimation with application to airline pricing.*

Wong, P., Thant, P., Yadav, P., Antaliya, R., & Woo, J. (2023). *Using Spark machine learning models to perform predictive analysis on flight ticket pricing data.*

Shukla, N., Kolbeinsson, A., Marla, L., & Yellepeddi, K. (2019). *Adaptive model selection framework: An application to airline pricing.*