Modelo de Machine Learning para previsão de DAU (Daily Active Users) em aplicativos

Adenis Segal Borel
10/07/2025

Coleta e tratamento dos dados

- A coleta de dados inicial foi realizada com SQL unindo todas as tabelas do banco de dados pela data e appId.

- Ao unir todas as tabelas, houveram 291 linhas duplicadas e 2990 (~7%) linhas com pelo menos 1 valor nulo.

- O ideal antes de tomar qualquer decisão sobre linhas duplicadas e valores ausentes é analisar o motivo, para que não seja necessário perder dados. Podemos até usar modelos de ML para prever os valores ausentes. Mas nesse caso, como não há a possibilidade de melhor compreensão e o percentual de linhas com valores ausentes era considerado baixo (7%), a decisão tomada foi a de excluir as linhas duplicadas e as linhas com dados ausentes.

Coleta e tratamento dos dados

- As variáveis “daily_ratings” e “daily_reviews” possuem valores negativos. Pela lógica, não é possível existirem valores negativos para este contexto, então optei por transformá-los em positivo (assumindo que poderia ter sido um erro de gravação no banco, ou algo durante a fase de ETL) ao invés de excluí-los e perder mais dados.

Modelagem e o treinamento

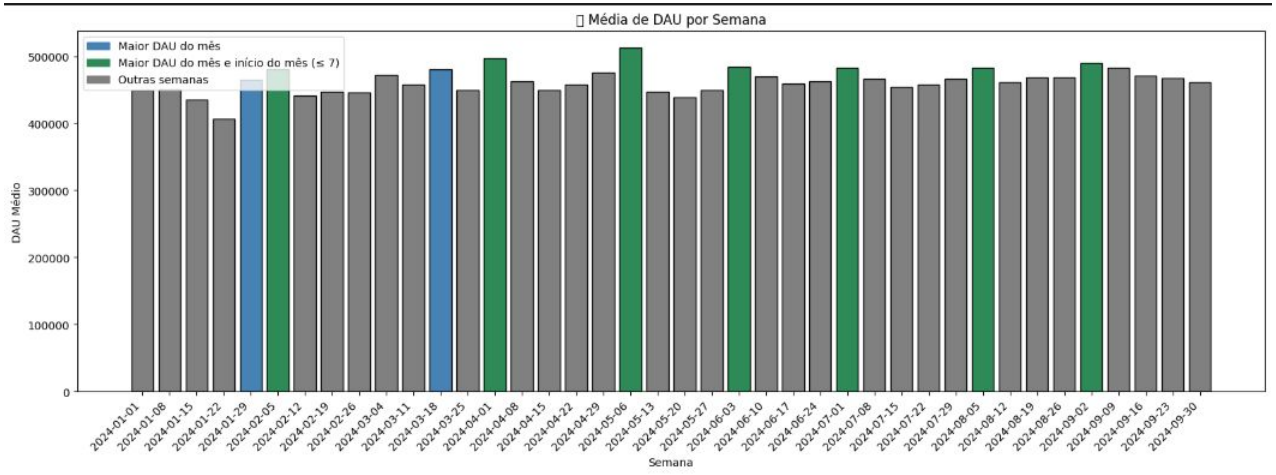
- Não foi notado nenhum valor extremamente fora da escala e diferente dos demais, mas ainda sim, a maioria das variáveis possuem valores extremos em sua distribuição, os quais poderiam ser considerados outliers. Entretanto, este é um comportamento natural dessa base, então optei por não removê-los.

- A variável alvo “daily_active_users” possui uma distribuição assimétrica, com poucos valores extremamente altos e muitos valores baixos. Os apps possuem comportamentos diferentes entre si, além do fato de que a maioria das variáveis possui uma alta variância, o que também dificulta para o aprendizado do modelo.

- Um modelo aprende melhor quando encontra padrões nos dados e isso acontece mais facilmente em relações mais lineares (ou estáveis). Uma das técnicas para ajudar o modelo neste caso é o uso do **np.log1p()**. Ele comprime os dados extremos e suaviza a variância, facilitando para o modelo aprender.

