PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

ANDRÉ LUIZ DE PAULA RODRIGUES GONÇALVES

Previsão de Vendas das lojas Rossmann com utilização de técnicas de Machine Learning

Belo Horizonte
2024

ANDRÉ LUIZ DE PAULA RODRIGUES GONÇALVES

Previsão de Vendas das lojas Rossmann com utilização de técnicas de Machine Learning

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte
2023

SUMÁRIO

1. Introdução

1.1. Contextualização

Gerir uma organização requer conhecimento sólido do mercado no qual ela está inserida e, além disso, ter a capacidade de antecipar as demandas dos clientes é um diferencial para as organizações que buscam planejamento estratégico. Neste contexto, uma das ferramentas que podem ser utilizadas é a previsão das vendas, que serve como base para a empresa estimar o faturamento futuro.

Previsão de vendas é um cálculo aproximado do volume de vendas realizados por uma empresa, dentro de um determinado período, essas informações são coletadas a partir de análises de fatores internos, como histórico de vendas e fatores externos, como as tendências do mercado.

Oliveira, 2021 diz que é com base na coleta dos dados que a empresa pode prosseguir para um planejamento eficiente visando ter um sistema de produção contínuo transformando insumos em produtos ou serviços acabados com intuito de satisfazer as necessidades de consumos dos clientes.

Ter uma previsão bem definida auxilia no controle de um orçamento empresarial e ajuda a empresa a entender fatores de mercado e concorrência. As projeções proporcionam aos gestores insights valiosos na tomada de decisões e otimiza o estoque e a produção, mas também influencia diretamente o desempenho financeiro e a competitividade no mercado.

Além dos benefícios citados, as previsões auxiliam nas estratégias de marketing, pois as empresas podem ajustar suas campanhas promocionais, lançamentos de produtos e estratégias de preços para maximizar o impacto e a eficácia. Isso não apenas impulsiona as vendas, mas também fortalece a fidelidade do cliente ao atender às suas expectativas de forma proativa.

Em resumo, as previsões de venda são ferramentas essenciais para a gestão eficiente do comércio, proporcionando uma visão estratégica que orienta nas

decisões operacionais. A capacidade de antecipar as necessidades do mercado confere uma vantagem competitiva significativa, permitindo que as empresas se adaptem rapidamente às mudanças e alcancem o sucesso sustentável a longo prazo.

O trabalho em questão visa analisar as informações históricas de vendas das lojas Rossmann. A Rossmann é uma grande rede de drogarias que opera principalmente na Europa. Fundada em 1972 na Alemanha, a Rossmann expandiu-se ao longo dos anos e tornou-se uma das principais cadeias de drogarias no continente. A empresa oferece uma ampla variedade de produtos, incluindo itens de cuidados pessoais, beleza, saúde, higiene e produtos para o lar.

Diante da importância das previsões de vendas, buscarei insights através de algoritmos de Machine Learning com objetivo de encontrar um modelo que realize as previsões das vendas baseados nos dados históricos das lojas Rossmann.

1.2. O problema proposto

O problema em questão visa entender, por meio de dados da Rossmann, os padrões e fatores que influenciam as vendas em suas lojas. Isso é fundamental para a tomada de decisões estratégicas, como a otimização de promoções, alocação eficiente de recursos e o desenvolvimento de estratégias de marketing direcionadas. A previsão de vendas utilizando modelos de Machine Learning também é um componente essencial, fornecendo à empresa novas percepções para o planejamento futuro.

(Why?) Por que esse problema é importante?

O problema é crucial para a Rossmann, pois através da análise de previsão das vendas e os seus fatores de influência, é possível verificar as vendas futuras e suas melhores estratégias para gestão de estoques, panejamento financeiro, produção e logística, marketing e promoções, entre outros.

(Who?) De quem são os dados analisados? De um governo? Um ministério ou secretaria? Dados de clientes?

Os dados analisados são da empresa Rossman. O conjunto de dados inclui informações sobre as vendas de 1.115 lojas ao longo de um período de 2013 a 2015, abrangendo detalhes como clientes, promoções, feriados, entre outros.

(What?): Quais os objetivos com essa análise? O que iremos analisar?

O objetivo principal é realizar uma análise profunda das vendas da Rossman, identificando padrões temporais, correlações entre variáveis e fatores que impactam as vendas. Além disso, busca-se criar modelos de Machine Learning capazes de prever as vendas futuras com precisão.

(Where?): Trata dos aspectos geográficos e logísticos de sua análise.

A análise é centrada nas 1.115 lojas da Rossman, distribuídas na região Europeia. A compreensão das características geográficas e logísticas pode revelar insights sobre o desempenho das lojas em diferentes regiões.

(When?): Qual o período está sendo analisado? A última semana? Os últimos 6 meses? O ano passado?

O período analisado compreende os anos de 2013 a 2015. A escolha desse intervalo de tempo permite uma análise abrangente, capturando variações sazonais, influências econômicas e eventos importantes que podem ter impactado as vendas.

1.3. Objetivos

O presente estudo tem como objetivo principal realizar uma análise abrangente dos dados de vendas das lojas Rossmann, compreendendo os anos de 2013 a 2015. Além disso, busca-se desenvolver modelos de Machine Learning capazes de prever as vendas futuras das lojas da Rossmann com alta precisão, compreendendo fatores que podem afetar as suas vendas, permitindo a implementação de estratégias mais eficazes, bem como a antecipação de tendências e a maximização do desempenho comercial.

2. Coleta de Dados

O conjunto de dados presente nesse estudo foram extraídos através da plataforma Kaggle, disponível em https://www.kaggle.com/competitions/rossmann-store-sales/data, em 21 de outubro de 2023.

A coleta de dados abrangeu registros de vendas da empresa Rossman, compreendendo o período de 2013 a 2015. Dois conjuntos de dados em formato CSV foram utilizados: o arquivo "train" no qual contém informações de vendas de 1.115 lojas, com 1.017.209 linhas e 9 colunas, enquanto o arquivo "store" fornece informações adicionais sobre as lojas, com 1.115 linhas e 10 colunas.

O relacionamento entre as duas tabelas se dá através da coluna Store que representa o ID único para cada loja.

