

***INTELIGÊNCIA ARTIFICIAL***

**UA02 / LABORATÓRIO # 3**

***Dataset usado:*** *CarPrice\_Assignment.csv*

# Importação de Bibliotecas

import pandas as pd

import seaborn as sns

# Leitura de Dataset - Car Price Assingment, do Kaggle

df = pd.read\_csv("CarPrice\_Assignment.csv",sep=",")

# Data Understanding - Compreensão dos Dados

df.head()

df.info()

df.describe()

df.describe(include='O')

list(df.carbody.unique())

print(df['carbody'].value\_counts())

print(df['fuelsystem'].value\_counts())

# Data Preparation - Preparação dos Dados (Limpeza, por exemplo)

df.isnull().sum() # para totalizar os valores nulos em cada feature (variável)

df[df.duplicated(keep='first')] # para identificar linhas duplicadas

# df.drop\_duplicates(keep='first',inplace=True) - remove linhas duplicadas, se houver

# Data Visualization

# Distribuição da Variável Price

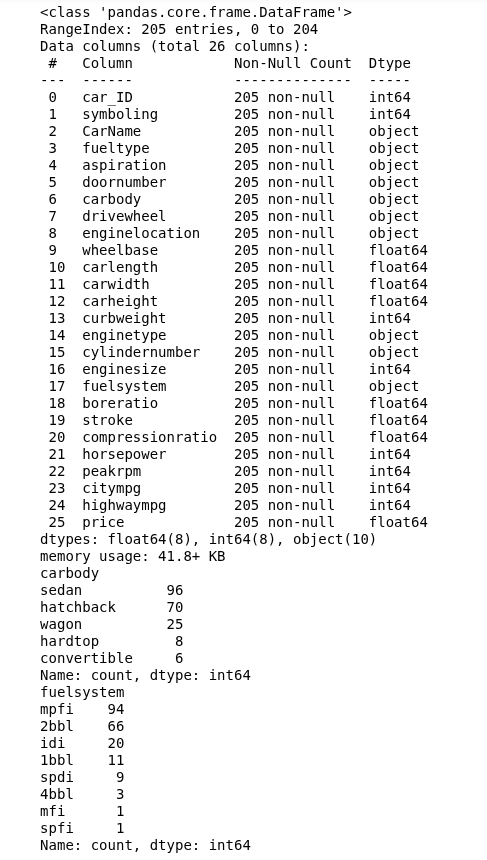
fig = sns.histplot(data=df, x = 'price', kde=True, color='b')

fig.set\_title('Distribuição da Variável Price',size=18)

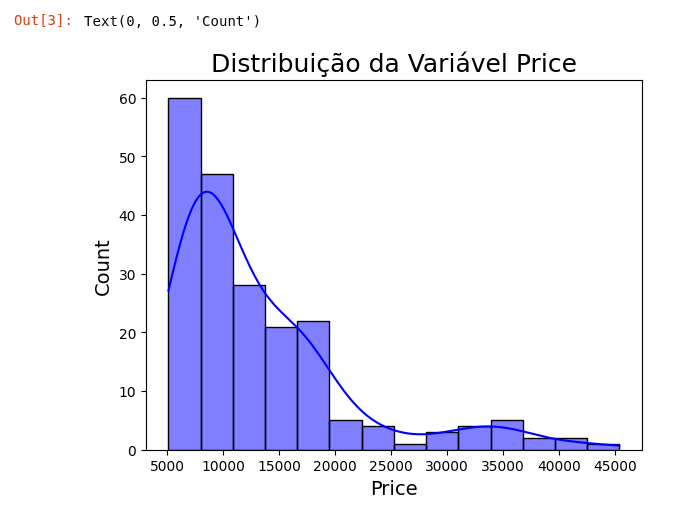
fig.set\_xlabel('Price', size=14)

fig.set\_ylabel('Count', size=14)

***Saída:***



***Saída:***

**

*1.) Importação de Bibliotecas*

*import pandas as pd*

*import seaborn as sns*

*- `pandas`: Biblioteca principal para manipulação e análise de dados (cria DataFrames)*

*- `seaborn`: Biblioteca para visualização de dados estatísticos (gráficos mais atraentes que matplotlib)*

*2.) Leitura dos Dados*

*df = pd.read\_csv("CarPrice\_Assignment.csv",sep=",")*

*- Lê o arquivo CSV contendo os dados de preços de carros*

*- Cria um DataFrame (`df`) que é uma tabela estruturada com linhas e colunas*

*- `sep=","` indica que o separador de colunas é a vírgula (padrão em CSVs)*

*3.) Análise Exploratória Básica*

***Primeiras linhas do dataset***

*df.head()*

*- Mostra as primeiras 5 linhas do DataFrame para inspeção visual*

***Informações gerais***

*df.info()*

*- Saída mostrada no seu resultado:*

*- Número de linhas (205 entradas)*

*- Nomes e tipos de todas as 26 colunas*

*- Contagem de valores não-nulos*

*- Uso de memória*

***Estatísticas descritivas***

*df.describe()*

*- Mostra estatísticas para colunas numéricas (média, desvio padrão, mín/máx, quartis)*

*df.describe(include='O')*

*- Mostra estatísticas para colunas categóricas (object)*

*- Contagem de valores únicos, valor mais frequente, sua frequência*

***Análise de valores únicos***

*list(df.carbody.unique())*

*- Lista todos os valores únicos na coluna 'carbody' (tipos de carroceria)*

***Contagem de valores***

*print(df['carbody'].value\_counts())*

*print(df['fuelsystem'].value\_counts())*

*- Mostra quantas vezes cada valor aparece nessas colunas categóricas*

*- Saída mostra que 'sedan' é o tipo mais comum (96) e 'mpfi' o sistema de combustível mais comum (94)*