Modelagem e o treinamento

- Existe uma certa sazonalidade na variável alvo que foi percebida nas análises. Aparentemente, existe um padrão de aumento de acessos no início do mês, mais especificamente nos primeiros 7 dias. Por conta disso, levei em consideração a criação dessa variável para utilizar no modelo.



Modelagem e o treinamento

- Uma abordagem que utilizei na modelagem foi criar variáveis de janela de tempo dos últimos x dias para algumas variáveis, como total_ratings dos últimos 3, 7, 15 e 30 dias, por exemplo.

- Poucas variáveis tiveram alto grau de correlação com a variável alvo. Percebi também que algumas variáveis que criei de janela de tempo poderiam gerar data leakage (vazamento de dados da variável alvo) e por fim acabei removendo variáveis de janela temporal quanto outras variáveis do próprio conjunto de dados inicial.

- Mais 3 variáveis foram criadas para trazer mais conhecimento para a base e auxiliar o modelo a encontrar padrões: “fim_de_semana” (True ou False), “delta_ratings” e “delta_reviews”. Esses valores delta, são o percentual que a quantidade do dia representa em relação ao total.

Modelagem e o treinamento

- Algumas variáveis também apresentaram alta colinearidade entre si, o que acaba deixando o modelo redundante e complexo sem necessidade. Essas foram removidas.

- De todas as variáveis utilizadas, menos da metade tiveram algum grau relevante de importância em relação ao target. Como a variável “inicio_do_mes” apresentou pouca relevância, ela também foi removida para o treinamento do modelo final.

- Como os dados não seguem uma relação linear e possuem alta variância, modelos como Regressão Linear e SVMs acabam tendo um desempenho pior, pois eles utilizam a premissa de que os dados seguem essa relação. Por isso, o foco foi nos modelos de árvore (Random Forest e Decision Tree) e também o XGBoost que lida melhor com outliers do que modelos lineares tradicionais.

Modelagem e o treinamento

- O modelo foi treinado com dados de 01/01/2024 até 31/08/2024, para que os dados a serem previstos fossem os do mês seguinte.

- Após algumas tentativas de ajuste e otimização com Gridsearch, o modelo final treinado ficou com as seguintes métricas:

  MAE médio: 210,722.26
  RMSE médio: 573,712.29
  R^2 médio: 0.6071

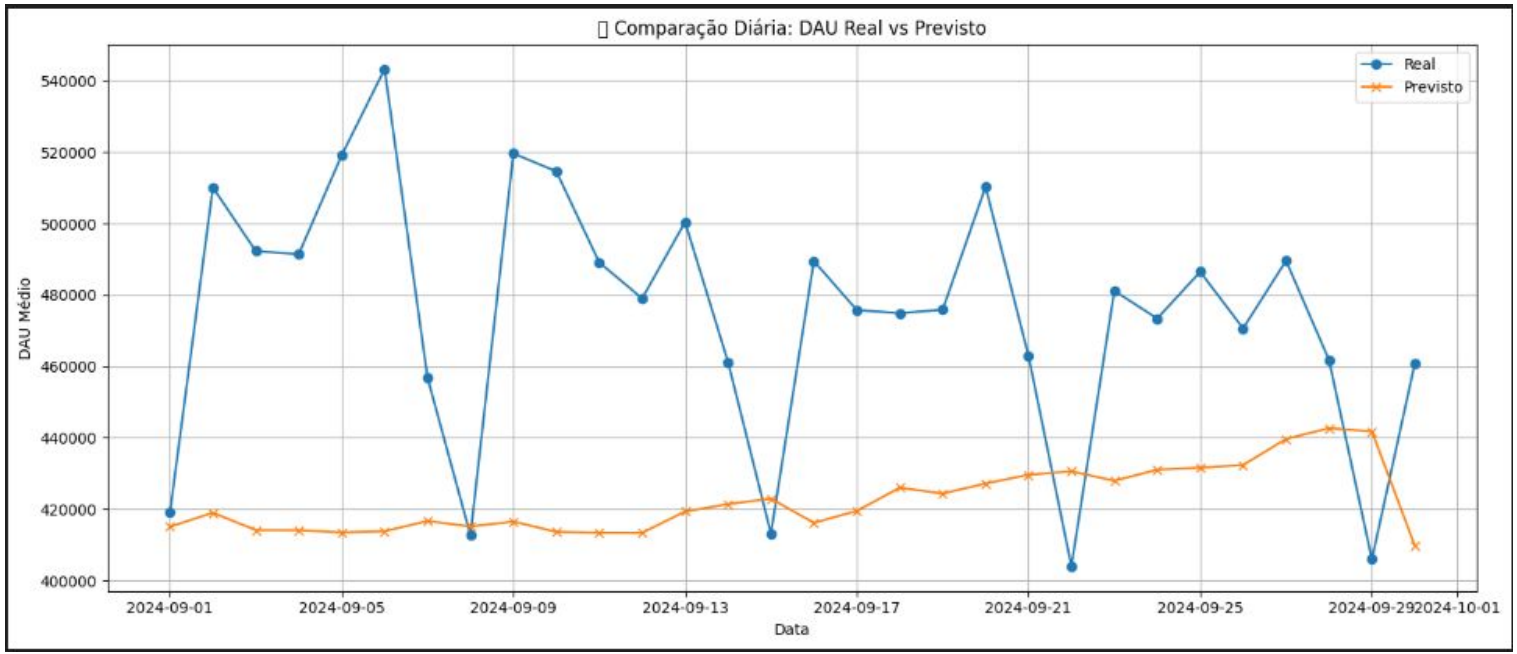
- Essas não são métricas muito positivas, os valores de erro ficaram relativamente altos e o modelo conseguiu generalizar bem (compreender os padrões) em apenas 60¨% dos casos.