Tabela 1: Relação de atributos da tabela train

ATRIBUTOS	**DESCRIÇÃO**	**TIPO**
Store	ID único para cada loja	Int64
DayOfWeek	Dia da Semana	Int64
Date	Data ocorrida	Date
Sales	Vendas/Dia	Int64
Customers	Número de clientes em determinado dia	Int64
Open	Indicador se a loja estava aberta ou fechada	Int64
Promo	Indicador que se refere se a loja estava com promoção ou não no dia	Int64
StateHoliday	Feriado Estadual	Object
SchoolHoliday	Feriado Escolar	Int64

A seguir, apresentaremos uma análise detalhada dos atributos contidos na tabela 'train'. Esta descrição detalhada visa proporcionar uma compreensão aprofundada das informações contidas na tabela, fornecendo uma base sólida para

análises precisas. O objetivo é extrair insights valiosos que poderão ser úteis para compreender e interpretar os dados, além de orientar decisões e estratégias futuras. Abaixo seguem os atributos da tabela “train”:

Store: Esse atributo representa o ID único da loja, no qual o total de lojas é de 1.115;

DayOfWeek: Esse atributo representa o dia da semana em que ocorreu a venda, sendo o número 1 sendo representado pelo início da semana (domingo) e o número 7 sendo representado pelo último dia da semana (sábado), e assim respectivamente;

Date: Data da Venda, sendo o período inicial de 01/01/2013 e o período final 31/07/2015;

Sales: Valor do faturamento diário;

Customers: Número de clientes que estiveram presentes na loja em um determinado dia, que realizaram alguma compra ou não;

Open: Esse atributo informa se a loja estava aberta ou fechada no dia, sendo 0 para fechada e 1 para aberta;

Promo: Esse atributo informa se a loja estava com uma promoção no dia, sendo 0 para sem promoção no dia e 1 para promoção no dia;

StateHoliday: Esse atributo informa se houve ou não um feriado, sendo classificado por variáveis nas quais representadas por a=feriado, b=feriado de Páscoa, c=natal e 0=nenhum;

SchoolHoliday: Esse atributo informa se houve ou não um feriado escolar, sendo 0 para sem feriado escolar no dia e 1 para feriado escolar no dia;

Tabela 2: Relação de atributos da tabela store

ATRIBUTOS	**DESCRIÇÃO**	**TIPO**
Store	ID único para cada loja	Int64
StoreType	Tipo de loja (a,b,c,d)	Object
Assortment	Variedade de produtos sendo a=basic, b=extra e c=extended	Object
CompetitionDistance	distância em metros para	Float64

	a loja concorrente mais próxima	
CompetitionOpenSinceMonth	fornece o mês que o concorrente mais próximo foi aberto	Float64
CompetitionOpenSinceYear	fornece o ano que o concorrente mais próximo foi aberto	Float64
Promo2	Promoção contínua e consecutiva para algumas lojas	Int64
Promo2SinceWeek	Semana que ocorreu a promoção contínua	Float64
Promo2SinceYear	Ano que ocorreu a promoção contínua	Float64
PromoInterval	Meses em que a Promo2 é reiniciada	Object

Faremos também a análise detalhada dos atributos para a tabela “store”. Abaixo seguem seus atributos:

Store: Esse atributo representa o ID único da loja, no qual o total de lojas é de 1.115;

StoreType: Esse atributo representa que há 4 categorias de lojas distintas sendo “a”,”b”,”c” e “d”.

Assortment: Esse atributo indica a variedade de produtos vendido na loja sendo identificada pelas seguintes variáveis: **a=basic** que indica a venda de produtos essenciais ou fundamentais para a loja na qual visam a atender as necessidades básicas dos clientes; **b=extra** que indica a venda de produtos de categorias ou características extras; **c=extended** que indica a venda de uma maior variedade de produtos na loja, podendo atender a uma maior variedade de opções.

CompetitionDistance: Esse atributo descreve a distância para a loja concorrente mais próxima em metros.

CompetitionOpenSinceMonth: Mês que foi feita a abertura da loja do concorrente mais próxima;

CompetitionOpenSinceYear: Ano que foi feita a abertura da loja do concorrente mais próxima;

Promo2: Esse atributo representa a participação ou não de determinadas lojas em uma promoção contínua, sendo 0 indicando a não participação e 1 indicando a participação;

Promo2SinceWeek: Esse atributo descreve a semana que a loja começou a participar da Promo2;

Promo2SinceYear: Esse atributo descreve o ano que a loja começou a participar da Promo2;

PromoInterval: Esse atributo descreve os intervalos que a Promo2 é iniciada, nomeando os meses em que a promoção é reiniciada.

3. Processamento/Tratamento de Dados

O estudo foi executado por meio da linguagem de programação Python [3] interpretada no Jupyter Notebook (que se refere a uma aplicação interativa de código aberto que permite criar e compartilhar documentos que contenham código, texto, visualizações). Para a importação e análise dos DataSets foram importadas bibliotecas importantes do Python para o contexto do estudo, como por exemplo, o **Pandas** que fornece estruturas de dados flexíveis para manipulação e análise eficiente dos dados, **Matplotlib** que oferece uma variedade de criar gráficos e visualizações**, Seaborn** que é uma biblioteca de criação de gráficos estatísticos baseada no Matplotlib.

```
In [16]: #bibliotecas
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sns
         import calendar
         from tabulate import tabulate
```

Figura 3.1: Exemplo de importação de bibliotecas utilizadas em Python

Inicialmente, foram importadas as duas bases de dados disponíveis para a análise das informações, a base “train” que contêm 1.017.209 linhas e 9 colunas, sendo a base relacionada as vendas no período.

```
In [17]: #importando a base de dados train (base de dados sobre as vendas)

dtypes= {'Store':'int64'
              ,'DayOfWeek':'int64'
              ,'Date': 'object'
              ,'Sales': 'int64'
              ,'Customers': 'int64'
              ,'Open': 'int64'
              ,'Promo': 'int64'
              ,'StateHoliday': 'object'
              ,'SchoolHoliday': 'int64'}


df_train = pd.read_csv('C:\\Users\\AL106341\\Desktop\\Pós Graduação\\16 - TCC\\TCC\\train.csv' ,encoding='ISO-8859-1', dtype=dtyp

In [24]: #quantidade de linhas e colunas do dataframe train
num_linhas, num_colunas = df_train.shape
print(f'O Dataframe train contêm {num_linhas} linhas e {num_colunas} colunas.')

O Dataframe train contêm 1017209 linhas e 9 colunas.
```

Figura 3.2: Código em Python para importação da base de vendas “train”.

