Relatório Técnico: Implementação e Análise do Algoritmo de Regressão Linear

Daniel de Souza Pereira
Victor Cesar do Rosario Calheiros

17 de novembro de 2024

Resumo:

O seguinte relatório irá mostrar todo o processo através do projeto que focamos em desenvolver sobre a taxa de engajamento dos principais influenciadores do Instagram, e para isso utilizamos a metodologia do algoritmo de regressão linear através de uma série de processos, como a análise exploratória dos dados, pré-processamento dos dados, implementação do modelo de regressão linear e a otimização e validação dele. Foi possível obter resultados satisfatórios do modelo, tendo um bom poder preditivo, e executando teste com diversos modelos diferentes até encontrar o que apresentou o melhor desempenho e consistência, com bons R^2, MSE e MAE, além de trazer estabilidade na validação cruzada.

Introdução:

Foi dado a missão de implementar e avaliar o desempenho do algoritmo de regressão Linear em um conjunto de dados real, então pensando nisso, buscamos uma situação da atualidade que se encaixava nos requisitos e foi chegado na conclusão de um processo de resolução do problema de inferência sobre a taxa de engajamento dos principais influenciadores do Instagram. Esse projeto foi feito utilizando da regressão linear por acreditar ser aquele que consiga estimar uma função que melhor se ajuste aos dados de entrada para prever a taxa de engajamento.

Todos os dados adquiridos e utilizados para poder criar o código com o modelo vieram de uma planilha csv com diversas informações úteis usadas para o processo que será explicado a seguir.

Metodologia:

Inicialmente, pegamos a planilha com diversas informações de influenciadores da plataforma do Instagram e separamos as informações importantes que serão utilizadas, divididas em:

- **posts**: Número de postagens (valores em string, com "k" indicando milhares).
- **followers**: Número de seguidores (também em formato de string com "m" ou "k").
- **avg_likes**: Média de curtidas por postagem (formato de string com "m" ou "k").
- **60_day_eng_rate**: Taxa de engajamento dos últimos 60 dias (em percentual, como string).
- **new_post_avg_like**: Média de curtidas em novas postagens.
- **total_likes**: Total de curtidas.

Primeiramente, convertemos as colunas (posts, followers, avg_likes, new_post_avg_like e total_likes) de contagem para valores numéricos, processo que também será feito com a coluna “60_day_eng_rate”, principal coluna a qual será realizada os cálculos para os valores dos modelos. Em seguida, é feita uma investigação das relações entre as variáveis principais, como:

Correlação entre seguidores e taxa de engajamento (followers vs. 60_day_eng_rate).

Média de curtidas por postagem e taxa de engajamento (avg_likes vs. 60_day_eng_rate).

Curtidas em novas postagens e taxa de engajamento (new_post_avg_like vs.60_day_eng_rate).

Com isso, o código vai testar os diferentes conjuntos de variáveis independentes para treinar o modelo e assim encontrar a melhor configuração. Nele usamos o SGDRegressor com uma taxa de aprendizado constante de 0.01 para realizar o gradiente descendente, além dos modelos L1 e L2 para observar o impacto da regularização sobre o ajuste, e o SelectKBest identifica as duas variáveis mais significativas, e o modelo de regressão linear é ajustado novamente para observar a performance, e o KFold sendo utilizado para a validação cruzada.

Nós vamos conseguir implementar o modelo do algoritmo de regressão linear inicialmente por sua função linear simples y = b0 + b1*x, sendo y a variável dependente e x a variável independente, então selecionamos as colunas independentes úteis (followers, avg_likes, new_post_avg_likes), e sendo usado o R^2, MSE e MAE como parâmetro para avaliar o desempenho geral dos modelos. Para os modelos baseados na regressão linear, também foi implementado a interpretação dos coeficientes, exibindo a contribuição de cada variável.

