Relatório Técnico: Implementação e Análise do Algoritmo de Regressão Linear

Lucca Fernandes Trancoso Nolasco, Rodrigo Ferreira Bento Aguiar

November 16, 2024

Resumo

Este relatório técnico apresenta o objetivo, metodologia e principais resultados de um projeto de análise com foco na implementação do algoritmo de Regressão Linear. Primeiramente, é realizada uma análise exploratória dos dados, seguida pela implementação do algoritmo, validação e ajustes de hiperparâmetros para otimizar o modelo e avaliar o seu desempenho com métricas apropriadas.

Introdução

O objetivo deste relatório é apresentar uma documentação e explicação da modelagem e treino de um modelo de Regressão Linear para resolver um problema de inferência sobre a taxa de engajamento dos principais influenciadores do Instagram. O presente trabalho consiste numa avaliação proposta pelo CEPEDI na Reseidência em Software Restic36, para a trilha de Ciência de Dados. A opção pelo algoritmo de Regressão Linear se deu por diversos fatores: apesar da capacidade de realizar regressão, o algoritmo de KNN é mais comumente usado em classificação; os dados aparentam seguir uma relação linear.

0.1 Contexto do Problema

O objetivo do modelo proposto é prever a taxa de engajamento de determinado perfil no Instagram baseado em outros atributos. Essa aplicação apresenta grande utilidade pois pode ser usada por patrocinadores, times de social-media dos influenciadores e equipes de marketing para decidir quando e quais perfis apresentam um melhor investimento.

0.2 Descrição do Conjunto de Dados

O conjunto de dados utilizado é o *Top Instagram Influencers Data (Cleaned)*, obtido no Kaggle por meio do link: https://www.kaggle.com/datasets/surajjha101/top-instagram-influencers-data-cleaned.

Algumas colunas numéricas estão representadas como Strings, para abreviação (200.000 vira 200k). Portanto, é necessário esse pré-processamento. As colunas do data-set são:

1. **rank:** É o rank do influenciador, onde 1 é o maior e 200 é o menor;
2. **channel_info:** É o nome do perfil no Instagram;
3. **influence_score:** Uma métrica de 0 a 100, expressando a porcentagem de usuários que reagirão a suas postagens;
4. **posts:** É o número de postagens até o momento;
5. **followers:** É o número de seguidores até o momento;
6. **60_day_eng_rate:** A taxa de engajamento do público neste perfil nos últimos 60 dias;
7. **new_post_avg_like:** Média de likes nos posts novos. Uma tupla possui *null*;
8. **total_likes:** Total de likes no perfil (em bilhões);
9. **country:** País de origem do perfil. Muitas tuplas com *null*.

Metodologia

Em nosso notebook, optamos por mostrar a evolução do modelo. Primeiramente fizemos o treinamento sem alterar os dados, para ver o nível de *overfitting* e o comportamento da reta. Também fizemos isso para exibir gráficos relevantes. À partir disso, mostramos o modelo final, que está separado de sua fase "evolutiva".

0.3 Análise Exploratória

Os dados foram obtidos do *Kaggle*. Inicialmente, transformamos o arquivo CSV em um Dataframe do Pandas, para melhor manipulação. Ao verificar sua head, obtemos:

Rank	**Channel Info**	**Influence Score**	**Posts**	**Followers**
1	cristiano	92	3.3k	475.8m
2	kyliejenner	91	6.9k	366.2m
3	leomessi	90	0.89k	357.3m
4	selenagomez	93	1.8k	342.7m
5	therock	91	6.8k	334.1m

Table 1: Primeira parte da tabela: Informações básicas

Avg Likes	**60 Day Eng Rate**	**Total Likes**	**Country**
8.7m	1.39%	29.0b	Spain
8.3m	1.62%	57.4b	United States
6.8m	1.24%	6.0b	NaN
6.2m	0.97%	11.5b	United States
1.9m	0.20%	12.5b	United States

Table 2: Segunda parte da tabela: Métricas de engajamento

Analisando o DataFrame, percebemos duas coisas: os valores numéricos são representados com Strings, que devem ser tratadas (porcentagens e números absolutos); existe um *NaN* já na Head. A coluna "country" possui diversos *NaNs*. Extraímos uma tabela apenas com as linhas onde não havia um valor para essa coluna, e observamos que essa tabela possuía 62 tuplas, o equivalente a 31% dos dados. Portanto, optamos por remover a coluna "country" para melhor funcionamento do algoritmo e para melhor análise dos dados. As outras alternativas eram: remover 31% da noss abasse de dados; adicionar os países manualmente. Ambas opções não são eficazes, a primeira por remover muitos

dados, a segunda por exigir um trabalho manual que dificultaria o uso do modelo em aplicações automáticas que desejassem prever esse target.

Para o tratamento das linhas numéricas, criamos um algoritmo em *Python* que extrai o valor numério absoluto e transforma o respectivo símbolo (k, m, b, %) num valor que possa ser trabalhado. Após isso, ocorre efetivamente a **análise exploratória dos dados**.