Após a importação da base "train", também importamos a base de dados "store", a qual contém 1.115 linhas e 10 colunas, fornecendo informações específicas sobre cada loja.

```
In [4]: #importando a base de dados store (base de dados sobre as vendas)

        df_store = pd.read_csv('C:\\Users\\AL106341\\Desktop\\Pós Graduação\\16 - TCC\\TCC\\store.csv' ,encoding='ISO-8859-1')

In [26]: #quantidade de linhas e colunas do dataframe store
         num_linhas, num_colunas = df_store.shape
         print(f'O Dataframe store contêm {num_linhas} linhas e {num_colunas} colunas.')

         O Dataframe store contêm 1115 linhas e 10 colunas.
```

Figura 3.3: Código em Python para importação da base de dados “store” com informações sobre as lojas.

Com o intuito de realizar a correlação entre as informações, foi criada uma base combinando os dados da tabela "store" com os da tabela "train", gerando o DataFrame denominado df_train_store. Este DataFrame manteve o número total de linhas da tabela "train" anterior (1.017.209), no entanto, foram adicionadas 8 colunas provenientes da tabela "store". É importante observar que a coluna "store" não foi incluída novamente na nova base, uma vez que ela foi utilizada como chave para a combinação dos dados (Figura 3.4).

```
In [33]: #realizando a junção da base de dados store para a base de dados train
         df_train_store = pd.merge(df_train, df_store, on='Store', how='left')
         a,b = df_train_store.shape
         print(f'O Dataframe store contêm {a} linhas e {b} colunas.')

         O Dataframe store contêm 1017209 linhas e 18 colunas.
```

Figura 3.4: Código em Python para junção da tabela “train” com a tabela “store”

Um ponto crítico a ser avaliado são os valores nulos, nos quais realizamos a verificação da quantidade de registros nulos por atributos. Nessa análise, observamos que os valores nulos se relacionam com a base de dados representadas pelas colunas CompetitionOpenSinceMonth, CompetitionOpenSinceYear, e às informações de promoções nas colunas Promo2SinceWeek, Promo2SinceYear, PromoInterval. Esses valores nulos serão substituídos pelo número 0, pressupondo que não houve informações disponíveis referentes às datas de abertura do concorrente mais próximo e que os atributos Promo2SinceWeek, Promo2SinceYear e PromoInterval não têm promoções associadas.

```
In [37]: #verificação dos valores nulos na base de dados
         df_train_store.isnull().sum()

Out[37]: Store                          0
         DayOfWeek                      0
         Date                           0
         Sales                          0
         Customers                      0
         Open                           0
         Promo                          0
         StateHoliday                   0
         SchoolHoliday                  0
         StoreType                      0
         Assortment                     0
         CompetitionDistance         2642
         CompetitionOpenSinceMonth  323348
         CompetitionOpenSinceYear   323348
         Promo2                         0
         Promo2SinceWeek            508031
         Promo2SinceYear            508031
         PromoInterval              508031
         dtype: int64
```

Figura 3.5: Código em Python demonstrando a quantidade de valores nulos dos atributos.

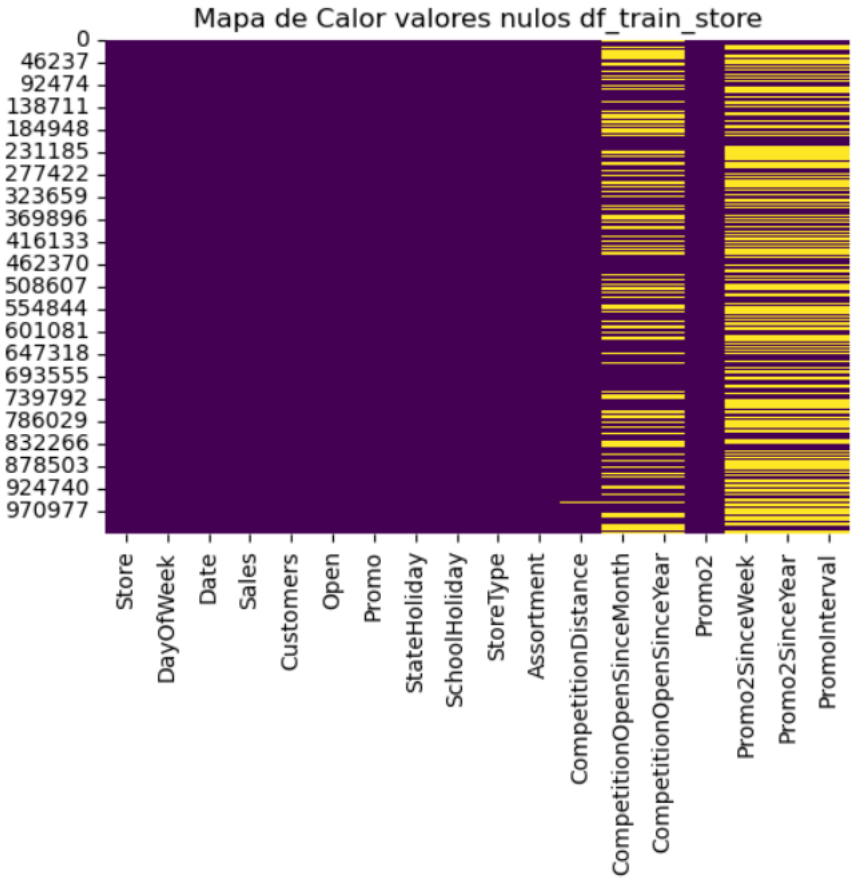

Figura 3.6: Mapa de calor representando os valores nulos.

Após a atualização dos valores verificou-se novamente o mapa de calor referente a quantidade de valores nulos na base de dados, e verificou-se que a base de dados obteve 99.98% de preenchimento total (Figura 3.7).

```
In [42]: #verificando o percentual de preenchimento das informações
         p = (1 - df_train_store.isnull().mean().mean()) * 100
         print(p)

         99.98557053881967
```

Figura 3.7: Código em Python demonstrando o % de preenchimento da base.