*4.) Preparação dos Dados*

***Verificação de valores nulos***

*df.isnull().sum()*

*- Verifica se há valores faltantes (NaN) em cada coluna*

*- No seu caso, todas as colunas têm 205 não-nulos, indicando nenhum valor faltante*

***Verificação de duplicatas***

*df[df.duplicated(keep='first')]*

*- Procura por linhas duplicadas no DataFrame*

*- `keep='first'` marca como duplicadas todas exceto a primeira ocorrência*

*- A linha comentada abaixo (`# df.drop\_duplicates...`) removeria as duplicatas se existissem*

*5.) Visualização de Dados - Histograma de preços*

*fig = sns.histplot(data=df, x = 'price', kde=True, color='b')*

*fig.set\_title('Distribuição da Variável Price',size=18)*

*fig.set\_xlabel('Price', size=14)*

*fig.set\_ylabel('Count', size=14)*

*- Cria um histograma da coluna 'price' (preço dos carros)*

*- `kde=True` adiciona uma linha de densidade de kernel (suavização da distribuição)*

*- Configura título e rótulos dos eixos com tamanhos de fonte específicos*

***Objetivo Geral do Código***

*Este é um típico fluxo de trabalho de análise exploratória de dados (EDA) para:*

*- Entender a estrutura e qualidade dos dados*

*- Identificar padrões e distribuições*

*- Preparar os dados para modelagem futura (possivelmente para prever preços de carros)*

*A saída mostra que:*

*- O dataset tem 205 carros com 26 características cada*

*- Não há valores faltantes ou duplicatas*

*- A maioria dos carros são sedans com sistema de combustível mpfi*

*- O histograma mostra a distribuição dos preços (que parece estar enviesada para valores mais baixos)*

***Continuação do Código:***

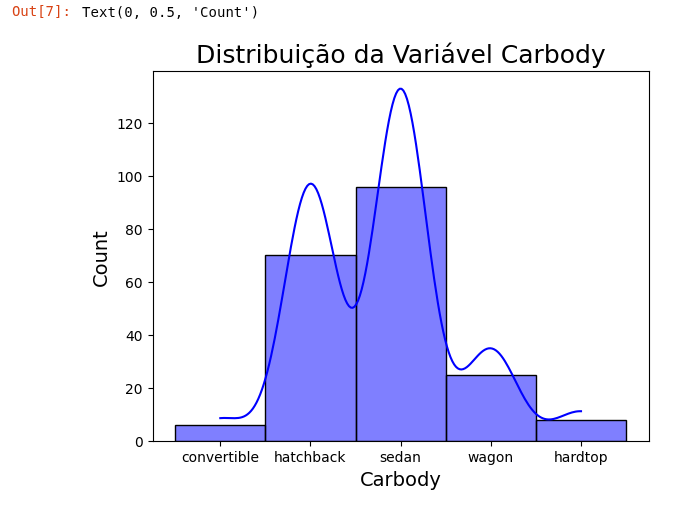
*# Distribuição da Variável Carbody*

*fig = sns.histplot(data=df, x = 'carbody', kde=True, color='b')*

*fig.set\_title('Distribuição da Variável Carbody',size=18)*

*fig.set\_xlabel('Carbody', size=14)*

*fig.set\_ylabel('Count', size=14)*

**

***Explicação do Código de Visualização da Distribuição de Carbody***

*1.) Criação do Gráfico*

*fig = sns.histplot(data=df, x='carbody', kde=True, color='b')*

*- `sns.histplot()`: Função do Seaborn para criar histogramas*

*- `data=df`: Especifica o DataFrame que contém os dados (`df`)*

*- `x='carbody'`: Define que usaremos a coluna 'carbody' no eixo X*

*- `kde=True`: Adiciona uma linha de densidade (Kernel Density Estimation) sobre as barras*

*- `color='b'`: Define a cor das barras como azul ('b' = blue)*

*2.) Personalização do Título*

*fig.set\_title('Distribuição da Variável Carbody', size=18)*

*- `set\_title()`: Método para adicionar título ao gráfico*

*- `'Distribuição da Variável Carbody'`: Texto do título*

*- `size=18`: Tamanho da fonte do título (18 pontos)*

*3.) Rótulo do Eixo X*

*fig.set\_xlabel('Carbody', size=14)*

*- `set\_xlabel()`: Define o rótulo do eixo X*

*- `'Carbody'`: Nome que aparecerá abaixo do eixo X*

*- `size=14`: Tamanho da fonte do rótulo*

*4.) Rótulo do Eixo Y*

*fig.set\_ylabel('Count', size=14)*

*- `set\_ylabel()`: Define o rótulo do eixo Y*

*- `'Count'`: Indica que o eixo Y mostra contagens de ocorrências*

*- `size=14`: Tamanho da fonte do rótulo*

*O que este gráfico mostra?*

*Este gráfico é um histograma com curva de densidade que visualiza:*

*- Quantos carros existem de cada tipo de carroceria no dataset*

*- A distribuição proporcional entre as categorias de carroceria*

*Interpretação Esperada*

*Baseado no resultado anterior que você mostrou (`df['carbody'].value\_counts()`), esperamos ver:*

*\* Barras mostrando a contagem absoluta de cada tipo:*

*- Sedan (mais alto, ~96)*

*- Hatchback (~70)*

*- Wagon (~25)*

*- Hardtop (~8)*

*- Convertible (~6)*

*\* Curva KDE (linha suave): Mostra a densidade de probabilidade da distribuição, que neste caso categórico acompanhará o formato das barras.*

*Por que usar este gráfico?*

*1.) Análise de distribuição: Entender quais tipos de carroceria são mais comuns*

*2.) Identificação de desbalanceamento: Ver se algumas categorias têm poucas amostras*

