Análise Preditiva Utilizando Modelos de Regressão e Desenvolvimento de um Sistema em Django

por Tiago Dutra Galvão

Resumo

Neste trabalho de análise de dados e desenvolvimento de um sistema web com inteligência Arfícial (IA) foram aplicados 7 modelos de Machine Learning (Regressão Linear, Regressão Ridge, Regressão Lasso, Random Forest Regressor, Gradiente Boosting Regressor, XGBoost Regressor e Rede Neural MLP Regressor) para estudar o efeito de oito variáveis de entrada (compacidade relativa, área de superfície, área de parede, área de telhado, altura total, orientação, área envidraçada e distribuição da área envidraçada) em duas variáveis de saída: carga de aquecimento (HL) e carga de resfriamento (CL) de edifícios residenciais. O estudo investigou sistematicamente a força de associação entre cada variável usando ferramentas estatísticas clássicas e não-paramétricas para identificar as variáveis mais fortemente relacionadas, identificando multicolinearidade. Os modelos foram avaliados e comparados entre si, por meio das métricas de RMSE, MAE e R^2. Os dados são provenientes de valores de 768 observações representando edifícios residenciais diversos, sendo um conjunto de dez variáveis, oito preditoras e duas alvo. Dentre as oito variáveis preditoras, seis são numéricas e duas são categóricas e suas relações mostram que é possível prever HL e CL com baixos desvios médios absolutos de erro em relação aos valores considerados reais. Dentre todos os modelos treinados e avaliados, o modelo XGBoost otimizado pelos hiperparâmetros foi o que obteve o melhor resultado com relação às métricas utilizadas.

1 Introdução

Existe uma quantidade considerável de pesquisas sobre desempenho energético de edifícios devido às crescentes preocupações com o desperdício de energia e seu impacto no meio ambiente. Os edifícios em países europeus são legalmente obrigados a cumprir requisitos mínimos de eficiência energética. Relatórios sugerem que o consumo de energia em edifícios aumentou constantemente nas últimas décadas, e sistemas de aquecimento, ventilação e ar condicionado (HVAC) representam a maior parte do uso de energia nos edifícios.

A computação da carga de aquecimento (HL) e da carga de resfriamento (CL) é necessária para determinar as especificações dos equipamentos necessários para manter condições confortáveis do ar interior. Para estimar as capacidades de resfriamento e aquecimento, arquitetos e projetistas precisam de informações sobre as características do edifício, o clima e o uso pretendido.

As ferramentas de simulação de energia para edifícios são amplamente utilizadas, mas podem ser demoradas e exigir experiência do usuário. Muitos pesquisadores recorrem a ferramentas de aprendizado de máquina para estudar o efeito de vários parâmetros de construção por ser mais fácil e rápido quando um banco de dados com as faixas de variáveis necessárias está disponível.

1.1 Objetivos do projeto

Uma empresa está implementando em seu ambiente de produção um sistema preditivo voltado para eficiência energética de edifícios. O objetivo é prever o consumo de energia para aquecimento e resfriamento com base em características arquitetônicas e construtivas dos imóveis.

1.2 Descrição do conjunto de dados

Para essa PoC (Proof of Concept), é proposta a criação de um sistema de regressão supervisionada utilizando o conjunto de dados público UCI Energy Efficiency Dataset, disponível no link: https://archive.ics.uci.edu/ml/datasets/energy+efficiency. O dataset possui 768 amostras de edifícios, com as seguintes características:

- Variáveis de entrada (8):
  - X1: Relative Compactness (Compacidade Relativa)
  - X2: Surface Area (Área da Superfície)
  - X3: Wall Area (Área da Parede)
  - X4: Roof Area (Área do Telhado)
  - X5: Overall Height (Altura Total)

  - X6: Orientation (Orientação)
  - X7: Glazing Area (Área Envidraçada)
  - X8: Glazing Area Distribution (Distribuição da Área Envidraçada)

- Variáveis de saída (2):
  - Y1: Heating Load (Carga de Aquecimento)
  - Y2: Cooling Load (Carga de Resfriamento)

Analisando o conjunto dos dados foi possível perceber que as variáveis X6 e X8 correspondem à variáveis categóricas, enquanto as demais são variáveis numéricas. Sendo assim, temos 8 variáveis numéricas: Relative_Compactness, Surface_Area, Wall_Area, Roof_Area, Overall_Height, Glazing_Area, Heating_Load e Cooling_Load e 2 variáveis são categóricas: Orientation e Glazing_Area_Distribution.

2 Metodologia

A metodologia aplicada aqui foi a padrão utilizada em geral para análise de dados de datasets, primeiramente uma Análise Exploratória dos Dados (EDA) para um primeiro contato de como é o comportamento dos dados, seus tipos e suas distribuições e como as variáveis se relacionam, são aplicadas ferramentas da estatística descritiva para tal. Depois é realizado um pré-processamento dos dados com técnicas que podem ser sugeridas pela etapa de EDA, como padronização das variáveis para posterior modelagem, aplicação de técnicas para lidar com multicolinearidade (PCA, seleção de features), verificação se é necessário tratamento de outliers até a possível exploraração de transformações não-lineares para capturar relações complexas.

Posteriormente, é realizada a modelagem dos dados, aqui são gerados modelos (de regressão neste caso específico) onde são carregados os dados pré-processados na etapa anterior, são aplicadas técnicas e métricas para a avaliação dos modelos. Os modelos são treinados e colocados em avaliação, são realizadas também técnicas de validação cruzada para validação dos modelos. Os modelos são testados para cada uma das variáveis alvo e é escolhido o melhor modelo para implementação no sistema web de predição que utiliza framework Django.

2.1 AT1 - Análise Exploratória dos Dados (EDA)

Por meio da análise exploratória dos dados (EDA) foi possível analisar com cautela os tipos corretos de dados (se numéricos ou se categóricos), colunas do dataset com dados faltantes, distribuição dos dados, as correlações entre as variáveis preditoras e principalmente as correlações das variáveis preditordas e as variáveis alvo.

O gráfico da Figura 1 mostra os histogramas das variáveis numéricas, mostrando uma distribuição não gaussiana (não normal).

Figura 1 - Distribuição das variáveis numéricas.

Já a Figura 2 mostra a distribuição para as variáveis categóricas. Já a Figura 3 mostra as densidades de distribuição para as variáveis numéricas por meio da análise kde.

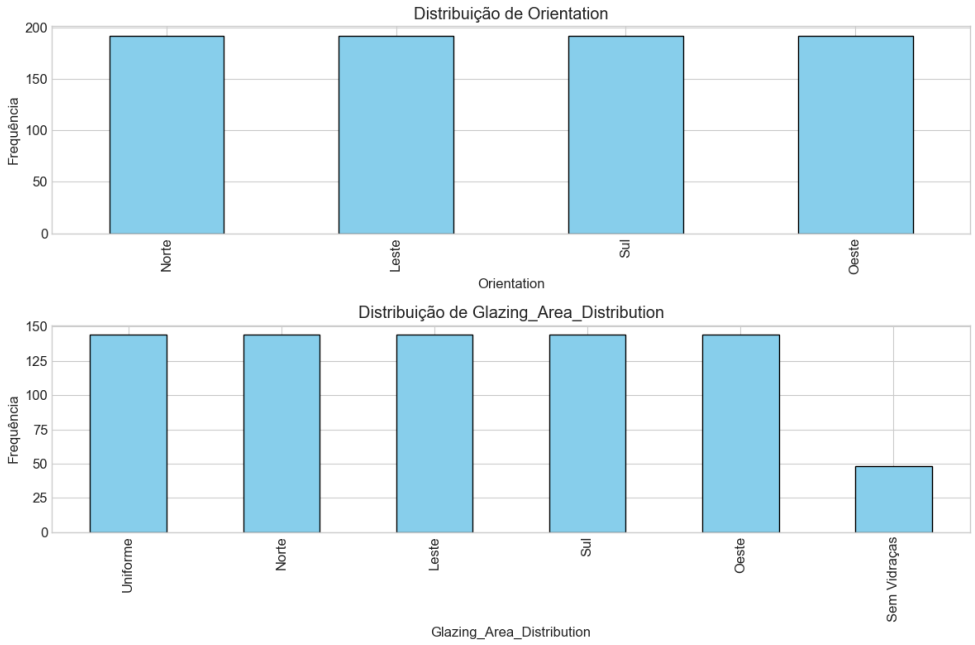

Figura 2 - Distribuição das variáveis categóricas.

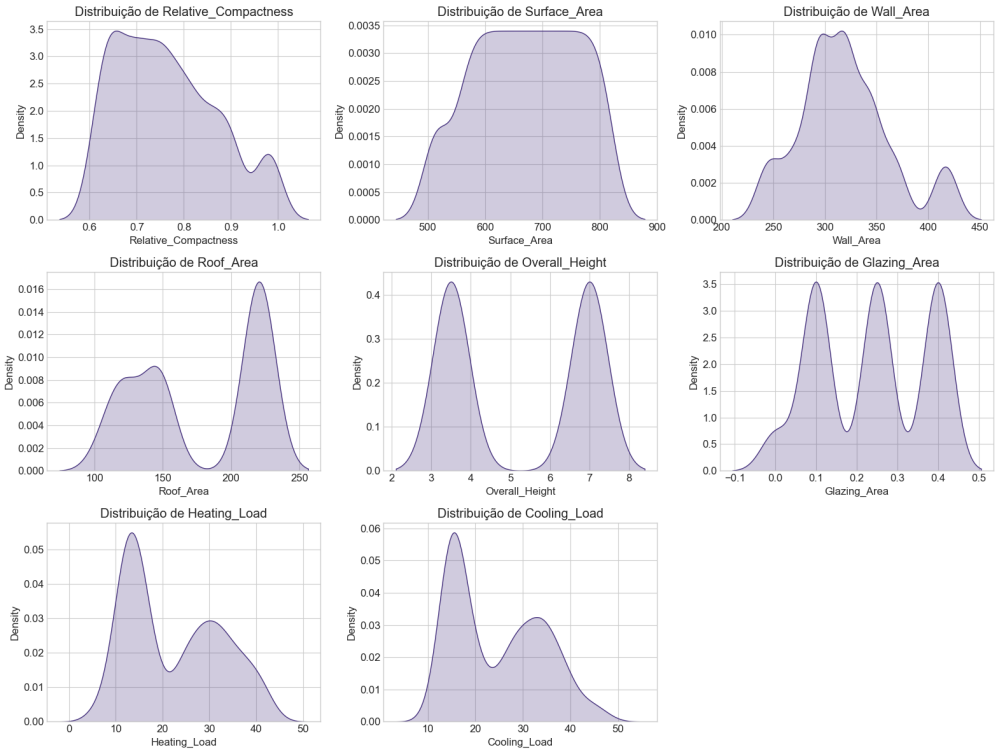

Figura 3 - Densidade de distribuição das variáveis numéricas.

Os gráficos de box-plot mostram que os dados não possuem outliers. É possível ver na Figura 4.