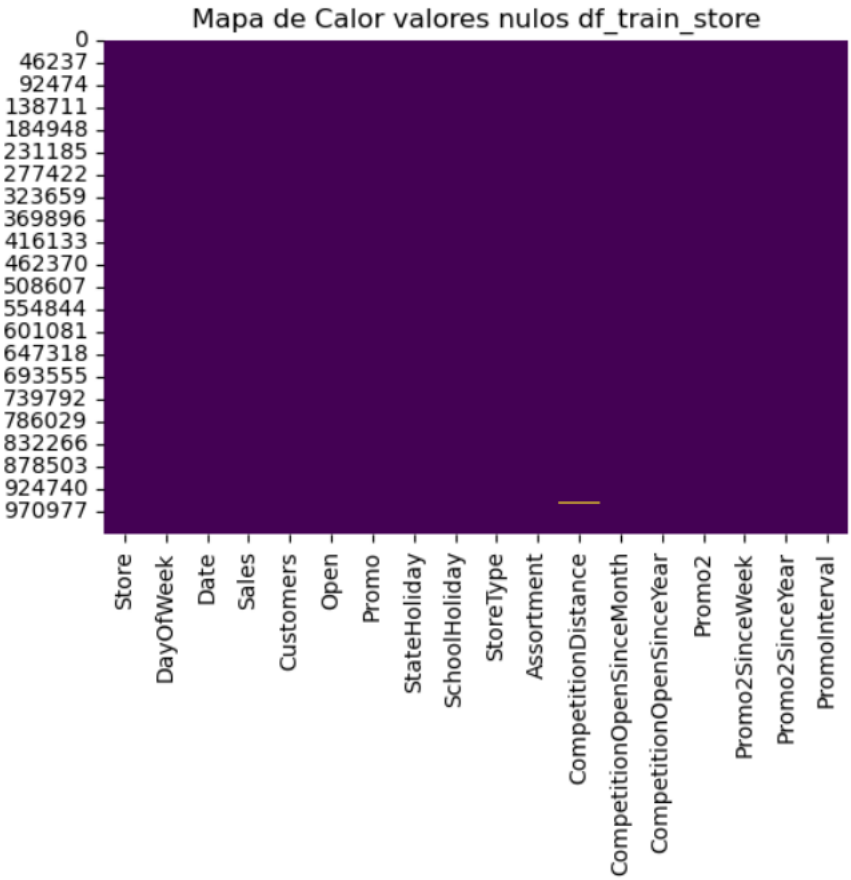

Figura 3.8: Mapa de calor representando os valores preenchidos.

A última consideração feita no tratamento da base de dados diz respeito às informações sobre quando a loja está fechada, ou seja, quando a coluna 'Open' é igual a 0. Durante essa análise, observou-se que não houve qualquer valor de venda (coluna 'Sales') associado aos momentos em que a loja estava fechada (coluna 'Open' igual a 0). Portanto, não faz sentido incluir esses registros na análise, uma

vez que o objetivo é investigar os fatores que influenciam as vendas. A quantidade de linhas na base após a exclusão da coluna Open = 0 totalizou-se em 844.391 linhas (Figura 3.9)

```
In [50]: #soma de vendas quando a loja estava fechada e aberta.
vendas_aberto_fechado = df_train_store.groupby('Open')['Sales'].sum().reset_index()

linhas_open_zero = df_train_store[(df_train_store['Open'] == 0) & (df_train_store['Sales'] == 0)].index

#excluir da base os dias que a loja estava fechada e não houve valor de vendas.
df_train_store = df_train_store.drop(linhas_open_zero)

# Resetando os índices
df_train_store.reset_index(drop=True, inplace=True)

contagem = df_train_store.shape[0]

print('A Quantidade de linhas após exclusão é de {}'.format(contagem))

A Quantidade de linhas após exclusão é de 844391
```

Figura 3.9: Código em Python demonstrando a quantidade de linhas após a exclusão das linhas em que a loja esteve fechada.

4. Análise e Exploração dos Dados

Na fase inicial da análise de dados, conduzi uma breve análise estatística de três variáveis críticas: Vendas (Sales), Clientes (Customers) e Distância da Concorrência (CompetitionDistance), com o intuito de obter insights que possam enriquecer as futuras métricas. Na variável vendas, nota-se que a média das vendas ao longo do período foi de 6.955, com o valor mínimo de 0 em um dia em que a loja esteve aberta e o valor máximo atingindo 41.551,00. Um ponto relevante evidenciado é que 75% das vendas diárias estão abaixo de 8.360,00, indicando a possibilidade de eventos como promoções ou feriados.

No que diz respeito à variável Clientes (Customers), observamos uma média de 762 visitantes diários, com o valor mínimo de visitas registrando 0 e o valor máximo alcançando 7.388. De outro modo, é notável que 75% das visitas diárias estão abaixo de 893 pessoas.

Na variável Distância da Concorrência (CompetitionDistance), constatamos que a média da distância entre as lojas concorrentes é de 5.457 metros, com o valor mínimo de 20 metros e o valor máximo de 75.860 metros. Outro aspecto importante é que 75% das distâncias para concorrentes estão abaixo de 6.890 metros. Os 25% restantes das informações relacionadas a essas três variáveis podem indicar a presença de outliers, sendo necessário investigar a relação com os dias da semana ou promoções que possam ter ocorrido no período, aspectos que serão analisados em etapas subsequentes da pesquisa.

Abaixo temos a figura 4.1 que representa uma breve análise sobre os dados estatísticos das variáveis Vendas (Sales), Clientes (Customers) e Distância da Concorrência (CompetitionDistance).

	Sales	Customers	CompetitionDistance
count	844392.000000	844392.000000	842206.000000
mean	6955.514291	762.728395	5457.979627
std	3104.214680	401.227674	7809.437311
min	0.000000	0.000000	20.000000
25%	4859.000000	519.000000	710.000000
50%	6369.000000	676.000000	2320.000000
75%	8360.000000	893.000000	6890.000000
max	41551.000000	7388.000000	75860.000000

Figura 4.1: Análise estatística referente a base de dados df_train_store

O próximo passo consiste em avaliar a correlação da coluna de vendas (Sales) com as outras variáveis numéricas. A matriz de correlação atribui valores no intervalo de -1 a 1, indicando uma correlação mais forte quando a variável se aproxima de -1 ou 1.

Na Figura 4.2, examinei a correlação da coluna de vendas (Sales) com as demais variáveis numéricas, resultando em uma correlação significativa de 0.82 com a coluna de clientes (Customers). Isso sugere que à medida que mais clientes frequentam a loja, mais as vendas aumentam.

Outra correlação relevante é aquela associada à coluna Promo, que apresentou um coeficiente de 0.36. Embora seja uma correlação moderada, ela exerce um impacto direto nas vendas.

```
Store                       0.007710
DayOfWeek                  -0.178736
Sales                       1.000000
Customers                   0.823597
Open                             NaN
Promo                       0.368145
SchoolHoliday               0.038617
CompetitionDistance        -0.036396
CompetitionOpenSinceMonth  -0.018370
CompetitionOpenSinceYear    0.005266
Promo2                     -0.127596
Promo2SinceWeek            -0.058476
Promo2SinceYear            -0.127621
Name: Sales, dtype: float64
```

Figura 4.2: Análise da correlação da coluna Sales com as demais variáveis

Na continuidade da análise exploratória, é essencial examinar os dados de vendas (Sales) em relação aos aspectos temporais, visando realizar projeções, especialmente associadas a essa variável. Na representação gráfica a seguir, houve a exploração da relação entre a coluna dos dias da semana ('DayofWeek') e a variável de vendas (Sales), distinguindo entre os dias que envolveram promoção e aqueles que não.

