Esse é o relatório para estimar o preço de um imóvel a partir de características tais como o tipo de imóvel (apartamento, casa, loft ou quitinete), bairro onde está localizado, número de quartos, número de vagas, área útil, área extra e presença de elementos diferenciais em relação a outros imóveis, tais como churrasqueira, estacionamento para visitantes, piscina, playground, quadra esportiva, campo de futebol, salão de festas, salão de jogos, sala de ginástica, sauna e vista para o mar, usando técnicas de regressão multivariável.

As importações são para:

Pré-processamento: Manipulação de dados, normalização e transformação polinomial.
Modelagem: Diversos algoritmos de regressão, como regressão linear, KNN e Random Forest.
Validação: Técnicas para avaliar o desempenho dos modelos de forma robusta.
Visualização e Análise: Gráficos e análise estatística para explorar e entender os dados.

Passo 1: Carregamento dos dados

Carrega os conjuntos de treino e teste em formato CSV.
Define o delimitador como vírgula (',') e usa o ponto ('.') como separador decimal.

Motivação:
Garantir que os dados sejam carregados corretamente para manipulação e análise.

Propósito:
Carregar os dados em DataFrames dados (treino) e dados_teste (teste), que serão utilizados para treinamento e validação do modelo.

Passo 2: Remoção da Coluna Id

Remove a coluna Id, que provavelmente contém identificadores únicos.

Motivação:
A coluna Id não possui significado preditivo para o modelo e pode introduzir ruído.

Propósito:
Limpar os dados removendo informações irrelevantes para a modelagem.

Passo 3: Subcategorização de Diferenciais

Define um dicionário (diferenciais) que agrupa diferentes combinações de características ou amenidades em categorias mais gerais.
Usa o método replace do pandas para substituir os valores nas colunas correspondentes do conjunto de treino (dados) e teste (dados_teste) de acordo com o mapeamento no dicionário.

Motivação:
Agrupar categorias reduz a cardinalidade, simplificando o processamento e análise.
Exemplo:
Combinações como campo de futebol e copa ou campo de futebol e estacionamento visitantes são mapeadas para a categoria genérica futebol+.

Propósito:
Facilitar a análise e o aprendizado do modelo ao reduzir o número de categorias e, consequentemente, as colunas geradas no processamento posterior (como One-Hot Encoding).

Carregamento: Carrega os conjuntos de dados de treino e teste para análise.
Limpeza: Remove colunas irrelevantes (Id).
Agrupamento: Simplifica categorias ao mapear combinações complexas para classes mais gerais.

Passo 4: Agrupamento de Bairros em Categorias Simbólicas

Substitui os valores na coluna de bairro nos conjuntos de treino e teste por categorias simbólicas (pobre, meiotermo, rico).
O dicionário bairros mapeia cada bairro para uma dessas categorias.

Motivação:
Os nomes dos bairros são de alta cardinalidade, tornando o processamento mais complexo.
Reduzir os bairros em categorias gerais ajuda a simplificar os dados e a análise.

Propósito:
Facilitar a análise e a modelagem ao reduzir o número de categorias únicas para algo mais gerenciável.

Passo 5: Codificação de Variáveis Categóricas com pd.get_dummies

Converte as variáveis categóricas mencionadas (tipo, bairro, tipo_vendedor, diferenciais) em variáveis dummy.
Cada categoria em uma coluna original se torna uma nova coluna, com valores binários (0 ou 1).

Motivação:
Os modelos de machine learning baseados em números não conseguem lidar diretamente com variáveis categóricas.
pd.get_dummies transforma essas variáveis em um formato numérico utilizável.

Propósito:
Garantir que as variáveis categóricas possam ser utilizadas pelos modelos durante o treinamento e a predição.

Passo 6: Exibição da Estrutura dos Dados

Retorna as dimensões do DataFrame dados como uma tupla (n_linhas, n_colunas).

Motivação:
Verificar o número de atributos (colunas) resultantes após a transformação e se o conjunto de dados é gerenciável.

Propósito:
Confirmar que o processo de codificação e substituição foi executado corretamente e avaliar a quantidade de variáveis geradas.

Essas linhas destacam um processo importante de preparação dos dados:

Agrupamento: Reduz categorias de alta cardinalidade (bairros) para classes mais simples e significativas.
Codificação: Transforma variáveis categóricas em variáveis dummy, adequando-as aos modelos.
Verificação: Usa o método shape para garantir que os dados estejam no formato esperado após as transformações.

Passo 7: Identificação de Colunas Comuns e Exclusivas

Obtém as colunas comuns (e) entre os DataFrames dados e dados_teste usando operações de interseção de conjuntos.

Motivação:
Garante que ambas as tabelas compartilhem um conjunto consistente de colunas, necessário para compatibilidade nos modelos.

Passo 8: Identificação de Colunas Exclusivas

Calcula as colunas exclusivas de cada DataFrame (f para dados_teste, g para dados).