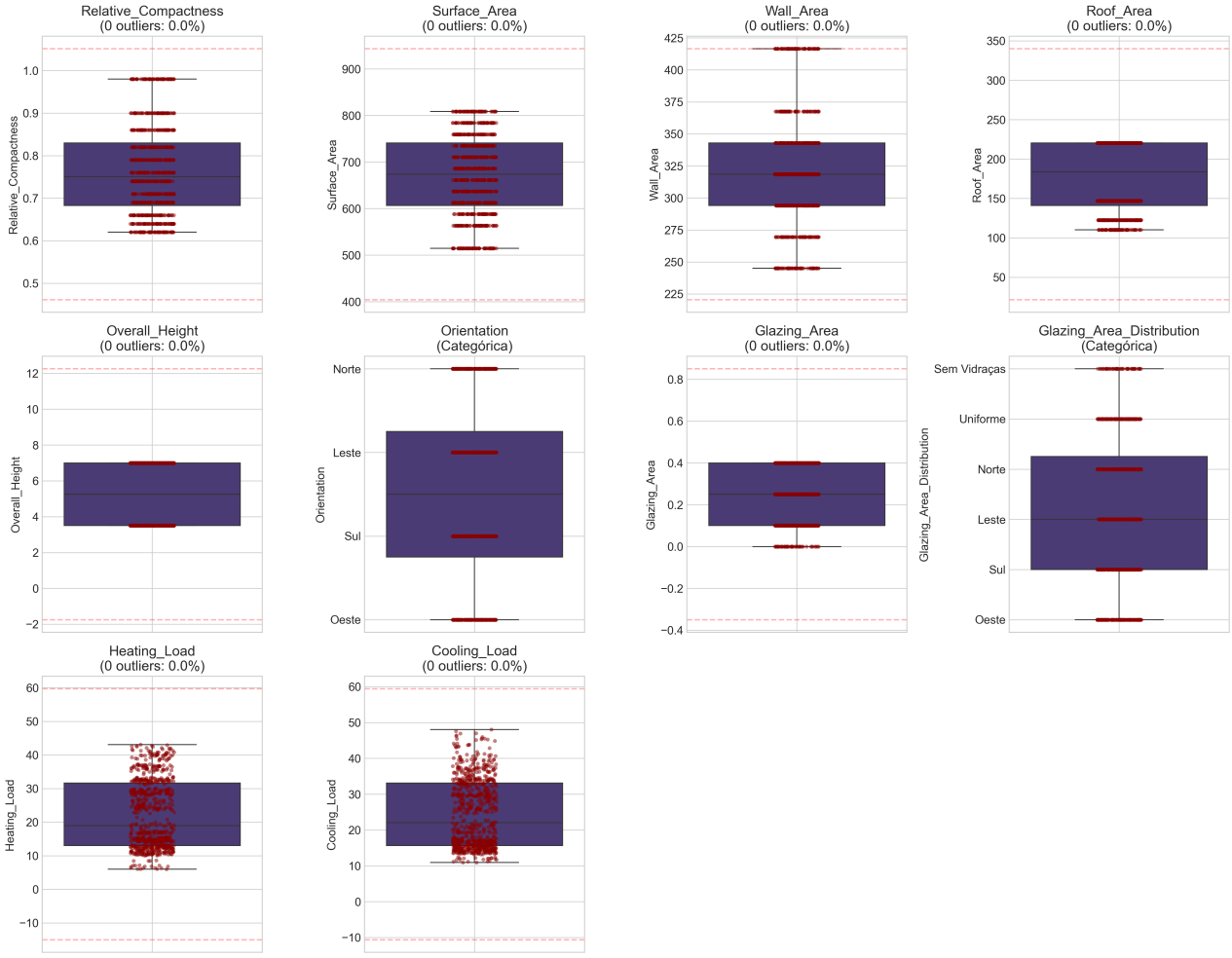

Figura 4 - Box-plot de distribuição das variáveis.

Isso pode ser evidenciado pela tabela de valores dos quartis, descritos na Tabela 1.

Tabela 1 - Densidade de distribuição das variáveis numéricas.

	Variável	Total Outliers	Porcentagem (%)	Outliers Abaixo	Outliers Acima	Limite Inferior	Limite Superior
0	Relative_Compactness	0	0.0	0	0	0.46125	1.05125
1	Surface_Area	0	0.0	0	0	404.25000	943.25000
2	Wall_Area	0	0.0	0	0	220.50000	416.50000
3	Roof_Area	0	0.0	0	0	21.43750	339.93750
4	Overall_Height	0	0.0	0	0	-1.75000	12.25000
5	Glazing_Area	0	0.0	0	0	-0.35000	0.85000
6	Heating_Load	0	0.0	0	0	-15.02000	59.68000
7	Cooling_Load	0	0.0	0	0	-10.64875	59.40125

Foi possível levantar o coeficiente de correlação por meio de um heatmap para as variáveis numéricas. No entanto, para saber a correlação para todas as variáveis, inclusive as categóricas, pode-se aplicar uma técnica para transformar o dataset. Codificação com One-Hot Encoding (OHE) transforma variáveis categóricas em colunas binárias e permite calcular correlações.

Por meio da Figura 5, pode-se perceber as variáveis de maior correlação. Pares de variáveis com alta correlação (>0.7):

- Relative_Compactness e Surface_Area: 0.9919

- Roof_Area e Overall_Height: 0.9725

- Surface_Area e Roof_Area: 0.8807

- Relative_Compactness e Roof_Area: 0.8688

- Surface_Area e Overall_Height: 0.8581

- Relative_Compactness e Overall_Height: 0.8277

Isso revela que tem-se um problema de multicolinearidade no conjunto de dados analisado. Uma solução para isso seria considerar técnicas para lidar com multicolinearidade como PCA e seleção de features (Como explicado em detalhe no notebook).

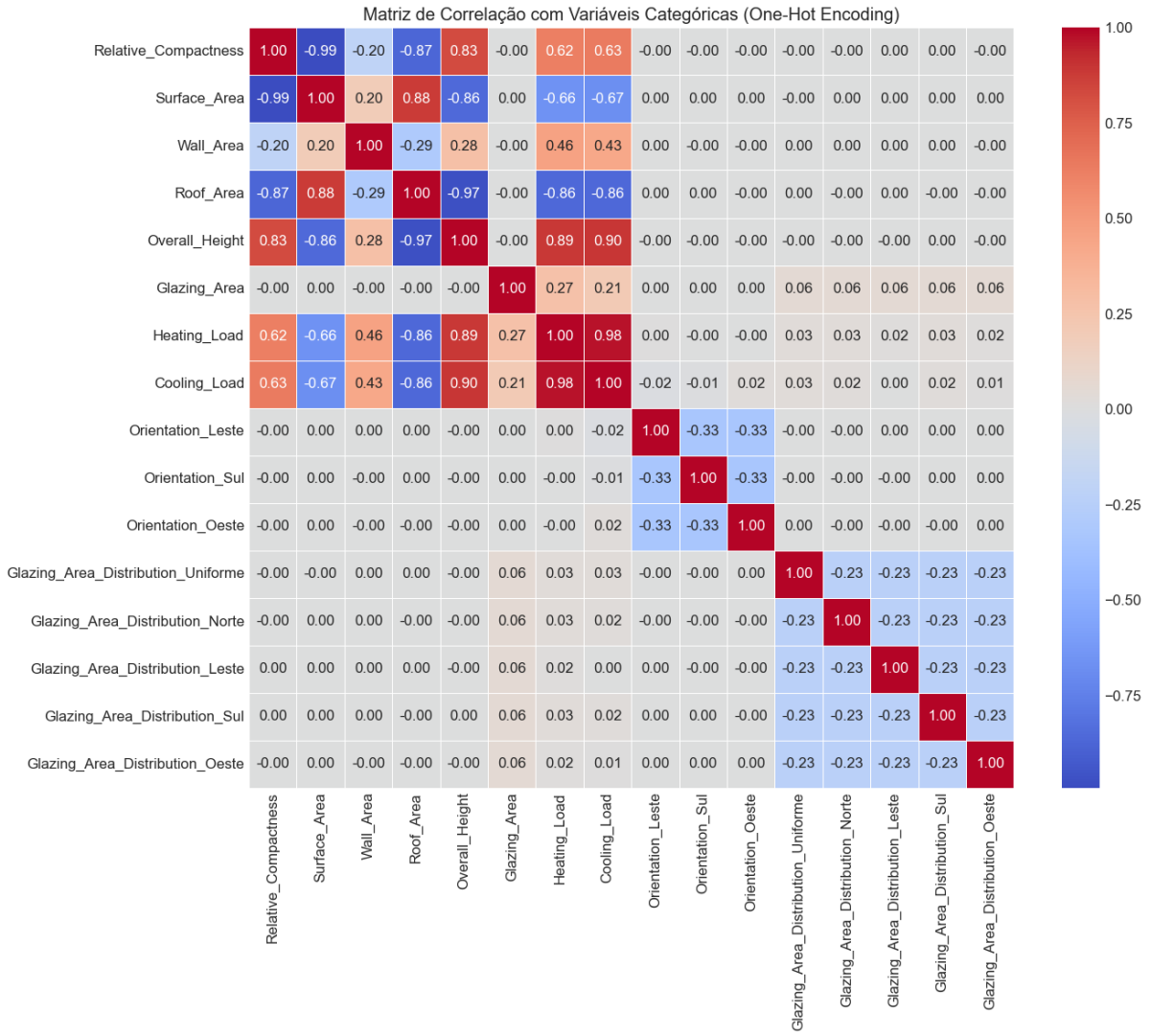

Figura 5 - Heatmap de correlação de Spearman.

Foi realizada uma análise de multicolinearidade chamada VIF. A análise de Variance Inflation Factor (VIF) é uma ferramenta útil para identificar problemas de multicolinearidade entre as variáveis independentes (ou de entrada) em um modelo de regressão.

Multicolinearidade ocorre quando uma ou mais variáveis independentes em um modelo estão altamente correlacionadas entre si. Isso pode causar instabilidade nos coeficientes da regressão, dificultando a interpretação e o entendimento da relação entre as variáveis independentes e a variável dependente.

O VIF mede o quanto a variância de um coeficiente de regressão é aumentada devido à colinearidade com outras variáveis. O cálculo do VIF para cada variável e tem as seguintes condições:

VIF = 1: Nenhuma multicolinearidade. A variável é independente das outras variáveis.

1 < VIF < 5: Multicolinearidade moderada, mas geralmente aceitável.

VIF ≥ 5 ou 10: Indica um problema sério de multicolinearidade, o que significa que a variável está altamente correlacionada com outras variáveis no modelo.

Inf (infinito): Quando o VIF é "inf" (infinito), isso significa que a variável é perfeitamente colinear com outras variáveis no modelo. Em outras palavras, há uma correlação linear perfeita entre as variáveis (Surface_Area, Wall_Area e Roof_Area). Isso pode ocorrer porque essas variáveis estão diretamente relacionadas e uma pode ser expressa como uma combinação linear das outras, o que é um claro indicativo de multicolinearidade forte.

Para resolver isso, podemos remover uma das variáveis colineares ou tentar combinar variáveis altamente correlacionadas (por exemplo, somando ou criando uma nova variável composta).

Relative_Compactness (166.93) e Overall_Height (133.97): Ambos têm um VIF muito alto, o que indica forte multicolinearidade com outras variáveis no modelo. Neste caso, pode-se tentar remover ou transformar essas variáveis para reduzir a multicolinearidade. Uma opção seria verificar a correlação entre essas variáveis para ver quais estão altamente correlacionadas e, se necessário, remover ou combinar algumas delas.

Glazing_Area (4.10): Este VIF está abaixo de 5, o que indica que não há multicolinearidade significativa com outras variáveis. Essa variável pode ser mantida no modelo sem grandes problemas de colinearidade. Mas o que pode ser realizado para resolver esse problema de multicolinearidade?

