CEPEDI - RESTIC 36

Relatório Técnico: Implementação e Análise do Algoritmo de Regressão Linear

Membros da equipe: Fabrício Soares Oliveira, Geisa Almeida dos Santos

Eunápolis 12 de novembro de 2024

Resumo

Neste trabalho, iremos elaborar uma análise de dados utilizando a Regressão Linear. O objetivo do trabalho é desenvolver um modelo de previsão usando Regressão Linear. Nossa base de dados possui informações relativas a influenciadores de uma rede social. Vamos analisar a taxa de engajamento desses influenciadores no dataset Instagram. O projeto envolve desde a análise exploratória dos dados até a otimização e validação do modelo de Regressão Linear.

Introdução

A área de Ciências de Dados envolve estatística, matemática, programação e machine learning (aprendizado de máquina). Trata-se de uma área que vem ganhando grande potencial de mercado nos últimos anos. Com a evolução das IAs, gerou-se a necessidade de mais analistas de dados, segundo Renata Cristina Gutierres Castanha (2021, p. 1).

> Profissionais da International Data Corporation (2021) apontam que a produção de dados dobra a cada dois anos. A quantidade de dados criados nos próximos três anos será maior do que os dados criados nos últimos 30 anos.

Diante desse cenário, um cientista de dados precisa desenvolver várias habilidades para melhor manipulação dos dados. Ao longo do curso de Ciências de Dados, aprendemos como carregar um banco de dados, visualizar um dataset, analisar e tratar esses dados. Como citado anteriormente, introduzimos técnicas de Machine Learning, como kNN, Árvore de Decisão, Regressão Logística, Regressão Linear, Redes Neurais, Redes Neurais Convolucionais, técnicas de regularização e Redes Competitivas.

No entanto, o objetivo deste trabalho é tratar apenas de Regressão Linear, uma técnica de análise de dados usada para entender a relação entre duas variáveis. Um de seus principais objetivos é prever o valor de uma variável dependente com base no valor de outras variáveis independentes.

A meta da Regressão Linear é encontrar uma linha reta que melhor represente essa relação entre as variáveis, reduzindo a distância entre os valores reais e os valores previstos pelo modelo, conforme Demetrios Coutinho.

Metodologia

Para iniciar a análise exploratória do projeto de Regressão Linear, primeiramente precisamos baixar o banco de dados com o qual iremos trabalhar. Em seguida, criamos uma pasta no Google Drive e a anexamos. Após isso, iniciamos o Colab e criamos um novo notebook. Para ter acesso ao banco de dados, precisamos conectar o Colab ao Drive, especialmente à base de dados que foi salva anteriormente. É necessário permitir que o Colab acesse o Drive.

Logo depois, devemos importar as bibliotecas, especialmente o Pandas e o Numpy, pois são essenciais na análise de dados com Python. A biblioteca Pandas é responsável pela manipulação de dados na análise de dados, sendo muito utilizada para trabalhar com dados em formato de tabela. No entanto, ela também é capaz de ler dados em outros formatos, como CSV, SQL, JSON, entre outros. Além disso, a Pandas permite a manipulação e análise estatística dos dados, possibilitando operações como a manipulação de linhas e colunas, além da exclusão ou adição de linhas no DataFrame de forma simples.

A Pandas é ideal para a manipulação de dados estruturados e análise de dados.

Já a biblioteca Numpy é utilizada para computação numérica, com foco em operações com arrays e matrizes de dados. Ela também possui funções matemáticas de alto desempenho e oferece operações como soma e cálculos em

arrays de maneira vetorizada, o que permite o acesso e manipulação eficiente dos elementos. A Numpy é ideal para cálculos numéricos e operações com arrays e matrizes.

Ao finalizar a importação das bibliotecas e a conexão com o Drive, o arquivo pode ser facilmente localizado e manipulado utilizando o comando gdown, que é muito útil para carregar arquivos grandes ou datasets, sem precisar fazer o upload manualmente para o Colab.

Ao carregar a base de dados, iniciamos uma análise para identificar aqueles que possuíam maior taxa de engajamento. Observamos uma boa quantidade de curtidas, os países, os nomes dos influenciadores, entre outros. No entanto, notamos também que a linguagem em que os dados foram fornecidos estava em inglês.

Visando uma melhor análise desses dados, optamos por realizar uma das primeiras modificações: a tradução desses atributos para o português. O método utilizado para realizar essa modificação foi o dataset.rename(columns={"Nome do atributo em inglês": "nome do atributo em português"}).