*3.) Tomada de decisão: Saber se precisamos balancear os dados para modelagem*

*Observação Importante*

*Para variáveis categóricas como 'carbody', um gráfico de barras (countplot) seria mais adequado que um histograma. Uma alternativa seria:*

*sns.countplot(data=df, x='carbody', color='b')*

*Mas o histograma funciona também, especialmente com `kde=True` para enfatizar a distribuição proporcional.*

***Continuação do Código:***

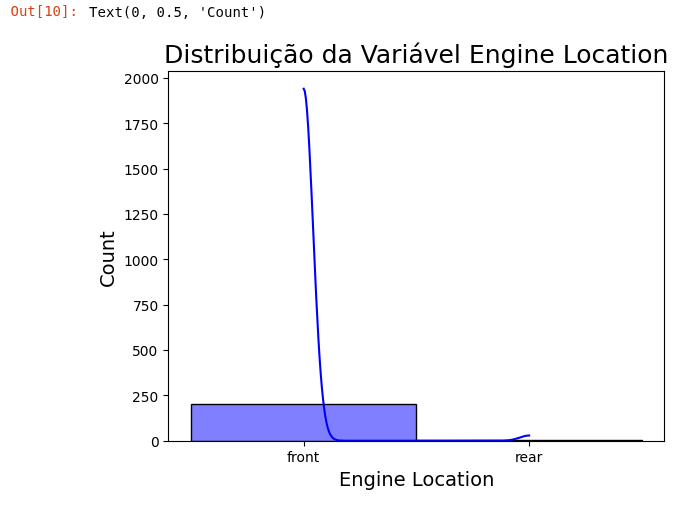
*# Distribuição da Variável EngineLocation*

*fig = sns.histplot(data=df, x = 'enginelocation', kde=True, color='b')*

*fig.set\_title('Distribuição da Variável Engine Location',size=18)*

*fig.set\_xlabel('Engine Location', size=14)*

*fig.set\_ylabel('Count', size=14)*

**

***Explicação do Código de Distribuição de Engine Location***

*1.) Criação do Gráfico de Distribuição*

*fig = sns.histplot(data=df, x='enginelocation', kde=True, color='b')*

*- `sns.histplot()`: Função do Seaborn para criar histogramas*

*- `data=df`: Especifica que estamos usando o DataFrame `df` como fonte de dados*

*- `x='enginelocation'`: Seleciona a coluna que mostra a localização do motor para análise*

*- `kde=True`: Adiciona uma linha de densidade (Kernel Density Estimation) sobre o histograma*

*- `color='b'`: Define a cor das barras como azul ('b' é a abreviação para blue)*

*2.) Configuração do Título*

*fig.set\_title('Distribuição da Variável Engine Location', size=18)*

*- `set\_title()`: Método para adicionar título ao gráfico*

*- Texto: 'Distribuição da Variável Engine Location' (em português)*

*- `size=18`: Define o tamanho da fonte do título como 18 pontos*

*3. Rótulo do Eixo X*

*fig.set\_xlabel('Engine Location', size=14)*

*- `set\_xlabel()`: Define o rótulo do eixo horizontal (X)*

*- Texto: 'Engine Location' (localização do motor)*

*- `size=14`: Tamanho da fonte do rótulo*

*4.) Rótulo do Eixo Y*

*fig.set\_ylabel('Count', size=14)*

*- `set\_ylabel()`: Define o rótulo do eixo vertical (Y)*

*- Texto: 'Count' (contagem, em inglês)*

*- `size=14`: Tamanho da fonte do rótulo*

*O que este gráfico específico revela?*

*Este gráfico mostra:*

*- A distribuição dos carros de acordo com a posição do motor*

*- Quantos carros têm motor dianteiro (front) vs. traseiro (rear)*

*Interpretação Esperada (com base em dados típicos):*

*\* Maioria esmagadora de motores dianteiros (provavelmente uma barra muito alta em "front")*

*\* Pouquíssimos carros com motor traseiro (barra pequena em "rear")*

*\* A curva KDE seguirá o mesmo padrão, mostrando pico acentuado em "front"*

*Observações Importantes:*

*1.) Para variáveis categóricas como 'enginelocation', um gráfico de barras simples seria mais adequado:*

*sns.countplot(data=df, x='enginelocation')*

*2.) O KDE (kernel density estimate) tem utilidade limitada em variáveis categóricas com poucas categorias*

*3.) Este gráfico é importante para:*

*- Identificar desbalanceamento extremo nos dados*

*- Verificar se há amostras suficientes de cada tipo para análises futuras*

*- Detectar possíveis erros nos dados (ex: valores inesperados)*

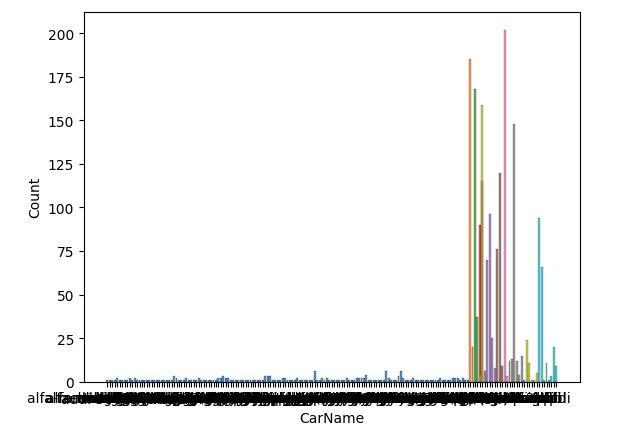
***Continuação do Código:***

*# Contagem de Ocorrências de Valores Categóricos - CarName*

*df\_categorical = df.select\_dtypes(include = 'object').columns*

*for i in df\_categorical:*

*fig = sns.histplot(data=df, x = i, shrink=.8)*

**

***Explicação Detalhada do Código de Análise de Variáveis Categóricas***

*1.) Seleção das Colunas Categóricas*

*df\_categorical = df.select\_dtypes(include='object').columns*

*- `select\_dtypes()`: Método do pandas para filtrar colunas por tipo de dado*

*- `include='object'`: Seleciona apenas colunas do tipo 'object' (que geralmente armazenam textos/categorias)*