Como Surface_Area, Wall_Area, e Roof_Area apresentam um VIF infinito, isso sugere que há colinearidade perfeita entre elas. Sendo assim, pode-se excluir uma delas ou combinar essas variáveis em uma nova varirável, por exemplo, somando essas áreas ou criando uma variável composta que represente a área total no caso. As análises da matriz de correlação e de PCA podem nos ajudar a tomar decisões em como resolver essa questão, sendo que a PCA pode ajudar a reduzir a multicolinearidade sem perder informações importantes.

Em resumo, o VIF ajuda a identificar variáveis que podem ser redundantes devido à sua alta correlação com outras. No seu caso, há variáveis que causam multicolinearidade significativa e precisam ser abordadas para melhorar a estabilidade e a interpretabilidade do seu modelo.

A Análise de Componentes Principais (PCA) é uma técnica de redução de dimensionalidade que projeta os dados em um novo espaço de características, de modo que as novas variáveis (os componentes principais) são combinações lineares das variáveis originais e são ordenadas pela quantidade de variância que capturam. As Figuras 6 e 7, mostras os PCA's para CL e HL, respectivamente.

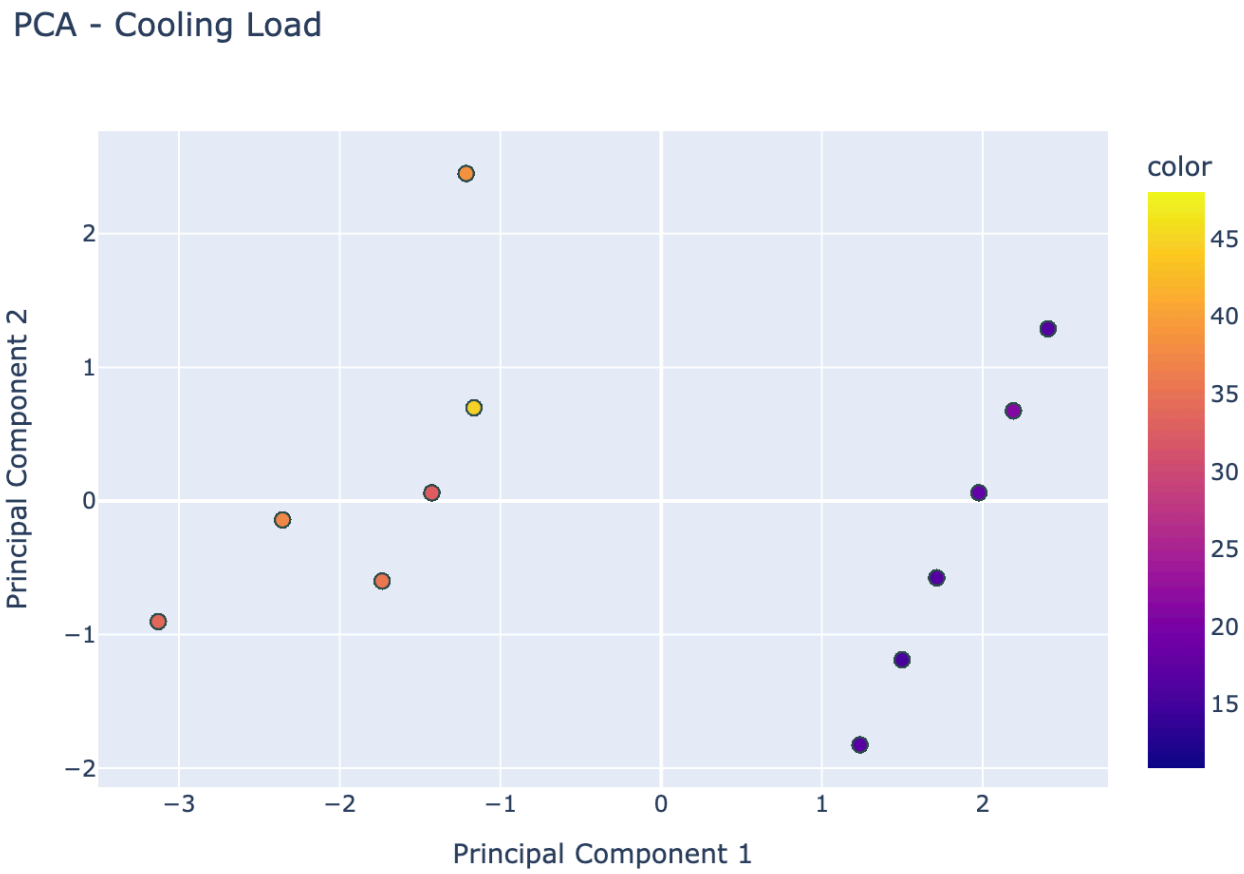

Figura 6 - PCA para Cooling Load.

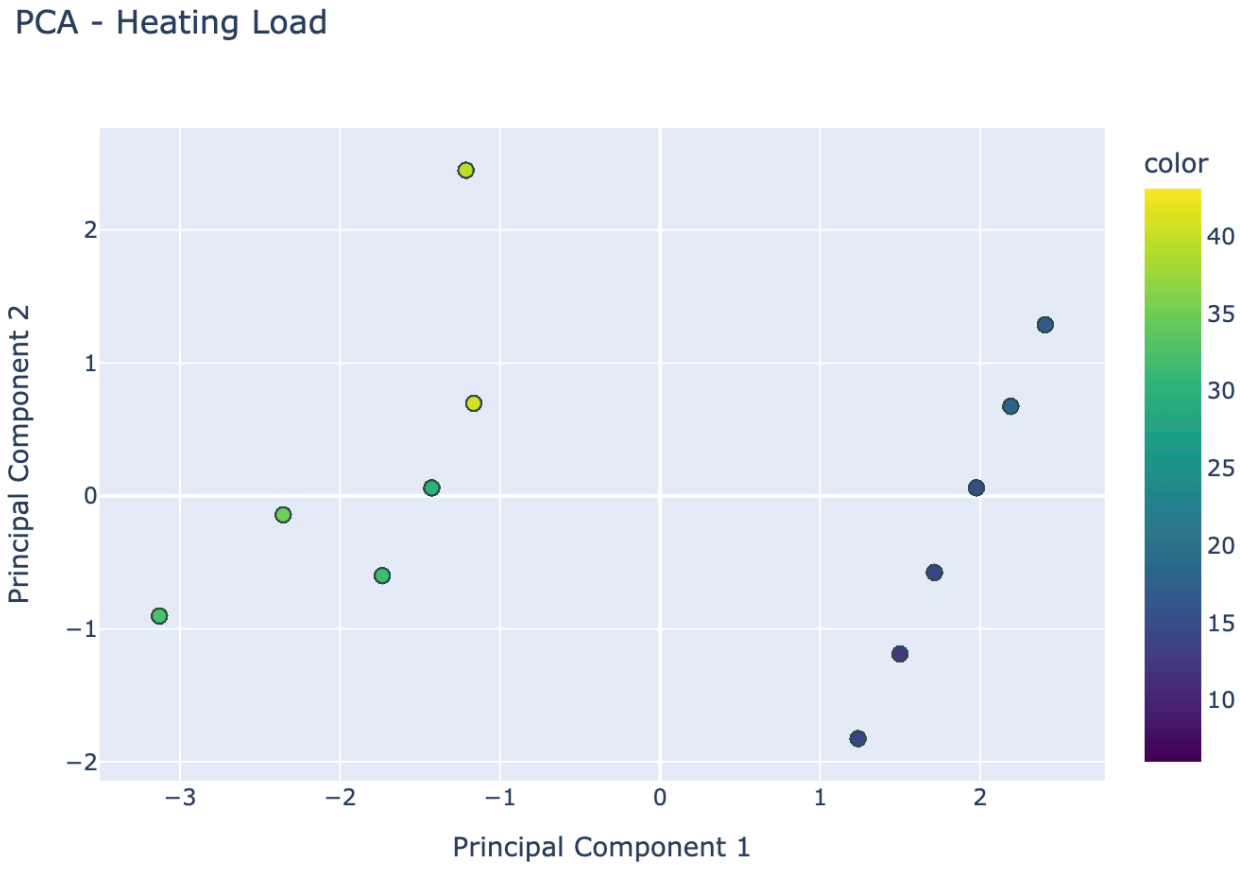

Figura 7 - PCA para Heating Load.

O primeiro componente principal (PC1) captura 61,72% da variância dos dados. Isso significa que PC1 é responsável por uma parte significativa da variação nas variáveis originais. Em outras palavras, se você projetar os dados no espaço formado pelo PC1, você estará mantendo mais de 60% da informação original. Esse componente provavelmente reflete a maior parte da estrutura subjacente dos dados.

O segundo componente principal (PC2) explica 20,66% da variância. Embora menos importante que o PC1, ele ainda captura uma quantidade significativa de variação nos dados. O PC2, junto com o PC1, representa 82,38% da variância total nos dados, o que significa que você pode representar a maior parte da informação original com apenas dois componentes principais.

A variância explicada total por PC1 e PC2 é de 82,38%, o que é bastante alto. Isso indica que, ao utilizar apenas os dois primeiros componentes principais, você pode reduzir significativamente a dimensionalidade dos dados sem perder muita informação. Em outras palavras, dois componentes principais (PC1 e PC2) são suficientes para descrever a maior parte da variabilidade nos dados.

Mas quais são as principais implicações desses resultados?

1 - Redução de Dimensionalidade: Você conseguiu reduzir as variáveis originais para 2 componentes principais que mantêm mais de 80% da informação original. Isso é útil quando você quer simplificar o modelo ou visualizar os dados de forma mais clara, mantendo a maior parte da variabilidade.

2 - Visualização: Com apenas dois componentes principais, você pode projetar seus dados em um gráfico 2D, facilitando a visualização da estrutura e das relações entre as observações. Isso é especialmente útil para explorar padrões, agrupamentos e outliers.

Pode-se criar um gráfico de dispersão (scatter plot) para visualizar como os dados se distribuem no novo espaço formado pelos dois componentes principais. Isso pode ajudar a identificar clusters, tendências ou padrões não aparentes nas variáveis originais.

Para entender melhor o que cada componente significa, pode-se olhar os coeficientes de cada variável nos componentes principais. Se olharmos os valores das componentes, poderemos ter uma ideia de como as variáveis originais influenciam os componentes principais.

Em resumo, os pontos representam observações do seu dataset e suas coordenadas no gráfico PCA mostram como essas observações se distribuem com base nas variáveis originais, enquanto as cores ajudam a entender como uma variável de interesse (como Heating_Load) se relaciona com essa distribuição.

Conclusões tiradas da análise exploratória EDA são:

1. Variáveis mais correlacionadas com Heating Load:

- Overall_Height: 0.8894
- Relative_Compactness: 0.6223
- Wall_Area: 0.4557
- Glazing_Area: 0.2698
- Surface_Area: -0.6581
- Roof_Area: -0.8618

2. Variáveis mais correlacionadas com Cooling Load:

- Overall_Height: 0.8958
- Relative_Compactness: 0.6343
- Wall_Area: 0.4271
- Glazing_Area: 0.2075
- Surface_Area: -0.6730
- Roof_Area: -0.8625

3. Correlação entre Heating Load e Cooling Load: 0.9759

4. Possível multicolinearidade detectada entre algumas variáveis de entrada.

Variáveis com VIF > 10 (indicativo de multicolinearidade): ['Relative_Compactness', 'Surface_Area', 'Wall_Area', 'Roof_Area', 'Overall_Height']

5. Presença de outliers não detectada nas variáveis.

6. Recomendações para pré-processamento:

- Padronização das variáveis para modelagem
- Considerar técnicas para lidar com multicolinearidade (PCA, seleção de features)
- Verificar se é necessário tratamento de outliers
- Explorar transformações não-lineares para capturar relações complexas

OBS.: Mais análises foram realizadas e se encontram junto aos códigos nos arquivos notebook, os da EDA mais especificamente em **01_exploracao_inicial.ipynb**, como gráficos das top 10 variáveis mais correlacionadas com as variáveis alvo, análise de relação entre variáveis de entrada e Heating Load e Cooling Load entre outras.

