Metodologia e resultados referentes ao modelo criado para RankMyApp

Nome: Diego Renan
Cargo: Cientista de Dados

Considerações Iniciais

O presente relatório traz um resumo acerca do desenvolvimento realizado nesse projeto. Vale ressaltar que nesse relatório estão descritos os procedimentos realizados bem como insights obtidos. É importante frisar que se trata de um relatório resumido, contendo as informações mais importantes, e que detalhes mais técnicos sempre deverão ser consideradas a presença do idealizador do projeto. Alguns detalhes também se encontram no próprio repositório do projeto. Foi criada uma branch denominada **develop_branch** para que o trabalho fosse desenvolvido, de modo a posteriormente ser inserida na branch **master**.

Consumo dos dados

O projeto foi estruturado no Github em um formato adequado para que outro desenvolvedor possa também reproduzir os resultados. Ou seja, foi criado uma virtual env, visando isolar todas as dependências do projeto. Essas dependências estão explícitas em um arquivo **requirements.txt.** Além disso, visando manter a integridade das informações informadas, foi criado um arquivo **.env** que contém as credenciais e diretórios do projeto. Assim todas elas ficam em segurança e não serão inseridas no repositório remoto pois estão mapeadas juntamente com a virtual env dentro do arquivo gitignore.

Inicialmente foi feito o consumo dos dados, no caso, consumimos as 4 tabelas principais que estão mapeadas no banco. Utilizou-se duas chaves primárias para juntar todas as tabelas: appId e a data. Isso foi feito para facilitar a seleção de variáveis. Vale ressaltar que para a junção das tabelas algumas manipulações de dados se fizeram necessárias, principalmente no que se trata de mudar o datatype de algumas variáveis.

Após a junção das tabelas, avaliou-se que algumas colunas apresentavam dados ausentes, bem como também pouca infomação. Ao final, escolheu-se as colunas: mauReal (essa como sendo a variável alvo), ratings, daily_ratings, reviews, daily_reviews, mês, dia, appId_encoded, category_encoded. Sendo que as duas últimas são oriundas de um processo de encodding, tendo em vista que representam variáveis categóricas, ou seja, representam categorias e que para os modelos, em geral, é melhor tratarmos as variáveis dessa forma. Algumas colunas foram removidas, umas apresentam poucos dados preenchidos, o que foi interessante removê-las; outras não possuiam nenhum dado devido a junção das tabela. No notebook denominado **extracao_dados_treino.ipynb** estão todos os detalhes dos motivos e também o passo a passo de como foram feitas as eliminações dessas colunas.

O conjunto de dados foi salvo dentro da pasta **data** e com nome de **data.csv**. Ele será utilizado para treinar um algoritmo de regressão que terá como objetivo prever a quantidade de mauReal baseado nas outras variáveis selecionadas. Outro artefato que foi salvo foi os encodings usados na criação dos dados, essa etapa é importante pois para previsões futuras é importante que os dados estejam trabalhados com as mesmas classes, evitando uma predição errônea. O arquivo é denominado **encodings.pkl**.

Metodologia e resultados referentes ao modelo criado para RankMyApp

Nome: Diego Renan
Cargo: Cientista de Dados

Criação do modelo

Os dados foram consumidos e utilizou-se de três algoritmos de regressão para o teste do modelo: regressão linear, árvores de regressão, florestas de regressão e SVR. Poderíamos ter escolhido outros modelos mais complexos como redes neurais, mas devido a natureza do problema e o tempo de execução de cada um, foi escolhido somente esses três – mais detalhes se encontram no notebook.

Dividiu-se os dados em treino e teste, com cerca de 25% para teste e o restante para treinamento. Utilizou-se de uma ténica de seleção aleatória com uma semente visando reprodutibilidade da separação dos dados. Treinou-se os algoritmos e calculou-se as métricas tanto no conjunto de treino no conjunto de testes visando entender um possível sobreajuste nos modelos.