Durante essa análise, decidimos verificar a existência de valores ausentes ou nulos, pois isso pode impactar nossos resultados finais. Precisamos realizar uma análise minuciosa dos dados que recebemos para tratar adequadamente as informações. Para visualizar a existência desses valores nulos, utilizamos o comando: dataset.isnull().sum().

O retorno foi interessante, pois apenas a coluna "País" apresentou valores nulos, totalizando 62 valores ausentes. Embora isso não deva impactar significativamente nosso resultado futuro, uma vez que não é uma variável essencial para o treinamento, optamos por tratar esses dados nulos. Decidimos, então, substituir esses valores pela mediana, a fim de evitar possíveis problemas. O código utilizado foi: dataset['País'].fillna("País", inplace=True).

Os passos a seguir foram realizados com o objetivo de melhorar o dataset. Primeiramente, verificamos os principais valores categóricos do dataset, buscando

identificar o mínimo, a média, a constância, entre outros valores presentes na tabela. O código utilizado foi: dataset.describe().

Após obtermos os resultados dessa consulta, realizamos uma alteração importante visando os próximos passos do processo. Fizemos a conversão de alguns valores para um formato adequado aos métodos subsequentes. Essa alteração foi necessária, pois alguns atributos estavam representados como strings, como é o caso de "m" (milhões), "k" (milhares) e "b" (bilhões). Utilizamos validações com estruturas condicionais (if, elif, else) para ler as strings e convertê-las para o formato float, como mostrado na Figura.

Figura 1 - Conversão de valores

```
analise_insta = pd.DataFrame()

# Aplicar a função à coluna "Seguidores"
analise_insta['Influenciador'] = dataset['Nome']
analise_insta['Classificação'] = dataset['Classificação']

def convert_postagens(value):
    if isinstance(value, str):  # Verifica se o valor é uma string
        if 'm' in value:
            return float(value.replace('m', '')) * 1000000
        elif 'k' in value:
            return float(value.replace('k', '')) * 1000
        elif 'b' in value:
            return float(value.replace('b', '')) * 1000000000
        else:
            return float(value)  # Caso não tenha 'm', 'k' ou 'b'
    else:
        return value  # Se já for numérico, retorna o valor como está

# Aplicar a função às colunas relevantes
analise_insta['Postagens'] = dataset['Postagens'].apply(convert_postagens)
analise_insta['Seguidores'] = dataset['Seguidores'].apply(convert_postagens)
analise_insta['Media_Curtidas'] = dataset['Media_Curtidas'].apply(convert_postagens)
analise_insta['Media_Curtidas_Novas'] = dataset['Media_Curtidas_Novas'].apply(convert_postagens)

analise_insta["Taxa_Engajamento"] = dataset["Taxa_Engajamento"].replace("%", "", regex=True)
analise_insta["Taxa_Engajamento"] = analise_insta["Taxa_Engajamento"].astype(float) / 100

analise_insta['País'] = dataset['País']

print(analise_insta)
```

Fonte: Elaborada pelos autores (2024)

Gostaríamos de destacar que todas as modificações, como ilustrado na Figura 1, foram realizadas em um novo dataset, o qual denominamos

analise_insta. Esse método é utilizado com o objetivo de trabalhar apenas com dados relevantes para a análise, sem a necessidade de manter todos os dados do dataset original. Portanto, agora estamos lidando exclusivamente com os dados necessários para a regressão linear e com um novo dataset.

Após realizar essas modificações no dataset (analise_insta), partimos para a criação da matriz de correlação. Essa matriz gera uma tabela onde é possível identificar as variáveis dependentes. Contudo, dependendo da quantidade de atributos em um banco de dados, pode ser difícil interpretar os resultados diretamente. Para facilitar essa análise, utilizamos um método de plotagem que gera um gráfico, permitindo observar a relação entre os atributos. Nesse gráfico, utilizamos cores mais quentes para dados ricos em informação e azul-escuro para dados com pouca informação.

Em nosso código, por exemplo, a taxa de engajamento, a média de curtidas novas e a média de curtidas são valores que apresentam uma boa proximidade. Quando plotados, ficou notável a relação entre eles. Também podemos observar que tanto a coluna "categoria" quanto a coluna "país" não impactariam significativamente a parte principal da nossa análise. Podemos ver isso na Figura 2.

Figura 2 - Relação entre as variáveis

```
# Relação entre as variaves, saber quais depende uma da outra.
correlation_matrix = analise_insta.corr()
print("\nMatriz de correlação:")
print(correlation_matrix)
print("\n")
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlação entre Variáveis')
plt.show()
```

Fonte: Elaborada pelos autores (2024).