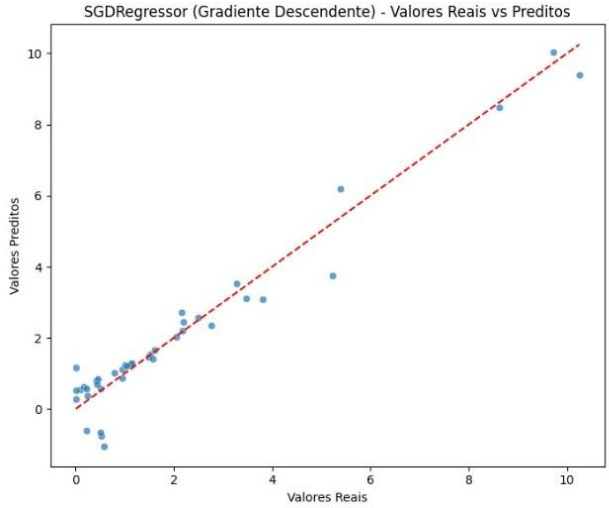
Ademais, o código também exibe a visualização gráfica dos resultados dos modelos, mostrando informações como os valores reais vs valores previstos e a distribuição de erros residuais, sendo um histograma dos erros para verificar se há desvios ou erros constantes no modelo.

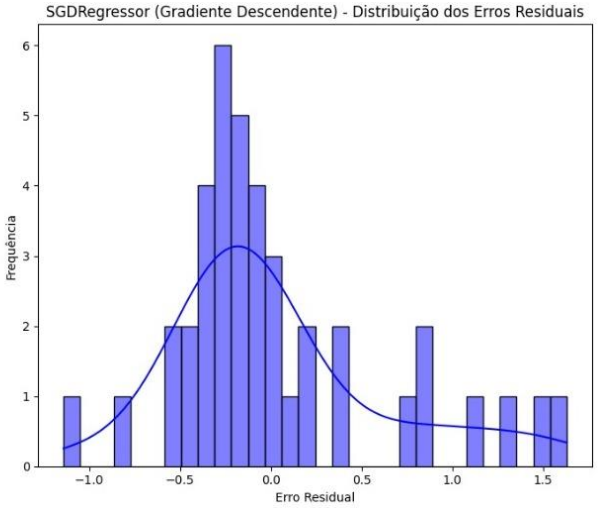
Resultados:

Pelo teste do código, foram testados 4 tipos de modelos diferentes pra encontrar os melhores resultados entre eles, sendo esses modelos, SGDRegressor (Gradiente Descendente), Lasso (L1 Regularization), Ridge (L2 Regularization) e Linear Regression com Seleção de Recursos. O parâmetro usado para medir os melhores resultados foram os que possuíam o maior R^2 e menores MSE (Erro Quadrátrico Médio) e MAE (Erro Absoluto Médio).

SGDRegressor (Gradiente Desconhecido):
R^2 Score: 0.9446
MSE: 0.3509
MAE: 0.4259

Coeficientes do Modelo:
followers: -1.4967
avg_likes: 0.4909
new_post_avg_like: 2.9409





Lasso (L1 Regularization):