	Model	Train MAPE	Train MedAPE	Train RMSE	Test MAPE	Test MedAPE	Test RMSE
0	Linear Regression	3.789551	81.640041	2.092886e+06	4.008198	81.610170	2.010108e+06
1	Random Forest	0.195930	0.104133	1.845751e+05	0.347273	0.299511	3.597777e+05
2	Decision Tree	0.000000	0.000000	0.000000e+00	0.409003	0.283083	6.051557e+05
3	SVR	3.438411	88.509615	5.195937e+06	3.695814	88.909668	5.151302e+06

É possível verificar que os modelos de árvore e ensemble apresentam desempenho muito melhor do que regressão linear e SVR. Porém, o modelo de árvore apresenta um possível sobreajuste, devido a métrica nos dados de treino haver pouco erro e um erro maior nos dados de teste – isso era esperado pois nessa etapa não fizemos nenhum ajuste dos hiperparâmetros. Portanto, o modelo escolhido foi o de floresta aleatória – mais detalhes encontram-se no notebook **modelo.ipynb.**

Após escolher o modelo de floresta aleatória, priorizou-se a melhoria do mesmo usando o GridSearchCV para fazer uma varredura nos hiperparâmetros. Utilizou-se dois dos principais hiperparâmetros: max_depth (profundidade das árvores) e n_estimators (número de árvores dentro das florestas). Vale ressaltar que poderíamos ter escolhido outros hiperparâmetros, mas devido a questões computacionais e de tempo, escolhemos apenas os principais visando melhorar o modelo de forma mais otimizada. A título de curiosidade, somente com esses hiperparâmetros foram realizados 60 ajustes de modelos diferentes. Se extendêssemso para outros hiperparâmetros, chegaríamos a 540, 1600 modelos distintos. O melhor modelo foi escolhido com um max_depth de 20 e um n_estimators de 50.

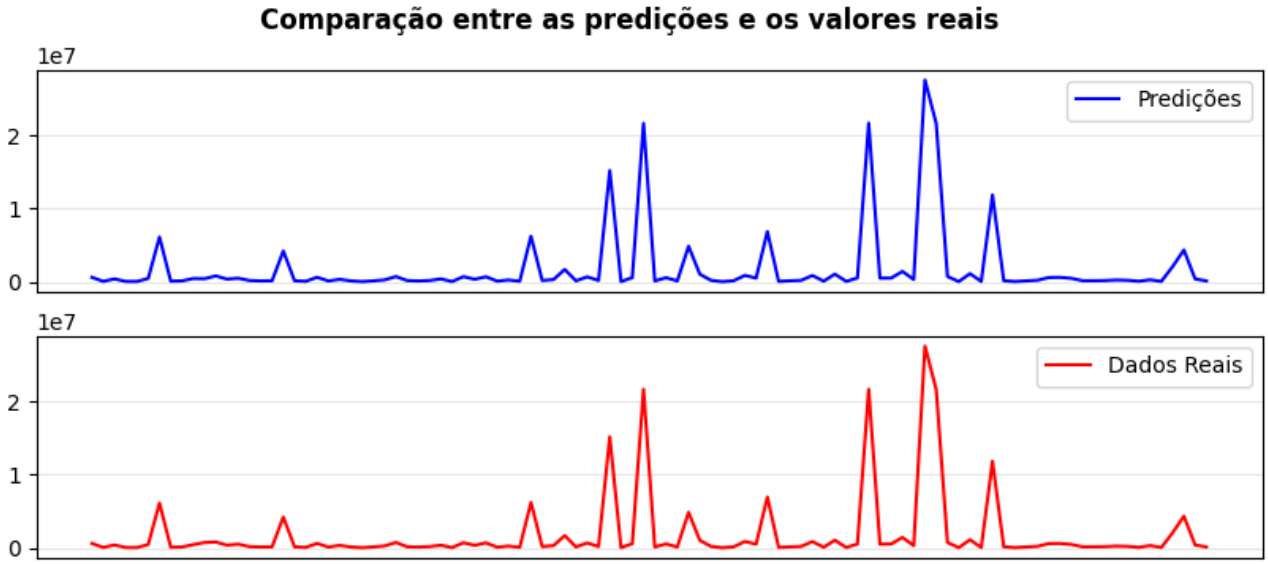
Após essa etapa o modelo foi exportado, bem como também os dados usados para o seu treinamento e os dados usados para o teste. Esse procedimento visa manter o procedimento de retreino do modelo de forma reprodutível e também em podermos comparar o seu desempenho ao longo do tempo e utilizar os mesmos dados para possíveis análises.