Outro método de análise interessante é verificar a distribuição da variável "taxa de engajamento", com o objetivo de identificar a existência de outliers. Isso é importante porque, caso sejam confirmados, e não tratados, os outliers podem impactar negativamente a eficiência do modelo, influenciando de forma

desproporcional e fornecendo métricas que não representam bem o conjunto de dados. Ao realizar esse método, foi possível identificar a existência de outliers, pois o gráfico apresentava um pico distinto em algumas áreas.

Figura 3 - outliers

```
# A taxa de engajamento está concentrada em um intervalo específico, pois podemos observar que ocorre picos
# Existe outliers por existir valor muito alto em comparação com os demais.
plt.figure(figsize=(10,6))
sns.histplot(analise_insta['Taxa_Engajamento'], kde = True)
plt.title('Distribuição da váriavel')
plt.xlabel('valor')
plt.ylabel('Frequência')
plt.show()
print (analise_insta ['Taxa_Engajamento']. describe())
```

Fonte: Elaborada pelos autores (2024).

Logo depois, buscamos tratar esses outliers, pois não queremos que eles impactem negativamente nossos resultados finais. Dentre as alternativas disponíveis, optamos por remover os outliers da coluna "taxa de engajamento", utilizando o intervalo interquartil (IQR). Escolhemos esse método por ser um dos mais utilizados quando os dados se distribuem de maneira aproximadamente normal.

Como o próprio nome sugere, o IQR divide os dados em quatro partes, sendo que cada uma delas contém 25% dos dados. Ele se baseia nos quartis e não na média, sendo mais eficaz quando os dados são assimétricos. Dessa forma, ele consegue identificar valores atípicos, ou seja, os outliers, como ilustrado na figura abaixo.

Figura 4 - Remoção dos outliers

```
#Remover outliers calculado o intervalo interquartil (IQR)
Q1 = analise_insta['Taxa_Engajamento'].quantile(0.25)
Q3 = analise_insta['Taxa_Engajamento'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
analise_insta = analise_insta[(analise_insta['Taxa_Engajamento'] >= lower_bound) & (analise_insta['Taxa_Engajamento'] <= upper_bound)]
```

Fonte: Elaborada pelos autores (2024).

Logo após o tratamento dos dados, removendo os outliers, buscamos verificar se eles realmente haviam sido excluídos. Para isso, visualizamos novamente o gráfico com o comando print(analise_insta['Taxa_Engajamento'].describe()). Verificamos que os dados já estavam livres dos valores atípicos.

Ainda com o objetivo de evitar eventuais problemas, realizamos um método para remover espaços invisíveis. Esses espaços, caso existentes, poderiam gerar conflitos em referências futuras, como ao acessar ou renomear as colunas. A técnica utilizada foi: analise_insta.columns = analise_insta.columns.str.strip().

Após realizar todos esses refinamentos nos dados, iniciamos a preparação para o treinamento, selecionando as variáveis dependentes e independentes. É com base nessas variáveis (X) que o modelo irá prever o valor da variável alvo, que neste caso é Y, o nosso target. As variáveis independentes escolhidas foram: média de curtidas, seguidores e postagens. A escolha dessas variáveis se deu devido ao gráfico de correlação citado anteriormente, no qual identificamos uma forte correlação entre elas.

O target foi definido com base na análise de correlação que fizemos previamente, e a partir dela notamos que a taxa de engajamento possui a maior correlação com as demais variáveis. Como mencionado antes, Y é o nosso target, ou seja, o valor que queremos prever, por isso buscamos as variáveis mais correlacionadas com ele para o treinamento. A seguir, apresentamos o código.

Figura 5 - Variáveis dependentes e independentes

```
# variáveis dependente e independentes
#Normatização dos dados utilizando os dados da correlação para um bom treinamento do modelo.

x = analise_insta[['Media_Curtidas', 'Seguidores', 'Postagens']]

# Target (variável alvo: se o solicitante é inadimplente ou não)
y = analise_insta['Taxa_Engajamento']
```

Fonte: Elaborada pelos autores (2024)

Posteriormente, realizamos a divisão do conjunto de dados para que pudéssemos construir e avaliar um modelo de aprendizado de máquina, lembrando que X e Y já contêm as variáveis dependentes e independentes, conforme mencionado anteriormente. Os dados foram divididos em 30% para teste e 70% para treinamento. Somente após essa etapa conseguimos treinar nosso modelo, como representado na imagem a seguir.