R² Score: 0.9575

MSE: 0.2695

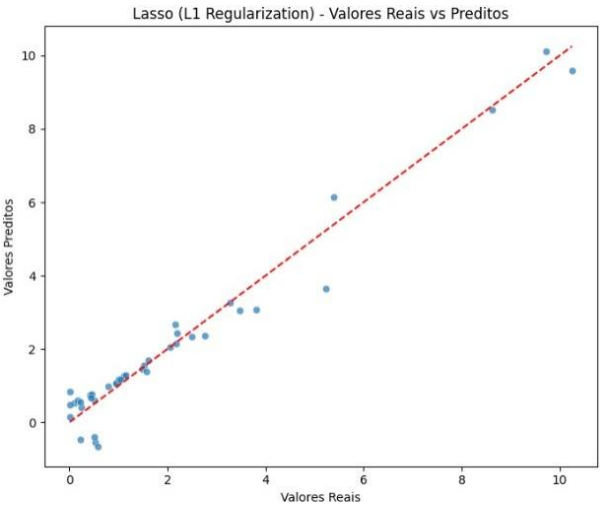
MAE: 0.3755

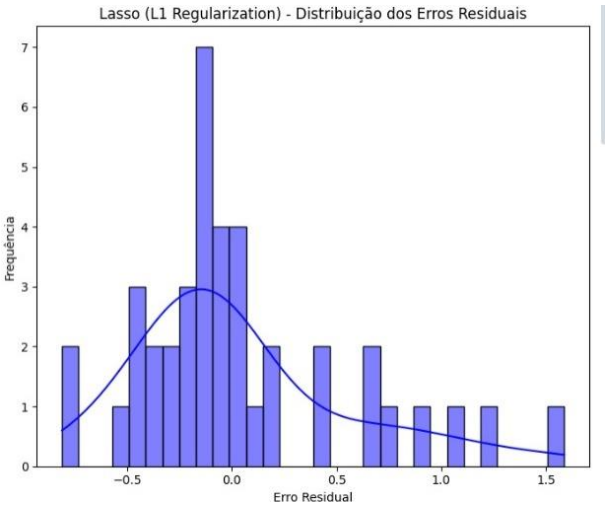
Coeficientes do modelo:

followers: -1.3304

avg_likes: 0.2487

new_post_avg_like: 3.1049





Ridge (L2 Regularization):

R² Score: 0.9487

MSE: 0.3248

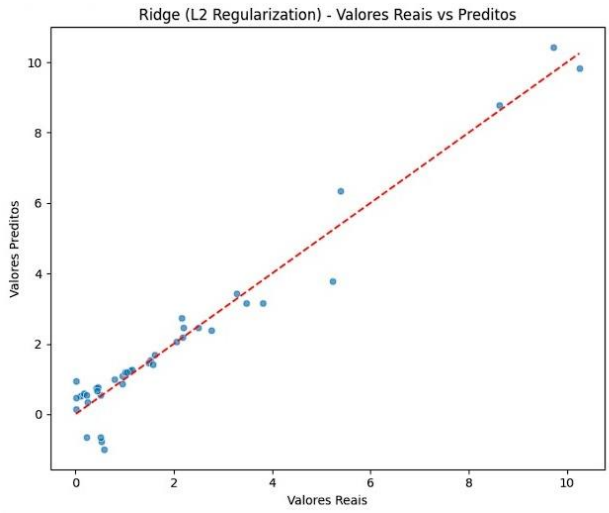
MAE: 0.3974

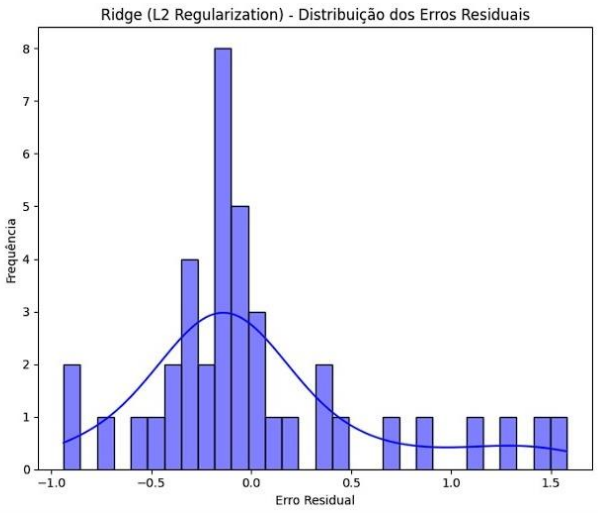
Coeficientes do modelo:

followers: -1.4867

avg_likes: 0.3561

new_post_avg_like: 3.1616





Linear Regression com Seleção de Recursos:

R² Score: 0.9010

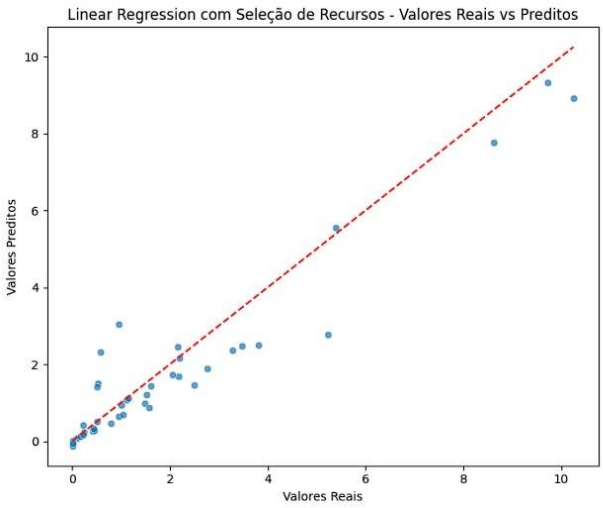
MSE: 0.6274

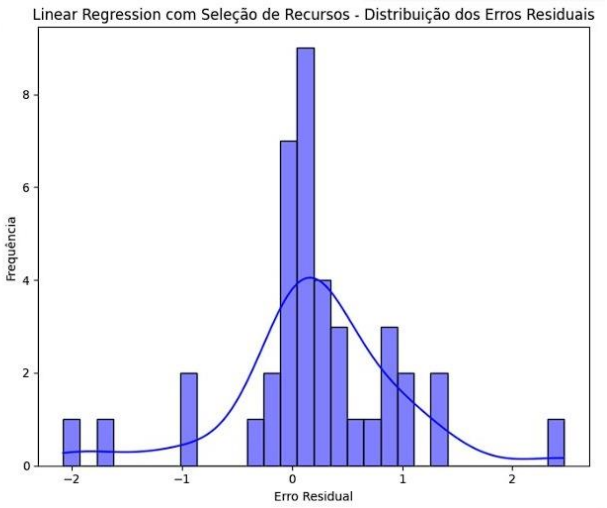
MAE: 0.5267

Coeficientes do modelo:

followers: -0.1123

avg_likes: 3.0819





Valores adquiridos no teste de Validação Cruzada:

SGDRegressor: 0.8161 ± 0.0906

Lasso: 0.8412 ± 0.0461

Ridge: 0.8316 ± 0.0687

Linear Regression com Seleção de Recursos: 0.8315 ± 0.069

Pelos valores adquiridos e gráficos dos 4 modelos que foram testados, aquele que apresentou os melhores resultados foi o Lasso, pelo fato dele possuir a maior média de R^2 com menores valores MSE e MAE, sendo o modelo mais estável. Além disso, o Lasso também se garantiu como o modelo mais consistente, tendo a menor variação nos resultados da validação cruzada.

Discussão:

Senti os resultados muito prazerosos, tendo em vista que ele conseguiu atender ao objetivo que tínhamos em mente quando selecionamos esse projeto para ser feito, porém ainda encontramos muitos desafios para poder fazê-lo funcionar corretamente e corrigimos diversos erros até chegar nesse ponto, como problema no programa conseguir ler os dados que estavam no arquivo csv, alguns problemas na exibição dos gráficos dos valores, entre outros. Além disso, o fato de termos escolhido os 4 modelos diferentes para serem testados tornou a tarefa mais complicada por termos

que desenvolver todos eles separadamente para depois os unirmos para apresentar um projeto bem-feito.

Conclusão:

Esse projeto até o momento está sendo uma experiência ótima para nós residentes da trilha de ciência de dados, por sintetizar tudo que estudamos durante a trilha e podemos utilizar tudo que aprendemos em um trabalho prático que nos aproxima do que realmente é o papel de um cientista de dados, ainda que houvesse muita margem para melhorias no nosso projeto, até o momento, nos sentimos satisfeitos com o ponto que conseguimos alcançar com ele.

Referências:

ROCKETSEAT. Aprenda a criar um modelo de regressão linear com Python. Blog Rocketseat, [s.d.]. Disponível em: https://blog.rocketseat.com.br/. Acesso em: 17 nov. 2024.

BRAINS.DEV. Como construir modelos de regressão linear em Python. Brains.dev, [s.d.]. Disponível em: https://brains.dev/. Acesso em: 17 nov. 2024.

ICHI.PRO. Guia passo a passo para regressão linear simples e múltipla em Python. Ichi.pro, [s.d.]. Disponível em: https://ichi.pro/. Acesso em: 17 nov. 2024.