Observações: Várias abordagens poderíam ter sido diferentes aqui, poderíamos também ter utilizado um maior tratamento nas variáveis previamente antes de treinar o modelo, fazer um standard scaler, ou demais técnicas de feature_enginner. Porém, devido ao tempo, não foi possível realizar a mesma. É claro, em um projeto real, essa etapa é de suma importância. Porém, algoritmos de floresta não tem a necessidade desse tipo de tratamento, e ela seria interessante em outros tipos de algoritmos como por exemplo KNN, que é sensível a escala dos dados.

Metodologia e resultados referentes ao modelo criado para RankMyApp
Nome: Diego Renan
Cargo: Cientista de Dados

Avaliação de desempenho

Após os dados e modelos serem salvos, foi utilizado os dados de teste (25%) para realizar a validação do modelo e geração de alguns gráficos. Vale ressaltar que esses dados não foram usados para o treinamento do modelo, o que mantêm os resultados fidedignos do ponto de vista que simulam realmente situações reais de utilização do modelo.

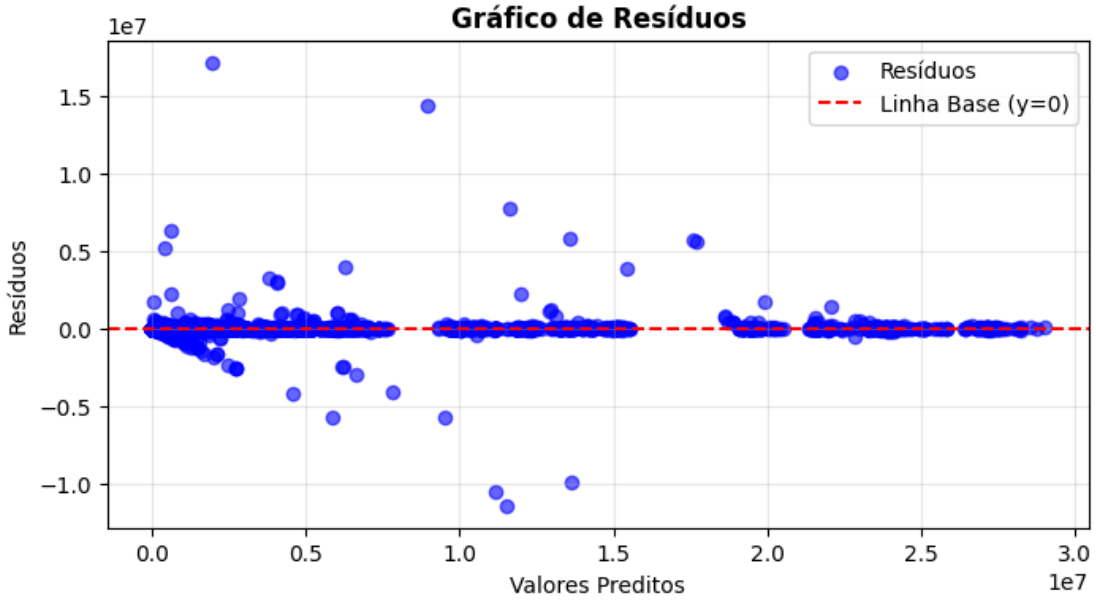


O gráfico acima mostra que o desempenho do modelo está se saindo muito bem. A curva em azul mostra as predições realizadas usando uma amostra de aproximadamente 100 dados apresentados ao modelo. A cuva em vermelho representa os mesmos 100 dados, porém, agora são os dados reais. Ou seja, é possível ver que o modelo apresenta um bom desempenho em dados desconhecidos, pois seu reultado se aproxima bastante dos resultados originais. É interessante também avaliar as métricas desse modelo.