Figura 6 - Target

```
x = analise_insta[['Media_Curtidas', 'Seguidores', 'Postagens']]

# Target (variável alvo: se o solicitante é inadimplente ou não)
y = analise_insta['Taxa_Engajamento']
```

Fonte: Elaborada pelos autores (2024).

Na parte posterior, realizamos o treinamento da nossa regressão. Optamos por usar o modelo de regressão Lasso para ajustar os dados. Decidimos usar esse modelo por ser uma técnica que elimina variáveis menos importantes, reduzindo a dimensionalidade. Isso significa que ele ajuda a evitar o overfitting e facilita a interpretação do modelo.

O overfitting é um problema comum em modelos de aprendizado de máquina, quando o modelo aprende não apenas os padrões relevantes dos dados, mas também os ruídos. Como mostra os códigos a seguir: lasso_model = Lasso(alpha=0.1), lasso_model.fit(x_treino, y_treino).

O passo seguinte após o treinamento é realizar a previsão. Esse processo é responsável por armazenar os resultados da previsão em y_treino_pred, enquanto y_teste_pred armazenará os resultados do teste. Em seguida, foi iniciado o cálculo das métricas MSE, RMSE, R^2 e MAE, que são utilizadas para avaliar o desempenho do nosso modelo. Para que essas métricas forneçam resultados o mais precisos possível, é necessário que os dados tenham sido tratados da melhor maneira possível.

Fatores como valores nulos, tratamento de outliers e remoção de variáveis altamente correlacionadas são essenciais para o sucesso do modelo. Por exemplo, em nosso modelo, não utilizamos a coluna Media_Curtidas_Novas, pois ela apresentou dados muito semelhantes à coluna Media_Curtidas e tinha alta correlação com ela. Essa medida contribuiu para um melhor resultado final.

Figura 7 - Previsão e métricas

```
#Previsão

y_treino_pred = lasso_model.predict(x_treino)
y_teste_pred = lasso_model.predict(x_teste)

#calcular metricas
train_mse = mean_squared_error(y_treino, y_treino_pred)
test_mse = mean_squared_error(y_teste, y_teste_pred)
train_rmse = np.sqrt(train_mse)
test_rmse = np.sqrt(test_mse)
train_r2 = r2_score(y_treino, y_treino_pred)
test_r2 = r2_score(y_teste, y_teste_pred)
train_mae = mean_absolute_error(y_treino, y_treino_pred)
test_mae = mean_absolute_error(y_teste, y_teste_pred)
print(f"MSE: {test_mse}")
print(f"RMSE: {test_rmse}")
print(f"R2: {test_r2}")
print(f"MAE: {test_mae}")
```

Fonte: Elaborada pelos autores (2024).

Além do que já havíamos feito anteriormente, também realizamos o cálculo do Erro Absoluto Médio (MAE) para os dados de treino e teste, o qual também exibe o coeficiente do nosso treinamento. O motivo de termos utilizado essa métrica é proporcionar uma melhor clareza sobre os dados, permitindo identificar se, mesmo após todo o tratamento que realizamos, nosso dataset ainda continha dados atípicos, que poderiam impactar negativamente em nossos resultados.

Figura 8 - Cálculo de impacto

```
#calcular imapctor da regularização
train_mae = mean_absolute_error(y_treino, y_treino_pred)
test_mae = mean_absolute_error(y_teste, y_teste_pred)
print(f"Erro de treinamento: {train_mae}")
print(f"Erro de teste: {test_mae}")

print("c", lasso_model.coef_)
print("b", lasso_model.intercept_)
```

Fonte: Elaborada pelos autores (2024).

Finalizado esse processo, vimos a necessidade de visualizar esses dados tratados em forma de gráficos, mais especificamente para avaliar a existência de resíduos, já que eles têm impacto em nosso resultado final. Utilizamos dois modelos para essa análise: o primeiro usa os valores previstos de X, enquanto o segundo usa os valores reais de X. Não há a necessidade de usar ambos, mas achamos interessante observar a forma como cada um analisa os dados, por isso decidimos manter ambos. A seguir, apresentaremos os dois modelos.

Figura 9 - Plotagem 1

```
# plotagem da Heterocedasticidade, ele detecta se está havendo uma dispersão ao longo que o  modelo se ajusta.
plt.scatter(y_teste, y_teste_pred)
plt.axhline(y=0, color='r', linestyle='--')
plt.xlabel('Valores Previstos')
plt.ylabel('Resíduos')
plt.title('Resíduos vs. Valores Previstos')
plt.show()
```

Fonte: Elaborada pelos autores (2024).