É possível observar que os valores numéricos possuem escalas discrepantes, o que torna a normalização uma técnica essencial. Sua importância será demonstrada ao analisar o modelo inicial sem alteração na base de dados. Antes disso, fizemos o *Box Plot* das variáveis para identificar outliers:

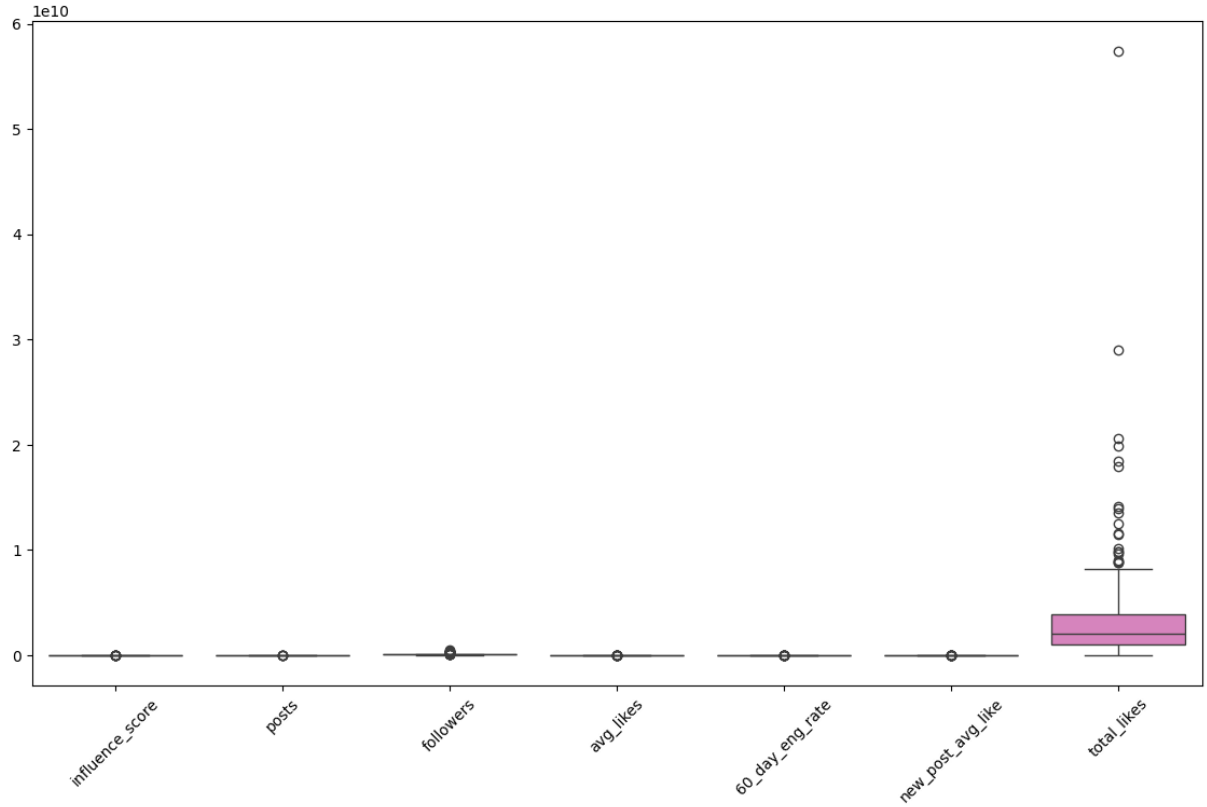

Figure 1: Box Plot das variáveis. Fonte: Própria

Observa-se que a variável *total_likes* possui uma variação muito mais ampla que as outras, já que possui o *Box Plot* visível em comparação com a dos outros, que tornou-se uma linha. Além disso, essa coluna possui uma quantidade elevada de outliers, com valores extremamente altos. Essa discrepância nos motivou a plotar a distribuição de frequência das variáveis, para verificar onde se concentram .