Nos dias em que não houve promoção, observamos que as vendas ficaram abaixo da média anteriormente identificada de 6.955 vendas por dia, sendo exceção apenas o sábado, devido ao caráter de final de semana. Por outro lado, nos dias com promoção, todas as jornadas semanais apresentaram vendas superiores à média, indicando que os 25% dos valores das vendas associados a outliers são, de fato, relativos aos dias de promoção.

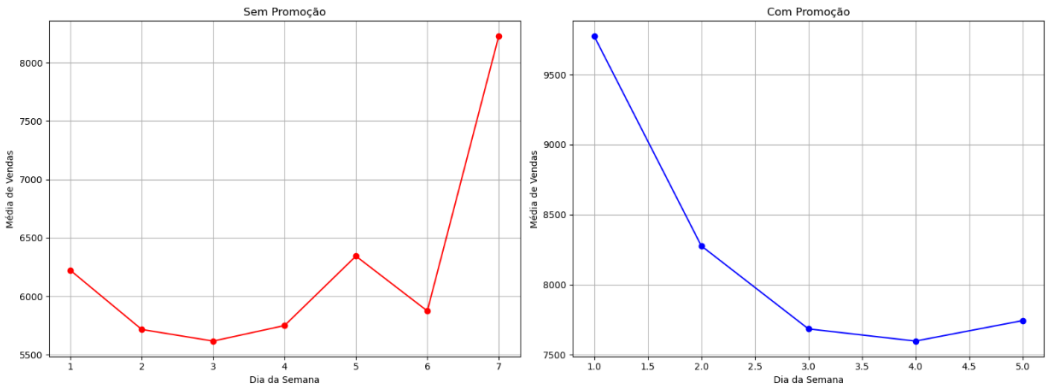

Figura 4.3 Análise da média diária de vendas por dia da semana

Na análise da média diária de vendas ao longo do mês, observa-se que nas vendas sem promoção há um pico mais elevado no início e no final do mês, com um período de baixa média entre os dias 02 e 17. Após essa data, a média de vendas se recupera gradualmente, atingindo seu pico no dia 30.

Para as vendas com promoção, nota-se uma média mais alta, com picos nos dias 01, 15 e 31 do mês. Essa observação sugere que nos dias do meio do mês e no final ou início do mês subsequente, pode ocorrer o recebimento de salários, pois tanto as vendas sem promoção quanto as com promoção atingem seus picos nesse período.

Isso pode indicar um aumento no estoque nesse intervalo, possivelmente incentivando a realização de promoções mais agressivas nos primeiros 15 dias do mês, quando observamos a curva descendente das vendas conforme Figura 4.4

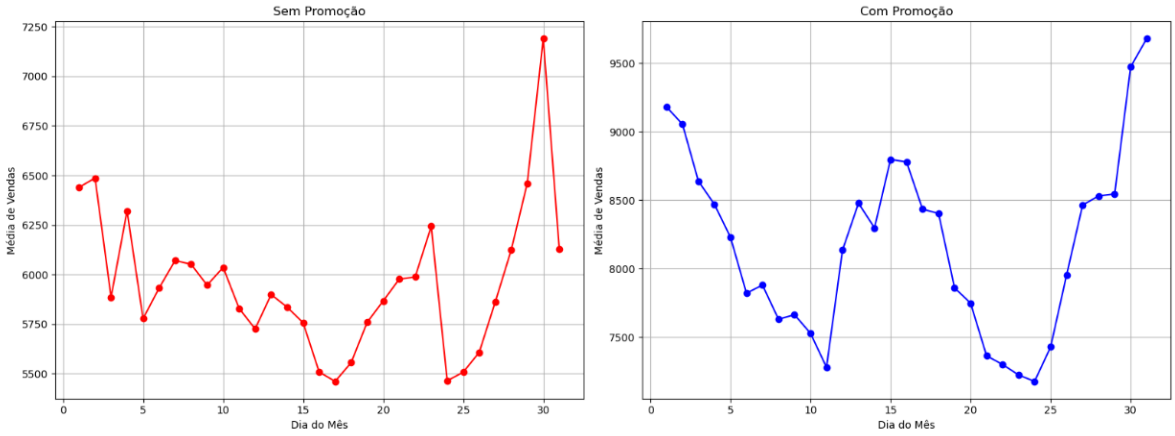

Figura 4.4: Análise da média diária de vendas por dia do mês

Na análise da média diária de vendas ao longo do mês, percebeu-se um considerável aumento na média de vendas no final do ano, especialmente a partir de outubro, coincidindo com as celebrações festivas de final de ano.

Por outro lado, também se nota uma tendência descendente tanto nas vendas sem promoção quanto com promoção nos meses de junho a setembro, sugerindo a possibilidade de ser vantajoso implementar promoções mais agressivas durante esse período, conforme Figura 4.5

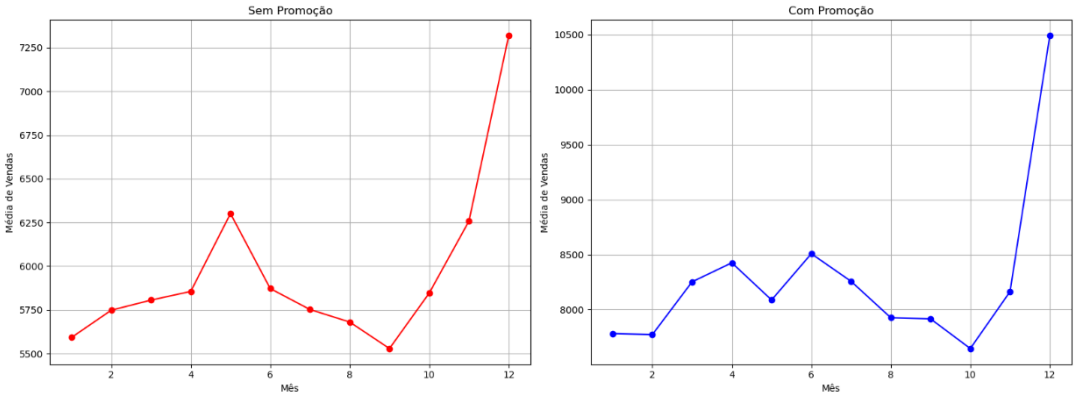

Figura 4.5: Análise da média diária de vendas por mês

Outra relação significativa presente no conjunto de dados diz respeito às vendas durante os feriados. A análise revelou que a média de vendas em dias de feriados é de 8.907, enquanto a média de vendas em dias sem feriados é de 6.895.