Figura 9 - Plotagem 2

```
#Visualização dos resultados, com valores independentes e juntos
plt.figure(figsize=(6, 4))
plt.scatter(y_teste, y_teste - y_teste_pred, c='blue', marker='o')
plt.axhline(y=0, color='r', linestyle='--')
plt.title('Resíduos')
plt.xlabel('Valores Reais')
plt.ylabel('Resíduos')
plt.show()
```

Fonte: Elaborada pelos autores (2024).

Por fim, realizamos a análise dos valores reais e previstos, buscando identificar se nosso modelo está prevendo corretamente os valores esperados. Verificamos se nossos dados foram tratados adequadamente, pois, caso isso não tenha sido bem executado, o modelo apresentará uma grande dispersão nos resultados. No entanto, tomamos todas as medidas necessárias para obter os melhores resultados possíveis.

Como mencionado anteriormente, realizamos a identificação e o tratamento de valores atípicos, incluindo valores nulos e outliers, além da utilização do modelo de regressão Lasso para simplificar o modelo, entre outros tratamentos. A seguir, apresentamos o código utilizado.

Figura 9 - Análise de valores

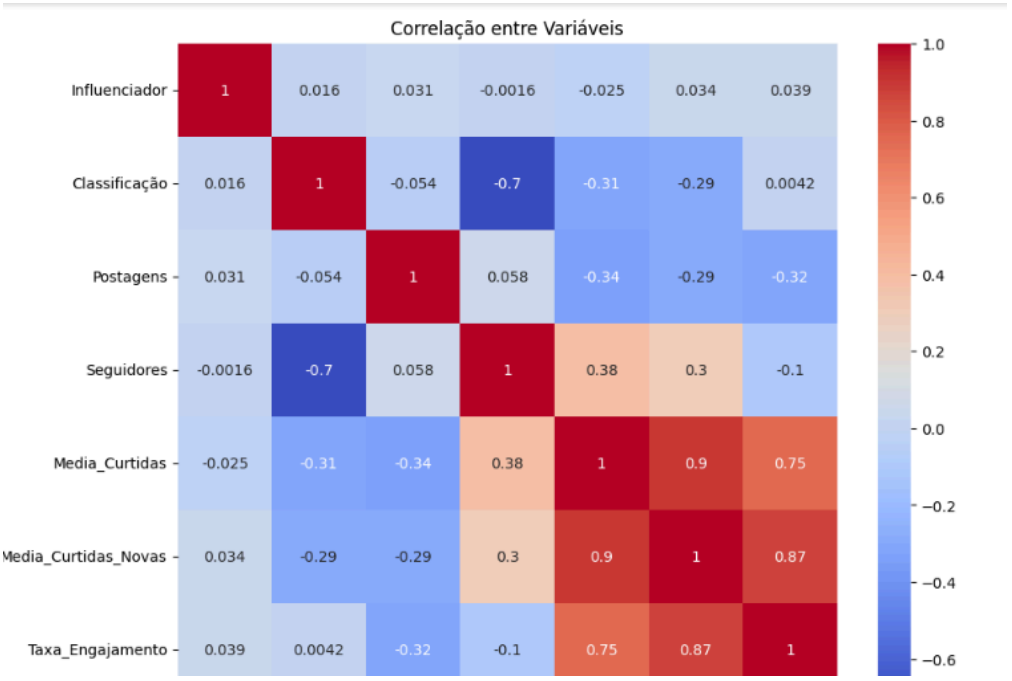
```
#Analise dos valores reais e os previstos
plt.scatter(y_teste, y_teste_pred)
m, b = np.polyfit(y_teste, y_teste_pred, 1)
plt.plot(y_teste, m * y_teste + b, color='red', linestyle='--')
plt.xlabel("Valores Reais")
plt.ylabel("Valores Previstos")
plt.title("Valores Reais vs. Valores Previstos")
plt.show()
```

Fonte: Elaborada pelos autores (2024).