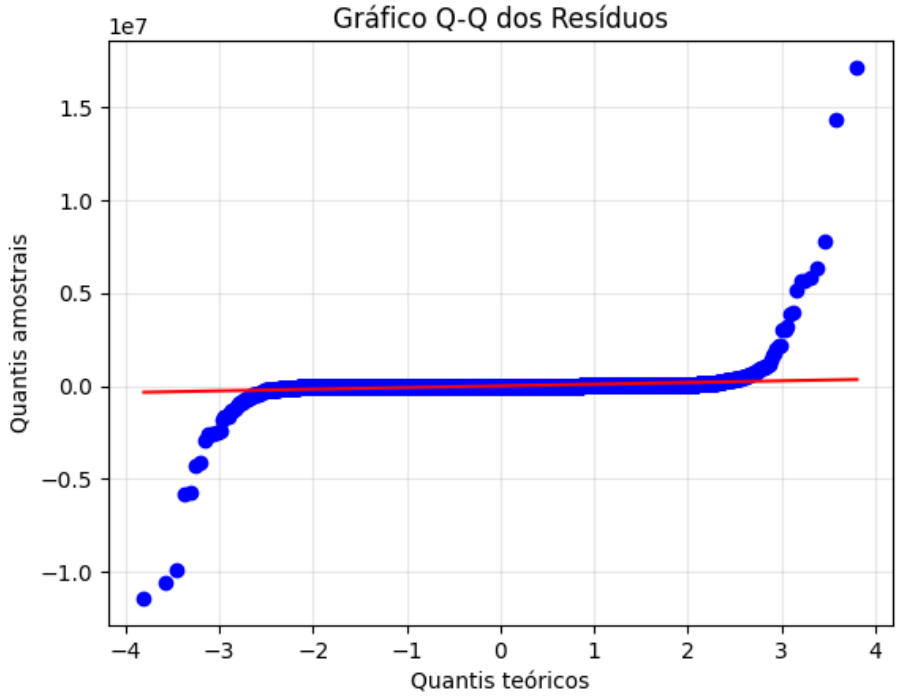
	Métrica	Valor
0	MAPE	34.814402
1	MedAPE	0.460846
2	RMSE	372971.112339

Essa tabela nos mostra que o modelo apresentou um MAPE de 34%. Ou seja, a média de erro do modelo é cerca de 34%. Esse valor pode ser considerado grande dependendo do modelo de negócio que estamos trabalhando. Porém, ao analisar a MedAPE, vemos que o erro cai drásticamente para 0,46%. Ou seja, o erro mediano é muito menor do que o erro médio, o que sugere que existem outliers interferindo no resultado do modelo. É importante avaliar tal ponto, e para isso, geramos um gráfico de resíduos do modelo.

Metodologia e resultados referentes ao modelo criado para RankMyApp
Nome: Diego Renan
Cargo: Cientista de Dados

