* NLP do campo descrição
* Valor real ou media da tipologia na variavel ALVO
* Melhor os graficos das analises (Seaborn, Plotly, PyPlot, Configs iguais dos exemplos da imersao)
* Crostab
* Valor medio dos Imoveis por Banheiros, Suites, Bairo, Quartos, Area, Piscina, etc …
* Quais outras metricas ??
* Unique() nos campos
* value\_counts() nos campos
* value\_counts().plot() nos campos
* rename das colunas por dicionário
* query com parâmetros(@)
* pegando dados do describe para plotar histogram(dados.loc[:,'g-0':'g-771'].describe().T['mean'].hist(bins=30)
* Usar pd.crosstab nos relacionamentos
* scatterplot das variaveis
* lmplot regressao linear
* metodo corr() para correlacionar colunas + explicacao do resultado
* Heatmap fracionado ao meio
* pd.merge entre dados
* query.value\_counts()
* Regressao Logistica
* Plotar a arvore de decisao em imagem
* Overfiting
* Plot dos resultados de teste e de previsao
* Plots de comparacao com boxplot um ao lado do outro
* Aplicar tecnica de remocao de outliers do curso (acima de 2x da max)
* Tunning dos modelos
* Exportacao pro github
* Implantacao do App no Streamlit na nuvem
* Synthetic features - campos calculados para machine learning rate
* Novas Features no modelo (Feature Engineering)
* As features \_Numero devem ser normalizadas para nao confundir o modelo
* Usar o OneHotencoder em vez do Label Encoder pra converter em numerico sem relacao de valor entre si (gera muitas colunas)
* Usar o getdummies em vez do onehotencoder(pra resolver o problema das colunas)
* Crescimento do valor medio do imovel de acordo com os bairros, suites, banheiros , psicinas
* Usar o GridSearch para fazer o tunning dos modelos de machine learning
* Usar os parametros de tunning da Arvore de decisao
* Aplicar a normalizacao dos atributos antes de treinar o modelo
* seleção de features mais importantes e teste do modelo(feature\_importances)
* sweetviz
* joinplot
* responder perguntas igual exemplo de aluguel
* usar map pra trocar categoricos por numeros
* regplot pra mostrar a regressao
* distplot pra ver a distribuição
* violinplot pra ver as concentracoes
* aplicar o DummyRegressor
* testar o KNN com Scaler antes
* usar o pycaret pra testar todos os modelos
* incluir o MLFlow pra fazer o track e deploy