2.2 AT2 - Modelagem e Avaliação de Algoritmos

Nesta PoC de análise de dados e desenvolvimento de um sistema web com inteligência Arfícial (IA) foram aplicados 7 modelos de Machine Learning (Regressão Linear, Regressão Ridge, Regressão Lasso, Random Forest Regressor, Gradiente Boosting Regressor, XGBoost Regressor e Rede Neural MLP Regressor) para estudar o efeito das oito variáveis de entrada em duas variáveis de saída: carga de aquecimento (HL) e carga de resfriamento (CL) de edifícios residenciais.

A escolha dos modelos empregados está associada as potencialidades dos mesmos para análises preditivas nesse contexto de dados.

A **Regressão Linear** é um dos algoritmos mais simples e fundamentais em machine learning, modelando a relação entre variáveis independentes e uma variável dependente através de uma equação linear. Suas potencialidades são simplicidade e facilidade de interpretação rápido treinamento e previsão, baixo custo computacional em comparação a outras técnicas, fornece coeficientes que indicam a importância e direção do efeito de cada variável. Além de todas essas vantagens ela serve de base para entender modelos mais complexos.

Suas desvantagens, assume inerentemente uma relação linear entre variáveis (limitação em problemas não-lineares), é bem sensível a outliers, possui desempenho inferior em dados com alta dimensionalidade, sendo vulnerável à multicolinearidade (alta correlação entre variáveis independentes). Portanto, não lida bem com interações complexas entre variáveis de interesse.

Por sua vez, a **Regressão Ridge** é uma extensão da regressão linear que adiciona regularização L2, penalizando a soma dos quadrados dos coeficientes, ajudando a reduzir a variância do modelo. Possui como vantagens a redução de overfitting ao penalizar coeficientes grandes, lida bem com multicolonearidade, mantém todas as features no modelo apenas reduzindo sua influência. Tem a capacidade de estabilizar modelos quando possuem muitas varia;aveis, tudo isso mantendo a interpretabilidade similar à regressão linear.

Por outro lado, ainda assume relacionamentos lineares, não realiza seleção de variáveis (mantém todas), requer ajuste do hiperparâmetro de regularização (alpha), pode possuir desempenho inferior em problemas altamente não-lineares e exige mais do poder computacional que a regressão linear simples.

Já a **Regressão Lasso** (Least Absolute Shrinkage and Selection Operator) utiliza regularização L1, penalizando a soma dos valores absolutos dos coeficientes, promovendo esparsidade no modelo. Tem como vantagens seleção automática de features (zerando coeficientes irrelevantes), reduz overfitting, cria modelos mais simples e interpretáveis, é eficaz para conjuntos de dados com muitas variáveis e bastante útil quando se suspeita que apenas algumas variáveis são importantes. Suas limitações estão em assumir relacionamentos lineares. Pode ser instável quando há features altamente correlacionadas, tende a selecionar arbitrariamente uma variável de um grupo correlacionado, requer ajuste cuidadoso do hiperparâmetro de regularização e pode remover variáveis que têm relações não-lineares importantes com a variável alvo, ocasionando perda de informação relevante.

Já **Random Forest** é um método ensemble baseado em múltiplas árvores de decisão, onde cada árvore é treinada em um subconjunto aleatório dos dados e features, combinando as previsões para gerar um resultado final. Suas potencialidades estão em capturar relações não-lineares e interações complexas, além da robustez contra overfitting. Não requer normalização dos dados, fornece medidas de importância das variáveis, além de lidar bem com dados faltantes e outliers, além de possuir um excelente desempenho em uma ampla gama de problemas. Suas fragilidades estão na menor interpretabilidade comparado a modelos lineares, pode ser computacionalmente intensivo para grandes conjuntos de dados, tem tendência a favorecer variáveis com muitos níveis em dados categóricos, não extrapola bem além dos limites dos dados de treinamento e pode requerer ajuste fino de vários hiperparâmetros.

O modelo **Gradient Boosting** é um método ensemble que constrói árvores sequencialmente, onde cada nova árvore corrige os erros das árvores anteriores, aprendendo gradualmente a melhorar a previsão. Vantagens desse modelo são maior precisão que Random Forest, excelente em capturar relações não-lineares, fornece medidas de importância das variáveis, flexibilidade para otimizar diferentes funções de

perda, além de ser eficaz em conjuntos de dados estruturados. Suas desvantagens se configuram em ser mais propenso a overfitting que Random Forest, computacionalmente intensivo, sensível ao ruído e outliers, também requer ajuste cuidadoso de hiperparâmetros, e o treinamento sequencial dificulta a paralelização.

Já o modelo **XGBoost** (Extreme Gradient Boosting) é uma implementação otimizada de Gradient Boosting que incorpora regularização, paralelização e outras otimizações algorítmicas para melhorar desempenho e velocidade. Potencialidades são a performance superior em muitas competições de machine learning, escalabilidade e eficiência computacional, além de incorporar regularização para controle de overfitting, lida bem com valores faltante, suporta paralelização e computação distribuída, versatilidade para customização de objetivos e métricas. Suas desvantagens são a complexidade em ajustes de hiperparâmetros (muitos parâmetros para otimizar), menor interpretabilidade que modelos lineares, pode ser overkill para problemas simples, requer cuidadoso pré-processamento para variáveis categóricas, além do consumo de memória pode ser alto com grandes conjuntos de dados.

Por fim, **Rede Neural MLP Regressor** (Multi-Layer Perceptron) é uma rede neural com pelo menos três camadas (entrada, oculta e saída) que pode aprender representações complexas e não-lineares através de funções de ativação e backpropagation. Suas vantagens estão na capacidade de modelar relações altamente complexas e não-lineares, potencial para alta precisão em conjuntos de dados grandes, possui bom desempenho em problemas de alta dimensionalidade, boa capacidade de aprender representações hierárquicas dos dados, flexibilidade para adaptação a diversos tipos de problemas. Como desvantagens tem-se a baixa interpretabilidade ("caixa preta"), requer grandes conjuntos de dados para bom desempenho, computacionalmente intensivo no treinamento, sensível à escala dos dados (requer normalização), propenso a overfitting sem regularização adequada. Além de tudo isso, requer muitos hiperparâmetros para ajustar (arquitetura, funções de ativação, otimizadores). Sua convergência pode ser desafiadora e sensível a inicializações aleatórias.

Antes de aplicar quaisquer um dos modelos, foi realizado o pré-processamento dos dados, sendo ele realizado pelo código que se encontra no notebook **01_preprocessamento.ipynb**. O código apresentado implementa uma sequência robusta e metodológica de pré-processamento para o conjunto de dados sobre eficiência

energética de edifícios. Este processo é fundamental para preparar os dados antes da aplicação de algoritmos de machine learning. Abaixo está uma descrição detalhada das etapas realizadas.

Inicialmente, o código importa bibliotecas essenciais para manipulação de dados (pandas, numpy), visualização (matplotlib, seaborn) e pré-processamento (sklearn). Em seguida, carrega o conjunto de dados previamente processado na etapa AT1, disponível no caminho '../../data/processed/energy_efficiency_processed.csv'. As primeiras linhas do dataset são exibidas para verificação inicial da estrutura dos dados, bem como os tipos de dados de cada coluna. Para a preparação das variáveis o dataset é então dividido em variáveis independentes (features) e variáveis dependentes (targets). As features são armazenadas na variável X, excluindo as colunas 'Heating_Load' e 'Cooling_Load', que representam as cargas de aquecimento e resfriamento, respectivamente. Estas duas variáveis alvo são armazenadas separadamente em y_heating e y_cooling.

Em seguida, é realizada a identificação automática de colunas categóricas e numéricas baseada nos tipos de dados presentes no dataset. Esta etapa é crucial para aplicar transformações específicas a cada tipo de variável.

Os dados são divididos em conjuntos de treino e teste na proporção 70-30, mantendo a mesma divisão para ambas as variáveis alvo. A função train_test_split com random_state=42 garante uma divisão consistente e reprodutível dos dados. São impressos os tamanhos resultantes dos conjuntos para verificação.

Um pipeline de pré-processamento é construído utilizando o ColumnTransformer, que permite aplicar transformações diferentes para tipos de dados diferentes:

Para variáveis numéricas: É aplicado o StandardScaler, que padroniza as variáveis numéricas subtraindo a média e dividindo pelo desvio padrão. Esta etapa é crucial para algoritmos sensíveis à escala das variáveis, como regressão linear, Ridge, Lasso e redes neurais.

Para variáveis categóricas: É aplicado o OneHotEncoder com a opção drop='first', que converte variáveis categóricas em variáveis binárias (dummies), eliminando a primeira categoria de cada variável para evitar multicolinearidade.

O preprocessador é aplicado aos conjuntos de treino e teste, garantindo que ambos passem pelas mesmas transformações, mas sem permitir vazamento de informações do conjunto de teste para o treino (fit apenas no conjunto de treino).

Para o tratamento de matrizes esparsas o código verifica se o resultado do pré-processamento é uma matriz esparsa (comum quando há muitas variáveis categóricas) e, em caso positivo, converte-a para um array denso para facilitar operações subsequentes.
Já a extração dos nomes das Features transformadas, logo após o pré-processamento são extraídos, incluindo os novos nomes gerados pelo OneHotEncoder para as variáveis categóricas. Isto é importante para manter a interpretabilidade do modelo após o pré-processamento.

Por fim, todos os dados pré-processados (conjunto de treino, conjunto de teste e variáveis alvo) são salvos em formato numpy (.npy) para uso posterior nas etapas de modelagem. O preprocessador também é persistido utilizando joblib, o que permitirá aplicar exatamente as mesmas transformações a novos dados no futuro, garantindo consistência na transformação.

Esta abordagem metodológica de pré-processamento garante que os dados estejam adequadamente preparados para a aplicação dos diversos algoritmos de machine learning (Regressão Linear, Ridge, Lasso, Random Forest, Gradient Boosting, XGBoost e Redes Neurais MLP), facilitando a comparação justa entre eles e potencializando seu desempenho preditivo.

Após o pré-processamento dos dados, os dados pré-processados são carregados e armazenados para o processo de modelagem. Aqui pode ser visto no código do notebook **02_modelagem.ipynb** que o código apresentado implementa um pipeline completo de modelagem para prever a eficiência energética de edifícios, focando nas variáveis-alvo de carga de aquecimento (Heating Load) e carga de resfriamento (Cooling Load). O processo segue uma abordagem sistemática e organizada, permitindo a comparação de diversos algoritmos de machine learning para identificar o modelo mais adequado para cada variável de interesse. O processo inicia com a importação das bibliotecas necessárias para modelagem, incluindo implementações de diversos algoritmos de regressão (desde abordagens lineares até modelos baseados em árvores e redes neurais), ferramentas de avaliação de desempenho e visualização.
Em seguida, são carregados os dados pré-processados da etapa anterior, incluindo as matrizes de features (X_train e X_test) e os vetores de variáveis-alvo para carga de aquecimento e resfriamento, tanto para treino quanto para teste. Este carregamento usa arquivos numpy previamente salvos, garantindo consistência com a etapa de pré-

processamento. Um componente central do código é a função *avaliar_modelo*, que implementa uma metodologia robusta e abrangente para avaliar o desempenho de cada algoritmo. Esta função: mensura o tempo de treinamento, um fator importante para aplicações práticas, treina o modelo com dados de treinamento, realiza predições nos conjuntos de treino e teste, calcula múltiplas métricas de desempenho, como RMSE (Root Mean Squared Error) que penaliza erros maiores, importante para detectar desvios significativos, MAE (Mean Absolute Error) que representa a magnitude média dos erros em escala original e R^2 (Coeficiente de Determinação) que indica a proporção da variância na variável dependente que é explicada pelo modelo. Além disso, executa validação cruzada com 5 folds para estimar o desempenho em dados não vistos de forma mais robusta, salva o modelo treinado em disco para uso posterior e registra todos os resultados em um formato estruturado. Esta abordagem garante uma avaliação consistente e multidimensional de todos os modelos.