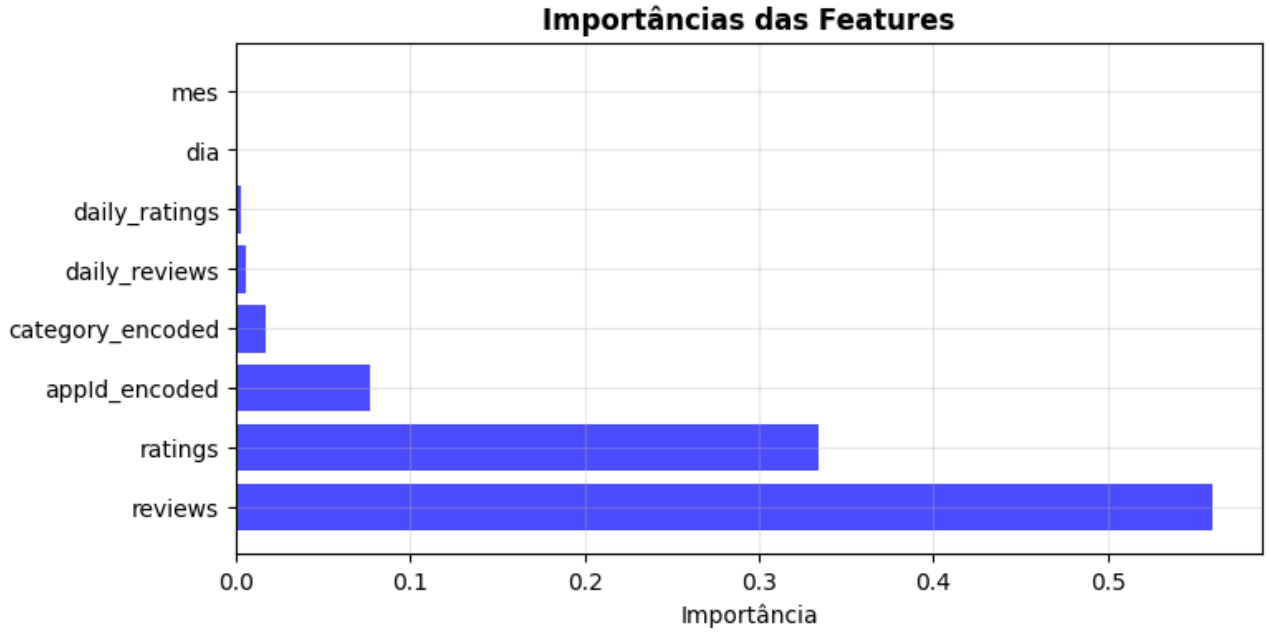


Observa-se nesse gráfico que o modelo em geral tem um resultado bem satisfatório, porém, existem alguns pontos que estão desviando bastante tanto positivamente quanto negativamente em relação aos valores preditos. Provavelmente esses pontos são responsáveis por gerar aquele aumento significativo na métrica MAPE, que é sensível a outliers. Para confirmar ainda mais esse resultado e entender de onde estão vindo essas divergências, vamos gerar um QQ-plot.



Como era de se esperar, o modelo apresenta um bom resultado no gráfico QQ, ou seja, as predições estão sobre a linha vermelha, porém, existem situações em que os pontos estão divergindo dessa

linha. Isso nos sugere que os resíduos seguem uma distribuição normal, porém, existem pontos distantes da média que estão divergindo e não seguem tal regularidade. Ou seja, fica confirmado que existem outliers em nossa base de dados e que esses outliers geralmente apresentam valores grandes ou pequenos. Por último e não menos importante, temos também o gráfico de importância das variáveis.



Observa-se que as variáveis reviews e ratings são responsáveis por mais de 80% da explicabilidade do modelo. Isso faz muito sentido se pensarmos que o problema de negócio em questão é sobre otimização de lojas de aplicativos. Esse tipo de informação é crucial para que novos aplicativos sejam instalados em smartphones. Esse gráfico sugere ainda que não há uma tendência mensal ou diária em explicar a nova instalação de aplicativos. Talvez o que poderia influenciar nessa variável seria o dia e mês do ano de lançamento do aplicativo, o que não está inserido na variável dia e mês utilizada aqui.

Metodologia e resultados referentes ao modelo criado para RankMyApp

Nome: Diego Renan
Cargo: Cientista de Dados

Conclusão e Próximos passos

O modelo criado seguiu boas práticas de programação, como:

1. criação de envs

2. requirements visando isolar o projeto e reprodutibilidade

3. criação de arquivo .env visando armazenar com segurança as credenciais

4. salvar os artefatos utilizados no treino do modelo (encodings, dados de treino e teste, etc)

Além disso, o desempenho do modelo foi bem satisfatório dado o tempo e a quantidade de informações passadas. O modelo pode e deve ser revisto visando melhoria. Algumas delas são:

1. utilizar talvez outras variáveis para incrementar seu desempenho

2. fazer um melhor tratamento das variáveis previamente a construção do modelo, talvez uma análise de correlação entre elas, visando removê-las antes mesmo do treinamento

3. fazer um tratamento nas variáveis no que diz respeito a escalonamento

4. talvez testar outros algoritmos visando performance

5. remover as variáveis que não foram úteis na explicabilidade do modelo visando remover intercorrelação entre elas e possíveis variáveis irrelevantes

6. fazer um tratamento nas variáveis no que diz respeito a remoção de outliers. Essa etapa seria crucial a presença do time de negócio auxiliando no entendimendo das variáveis e o que pode e não pode ser considerado um outlier.

A lista acima pode ser infinita, a depender das necessidades do time de negócio e também da empresa. O cientista de dados precisa estar em contato com o time de negócio e produto para que o modelo possa ser feito com uma maior assertividade, pois ciência de dados é dependente do time de negócio e também de produto, visando a criação de bons produtos de dados que realmente tragam valor ao negócio.