Validação do modelo e métricas de performance

- Como dito anteriormente, para a validação do modelo foram utilizados dados do último mês presente na base, 01/09/2024 em diante, para garantir que os dados a serem previstos não foram usados no treinamento.

- O gráfico a seguir que compara a média do valor real com a média do valor predito, mostra claramente uma grande diferença entre estes valores. principalmente nos picos e quedas abruptas de acessos. Isso acontece quando existe uma alta variabilidade nos dados e picos sazonais, mas o modelo ainda não conseguiu captar bem essas nuances.

Validação do modelo e métricas de performance



Validação do modelo e métricas de performance

- No entanto, as métricas MAE, RMSE e R^2 indicam uma performance melhor no geral, do que a apresentada no gráfico.
  MAE ~ 107 mil em um volume de DAU médio de ~400 a 500 mil → erro absoluto de ~20–25%.
  RMSE ~ 324 mil mostra que há erros maiores em alguns dias (picos).
  R^2 = 0.9462 → o modelo está explicando mais de 94% da variância, o que é muito bom.

Interpretações dos resultados

- O modelo generaliza bem a tendência geral, visto que a linha laranja segue um movimento suave e não previsões totalmente discrepantes entre si, porém, ele não conseguiu compreender e acompanhar as oscilações abruptas de um dia para o outro.

- Boas métricas e um gráfico que apresenta maus resultados geralmente acontecem quando o modelo acerta bem a média geral (boa performance global), mas erra em picos extremos (impactando menos o R^2, porém, mais visível no gráfico)

Sugestões de melhoria e próximos passos

- **Adicionar outras variáveis temporais:** a variável inicio de mês até foi criada, mas não foi utilizada no modelo final pelo fato de não ter apresentado relevância pelo Feature Importances.
- **Adicionar variáveis de médias móveis**.
- Entender melhor como a sazonalidade se comporta nestes dados e aplicar ajustes para lidar melhor com estes eventos.
- Realizar uma clusterização para agrupar aplicativos com comportamento semelhante e para cada cluster, treinar um modelo diferente, garantindo assim, comportamentos mais semelhantes dentro da mesma base.

Considerações

- Como este é um teste que está sendo desenvolvido em apenas 3 dias, que é um período curto de tempo e somente no tempo livre do desenvolvedor, acredito muito que seja possível melhorá-lo com mais tempo de desenvolvimento para atacar os pontos que foram citados e outros que também podem ser testados e validados posteriormente.

- Apesar dos desafios, fazer este teste técnico foi divertido e empolgante e gostaria de vivenciá-los na vida real com a RankMyApp :)

Obrigado!

Adenis Segal Borel

10/07/2025