Após isso, os sete algoritmos de regressão diferentes são avaliados para cada variável-alvo (Heating Load e Cooling Load). Cada modelo é configurado com hiperparâmetros básicos, estabelecendo uma base de comparação justa. Esta diversidade de algoritmos permite avaliar desde abordagens lineares simples até modelos não-lineares complexos.

O código executa sistematicamente o treinamento e avaliação de cada modelo para ambas as variáveis-alvo, proporcionando feedback durante o processo. Os resultados são coletados em estruturas de dados organizadas para posterior análise e visualização.

Após o treinamento de todos os modelos, os resultados são consolidados em um DataFrame e salvos em formato CSV para documentação e análise posterior. Adicionalmente, são geradas visualizações comparativas gráficos de barras comparando o RMSE dos diferentes modelos para cada variável-alvo, gráficos de barras comparando o R^2 dos diferentes modelos, identificação e destaque dos melhores modelos para cada variável-alvo baseado no R^2.

Estas visualizações são salvas como arquivos de imagem, facilitando sua inclusão em relatórios e apresentações. Também é realizada a identificação dos melhores modelos.

O código finaliza com a identificação explícita dos melhores modelos para cada variável-alvo baseado no coeficiente de determinação (R^2) no conjunto de teste, fornecendo também os valores de RMSE correspondentes. Esta análise direta permite selecionar rapidamente os modelos mais adequados para implementação em produção.

Este processo de modelagem representa uma abordagem metodológica completa e robusta, avaliando sistematicamente múltiplos algoritmos para encontrar a melhor solução para o problema de predição de eficiência energética em edifícios, alinhando-se às melhores práticas em ciência de dados e aprendizado de máquina.

Tabela 2 - Modelos e os valores obtidos de suas métricas.

	Modelo	Target	RMSE_Train	RMSE_Test	MAE_Train	MAE_Test	R2_Train	R2_Test	CV_RMSE	Tempo_Treino
0	LinearRegression	Heating_Load	2.746273	2.887576	1.974129	2.101903	0.925754	0.917768	2.843623	0.001460
1	Ridge	Heating_Load	2.749982	2.886009	1.981527	2.100602	0.925554	0.917857	2.846977	0.001593
2	Lasso	Heating_Load	2.881241	2.960829	2.078622	2.136737	0.918277	0.913543	2.919405	0.001740
3	RandomForest	Heating_Load	0.205954	0.507999	0.134452	0.341333	0.999582	0.997455	0.590152	0.062200
4	GradientBoosting	Heating_Load	0.405906	0.494623	0.283330	0.359827	0.998378	0.997587	0.486084	0.033783
5	XGBoost	Heating_Load	0.067280	0.404146	0.042418	0.291071	0.999955	0.998389	0.513470	0.077270
6	NeuralNetwork	Heating_Load	0.291486	0.615403	0.203502	0.436592	0.999164	0.996265	0.644033	1.827334
7	LinearRegression	Cooling_Load	3.138550	3.204980	2.222192	2.287451	0.890782	0.886043	3.245385	0.000990
8	Ridge	Cooling_Load	3.142225	3.204162	2.225101	2.285438	0.890527	0.886101	3.248444	0.000279
9	Lasso	Cooling_Load	3.272394	3.300865	2.333458	2.338899	0.881269	0.879123	3.323006	0.000630
10	RandomForest	Cooling_Load	0.736338	1.882096	0.433007	1.159472	0.993988	0.960702	2.005827	0.116220
11	GradientBoosting	Cooling_Load	1.361287	1.515806	0.903960	1.038504	0.979454	0.974510	1.649304	0.028863

As Figuras 8 e 9, mostram, respectivamente, a comparação das métricas R^2 e RMSE para os modelos com relação as duas variáveis alvo.

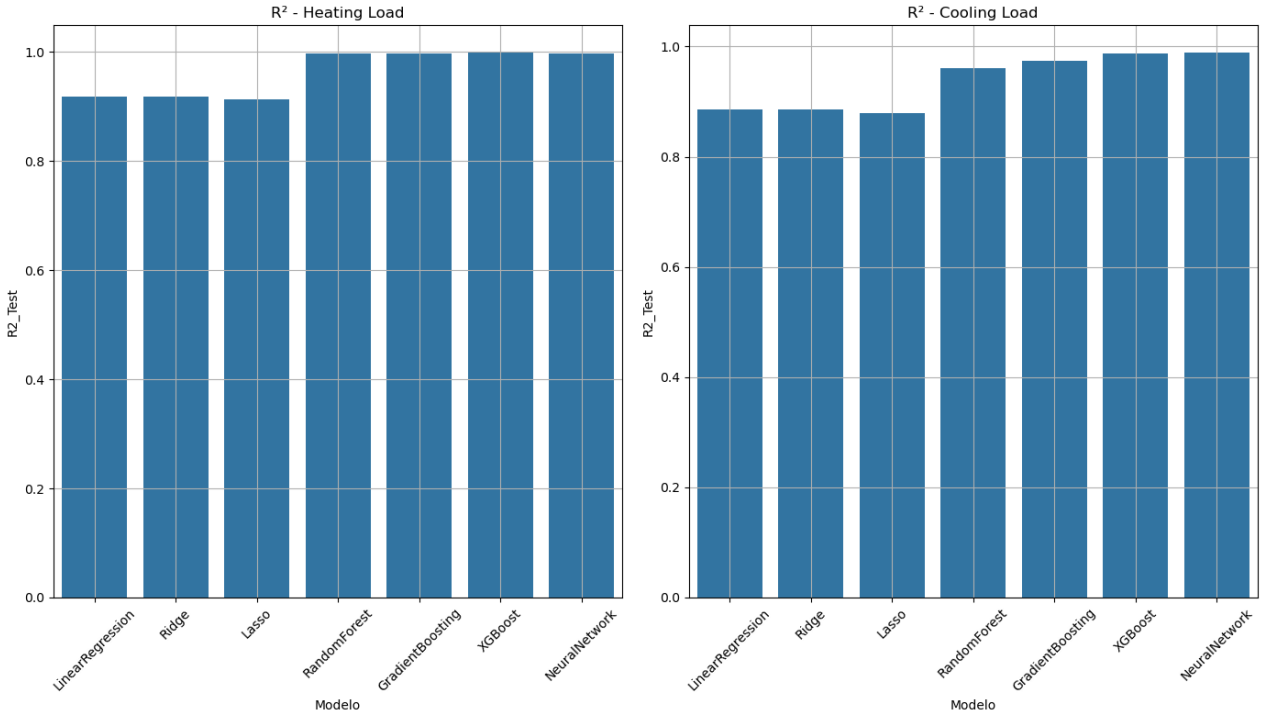

Figura 8 - Comparação dos coeficientes para os modelos.

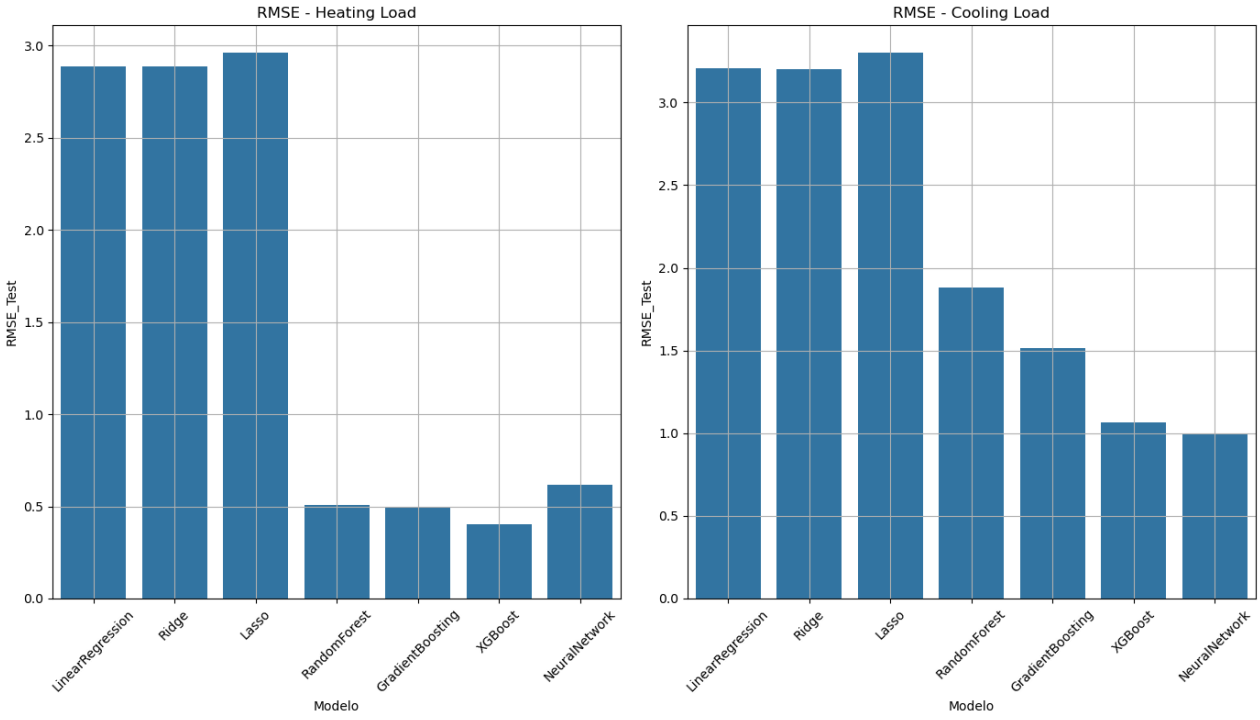

Figura 9 - Comparação dos RMSE para os modelos.

Os modelos também passaram por uma otimização dos hiperparâmetros visto no notebook **03_otimizacao_hiperparametros.ipynb**. O código apresentado implementa um processo sistemático de otimização de hiperparâmetros para aprimorar o desempenho dos modelos de machine learning previamente identificados como os mais promissores na etapa de modelagem. Esta otimização representa uma fase crítica no desenvolvimento de modelos preditivos robustos para eficiência energética de edifícios, permitindo um ajuste fino dos algoritmos para maximizar sua capacidade preditiva. O processo inicia com o carregamento dos dados pré-processados (conjuntos de treino e teste) e dos resultados da avaliação inicial dos modelos. Uma análise inteligente é realizada para identificar os três melhores modelos para cada variável-alvo (carga de aquecimento e carga de resfriamento) com base no coeficiente de determinação (R^2), estabelecendo um foco estratégico para o processo de otimização.