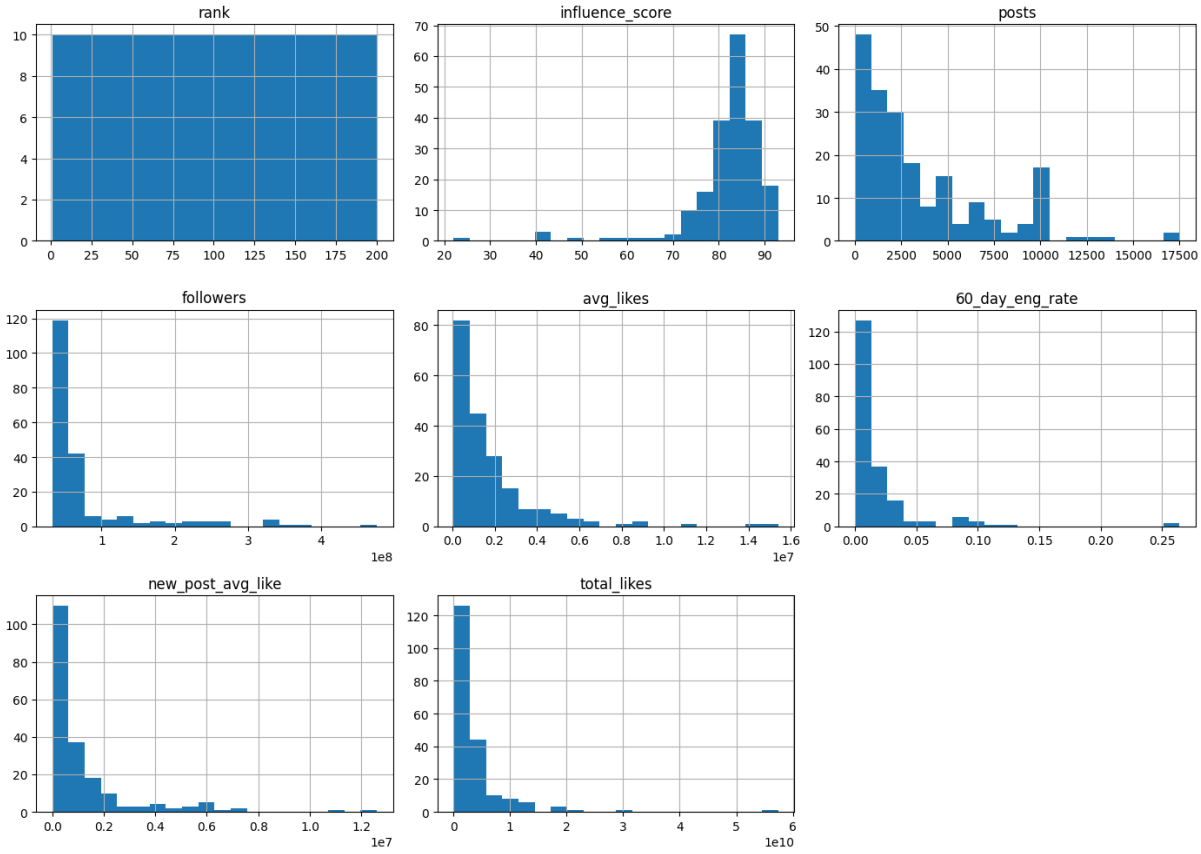

Figure 2: Distribuição de Frequência das Variáveis. Fonte: Própria.

O *rank* possui uma concentração total pois possui um de cada, de 1 a 200. O *influence_score* se concentra por volta de 85. Já as outras variáveis concentram-se nos valores mais baixos, e seus outliers são sempre para quantidades maiores. Por meio da ordem de grandeza, observamos o quanto a variável *total_likes* possui uma variação extrema. Além disso, ao realizar uma análise via IQR nas variáveis, observamos a quantidade de outliers em cada uma. Vale reassaltar que a soma não representa a quantidade total de outliers, tendo em vista que muitas amostras o são em mais de uma variável, e portanto, são contadas mais de uma vez.

Variável	**Quantidade**
rank	0 outliers
influence_score	8 outliers
posts	3 outliers
followers	27 outliers
avg_likes	12 outliers
60_day_eng_rate	16 outliers
new_post_avg_like	20 outliers
total_likes	15 outliers

Table 3: Outliers por variável

Por fim, foi feito um gráfico de correlação entre as variáveis. Observou-se que *avg_likes* e *new_post_avg_like* possuem uma forte correlação positiva com nosso target, enquanto

que *posts* possui uma correlação negativa moderada. As outras variáveis não apresentam correlação relevante.