Em outras palavras, as vendas durante os feriados superam a média normal, indicando que os feriados desempenham um papel importante como variável, sugerindo uma demanda maior nesses dias. Esses resultados sugerem que é vantajoso manter as lojas abertas durante os feriados (Figura 4.6).

```
df_train_store_filtrado = df_train_store.loc[(df_train_store['StateHoliday'] == '0') & (df_train_store['SchoolHoliday'] == 0)]
df_train_store_filtrado_2 = df_train_store.loc[(df_train_store['StateHoliday'] != '0') & (df_train_store['SchoolHoliday'] != 0)]

media_vendas_sem_feriado = df_train_store_filtrado['Sales'].mean()
media_vendas_com_feriado = df_train_store_filtrado_2['Sales'].mean()

print('A média de vendas nos dias sem feriado é {}'.format(media_vendas_sem_feriado))
print('A média de vendas nos dias com feriado é {}'.format(media_vendas_com_feriado))
```

```
A média de vendas nos dias sem feriado é 6895.153041351444
A média de vendas nos dias com feriado é 8907.162420382165
```

Figura 4.6: Código Python demonstrando a análise média das vendas nos dias com e sem feriado.

Ao examinar igualmente qual loja apresenta a maior média de vendas ao longo do período, o resultado revelou que a loja "b" lidera com uma média de vendas de 10.231, superando a média normal das lojas. Por outro lado, o tipo "d" é caracterizado pela menor média de vendas, totalizando 6.822 (Figura 4.7).

```
#qual tipo de loja tem a maior média de vendas?

media_vendas_tipo_loja = df_train_store.groupby('StoreType')['Sales'].mean()
print(media_vendas_tipo_loja)
```

```
StoreType
a     6925.167661
b    10231.407505
c     6932.512755
d     6822.141881
Name: Sales, dtype: float64
```

Figura 4.7: Código Python com a análise média das vendas por loja

Analisei também a associação entre o valor médio das vendas em lojas que mantêm promoções contínuas. Os resultados indicaram que essas lojas apresentaram uma média de vendas de 6.558, abaixo da média normal de vendas. Isso sugere que as promoções contínuas têm um impacto limitado nas vendas (Figura 4.8)

```
# impacto das promoções
#Explore mais a fundo o impacto das promoções nas vendas, considerando a duração das promoções e se existe uma diferença signific

df_train_store.head()

promo_continua = (df_train_store['Promo2'] == 1)

vendas_promo_continua = df_train_store.loc[promo_continua,'Sales']

media_vendas_promo_continua = vendas_promo_continua.mean()

print('A média de vendas das lojas com promoção contínua é de {}'.format(media_vendas_promo_continua))


promo_nao_continua = (df_train_store['Promo2'] == 0)

vendas_promo_nao_continua = df_train_store.loc[promo_nao_continua,'Sales']

media_vendas_nao_continua = vendas_promo_nao_continua.mean()

print('A média de vendas das lojas que não possuem promoção contínua é de {}'.format(media_vendas_nao_continua))
```

```
A média de vendas das lojas com promoção contínua é de 6558.386062196469
A média de vendas das lojas que não possuem promoção contínua é de 7350.557935493625
```

Figura 4.8: Código Python com a média de vendas das lojas com e sem promoções contínuas

Ao analisar a relação entre a média de vendas e a distância entre os concorrentes, constatei que não há uma correlação significativa entre essas variáveis. Mesmo com o aumento da distância até a primeira loja concorrente, a média de vendas permanece bastante constante quando consideramos distâncias entre 1.000 e 10.000 metros, dado que a média geral da distância das lojas é de 5.457 metros (Figura 4.9)

```
# Relação entre Distância da Concorrência e Vendas:
correlacao = df_train_store['CompetitionDistance'].corr(df_train_store['Sales'])
print('Correlação entre Distância da Concorrência e Vendas:', correlacao)

# Criar intervalos de distância
bins = [0, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, float('inf')]
labels = ['0-1000', '1001-2000', '2001-3000', '3001-4000', '4001-5000', '5001-6000', '6001-7000', '7001-8000', '8001-9000', '9001

# Adicionar coluna 'DistanceRange' ao DataFrame
df_train_store['DistanceRange'] = pd.cut(df_train_store['CompetitionDistance'], bins=bins, labels=labels, right=False)

# Calcular a média de vendas em cada intervalo
avg_sales_by_distance = df_train_store.groupby('DistanceRange')['Sales'].mean()
print(avg_sales_by_distance)
```

```
Correlação entre Distância da Concorrência e Vendas: -0.03639608890500295
DistanceRange
0-1000        7309.679305
1001-2000     7060.500991
2001-3000     6653.722201
3001-4000     6332.933297
4001-5000     7184.177899
5001-6000     6681.395044
6001-7000     6477.421416
7001-8000     6956.958325
8001-9000     6956.345576
9001-10000    6778.205510
10001+        6824.760993
Name: Sales, dtype: float64
```

Figura 4.9: Código Python demonstrando a correlação entre a distância da concorrência e vendas, como também a média de vendas por distância de 0 a 10.000 metros

Com os resultados obtidos em nossa análise e exploração dos dados retiramos os seguintes insights resumidos:

Vendas, Clientes e Distância da Concorrência:

- Média de vendas diárias: 6.955, com mínimo de 0 e máximo de 41.551.
- Média diária de clientes: 762, com mínimo de 0 e máximo de 7.388.
- Média da distância da concorrência: 5.457 metros, com mínimo de 20 e máximo de 75.860.

Correlações relevantes:

- Forte correlação positiva entre vendas e clientes (0.82)
- Correlação moderada entre vendas e Promo (0.36)

Análise Temporal:

- Promoções têm impacto significativo nas vendas.
- Médias diárias de vendas variam ao longo do mês, sugerindo influência de eventos salariais.
- Aumento notável nas vendas no final do ano.

Feriados e Tipo de Loja:

- Vendas durante feriados superam a média normal.
- Lojas tipo "b" lideram com a maior média de vendas (10.231), enquanto tipo "d" tem a menor (6.822).
- Promoções contínuas têm impacto limitado nas vendas médias.