Os resultados são exibidos em formato tabular, proporcionando uma visão clara dos modelos candidatos à otimização e seu desempenho atual, servindo como linha de base para avaliar as melhorias obtidas após o processo de otimização. Um componente fundamental do código é a definição estruturada dos espaços de busca de hiperparâmetros para cada tipo de algoritmo. Para cada modelo, são especificados conjuntos criteriosos de hiperparâmetros a serem explorados.

Ridge: Variações do parâmetro alpha de regularização (0.01 a 100.0).

Lasso: Diferentes níveis de regularização alpha (0.001 a 10.0).

Random Forest: Combinações de número de árvores, profundidade máxima e critérios de divisão de nós.

Gradient Boosting: Variações em taxa de aprendizado, número de estimadores, profundidade e taxa de subamostragem.

XGBoost: Configurações de árvores, taxa de aprendizado e amostragem de colunas

Redes Neurais: Diferentes arquiteturas, funções de ativação e parâmetros de aprendizado

Estes espaços de busca foram cuidadosamente projetados para cobrir um espectro significativo de configurações possíveis, mantendo a viabilidade computacional.

A função *otimizar_modelo* implementa um processo abrangente de otimização utilizando busca em grade (Grid Search) com validação cruzada. Esta abordagem:

instancia o modelo base com configuração inicial, configura a busca em grade com o espaço de hiperparâmetros específico do modelo, utiliza validação cruzada com 5 folds para garantir robustez na avaliação, paraleliza o processo para otimizar o tempo de execução, identifica a melhor combinação de hiperparâmetros com base no R^2, avalia o modelo otimizado nos conjuntos de treino e teste, registra métricas detalhadas de desempenho (RMSE, MAE, R^2), persiste o modelo otimizado para uso posterior Esta metodologia garante uma exploração sistemática e estatisticamente robusta do espaço de hiperparâmetros.

O código executa sequencialmente a otimização dos melhores modelos para cada variável-alvo, com feedback em tempo real sobre o progresso e resultados. Ao término de cada otimização, são registrados os parâmetros ótimos e as métricas de desempenho correspondentes, permitindo uma análise imediata das melhorias obtidas.

Após a otimização de todos os modelos candidatos, o código implementa uma análise comparativa abrangente: consolida os resultados em um DataFrame estruturado, salva os resultados em formato CSV para documentação e análise posterior, gera visualizações comparativas em formato de gráficos de barras, destacando comparação de RMSE entre modelos originais e otimizados para carga de aquecimento, comparação de RMSE entre modelos originais e otimizados para carga de resfriamento, comparação de R^2 entre modelos originais e otimizados para ambas as variáveis-alvo.

Estas visualizações são organizadas em um painel de 2x2 gráficos e salvas como imagem, proporcionando uma representação visual clara do impacto da otimização no desempenho dos modelos.

O código conclui com a identificação explícita do melhor modelo otimizado para cada variável-alvo, destacando: o nome do modelo com melhor desempenho, as métricas de desempenho (R^2 e RMSE) alcançadas, os hiperparâmetros ótimos encontrados durante a otimização.

Esta informação final é crítica para a seleção do modelo a ser implementado em produção, fornecendo uma visão clara e objetiva do resultado do processo de otimização.
Este processo de otimização de hiperparâmetros representa uma abordagem metodológica rigorosa para refinar os modelos de machine learning, maximizando seu poder preditivo para a estimativa de eficiência energética de edifícios. A estrutura sistemática, a documentação detalhada e as análises comparativas fornecem uma base

sólida para decisões informadas sobre a seleção de modelos para implementação. A Figura 10 mostra a comparação das métricas RMSE e R² para os modelos originais e otimizados.

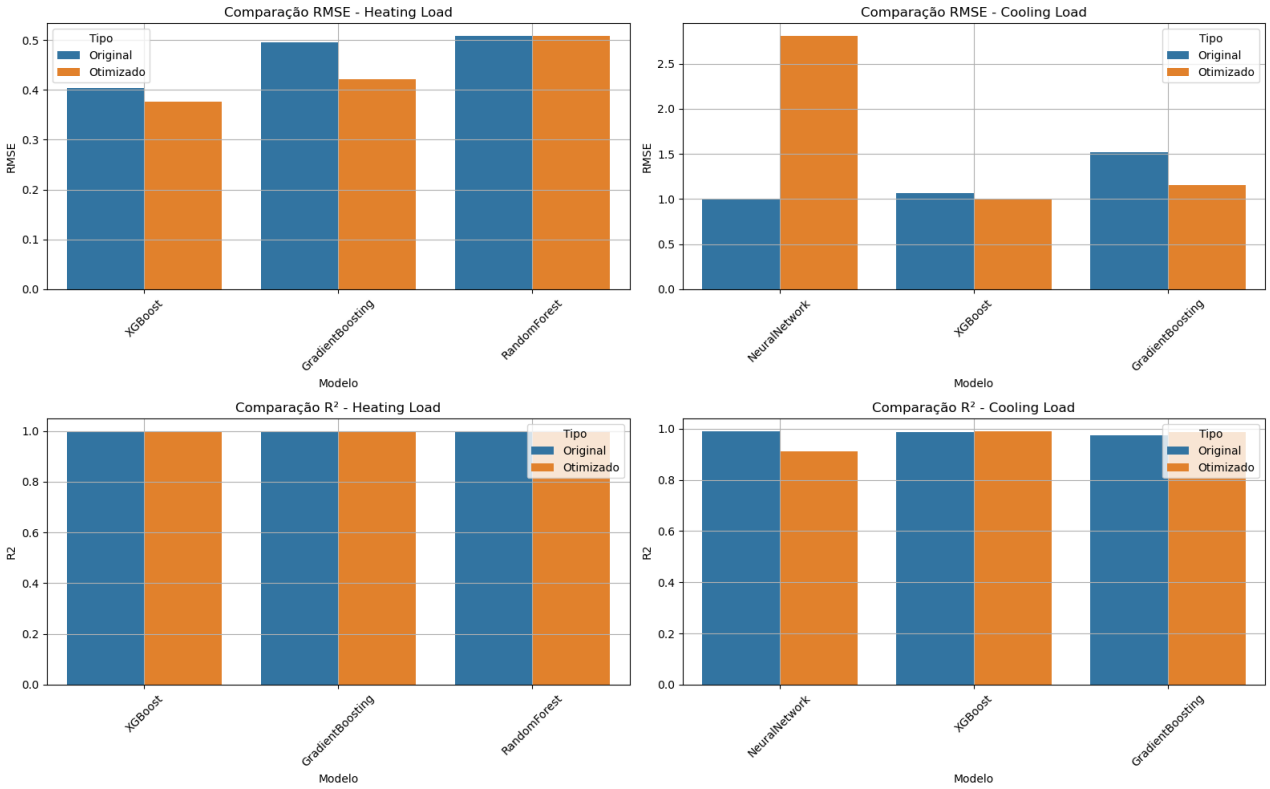

Figura 10 - Comparação das métricas entre os modelos originais e otimizados.

A Figura 11 mostra uma comparação das métricas de todos os modelos para as variáveis de saída.

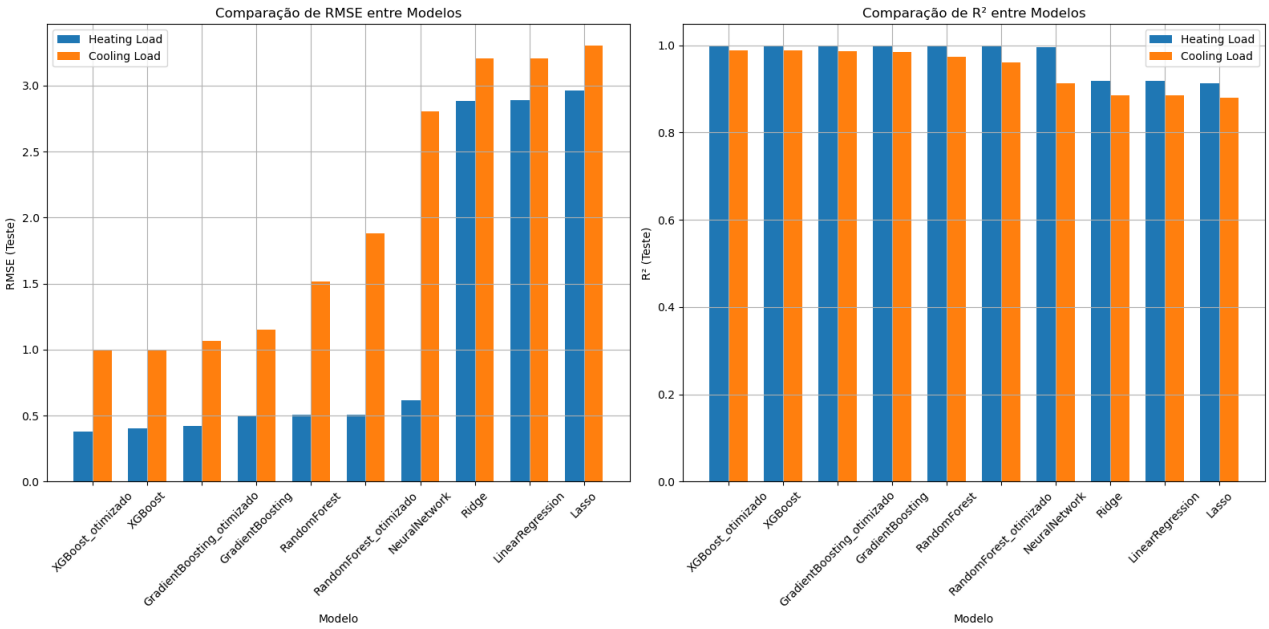

Figura 11 - Comparação das métricas entre os modelos finais para as variáveis alvo.

3 Resultados

O código apresentado no notebook **04_analise_modelos.ipynb** implementa uma análise avançada dos modelos preditivos para eficiência energética de edifícios, com foco em técnicas de interpretabilidade, diagnóstico de desempenho e avaliação comparativa.

Esta etapa final do pipeline de modelagem proporciona insights profundos sobre o comportamento dos modelos selecionados e valida sua adequação para implementação em produção.

O processo inicia com uma fase robusta de carregamento de dados e identificação dos melhores modelos, implementando verificações de integridade e mecanismos de fallback:

- Carregamento dos dados pré-processados de treino e teste
- Verificação da existência dos arquivos de resultados dos modelos base e otimizados
- Identificação automática dos melhores modelos para cada variável-alvo (Heating Load e - Cooling Load), baseada no R^2.
- Verificação da existência física dos arquivos dos modelos selecionados.
- Implementação de estratégias de contingência em caso de arquivos ausentes:

Busca por modelos alternativos disponíveis no diretório de modelos:

- Criação de modelos dummy como último recurso

Esta abordagem defensiva garante que a análise prossiga mesmo em cenários onde a integridade dos arquivos intermediários possa ter sido comprometida.

Uma análise detalhada dos resíduos dos modelos é realizada através da função analisar_residuos, que:

- Gera histogramas da distribuição dos resíduos para verificar normalidade.
- Plota resíduos vs. valores previstos para verificar a homocedasticidade (variância constante).
- Cria QQ-plots para avaliar a normalidade dos resíduos.
- Calcula e exibe estatísticas descritivas dos resíduos (média, desvio padrão, RMSE).

Esta análise é crucial para verificar se os pressupostos dos modelos são atendidos e para identificar potenciais problemas como heterocedasticidade ou viés sistemático.

O código realiza uma análise abrangente da importância das variáveis, adaptando-se automaticamente aos diferentes tipos de modelos:

- Para modelos baseados em árvores: extração da importância nativa (feature_importances_).
- Para modelos lineares: uso dos coeficientes absolutos (coef_).
- Para outros modelos: cálculo da importância por permutação (permutation_importance).