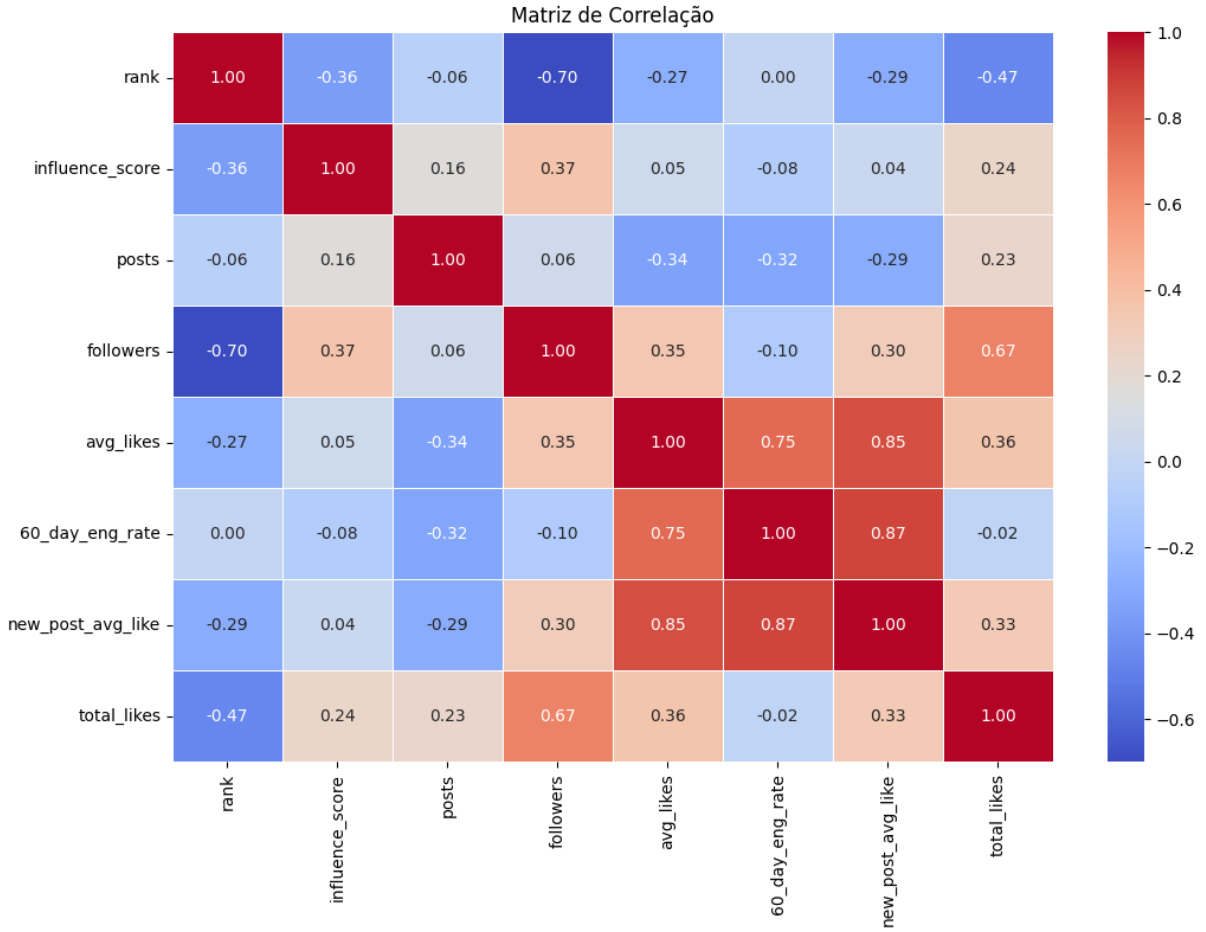

Figure 3: Matriz de Correlação. Fonte: Própria

0.4 Implementação Preliminar do Algoritmo

Realizamos primeiramente a implementação do algoritmo sem alteração na base de dados, para verificar as estatísticas e onde as melhorias deveríam ser aplicadas. Por isso, a descrição detalhada da implementação encontra-se na próxima seção "Implementação Final do Algoritmo" e aqui, as métricas preliminares e conclusões.

Métricas	**Treino**	**Teste**
R^2	0.913351304	0.923989345
MSE	0.0001061678091	4.68814715e-05
MAE	0.00607152630	0.0050298049

Table 4: Métricas Treino x Teste

A comparação das métricas mostra que o modelo consegue generalizar bem. Porém, as os valores não foram normalizados, e pode haver overfitting no algoritmo. Por isso, analisamos as curvas de aprendizado, e percebemos que o desempenho do modelo piora por volta de 110 amostras.

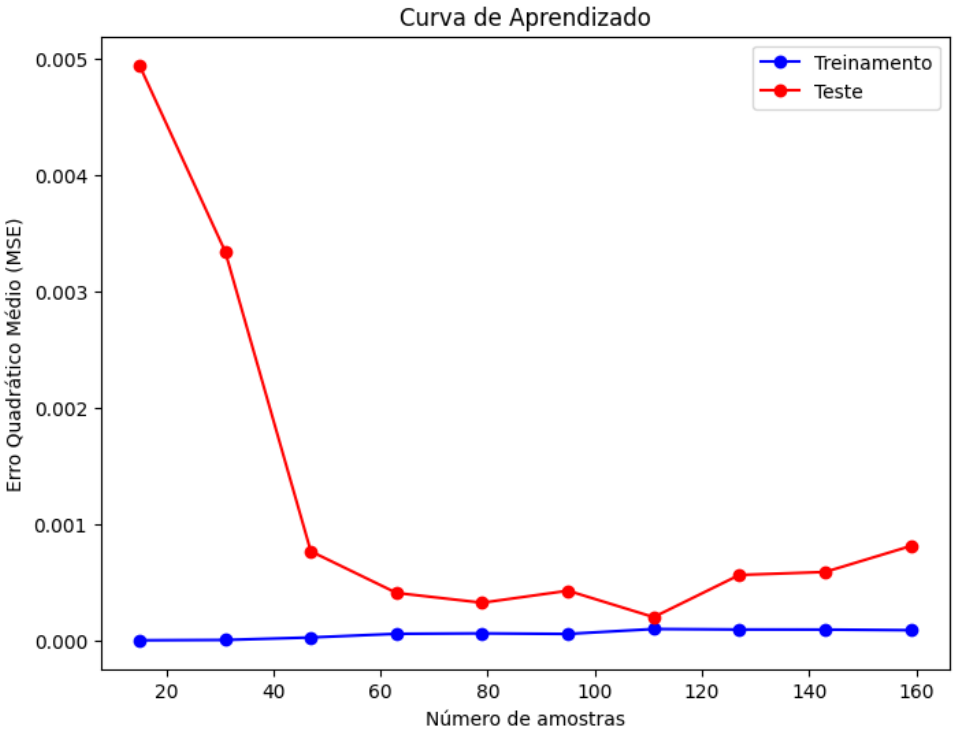

Figure 4: Curva de Aprendizado preliminar. Fonte: própria

0.5 Implementação Final do Algoritmo

Para a implementação do algoritmo, utilizou-se a biblioteca *Scikit-Learn*. Para a manipulação de valores numéricos e dataframes, foram empregadas as bibliotecas *Pandas* e *NumPy*. Já para a visualização, as bibliotecas *Matplotlib* e *Seaborn* foram as escolhidas. Os dados foram obtidos no Kaggle, por meio da biblioteca *KaggleHub*.

Na etapa inicial, os dados foram transformados de um arquivo CSV para um dataframe do Pandas. A variável *country* foi removida, assim como os outliers, eliminados por meio do método IQR. Dados numéricos representados como strings foram devidamente convertidos utilizando a função mencionada anteriormente. Em seguida, os valores numéricos foram normalizados com o *MinMaxScaler*, ajustando-os para o intervalo entre 0 e 1. Após a remoção dos outliers, restaram apenas 133 amostras, uma quantidade limitada para treinamento e teste. Para ampliar a base de dados, foram geradas variáveis artificiais: 50% das tuplas foram selecionadas aleatoriamente, e um ruído de 3% foi adicionado a essas amostras, criando um novo dataframe que foi concatenado ao original, resultando em um total de 199 amostras.