Motivação:
Identificar inconsistências entre os conjuntos de treino e teste após a aplicação de transformações (como pd.get_dummies).

Passo 9: Remoção de Colunas Não Compartilhadas

Remove as colunas exclusivas de cada DataFrame para alinhar a estrutura de ambos.

Motivação:
Garante que os conjuntos de treino e teste tenham as mesmas colunas, evitando erros nos modelos.

Passo 10: Tratamento de Outliers

Filtra os dados para manter apenas registros cujo preço esteja no intervalo de 50.000 a 5.000.000.

Motivação:
Excluir valores extremos que poderiam distorcer o treinamento e a avaliação do modelo.

Propósito:
Melhorar a robustez e a generalização do modelo.

Passo 11: Análise de Variáveis Categóricas

Identifica colunas categóricas e exibe o número de categorias únicas em cada uma, além dos valores possíveis.

Motivação:
Entender a estrutura das variáveis categóricas para decisões de codificação ou exclusão.

Passo 12: Cálculo do Coeficiente de Correlação de Pearson

Calcula o coeficiente de Pearson entre cada coluna e o preço (preco), medindo a correlação linear.

Motivação:
Identificar variáveis com baixo impacto na predição do preço, para possíveis exclusões.

Passo 13: Remoção de Variáveis com Baixa Correlação

Remove colunas com baixo coeficiente de correlação com a variável-alvo (preco).

Motivação:
Reduzir a dimensionalidade dos dados, eliminando variáveis irrelevantes.

Propósito:
Simplificar o modelo e melhorar sua capacidade de generalização.

Passo 14: Exclusão Adicional de Colunas

Remove colunas adicionais com base em critérios específicos (como baixa relevância ou redundância).

Motivação:
Continuar a limpeza dos dados para melhorar o desempenho do modelo.

Essas linhas abrangem um processo avançado de preparação dos dados:

Harmonização: Garante que treino e teste compartilhem a mesma estrutura.
Outliers: Remove valores extremos que podem distorcer o aprendizado.
Correlação: Usa métricas estatísticas para identificar e excluir variáveis de baixa relevância.
Simplificação: Exclui colunas redundantes ou com baixa utilidade preditiva.

Passo 15: Separação dos Conjuntos de Treino e Teste

x: Matriz com os atributos preditivos (exclui a coluna preco).
y: Vetor com a variável alvo (preco).
Divide os dados em conjuntos de treino e teste com a função train_test_split.
test_size=1500: Define que o conjunto de teste terá 1500 amostras.
random_state=0: Garante que a divisão seja reproduzível.

Motivação:
Separar os dados para avaliar o modelo em dados que ele não viu durante o treinamento.

Propósito:
Garantir que o modelo seja avaliado de forma justa em um conjunto de dados independente.

Passo 16: Normalização dos Dados

Ajusta os dados para terem média 0 e desvio padrão 1 com StandardScaler.
Aplica a normalização ao conjunto de treino e ao conjunto de teste.

Motivação:
Melhorar o desempenho de algoritmos sensíveis à escala dos dados, como Regressão Linear e Polinomial.

Propósito:
Garantir que os atributos preditivos estejam na mesma escala para evitar vieses.

Passo 17: Regressão Linear

Cria um modelo de Regressão Linear com LinearRegression.
Treina o modelo no conjunto de treino (fit).
Faz previsões nos conjuntos de treino e teste.

Motivação:
Avaliar o desempenho da Regressão Linear como modelo base.

Propósito:
Obter previsões e avaliar a performance do modelo linear.

Passo 18: Avaliação do Modelo Linear

Calcula métricas de avaliação:
MSE (Erro Quadrático Médio).
RMSE (Raiz do Erro Quadrático Médio).
R^2 (Coeficiente de Determinação).
RMSPE (Raiz do Erro Percentual Médio Quadrático).
Separa os cálculos para os conjuntos de treino (in) e teste (out).

Motivação:
Avaliar a qualidade do modelo em termos de erro absoluto, erro percentual e ajuste.

Propósito:
Identificar se o modelo está generalizando bem ou superajustando (overfitting).

Passo 20: Regressão Polinomial

Itera por diferentes graus de polinômios (k=1, k=2, k=3).
Expande os atributos preditivos para incluir termos polinomiais.
Treina um modelo de Regressão Linear nos dados transformados.

Motivação:
Explorar modelos mais complexos para capturar relações não lineares entre atributos e a variável alvo.

Propósito:
Testar se a introdução de termos polinomiais melhora a precisão do modelo.

Passo 21: Avaliação do Modelo Polinomial

Avalia os modelos polinomiais usando as mesmas métricas da Regressão Linear.
Exibe os resultados para diferentes graus de polinômios.
Motivação:

Comparar o desempenho da Regressão Polinomial com a Linear para avaliar se a complexidade adicional é justificada.