Os resultados são visualizados por meio de gráficos de barras e tabulados para identificação das variáveis mais influentes na previsão de cada variável-alvo, fornecendo insights valiosos sobre os fatores determinantes da eficiência energética.

Uma avaliação visual da capacidade preditiva dos modelos é realizada através da função plot_previsoes_vs_reais, que:

- Plota os valores previstos contra os valores reais.
- Adiciona uma linha de referência representando a previsão perfeita.
- Calcula e exibe o coeficiente de determinação (R^2).

Esta visualização permite identificar rapidamente se o modelo tende a subestimar ou superestimar os valores, se existem padrões de erro para diferentes faixas de valores, e quão próximas as previsões estão dos valores reais.

O código implementa uma análise SHAP (SHapley Additive exPlanations), uma técnica avançada de interpretabilidade de modelos, que:

- Adapta-se automaticamente ao tipo de modelo utilizado (modelos baseados em árvores vs. outros).
- Calcula valores SHAP para uma amostra representativa dos dados de teste
- Gera visualizações que mostram como cada feature contribui para as previsões

Esta técnica proporciona uma explicação detalhada do funcionamento interno dos modelos, identificando não apenas quais variáveis são importantes, mas também como e em que condições elas influenciam as previsões.

Uma análise comparativa abrangente é realizada para avaliar os diferentes modelos:

- Visualização lado a lado do RMSE para Heating Load e Cooling Load.
- Comparação do R^2 entre os diferentes modelos.
- Identificação explícita dos melhores modelos com base nas métricas de desempenho.

Esta análise proporciona uma visão holística do desempenho relativo dos diferentes algoritmos, ajudando a validar a seleção dos melhores modelos.

O código tenta realizar uma análise de desempenho por categorias, verificando se existem variáveis categóricas nos dados originais. Para cada categoria identificada:

- Calcula o RMSE específico para cada valor categórico.
- Visualiza o desempenho do modelo em cada categoria.
- Identifica potenciais diferenças de desempenho entre categorias.

Esta análise pode revelar se os modelos funcionam melhor ou pior para determinados subgrupos dos dados, proporcionando insights sobre potenciais vieses ou oportunidades de melhoria.

O código conclui com a preparação dos melhores modelos para implementação e cria um arquivo JSON contendo metadados sobre os melhores modelos, incluindo:

- Nomes dos modelos selecionados.
- Referências aos arquivos de modelo e preprocessador.
- Métricas de desempenho (RMSE, MAE, R^2).
- Lista de nomes das features
- Copia os arquivos dos melhores modelos para localizações padronizadas, facilitando sua integração na aplicação Django (AT3).

Esta etapa final garante que todos os artefatos necessários estejam organizados e documentados para uma transição suave para a fase de implementação.

A análise avançada de modelos implementada neste código representa uma abordagem abrangente e rigorosa para validar e compreender os modelos preditivos de eficiência energética, garantindo não apenas seu desempenho quantitativo, mas também sua interpretabilidade e adequação para uso em cenários reais.

Os resultados obtidos foram:

- Melhor modelo para Heating Load: XGBoost_otimizado
- Melhor modelo para Cooling Load: XGBoost_otimizado
- Features utilizadas: ['Relative_Compactness', 'Surface_Area', 'Wall_Area', 'Roof_Area', 'Overall_Height', 'Glazing_Area', 'Orientation_Norte', 'Orientation_Oeste', 'Orientation_Sul', 'Glazing_Area_Distribution_Norte', 'Glazing_Area_Distribution_Oeste', 'Glazing_Area_Distribution_Sem Vidraças', 'Glazing_Area_Distribution_Sul', 'Glazing_Area_Distribution_Uniforme']

Análise de resíduos para o melhor modelo de Cooling Load e Heating Load, respectivamente, são mostrados nas Figuras 12 e 13:

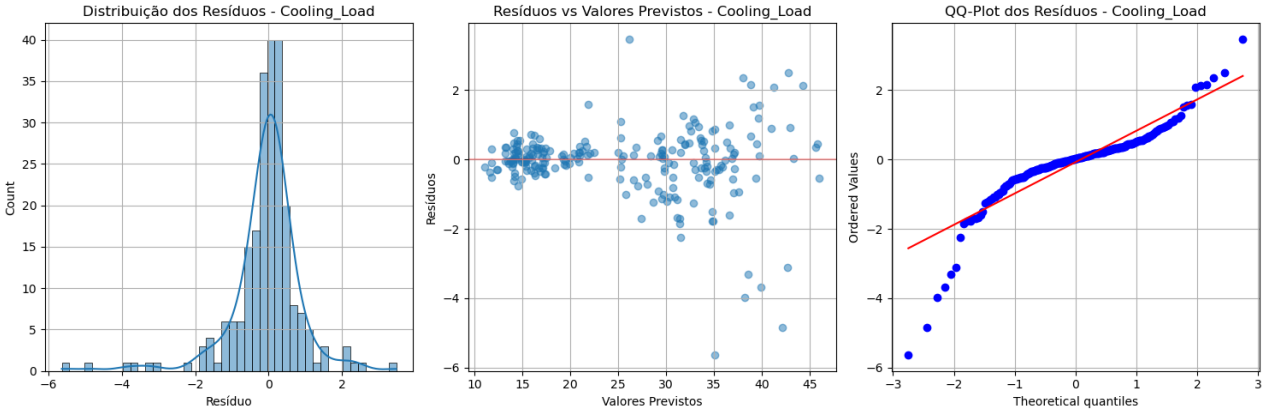

Figura 12 - Análise de distribuição de resíduos para Cooling Load.

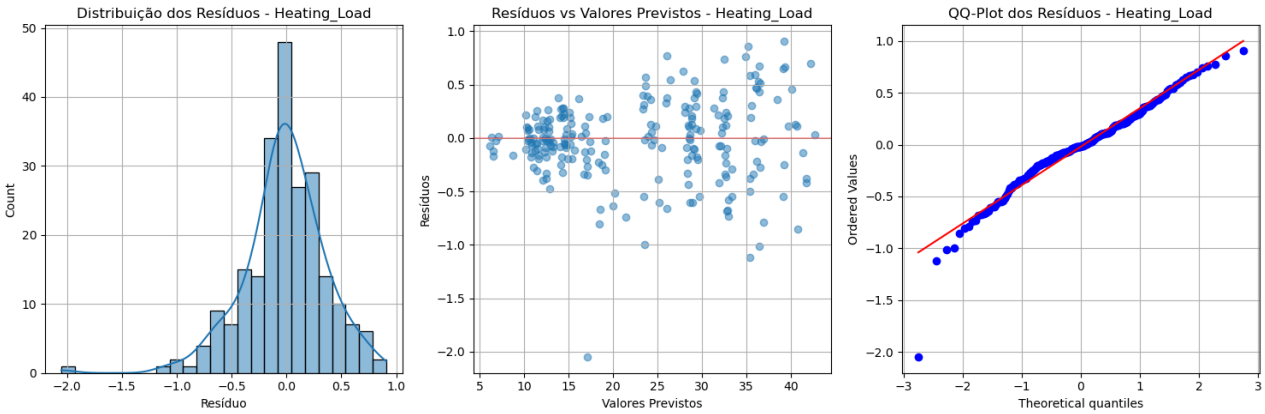

Figura 13 - Análise de distribuição de resíduos para Heating Load.

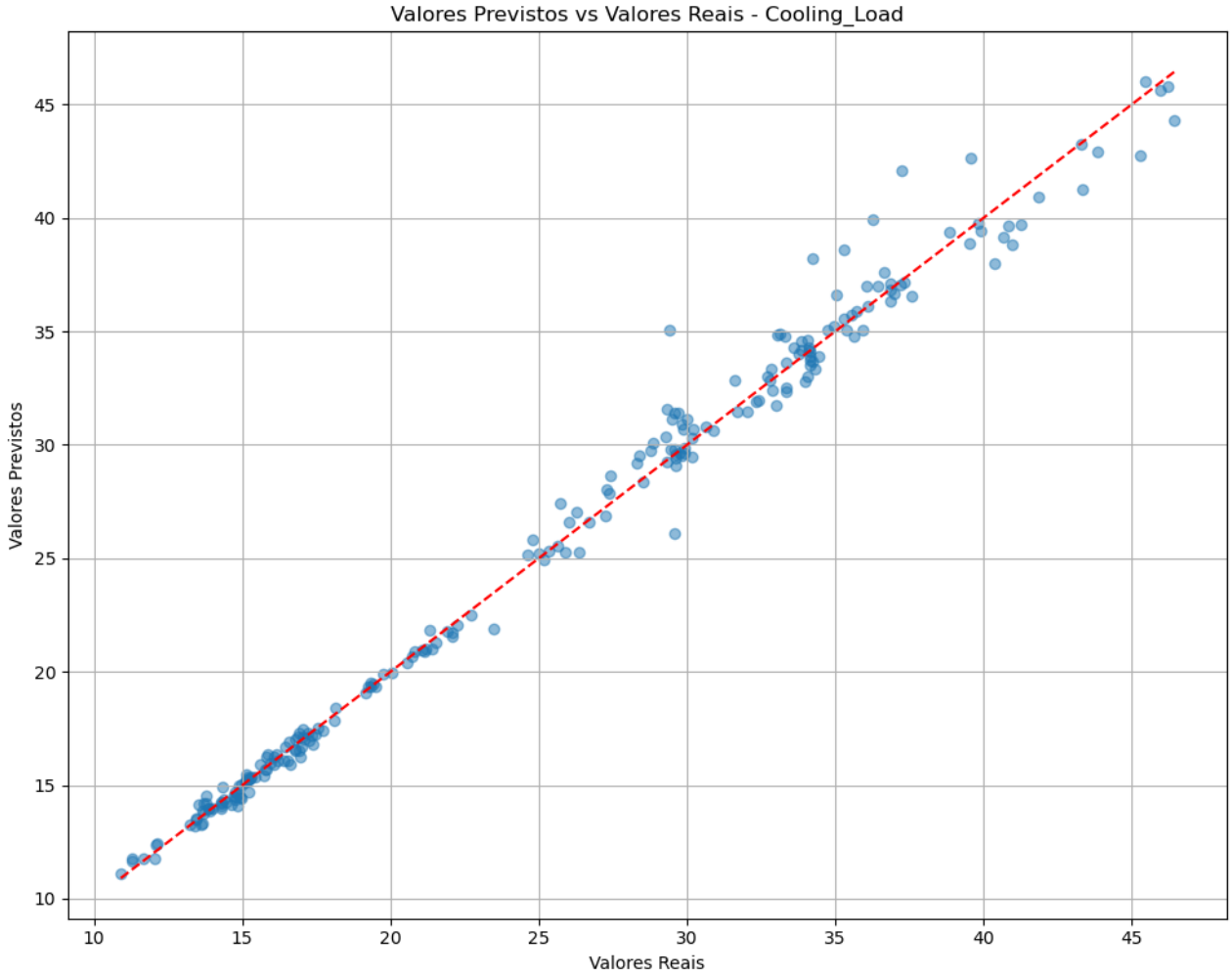

Figura 14 - Valores previstos vs valores reais para Cooling Load.