Posteriormente, os dados passaram por uma Análise de Componentes Principais (PCA). Após vários testes, identificou-se que a quantidade ideal de componentes para o modelo era 5. Com 4 componentes, a variância explicada acumulada não era suficiente, enquanto 5 componentes permitiram um desempenho semelhante ao modelo ideal com todas as variáveis.

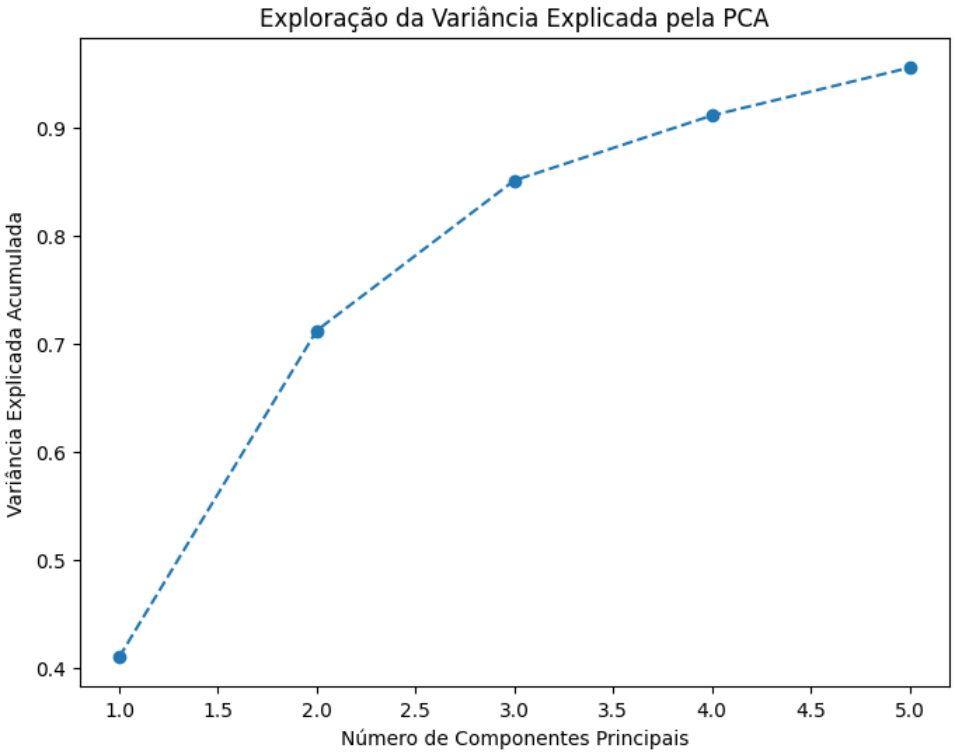

Figure 5: Variância Explicada Acumulada (mais de 95%). Fonte: Própria

Após isso, os dataframes com as componentes e com o target foram divididos entre treino e teste (80% treino e 20% teste). Os resultados do modelo serão explorados na seção "Resultados".

0.6 Validação e Ajuste de Hiperparâmetros

Para validação e teste do modelo, foram escolhidas as seguintes métricas e fatores:

- R^2 (Coeficiente de Determinação): Mede a proporção da variabilidade na variável dependente em relação às variáveis independentes;
- MSE (Erro Quadrático Médio): Mede a média dos quadrados das diferenças entre os valores observados e os valores previstos pelo modelo. Por ser quadrático, os valores são todos positivos, e erros maiores são mais penalizados;
- MAE (Erro Absoluto Médio): Mede a média das diferenças absolutas entre os valores reais e os valores previstos. Dá uma medida linear, e não penaliza erros maiores.
- Validação Cruzada (5 Folds):Divide os dados em folds (nesse caso, 5), e usa alguns para treinar o modelo e outros para testá-lo, o processo é repetido 5 vezes. Garante um teste mais robusto ao modelo;
- Análise de Resíduos vs Valores previstos: Caso os resíduos sigam um padrão nesse gráfico, pode significar a presença de *overfitting*;
- Histograma dos Resíduos: Uma distribuição normal nos resíduos aponta falta de *overfitting*;

- QQ Plot dos Resíduos: Resíduos próximos à reta apontam uma distribuição próxima da normal;
- Distribuição do Erro absoluto: mostra a quantidade de erros e como o modelo erra;

Resultados

O R^2, o MSE, o MAE e a validação cruzada apontam valores promissores, muito semelhantes aos valores de treino do modelo completo. Caso não haja *overfitting*, o modelo está bem treinado. Média dos scores de validação cruzada baseados em MSE: 0.00594671088.

Métricas	**Treino**	**Teste**
R^2	0.916448608	0.912623927
MSE	0.004880089	0.006371481
MAE	0.0494520440	0.057488486

Table 5: Métricas Treino x Teste

Observa-se que tanto no treino quanto no teste, os erros são relativamente baixos e próximos, e o coeficiente de determinação é alto e próximo. Isso é demonstrado pela análise da curva de aprendizado. Agora, a curva para o teste vai diminuindo com o aumento dos dados e, ao analisar a ordem de grandeza do eixo Y, observa-se o quão próximo está da curva de teste em relação ao gráfico do modelo preliminar.