Distância da Concorrência:

- Não há correlação significativa entre a distância da concorrência e as vendas.
- Mesmo com aumento da distância, média de vendas permanece constante entre 1.000 e 10.000 metros.

5. Criação de Modelos de Machine Learning

Nos últimos anos, o campo do Machine Learning emergiu como uma força motriz revolucionária, capacitando sistemas computacionais a aprender padrões complexos e fazer previsões ou tomar decisões sem serem explicitamente programados. Essa disciplina, profundamente enraizada na inteligência artificial, busca capacitar máquinas a extrair insights significativos a partir de dados, adaptar-se a novas informações e aprimorar seu desempenho ao longo do tempo.

Em sua essência, o Machine Learning representa uma mudança paradigmática em relação à programação tradicional. Em vez de depender de regras e lógica de código explicitamente definidas, os modelos de machine learning são treinados em dados para aprender padrões e tomar decisões de maneira autônoma. Esses modelos têm a capacidade de generalizar a partir de exemplos, o que significa que podem fazer previsões ou realizar tarefas em dados não vistos durante o treinamento.

O Machine Learning é aplicado em uma variedade de setores, desde reconhecimento de voz e visão computacional até recomendação de produtos, diagnóstico médico e previsão de tendências. Algoritmos de aprendizado de máquina são capazes de processar grandes volumes de dados, identificar correlações complexas e automatizar tarefas que anteriormente demandavam programação explícita.

Para realizar a previsão das vendas, testei 4 modelos de machine learning baseados em árvore de decisão, para verificar qual modelo mais se adequa as nossas informações.

Os modelos utilizados foram **DecisionTreeRegressor** no qual são modelos flexíveis e fáceis de interpretar e podem capturar padrões complexos nos dados e fornecer insights sobre as relações entre variáveis.

O **Bagging Regressor** que melhora a estabilidade e a generalização do modelo ao combinar as previsões de várias árvores de decisão treinadas em subconjuntos aleatórios dos dados.

O **RandomForestRegressor** conhecido por sua robustez e capacidade de fornecer previsões precisas, mesmo em conjuntos de dados grandes e complexos.

E por fim, o **GradientBoostingRegressor** que constrói árvores de decisão sequencialmente, cada uma corrigindo os erros do modelo anterior. Isso permite um ajuste mais fino e uma adaptação mais precisa aos padrões nos dados.

No código abaixo foi realizado o pré-processamento dos dados (tratando valores ausentes e convertendo as variáveis categóricas em numéricas), e em seguida realizou-se os treinamentos dos modelos de regressão para a escolha do melhor modelo. (Figura 5.1)

```
In [*]: from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import cross_validate
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import BaggingRegressor, RandomForestRegressor, GradientBoostingRegressor

X_encoded = X.copy()
non_numeric_cols = X_encoded.select_dtypes(exclude=['number']).columns
numeric_cols = X_encoded.select_dtypes(include=['number']).columns
imputer_numeric = SimpleImputer(strategy='mean')
X_encoded[numeric_cols] = imputer_numeric.fit_transform(X_encoded[numeric_cols])

imputer_non_numeric = SimpleImputer(strategy='constant', fill_value='missing')
X_encoded[non_numeric_cols] = imputer_non_numeric.fit_transform(X_encoded[non_numeric_cols])

label_encoder = LabelEncoder()
X_encoded[non_numeric_cols] = X_encoded[non_numeric_cols].apply(label_encoder.fit_transform)

# Continue com a divisão dos dados e treinamento do modelo
for model_name, model in zip(modelos, regressao):
    cv_results = cross_validate(model, X_encoded, y, cv=5, scoring=['r2', 'neg_mean_squared_error'])
    print("############## {} ###############".format(model_name))
    print("R_score: {}".format(round(cv_results["test_r2"].mean(), 2)))
    print("RMSE: {}".format(round(cv_results["test_neg_mean_squared_error"].mean(), 2)))
```

Figura 5.1: Código Python para geração dos resultados dos modelos de regressão

De acordo com o resultado do R_Score no qual representa a porcentagem da variabilidade nos dados que é explicada pelo modelo, quanto mais próximo de 1, melhor. Nesse caso, RandomForestRegressor obteve o R_score mais alto (0.97), seguido por BaggingRegressor (0.96) e DecisionTreeRegressor (0.94). GradientBoostingRegressor tem um R_score um pouco mais baixo (0.88). (Figura 5.2)

```
############## DecisionTreeRegressor ##############
R_score: 0.94
RMSE: -0.06
############## BaggingRegressor ##############
R_score: 0.96
RMSE: -0.04
############## RandomForestRegressor ##############
R_score: 0.97
RMSE: -0.03
############## GradientBoostingRegressor ##############
R_score: 0.88
RMSE: -0.12
```

Figura 5.2: Representação dos resultados dos modelos de regressão utilizados

O menor RMSE que mede a precisão das previsões também foi o do RandomForestRegressor indicando um menor erro nas previsões.

Ao efetuarmos a previsão de vendas e compararmos os resultados reais com os previstos nas primeiras 100 instâncias, observamos, no gráfico da Figura 5.3 uma boa concordância entre o modelo de previsão e os valores reais. Isso sugere que o modelo é capaz de realizar projeções precisas das vendas futuras.