*- `.columns`: Acessa apenas os nomes das colunas (não os dados em si)*

*- Resultado: `df\_categorical` será uma lista com todos os nomes de colunas categóricas do DataFrame*

*2.) Loop pelas Variáveis Categóricas*

*for i in df\_categorical:*

*..- Itera sobre cada nome de coluna categórica armazenado em `df\_categorical`*

*..- Em cada iteração, `i` assumirá o nome de uma coluna diferente*

*3. Criação dos Histogramas para Cada Variável*

*fig = sns.histplot(data=df, x=i, shrink=.8)*

*- \*\*`sns.histplot()`\*\*: Cria um histograma usando Seaborn*

*- \*\*`data=df`\*\*: Especifica o DataFrame de origem*

*- \*\*`x=i`\*\*: Usa a coluna atual do loop (i) no eixo x*

*- \*\*`shrink=.8`\*\*: Controla a largura das barras (80% do espaço disponível)*

*- Isso evita que as barras fiquem coladas umas nas outras*

*- Melhora a visualização quando há muitas categorias*

*O que Este Código Faz?*

*\* Identifica automaticamente todas as colunas categóricas do DataFrame*

*\* Gera um histograma separado para cada uma dessas colunas*

*\* Mostra a distribuição de valores em cada variável categórica*

*Exemplo de Variáveis Analisadas (baseado no dataset anterior):*

*- CarName (nome do carro)*

*- fueltype (tipo de combustível)*

*- aspiration (tipo de aspiração)*

*- doornumber (número de portas)*

*- carbody (tipo de carroceria)*

*- drivewheel (tipo de tração)*

*- enginelocation (posição do motor)*

*- enginetype (tipo de motor)*

*- cylindernumber (número de cilindros)*

*- fuelsystem (sistema de combustível)*

*Observações Importantes:*

*1.) Para variáveis com muitas categorias (como CarName), o gráfico pode ficar poluído*

*- Solução possível: `plt.xticks(rotation=90)` para girar os rótulos*

*2.) Alternativa mais eficiente para variáveis categóricas:*

*for col in df\_categorical:*

*plt.figure(figsize=(10,4))*

*sns.countplot(data=df, x=col)*

*plt.title(f'Distribuição de {col}')*

*plt.xticks(rotation=45)*

*plt.show()*

*3.) O parâmetro `shrink` é particularmente útil quando:*

*- Há poucas categorias (para evitar barras muito largas)*

*- Quando se quer comparar visualmente a proporção entre categorias*

*Este código é uma forma eficiente de explorar rapidamente todas as variáveis categóricas do dataset, identificando:*

*- Distribuição de frequências*

*- Possíveis desbalanceamentos*

*- Valores atípicos ou categorias com poucas observações*

*- Necessidade de agrupamento de categorias raras*

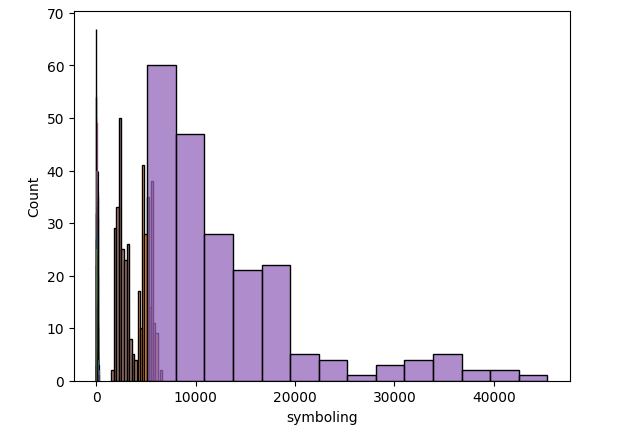
***Continuação do Código:***

# Contagem de Ocorrências de Valores Numéricos

df\_numerical = df.select\_dtypes(exclude = 'object').columns

for i in df\_numerical:

fig = sns.histplot(data=df, x = i)



**Explicação Detalhada do Código de Análise de Variáveis Numéricas**

1.) Seleção das Colunas Numéricas

df\_numerical = df.select\_dtypes(exclude='object').columns

- `select\_dtypes()`: Método do pandas para filtrar colunas por tipo de dado

- `exclude='object'`: Exclui colunas do tipo 'object' (texto/categorias), mantendo apenas numéricas

- `.columns`: Extrai apenas os nomes das colunas (não os dados em si)

- Resultado: `df\_numerical` contém uma lista com todos os nomes de colunas numéricas do DataFrame

2.) Loop pelas Variáveis Numéricas

for i in df\_numerical:

- Itera sobre cada nome de coluna numérica armazenado em `df\_numerical`

- Em cada iteração, `i` assumirá o nome de uma coluna numérica diferente

3.) Criação dos Histogramas para Cada Variável Numérica

fig = sns.histplot(data=df, x=i)

- `sns.histplot()`: Cria um histograma usando Seaborn

- `data=df`: Especifica o DataFrame de origem dos dados

- `x=i`: Usa a coluna numérica atual do loop (i) no eixo x

- Sem parâmetros adicionais\*\*: Usa configurações padrão do Seaborn

O que Este Código Faz?