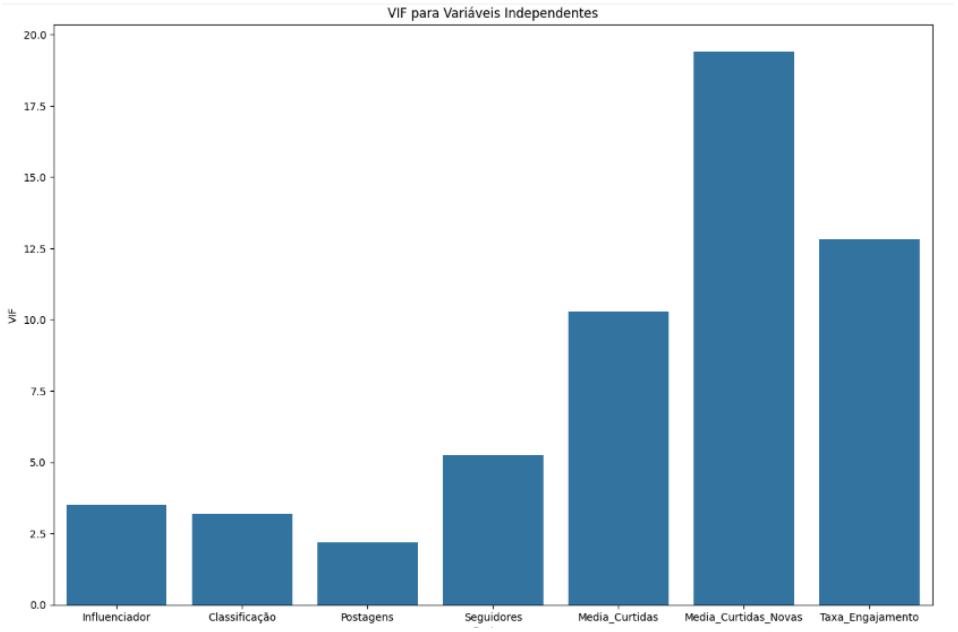
Resultados

Ao longo deste trabalho, a meta foi aplicar técnicas de análise de dados, modelagem e tratamento, focando particularmente no modelo de Regressão Linear, buscando prever a taxa de engajamento de influenciadores da rede social Instagram. Nos aprofundamos em métodos de tratamento, como limpeza dos dados, preparação e manipulação de dados categóricos.

Houve também análise exploratória, que nos ajudou a encontrar relações significativas entre as variáveis, sendo de extrema importância para a seleção das variáveis independentes. Como mostram os gráficos a seguir.

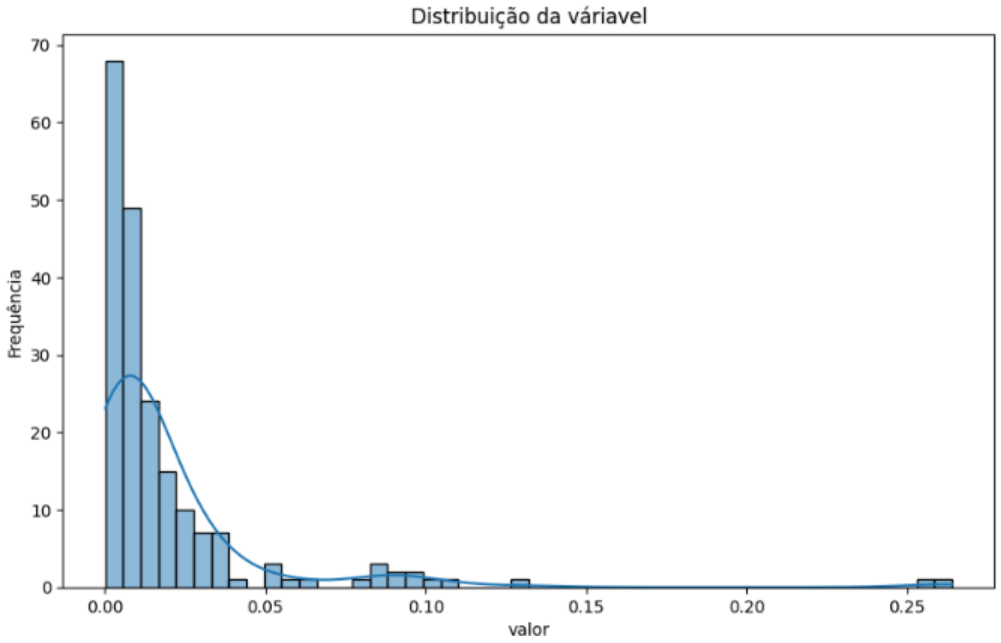


Fonte: Elaborada pelos autores (2024).



Fonte: Elaborada pelos autores (2024).