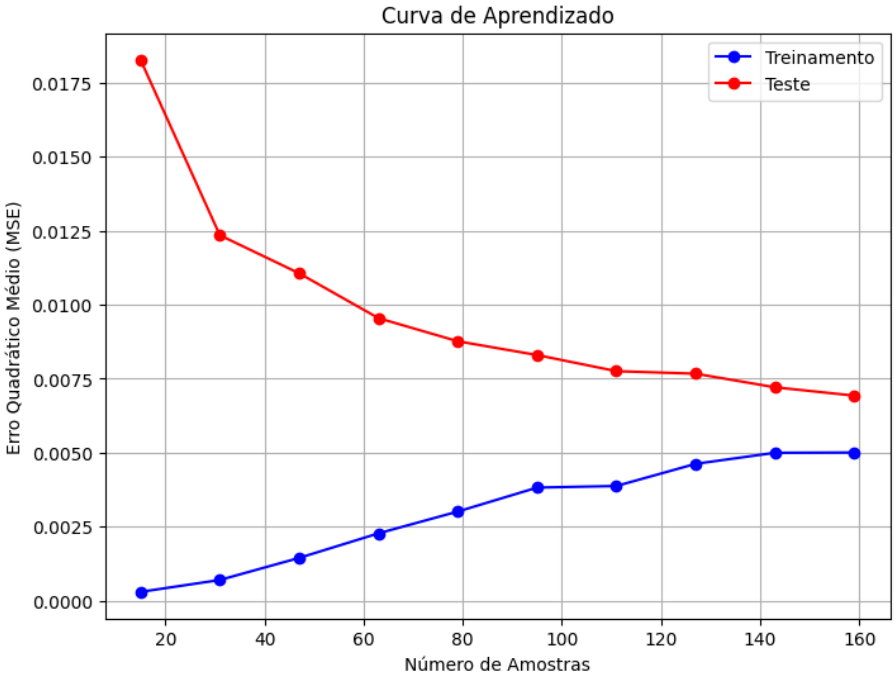

Figure 6: Curva de aprendizado final. Fonte: Própria

Resta agora analisar o modelo quanto à existência de *overfitting*. Analisando a os resíduos vs valores previstos, bem como o histograma e o QQ Plot dos mesmos, o modelo aparenta estar bem treinado e calibrado.