\* Identifica automaticamente todas as colunas numéricas do DataFrame

\* Gera um histograma separado para cada variável numérica

\* Mostra a distribuição dos valores em cada variável

Variáveis Numéricas Analisadas (baseado no dataset anterior):

- car\_ID (identificador)

- symboling (classificação de risco)

- wheelbase (distância entre eixos)

- carlength (comprimento)

- carwidth (largura)

- carheight (altura)

- curbweight (peso)

- enginesize (tamanho do motor)

- boreratio (relação de furo)

- stroke (curso)

- compressionratio (taxa de compressão)

- horsepower (potência)

- peakrpm (RPM máximo)

- citympg (consumo urbano)

- highwaympg (consumo na estrada)

- price (preço)

Visualização Gerada:

Para cada variável numérica, o código cria:

- Um histograma mostrando a frequência dos valores

- Eixo X: valores da variável (agrupados em bins automáticos)

- Eixo Y: contagem de ocorrências em cada intervalo

Observações Importantes:

1.) Para melhorar a visualização, poderíamos adicionar:

plt.figure(figsize=(8,4)) # Controla o tamanho da figura

plt.title(f'Distribuição de {i}') # Adiciona título dinâmico

2.) Parâmetros úteis para `histplot()`:

sns.histplot(data=df, x=i, bins=20, kde=True, color='blue')

- `bins`: Controla o número de intervalos

- `kde`: Adiciona linha de densidade

- `color`: Define a cor das barras

3.) Alternativa para distribuições complexas:

sns.displot(data=df, x=i, kind='kde') # Mostra apenas a curva de densidade

Este código é particularmente útil para:

- Identificar distribuições (normal, skewness, bimodal)

- Detectar outliers (valores extremos)

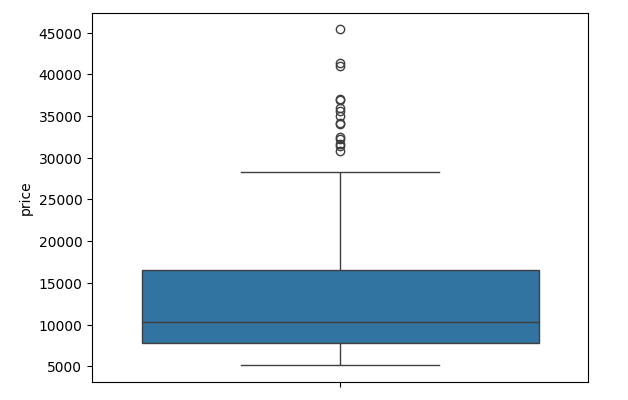
- Verificar necessidade de transformações (log, normalização)

- Entender a dispersão dos dados numéricos

***Continuação do Código:***

# Boxplot o atributo price

fig = sns.boxplot(data=df, y = 'price')



**Explicação do Boxplot do Preço (Price)**

fig = sns.boxplot(data=df, y='price')

`sns.boxplot()`:

- Função do Seaborn para criar gráficos de caixa (boxplots)

- Ideal para visualizar a distribuição de dados numéricos

`data=df`:

- Especifica o DataFrame que contém os dados (`df`)

`y='price'`:

- Define que queremos analisar a coluna 'price' no eixo Y

- Usamos `y` em vez de `x` para que o boxplot fique na vertical

**Interpretação do Boxplot Gerado:**

Um boxplot mostra a distribuição dos dados através de 5 números principais:

Linha Inferior (Bigode Inferior):

- Representa o mínimo (Q1 - 1.5\*IQR)

- Mostra o menor valor que não é considerado outlier

Borda Inferior da Caixa (Q1 - Primeiro Quartil):

- 25% dos dados estão abaixo deste valor

- No seu caso, 25% dos carros têm preço abaixo de Q1

Linha do Meio (Q2 - Mediana):

- Divide os dados em duas partes iguais

- 50% dos carros têm preço abaixo deste valor

Borda Superior da Caixa (Q3 - Terceiro Quartil):

- 75% dos dados estão abaixo deste valor

- Apenas 25% dos carros têm preço acima de Q3

Linha Superior (Bigode Superior):

- Representa o máximo (Q3 + 1.5\*IQR)

- Mostra o maior valor que não é considerado outlier

Pontos Individuais:

- Qualquer ponto além dos bigodes é considerado outlier (valor extremo)

- Mostra carros com preços excepcionalmente altos

Como Interpretar no Contexto de Preços de Carros:

Distribuição Geral:

- A posição vertical da caixa mostra a faixa de preços mais comum

- A altura da caixa mostra a variação entre Q1 e Q3 (dispersão dos preços)

Mediana (Linha do Meio):

- Se a mediana está mais próxima de Q1, significa que há mais carros com preços baixos

- Se está mais próxima de Q3, significa mais carros com preços altos

Outliers:

- Carros com preços muito acima do normal aparecem como pontos isolados

- Podem representar veículos de luxo ou dados incorretos

4. Assimetria:

- Se os bigodes têm comprimentos diferentes, indica distribuição assimétrica

- No caso de preços, é comum ter assimetria positiva (mais valores baixos)

Exemplo Prático de Interpretação:

Se o boxplot mostrar:

- Bigode inferior: $5,000

- Q1: $10,000

- Mediana: $15,000

- Q3: $20,000

- Bigode superior: $30,000

- Alguns pontos acima de $50,000

Significa que:

- A maioria dos carros (50%) custa entre $10,000 e $20,000

- O preço típico (mediana) é $15,000

- Carros abaixo de $5,000 ou acima de $30,000 são incomuns

- Os pontos acima de $50,000 são outliers (possivelmente carros de luxo)

Dica para Melhor Visualização:

Você pode melhorar o gráfico adicionando:

plt.figure(figsize=(8,6)) # Tamanho da figura

sns.boxplot(data=df, y='price', color='lightblue') # Cor personalizada

plt.title('Distribuição de Preços de Carros', fontsize=14)

plt.ylabel('Preço (USD)', fontsize=12)

plt.show()

Isso tornará a visualização mais clara e profissional.