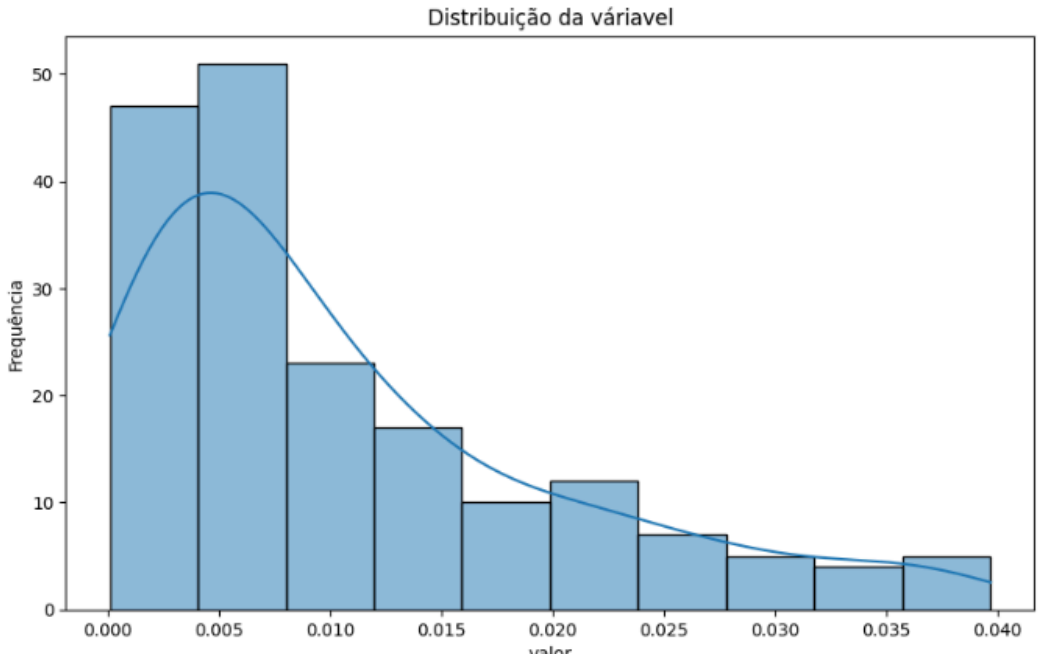
Após obter esse resultado da relação entre as colunas, decidimos explorar ainda mais nosso dataset em busca da presença de outliers, pois sua presença impacta negativamente em nossos resultados.



Fonte: Elaborada pelos autores (2024).

Com base no resultado acima, vimos a necessidade de realizar o tratamento outliers, já que como podemos ver o gráfico aponta a existência de dados mudo distante dos demais e com um valor significativo. Visando o melhor resultado

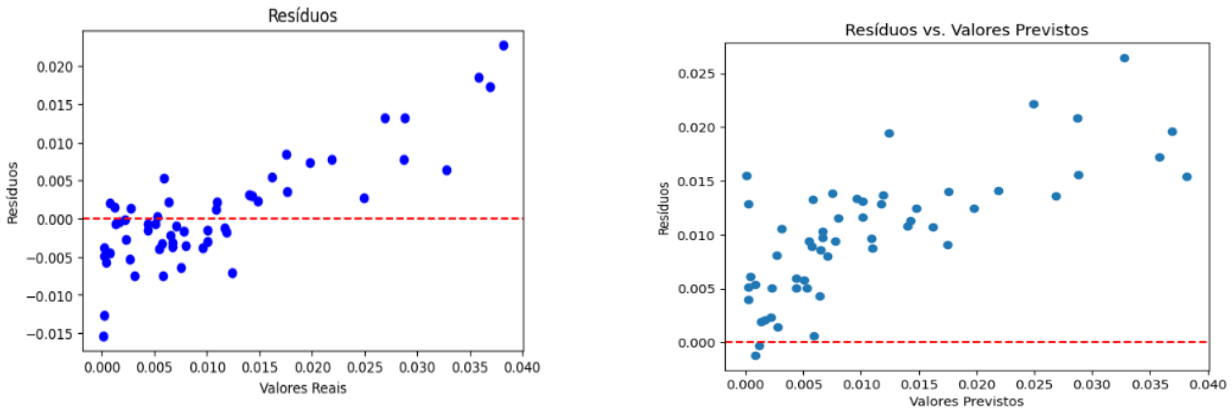
Remover outliers calculado o intervalo interquartil (IQR). Como podemos visualizar no gráfico seguinte.



Fonte: Elaborada pelos autores (2024).