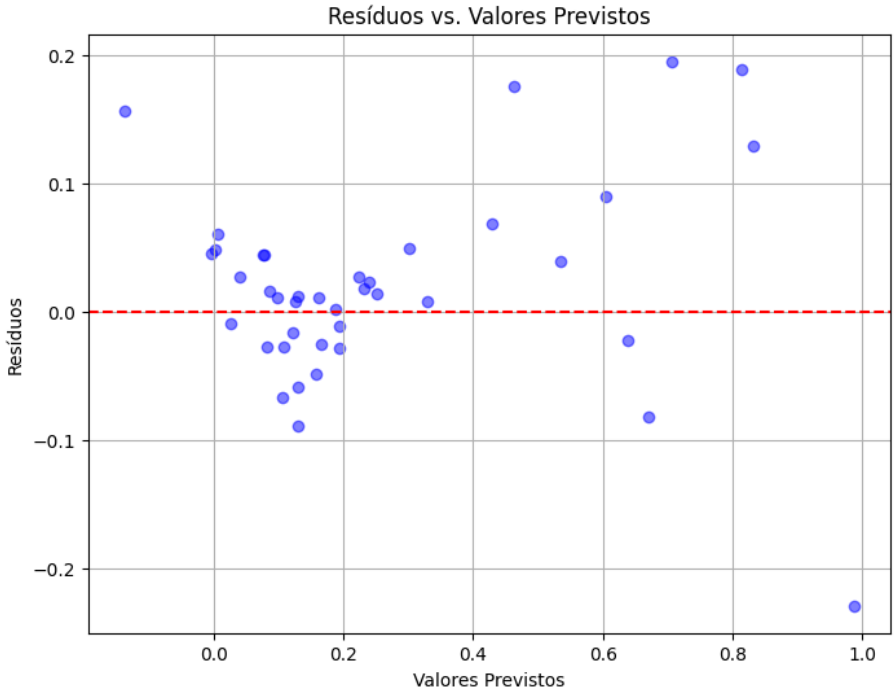

Figure 7: Resíduos vs Valores previstos. Fonte: Própria

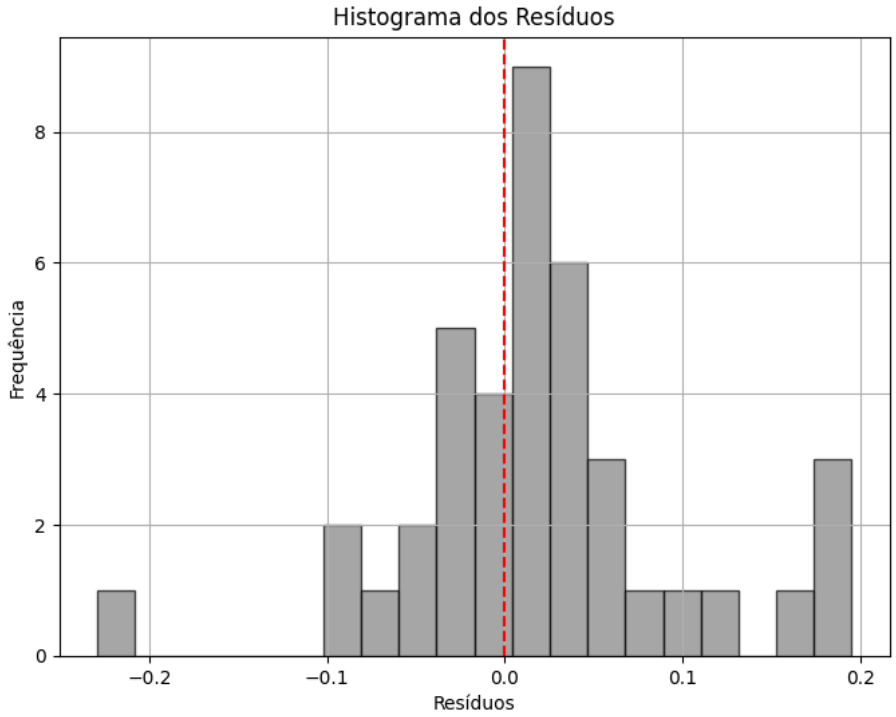

Figure 8: Distribuição de Resíduos (semelhante à Normal). Fonte: Própria

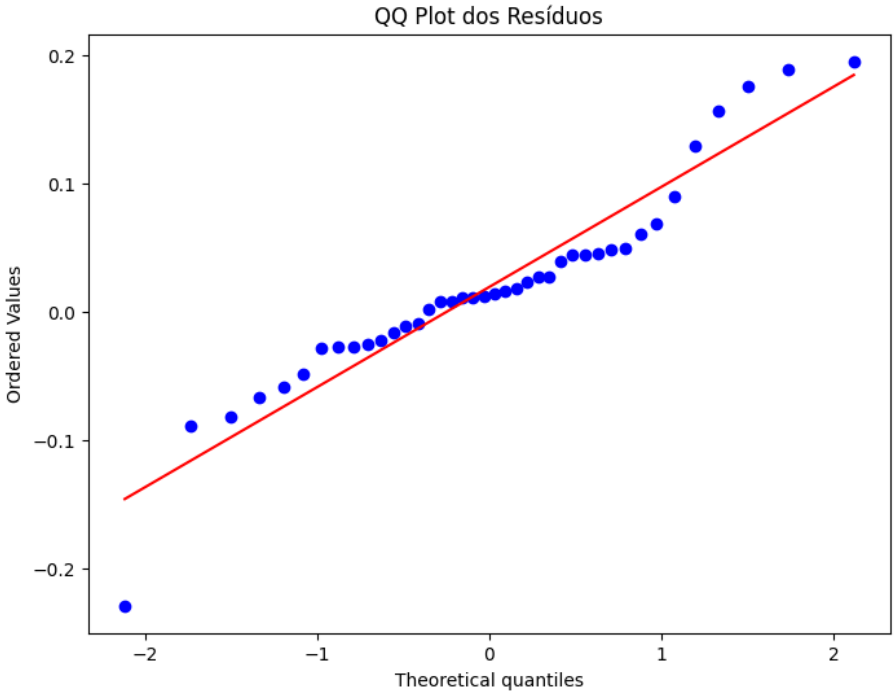

Figure 9: Próximo da reta é próximo da Distribuição Normal. Fonte: Própria

Discussão

Um dos desafios ao realizar a Análise de Componentes Principais é a perda da capacidade de identificar diretamente o impacto de cada variável sobre o target. No entanto, é possível analisar os coeficientes das componentes principais para obter uma estimativa aproximada da influência de cada parâmetro, considerando também a porcentagem de variância explicada por cada componente.