***Continuação do Código:***

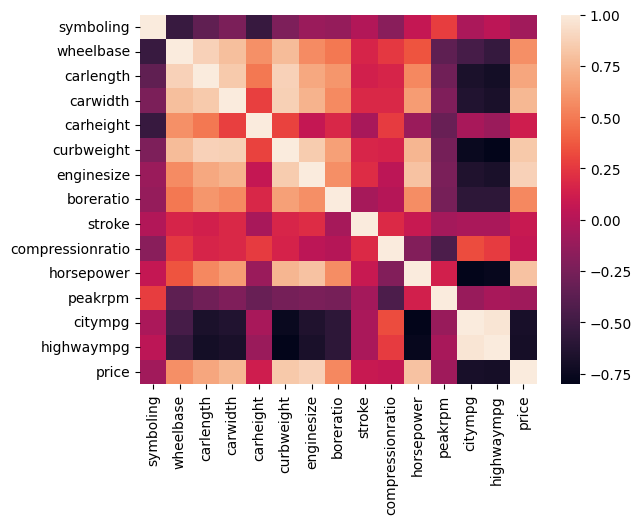
# Gráfico de Calor (heatmap) de todos os atributos

# Correlação entre atributos - quanto mais clara a cor (mais perto do valor 1.0), mais alta a correlação

# Seleciona apenas colunas numéricas

df\_numeric = df.select\_dtypes(include=['int64', 'float64'])

sns.heatmap(df\_numeric.corr())



Linha 1: Seleção de Colunas Numéricas

df\_numeric = df.select\_dtypes(include=['int64', 'float64'])

- `select\_dtypes()`: Método do pandas que filtra colunas por tipo de dado

- `include=['int64', 'float64']`: Especifica que queremos apenas colunas numéricas

- `int64`: números inteiros

- `float64`: números decimais

- Resultado: `df\_numeric` é um novo DataFrame contendo apenas as colunas numéricas do DataFrame original

Linha 2: Criação do Heatmap de Correlação

sns.heatmap(df\_numeric

- `df\_numeric.corr()`: Calcula a matriz de correlação entre todas as colunas numéricas

- Retorna uma matriz quadrada onde cada célula mostra a correlação entre duas variáveis

- Valores variam de -1 (correlação negativa perfeita) a +1 (correlação positiva perfeita)

- `sns.heatmap()`: Função do Seaborn que cria uma representação visual da matriz

- Por padrão, usa uma escala de cores que vai do azul (valores negativos) ao vermelho (valores positivos)

- Cores mais intensas indicam correlações mais fortes4444

Visualização Resultante

O gráfico gerado terá:

- Eixo X e Y: Nomes das colunas numéricas

- Quadrados coloridos: Intensidade e direção da correlação entre cada par de variáveis

- Barra de cores (legend): Mostra a escala de valores de correlação

Versão Aprimorada do Código

Para um heatmap mais informativo, recomendo:

plt.figure(figsize=(12, 10)) # Tamanho da figura

# Cálculo da correlação

corr\_matrix = df\_numeric.corr()

# Heatmap com customizações

sns.heatmap(

corr\_matrix,

annot=True, # Mostra valores dentro dos quadrados

fmt=".2f", # Formato com 2 casas decimais

cmap='coolwarm', # Mapa de cores (azul para negativo, vermelho para positivo)

center=0, # Centraliza o branco no zero

vmin=-1, vmax=1, # Limites fixos para a escala de cores

linewidths=0.5, # Linhas finas entre as células

linecolor='white', # Cor das linhas divisórias

square=True # Mantém as células quadradas

)

# Customizações adicionais

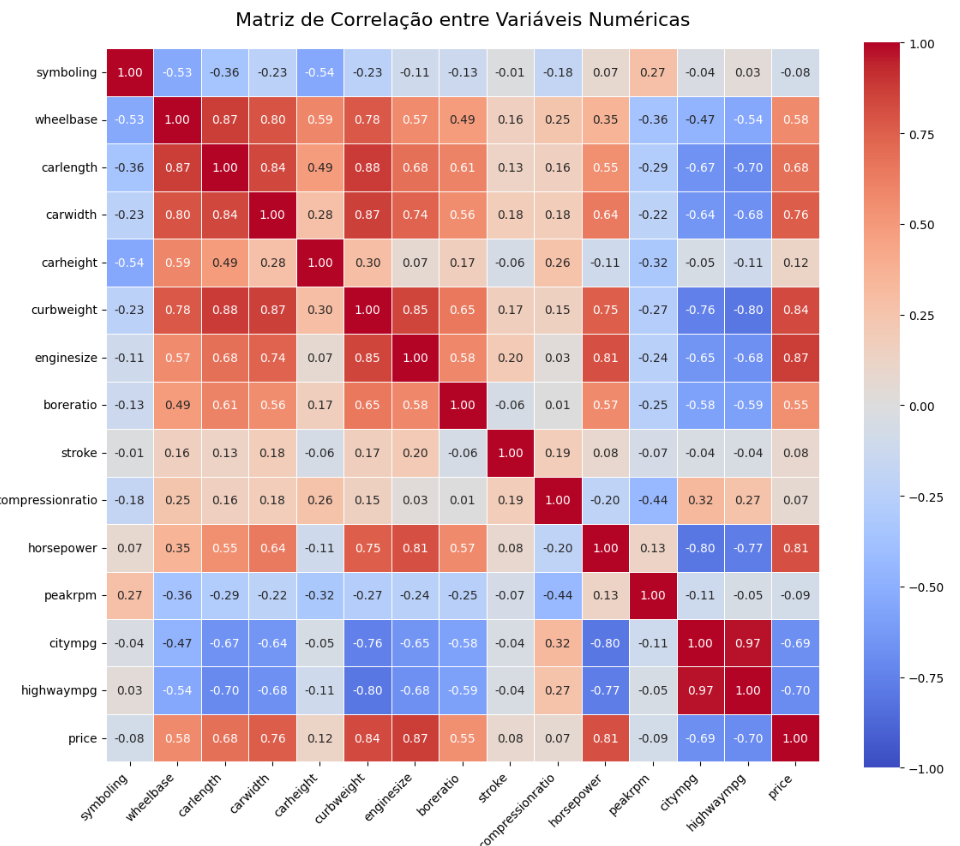
plt.title('Matriz de Correlação entre Variáveis Numéricas', pad=20, fontsize=16)