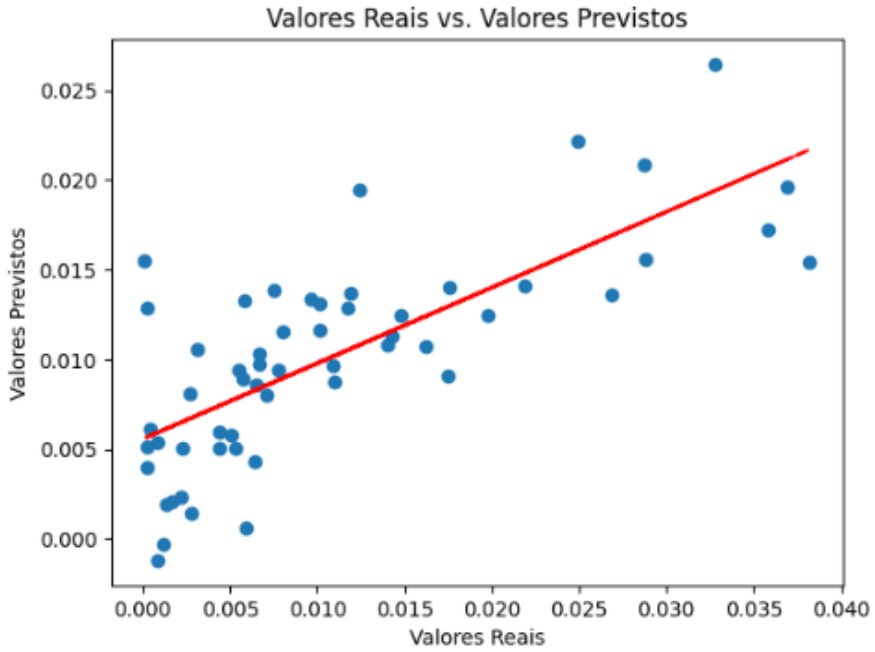
Seguimos realizando o tratamento dos dados, como a escolha do modelo de regressão Lasso, um modelo muito interessante, pois não só nos ajudou a avaliar o desempenho do nosso modelo, mas também a evitar o overfitting.

Obtivemos aprendizados muito importantes ao longo deste trabalho. Durante esse percurso, enfrentamos desafios como a presença de dados nulos e outliers, mas, por fim, conseguimos realizar o tratamento de forma eficaz, promovendo assim um bom resultado para a performance do modelo, como podemos visualizar nos gráficos a seguir.



Fonte: Elaborada pelos autores (2024).

A utilização dessas técnicas de pré-processamento e a escolha das variáveis dependentes e independentes nos proporcionaram uma melhor precisão, sendo de grande importância para a construção de um modelo mais eficiente e preciso, como mostra o gráfico abaixo.



Fonte: Elaborada pelos autores (2024).

Discussão

Embora tenhamos obtido um bom resultado, ao longo do percurso encontramos algumas dificuldades. Algumas vezes nos vimos precisando retroceder para, somente depois, conseguir dar continuidade ao tratamento dos dados. Algumas abordagens utilizadas só foram realmente elaboradas após o resultado do treinamento. Nos deparamos com resultados muito dispersos, com muitos ruídos, e fomos obrigados a realizar um novo tratamento dos dados. No entanto, conseguimos implementar métodos que nos trouxeram uma previsão mais eficaz. Uma dessas decisões foi a utilização do modelo de regressão Lasso, que nos possibilitou obter um melhor resultado final.

Conclusão e Trabalhos Futuros

Este trabalho nos proporcionou uma compreensão mais aprofundada das etapas necessárias para a construção de um modelo mais eficaz. Realizamos uma análise exploratória dos dados, buscando o melhor tratamento, assim como a aplicação de técnicas de modelagem para a interpretação das informações. Essas abordagens, sem dúvida, servirão como base para trabalhos futuros.

Como sugestão para trabalhos posteriores, consideramos interessante substituir um relatório tão detalhado por uma apresentação em vídeo mais técnica, explicando os métodos e as abordagens utilizadas na elaboração do projeto.

Referências

Aprendizado de Máquina. [s.l: s.n.]. Disponível em: <https://www.maxwell.vrac.puc-rio.br/47049/47049_3.PDF>.

Castanha, R. C. G. (2021). A ciência de dados e a cientista de dados. AtoZ: novas práticas em informação e conhecimento, 10(2), 1 – 4. Recuperado de: http://dx.doi.org/10.5380/atoz.v10i2.79822

ANSARI, Y. Linear Regression (An Overview). Disponível em: <https://medium.com/@novus_afk/linear-regression-an-overview-13d37a6bc4dd>.