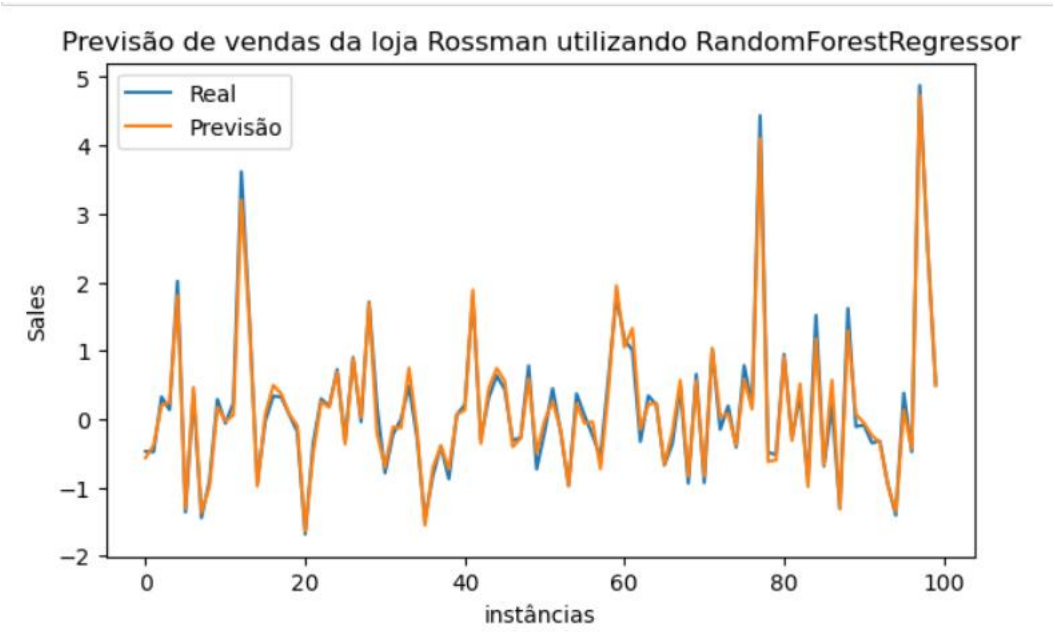

Figura 5.3: Previsão das Vendas com RandomForestRegressor Real x Previsto

Outro ponto importante é analisar a importância relativa de cada característica (feature) no modelo de previsão utilizado, portanto ao analisar essas características temos como resultado a coluna de Customers (clientes) como a mais influente, representando 74.12% para a previsão do modelo. Isso indica que a quantidade de clientes é o fator crucial na previsão das vendas. (Figura 5.4)

```
Feature Ranking

Feature 2 , Customers 0.7412
Feature 7 , StoreType 0.0578
Feature 9 , CompetitionDistance 0.0561
Feature 0 , Store 0.0346
Feature 4 , Promo 0.034
Feature 13 , Promo2SinceWeek 0.0123
Feature 1 , DayOfWeek 0.0116
Feature 11 , CompetitionOpenSinceYear 0.0116
Feature 10 , CompetitionOpenSinceMonth 0.0108
Feature 8 , Assortment 0.0094
Feature 16 , Month 0.0086
Feature 14 , Promo2SinceYear 0.0063
Feature 15 , PromoInterval 0.0032
Feature 6 , SchoolHoliday 0.0013
Feature 12 , Promo2 0.0009
Feature 5 , StateHoliday 0.0002
Feature 3 , Open 0.0
```

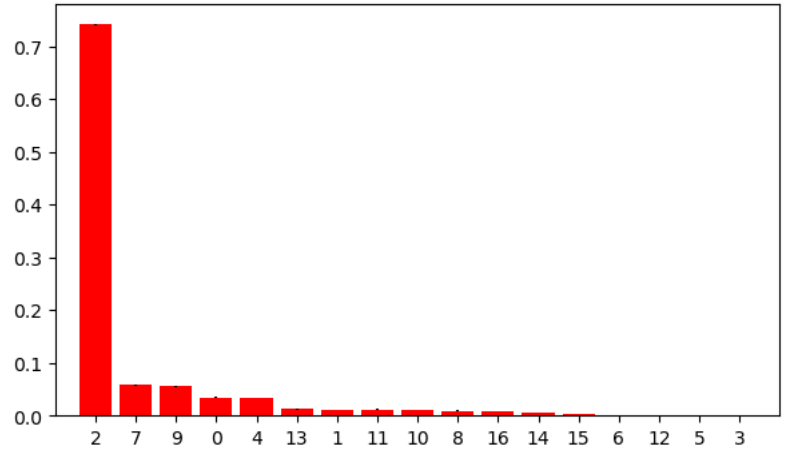

Figura 5.4: Classificação das Features mais importantes no modelo de RandomForestRegressor

6. Interpretação dos Resultados

A feature "Customers" mostrou-se altamente relevante na previsão das vendas, representando aproximadamente 74.12% da importância relativa no modelo. Isso sugere que o número de clientes tem uma forte influência nas vendas, indicando a importância de estratégias de marketing e fidelização.

A aplicação dos algoritmos de Machine Learning resultou em modelos capazes de realizar previsões precisas das vendas, com destaque para o RandomForestRegressor, que obteve o R_score mais alto (0.97) e o menor RMSE, indicando uma boa capacidade de generalização e ajuste aos padrões dos dados.

Ao comparar as previsões com os valores reais nas primeiras 100 instâncias, observamos uma boa concordância entre o modelo de previsão e os valores reais, fortalecendo a confiança na capacidade do modelo de realizar projeções precisas das vendas futuras.

7. Apresentação dos Resultados

Após o estudo referente à previsão das vendas relacionados as lojas Rossmann, podemos destacar pontos importantes que nos levaram as seguintes conclusões:

Importância da Feature "Customers": A variável "Customers" se destacou como a mais influente na previsão das vendas, representando 74.12% da importância relativa. Isso destaca a necessidade de estratégias voltadas para o aumento do número de clientes, como programas de fidelização e ações de marketing direcionadas.

Com referência aos modelos de Machine Learning RandomForestRegressor apresentou o melhor desempenho, com um R_score de 0.97. Os modelos BaggingRegressor e DecisionTreeRegressor também demonstraram bom desempenho, com R_scores de 0.96 e 0.94, respectivamente. e o menor RMSE, indicando uma capacidade sólida de prever as vendas. O modelo GradientBoostingRegressor obteve um R_score um pouco inferior (0.88), indicando um desempenho um pouco menos preciso em comparação com os outros modelos.

Previsões: A comparação entre as previsões e os valores reais nas primeiras 100 instâncias mostrou uma concordância satisfatória, reforçando a confiança na capacidade do modelo de fazer projeções precisas das vendas.

Influência Temporal nas Vendas: A análise temporal revelou padrões significativos, destacando a influência das promoções nas vendas e a variação ao longo do mês e do ano. A identificação de aumentos nas vendas durante feriados sugere a importância de estratégias específicas para essas ocasiões.

Demais Considerações: A análise da correlação entre a distância da concorrência e as vendas não mostrou uma relação significativa, indicando que outros fatores podem ser mais relevantes na determinação do desempenho das lojas. A importância relativa das features fornece insights valiosos sobre os fatores críticos para o modelo.

Por fim esses resultados proporcionaram uma base sólida para a orientação de decisões e estratégias futuras, como investimentos em marketing direcionado a clientes, otimização de estratégias de promoções e consideração de padrões temporais nas operações das lojas.

8. Links

Aqui você deve disponibilizar os links para o vídeo com sua apresentação de 5 minutos e para o repositório contendo os dados utilizados no projeto, scripts criados, etc.

Link para o vídeo: https://www.youtube.com/watch?v=jZ0kERjihU4
Link para o repositório: https://github.com/andrelprg/TCC-Puc-Minas