plt.xticks(rotation=45, ha='right') # Rótulos do eixo X em 45 graus

plt.yticks(rotation=0) # Rótulos do eixo Y horizontais

plt.tight\_layout() # Ajuste automático do layout

plt.show()



Como Interpretar os Resultados

Correlação Positiva (valores próximos de +1):

- Quando uma variável aumenta, a outra tende a aumentar

- Exemplo: `enginesize` e `price` geralmente têm alta correlação positiva

Correlação Negativa (valores próximos de -1):

- Quando uma variável aumenta, a outra tende a diminuir

- Exemplo: `citympg` e `enginesize` geralmente têm correlação negativa

Sem Correlação (valores próximos de 0):

- Não há relação linear aparente entre as variáveis

Diagonal Principal (valores = 1):

- Mostra a correlação de cada variável com ela mesma (sempre perfeita)

Aplicações Práticas

Esta análise é útil para:

- Identificar relações fortes entre variáveis

- Detectar multicolinearidade em modelos estatísticos

- Selecionar features para modelos de machine learning

- Entender a estrutura dos dados

**Desafio:**

Desafio 2: como fazer um gráfico que compare tendências de dois dados numéricos (price e mais outro dado, como cardwidth por exemplo) com alta correlação, positiva ou negativa?

1.) Gráfico de Dispersão (Scatter Plot) - Ideal para correlações

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(10, 6))

sns.scatterplot(data=df, x='carwidth', y='price', hue='carbody', style='fueltype', s=100)

plt.title('Relação entre Largura do Carro e Preço', fontsize=14)

plt.xlabel('Largura do Carro (cm)', fontsize=12)

plt.ylabel('Preço (USD)', fontsize=12)

plt.grid(True, alpha=0.3)

plt.tight\_layout()

# Adiciona linha de tendência

sns.regplot(data=df, x='carwidth', y='price', scatter=False, color='red', line\_kws={'linestyle':'--'})

plt.show()

O que este gráfico mostra:

- Cada ponto representa um carro

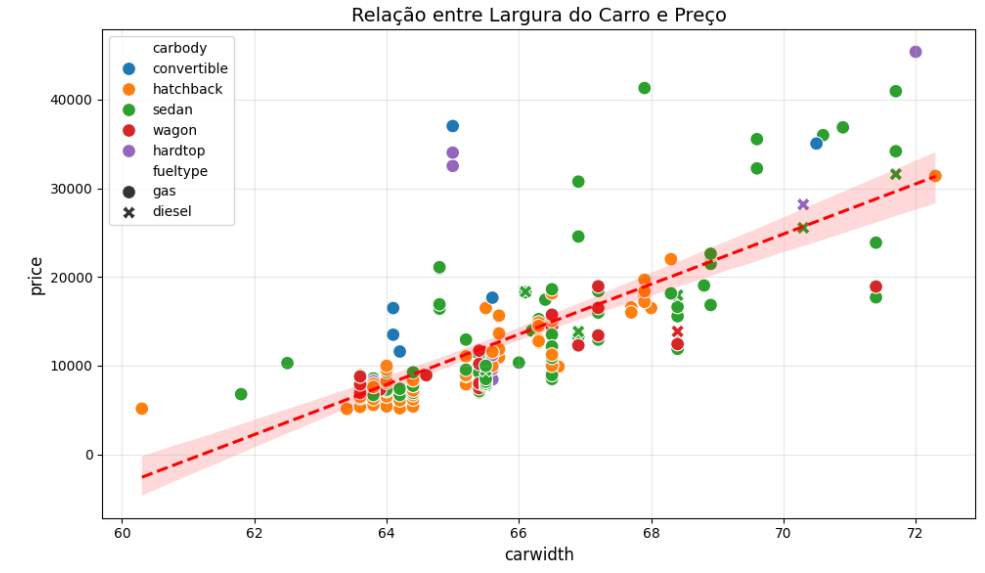
- Eixo X: largura do carro

- Eixo Y: preço

- Cores: tipo de carroceria (categórico adicional)

- Formas: tipo de combustível (outro dado categórico)

- Linha vermelha tracejada: linha de tendência linear



2.) Gráfico de Linha para Tendências Temporais (se tiver ano/mês)

Se seus dados tiverem uma dimensão temporal:

plt.figure(figsize=(12, 6))

sns.lineplot(data=df, x='carwidth', y='price', hue='carbody', style='fueltype',

markers=True, dashes=False, linewidth=2.5)

plt.title('Tendência: Largura vs Preço por Tipo de Carroceria', fontsize=14)

plt.xlabel('Largura do Carro (cm)', fontsize=12)