Essas linhas implementam as etapas finais do pipeline de modelagem:

Divisão: Garante conjuntos de treino e teste independentes.
Normalização: Prepara os dados para modelos sensíveis à escala.
Modelagem: Avalia Regressão Linear como modelo base e Regressão Polinomial para capturar relações não lineares.
Avaliação: Usa métricas abrangentes para comparar modelos e identificar o mais adequado.

Passo 22: Embaralhamento e Preparação para Modelagem

dados_embaralhados: Embaralha os dados para evitar possíveis padrões na ordem original.
x e y: Matriz de atributos preditivos (x) e vetor alvo (y) para o conjunto de treino.
x_final: Matriz de atributos do conjunto de teste.

Motivação:
Garantir que o modelo seja treinado com dados aleatórios e não influenciado por padrões de ordenação.

Passo 23: Regressão KNN (Vizinhos Mais Próximos)

Itera por diferentes valores de k (número de vizinhos).
Ajusta o modelo KNN com pesos baseados na distância entre pontos (weights='distance').

Motivação:
Comparar o desempenho do KNN com diferentes números de vizinhos.

Propósito:
Identificar o melhor valor de k para o modelo.

Passo 24: Geração do CSV com Resultados do KNN

Treina o modelo KNN com k=19 no conjunto de treino final.
Gera previsões no conjunto de teste final.

Passo 25: Random Forest (Seleção de Parâmetros)

Define uma grade de parâmetros para testar com o Random Forest.

Motivação:
Explorar diferentes combinações de hiperparâmetros para otimizar o modelo.

Passo 26: Random Forest com Hiperparâmetros Específicos

Treina o modelo Random Forest com hiperparâmetros fixados.

Motivação:
Ajustar o modelo com parâmetros conhecidos para avaliar seu desempenho.

Passo 27: Validação Cruzada com K-Fold

Usa validação cruzada K-Fold para avaliar o desempenho do Random Forest.

Motivação:
Obter uma avaliação robusta do modelo em diferentes subconjuntos de dados.

Passo 28: Geração do CSV com Resultados do Random Forest

Gera um arquivo CSV com as previsões do Random Forest para o conjunto de teste final.

Motivação:
Produzir um arquivo de saída para envio ou avaliação externa.

Essas linhas abrangem:

Modelagem: Exploram KNN com diferentes valores de k e Random Forest com parâmetros ajustados.
Avaliação: Usam métricas como MSE, RMSE, R^2 e validação cruzada K-Fold.
Exportação: Geram arquivos CSV com previsões para o conjunto de teste.

Execução:

Coeficiente de Correlação de Pearson:

O coeficiente de Pearson mostra a força da relação linear entre cada variável e a variável alvo (preco).
Variáveis como suites (0.710) e quartos (0.589) possuem correlação moderada a forte, enquanto bairro_rico (0.208) possui uma correlação mais fraca.

Análise:
Variáveis com baixa correlação (próximas de 0) foram removidas para reduzir a dimensionalidade dos dados e evitar ruído.

Previsões e Avaliação do Modelo Linear:

Resultados para o modelo de Regressão Linear:
MSE: O erro médio quadrático é alto, indicando que as previsões têm variações consideráveis.
RMSE: Os erros são, em média, da ordem de centenas de milhares, o que reflete a escala dos preços.
R^2: O modelo explica cerca de 65% da variabilidade no conjunto de teste.

RMSPE: O erro percentual médio é de aproximadamente 40%, sugerindo margem de erro significativa.

Análise:
A Regressão Linear apresenta desempenho razoável, mas existem limitações devido à natureza possivelmente não linear dos dados.

Previsões com Regressão Polinomial:

Para graus polinomiais maiores, o RMSE interno diminui (indicando melhor ajuste aos dados de treino).
Contudo, o RMSE externo e o RMSPE explodem, sugerindo overfitting para k=2 e k=3.

Análise:
O modelo polinomial começa a superajustar os dados, resultando em previsões ruins no conjunto de teste.
Aumentar a complexidade sem controle piora a generalização.

Resultados do Random Forest:

Random Forest apresenta uma média de R^2 superior a 81% na validação cruzada, indicando um bom desempenho em explicar a variância dos dados.

Análise:
Random Forest é robusto a outliers e captura relações não lineares, apresentando desempenho superior ao modelo linear.

Iteração por n_estimators:

Com maior número de árvores (n_estimators), o desempenho melhora, mostrando maior estabilidade.

Análise:
A adição de mais árvores reduz a variância do modelo, mas o ganho marginal diminui após certo ponto.

Modelos Comparados:

Regressão Linear: Simples, com desempenho moderado ($R^2 \approx 65\%$)

Regressão Polinomial: Problemas de overfitting em graus mais altos, prejudicando a generalização.

Random Forest: Melhor desempenho, com $R^2 \approx 81\%$ na validação cruzada K-Fold.

Métricas de Avaliação: Random Forest mostrou maior robustez, enquanto a regressão linear foi limitada pela suposição de linearidade.