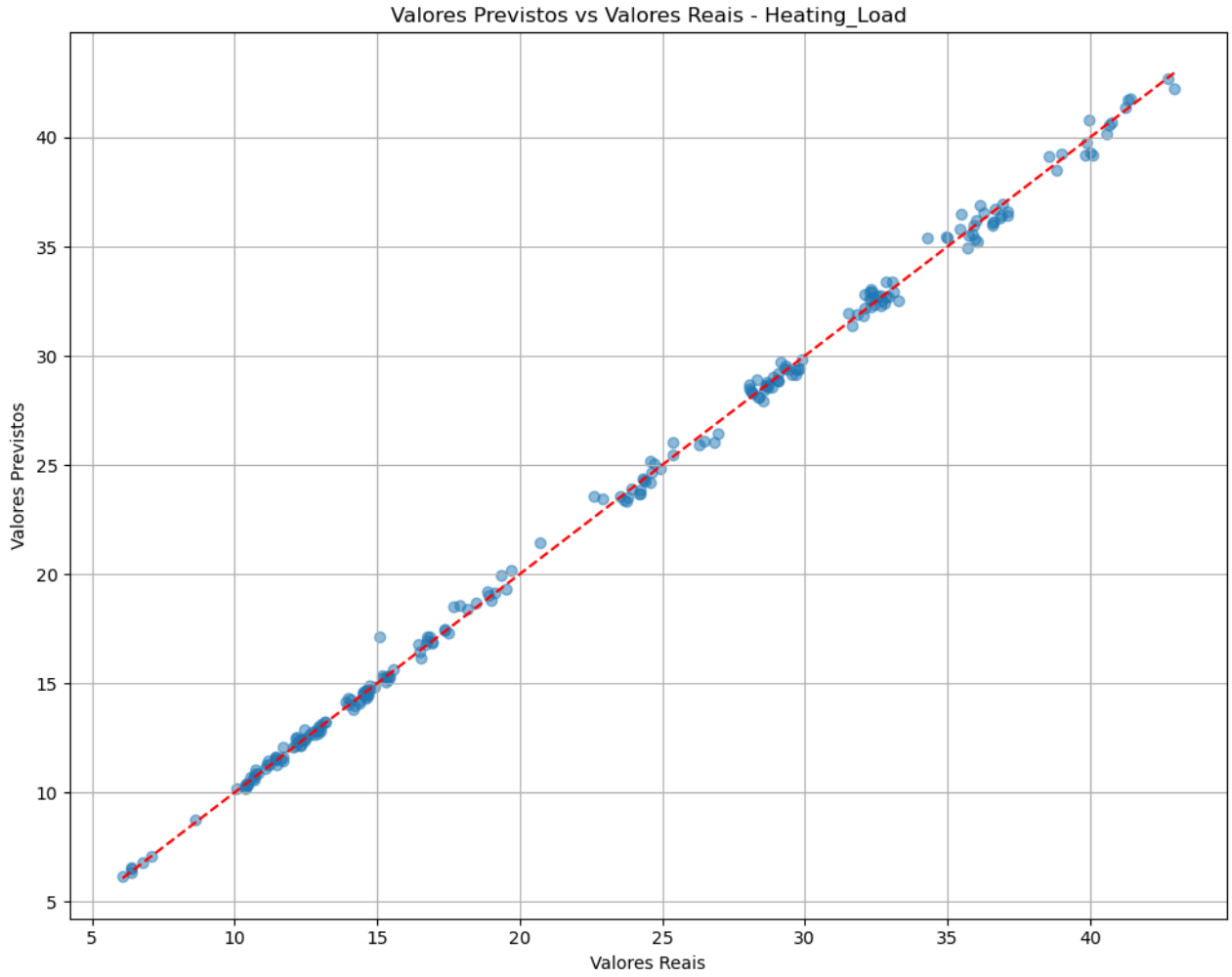

Figura 15 - Valores previstos vs valores reais para Heating Load.

Análise de resíduos para o melhor modelo de Cooling Load e Heating Load, mostram distribuição normal e os ajustes dos gráficos das Figuras 14 e 15, mostram bons ajustes lineares entre os valores previstos pelo modelo XGBosst otimizado e os valores tomados como valores reais.

4 Aplicação desenvolvida

Após todo o processo de análise de dados e construção de modelos que acabamos de ver, foi desenvolvido um sistema web prático e intuitivo para ajudar arquitetos, engenheiros e construtores a estimar a eficiência energética de edifícios ainda na fase de projeto. Esse sistema funciona de forma simples e acessível. Imagine que você está projetando um novo edifício e quer saber quanto ele vai consumir de energia para aquecimento no inverno e resfriamento no verão antes mesmo de construí-lo. Este sistema permite justamente isso!

Trata-se de uma plataforma online onde você insere as características básicas do edifício que está projetando, como tamanho, formato, altura, orientação e áreas envidraçadas (janelas), e o sistema calcula automaticamente a carga de aquecimento e resfriamento estimada. Como Funciona na Prática?

- Acesso ao Sistema: Você acessa um site através do seu navegador, onde faz login com suas credenciais.
- Inserção de Dados: Na tela principal, você encontra um formulário amigável onde pode
- Inserir as características do edifício:
- Compacidade relativa (o quão compacto é o edifício)
- Área da superfície externa total
- Área das paredes
- Área do telhado
- Altura total
- Orientação (norte, sul, leste ou oeste)
- Área de janelas (vidros)
- Como as janelas estão distribuídas pelo edifício
- Resultado da Previsão: Após preencher os dados e clicar em "Calcular", o sistema apresenta duas informações principais:
- Carga de Aquecimento: quanto de energia será necessário para manter o ambiente aquecido.

- Carga de Resfriamento: quanto de energia será necessário para manter o ambiente refrigerado.

- Histórico de Projetos: O sistema salva suas previsões, permitindo que você compare diferentes designs ou volte a consultar projetos anteriores.

Embora a interface seja simples para o usuário, o sistema utiliza modelos matemáticos sofisticados (como o XGBoost) que foram treinados usando centenas de exemplos de edifícios reais.

Estes modelos aprenderam a reconhecer padrões complexos que nos humanos seria difícil identificar, como:

- Como a altura do edifício afeta a eficiência energética.

- O impacto da orientação solar no consumo de energia.

- Como diferentes distribuições de janelas podem influenciar no aquecimento e resfriamento.

Economia de tempo e recursos:

- Você não precisa criar simulações complexas em softwares especializados como o Ecotect, que exigem conhecimento técnico avançado e muito tempo.

- Tomada de decisão informada: Antes mesmo de finalizar o projeto, você pode comparar diferentes designs e escolher o mais eficiente energeticamente.

- Sustentabilidade: Ao projetar edifícios mais eficientes, você contribui para a redução do consumo de energia e emissões de carbono.

- Economia financeira: Prever o consumo energético permite estimar custos operacionais futuros e fazer ajustes para reduzi-los.

O sistema pode ser acessado de duas formas:

- Acesso Online: Para quem não quer se preocupar com instalações, basta acessar o sistema pela internet, criar sua conta e começar a usar.

- Instalação Local: Para organizações maiores ou quem precisa de uma solução interna, é possível instalar o sistema nos próprios servidores usando Docker, uma tecnologia que empacota todo o sistema em containers fáceis de instalar.

No final das contas, este sistema transforma conhecimentos complexos de eficiência energética e algoritmos avançados de inteligência artificial em uma ferramenta simples e acessível, permitindo que profissionais da construção civil projetem edifícios mais sustentáveis e econômicos com facilidade.

5 Conclusão

Durante o desenvolvimento deste projeto de modelagem preditiva para eficiência energética de edifícios, enfrentamos diversos desafios significativos, como:

- Complexidade das Relações Não-lineares: Um dos maiores desafios foi capturar adequadamente as relações não-lineares complexas entre as características arquitetônicas dos edifícios e suas cargas térmicas. Os modelos lineares simples provaram ser insuficientes para representar essas interações, exigindo abordagens mais sofisticadas.
- Balanceamento entre Precisão e Interpretabilidade: Encontrar o equilíbrio ideal entre modelos de alta precisão (como XGBoost e Random Forest) e a necessidade de interpretabilidade para profissionais da construção civil foi um desafio constante. Técnicas avançadas como SHAP foram fundamentais para tornar "caixas pretas" mais transparentes.
- Otimização de Hiperparâmetros: O ajuste fino dos modelos exigiu extensas buscas em grades de hiperparâmetros, demandando recursos computacionais significativos e implementação de estratégias eficientes para encontrar as configurações ideais.
- Representatividade dos Dados: Embora o conjunto de dados UCI Energy Efficiency seja valioso, ele representa um subconjunto limitado de possíveis configurações de edifícios. Garantir que os modelos possam generalizar para designs arquitetônicos diversos permanece um desafio.
- Integração entre Análise e Aplicação: Transformar modelos complexos treinados em ambientes de análise em uma aplicação web prática e acessível exigiu superar diversos desafios técnicos de integração e otimização.

Perspectivas e Melhorias Futuras:

- Apesar dos resultados promissores, existem diversas oportunidades de aprimoramento para o sistema:
- Expansão do Conjunto de Dados: Incorporar dados de edifícios reais de diferentes regiões climáticas e estilos arquitetônicos poderia melhorar significativamente a robustez

e aplicabilidade dos modelos. A coleta contínua de dados de edifícios já construídos criaria um ciclo de feedback valioso.

- Incorporação de Variáveis Adicionais: Expandir o modelo para incluir características como materiais de construção, sistemas de isolamento, tipo de vidro e variáveis climáticas locais poderia aumentar a precisão e utilidade das previsões.

- Aprendizado Federado: Implementar técnicas de aprendizado federado permitiria que diferentes organizações contribuíssem com dados de seus projetos sem comprometer a privacidade, melhorando continuamente os modelos.

- Modelos Adaptativos Regionais: Desenvolver modelos específicos para diferentes zonas climáticas e contextos regionais, reconhecendo que as características que influenciam a eficiência energética podem variar significativamente entre regiões.

Interfaces de Visualização Avançadas: Incorporar visualizações 3D e simulações interativas permitiria que os usuários visualizassem o impacto de suas escolhas de design na eficiência energética em tempo real.

- Integração com BIM: Conectar o sistema a plataformas de Building Information Modeling (BIM) facilitaria a transferência automática de especificações de edifícios para o modelo preditivo, eliminando a entrada manual de dados.

- Recomendações Automatizadas: Expandir o sistema para não apenas prever consumo energético, mas também sugerir modificações específicas no design para melhorar a eficiência com base em algoritmos de otimização.

- Aplicativo Móvel: Desenvolver uma versão móvel do sistema permitiria que profissionais fizessem avaliações rápidas em campo ou apresentassem resultados em reuniões com clientes.

- Certificação Automática de Eficiência: Integrar os requisitos de certificações de sustentabilidade (como LEED ou BREEAM) para indicar automaticamente se um projeto atende aos critérios necessários.

Este projeto demonstra como a intersecção entre ciência de dados e arquitetura sustentável pode criar ferramentas poderosas para enfrentar desafios ambientais contemporâneos. À medida que avançamos, a incorporação contínua de novas tecnologias e dados promete transformar ainda mais profundamente como projetamos e construímos edificações energeticamente eficientes, contribuindo para um futuro mais sustentável.

ANEXOS

Roteiro de Desenvolvimento do Projeto de Análise Preditiva e todos os códigos estão disponíveis em um repositório remoto do GitHub, dado por https://github.com/tdggaltra/CaseIA_RP_16497_16499_16500_SENAI_Londrina_PR.git.

Lá podem ser encontrados todas as análises detalhadas, os gráficos e os códigos e as descrições de README.md tanto para o projeto geral de análise de dados dos scripts notebook quanto para o sistema web desenvolvido em Django