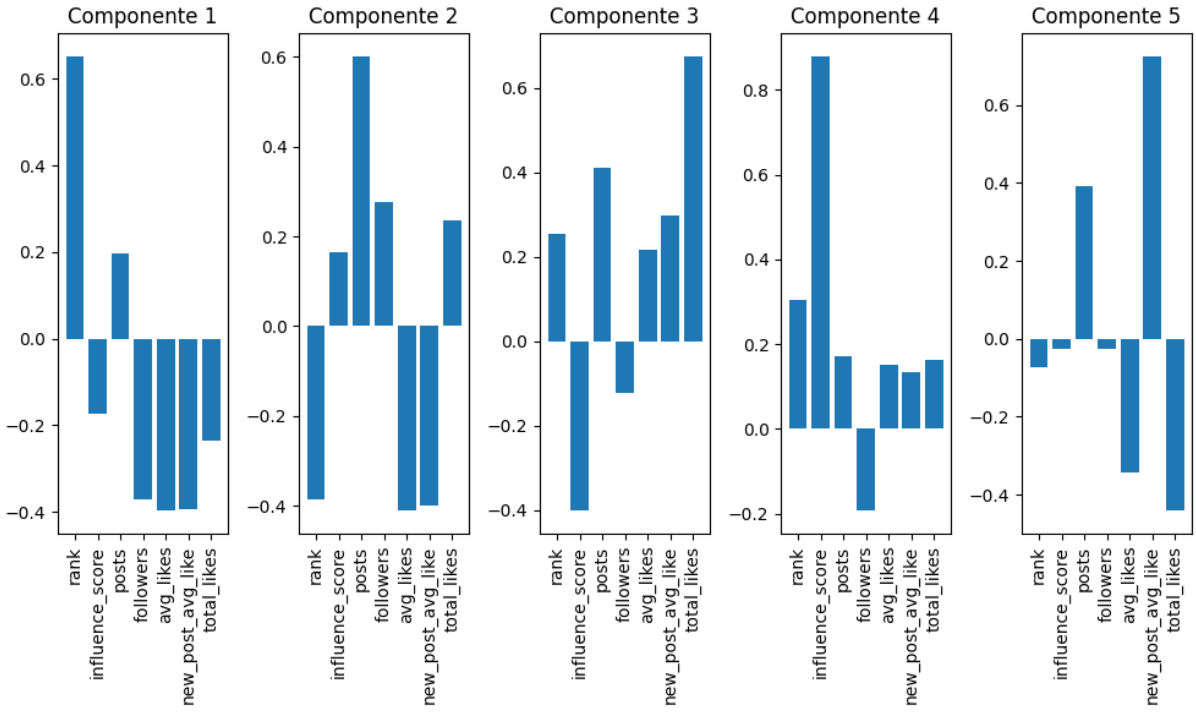

Figure 10: Fonte: Própria

Como demonstrado no gráfico da figura 5, as três primeiras componentes explicam mais de 80% da variância, tornando-se as mais relevantes para o modelo. Ao examinar os coeficientes das variáveis nessas componentes, pode-se concluir, de forma mais abstrata, que os fatores mais importantes para a taxa de engajamento são, em ordem decrescente: popularidade e ranking geral; atividade da conta (número de postagens) e média de likes; e, por fim, o total de likes acumulados em relação à influência da conta (mais likes associados a menor influência indicam maior engajamento).

Contudo, como observado na figura 9 e na análise da distribuição do erro absoluto, o modelo demonstrou limitações na captura de outliers.

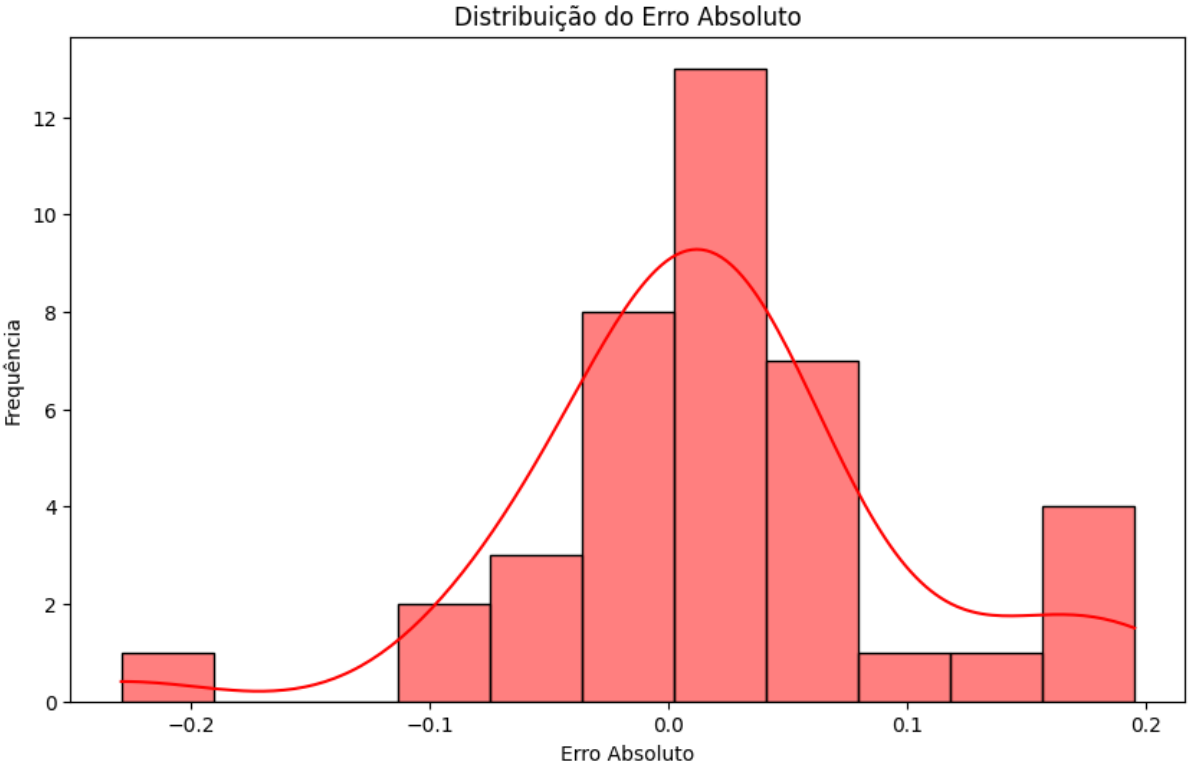

Figure 11: Fonte Própria

Conclusão e Trabalhos Futuros

A comparação entre o modelo preliminar e o final evidenciou a importância da normalização e da remoção de outliers na base de dados utilizada para o treinamento. Além disso, por meio da Análise de Componentes Principais (PCA), observou-se que, embora o modelo original incluísse 7 variáveis independentes (9, considerando o nome do perfil e o país), apenas 5 componentes foram suficientes para alcançar um coeficiente de determinação praticamente idêntico.

Contudo, percebeu-se que o modelo não foi eficaz na previsão de outliers. Essa limitação pode ser problemática no contexto analisado, já que redes sociais são ambientes propensos à ocorrência de "pontos fora da curva": celebridades; perfis com muitos seguidores, mas quase nenhum engajamento; e perfis grandes, com alta ou baixa frequência de postagens (como influenciadores ou marcas geridas por equipes de marketing).

Portanto, para uma análise estatística mais robusta, seria promissor combinar este modelo com outro mais calibrado para lidar com perfis que apresentem tais discrepâncias. Dessa forma, empresas que buscam patrocinados poderiam acessar dados mais representativos para perfis médios, grandes e gigantes.

Bibliography

[1] GRUS, Joel. *Data Science do Zero: Noções fundamentais com Python*. Tradução de Welington Nascimento. 2. ed. Rio de Janeiro: Alta Books 2021. Título Original: Data Science from Scratch. Publicado originalmente em Nova York: O'Reilly Media, 2019.