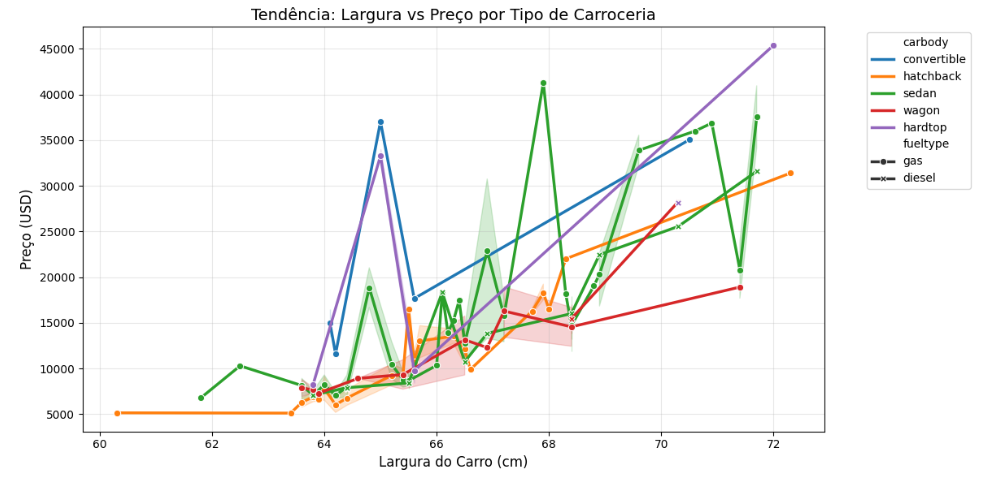
plt.ylabel('Preço (USD)', fontsize=12)

plt.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.grid(True, alpha=0.3)

plt.tight\_layout()

plt.show()



3.) Pair Plot para Múltiplas Comparações

Para comparar várias variáveis correlacionadas de uma vez:

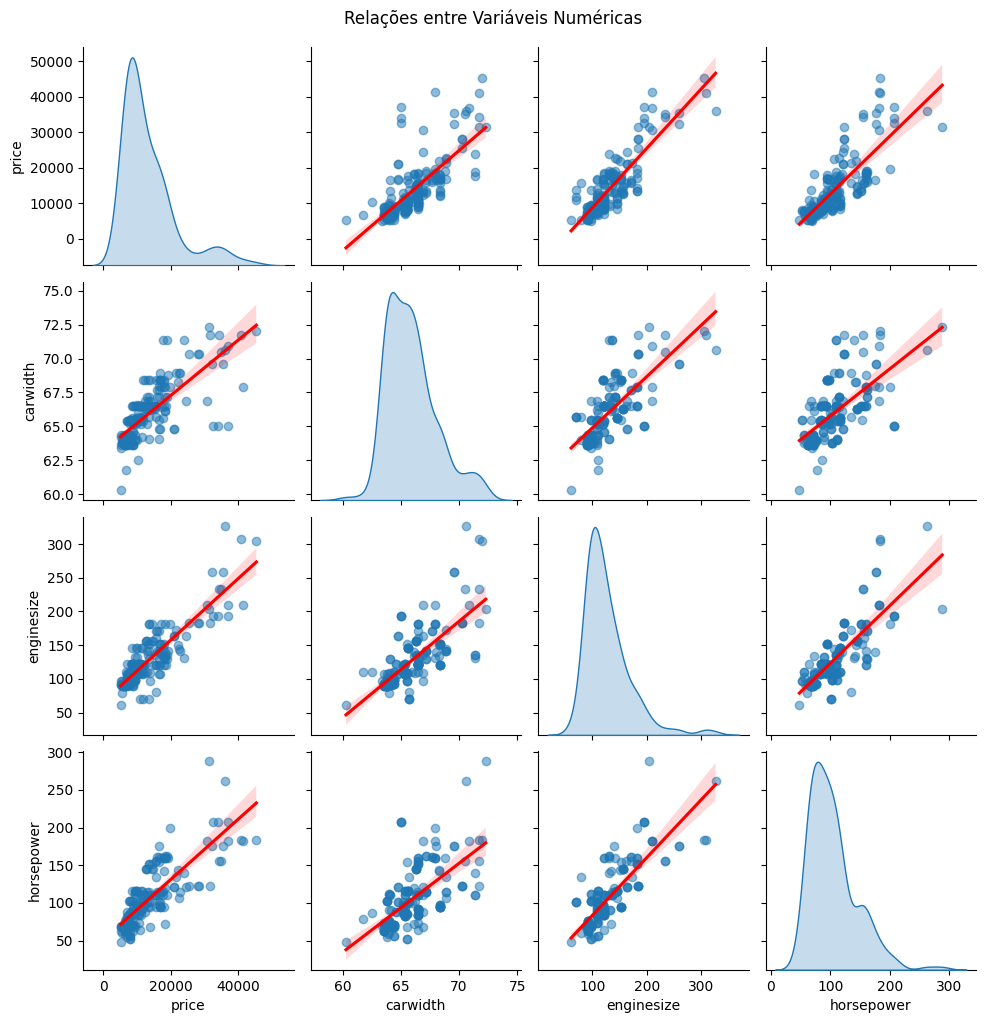
sns.pairplot(df[['price', 'carwidth', 'enginesize', 'horsepower']],

kind='reg', diag\_kind='kde',

plot\_kws={'line\_kws':{'color':'red'}, 'scatter\_kws': {'alpha': 0.5}})

plt.suptitle('Relações entre Variáveis Numéricas', y=1.02)

plt.show()



4.) Gráfico de Bolhas (Bubble Plot) - 3 Dimensões

plt.figure(figsize=(12, 8))

scatter = sns.scatterplot(data=df, x='carwidth', y='price',

size='enginesize', hue='horsepower',

sizes=(50, 500), alpha=0.7, palette='viridis')

plt.title('Relação Preço x Largura x Tamanho do Motor x Potência', fontsize=14)

plt.xlabel('Largura do Carro (cm)', fontsize=12)

plt.ylabel('Preço (USD)', fontsize=12)

# Ajusta a legenda

plt.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left', title='Potência (HP)')

# Adiciona valor de correlação no gráfico

corr = df[['price', 'carwidth']].corr().iloc[0,1]

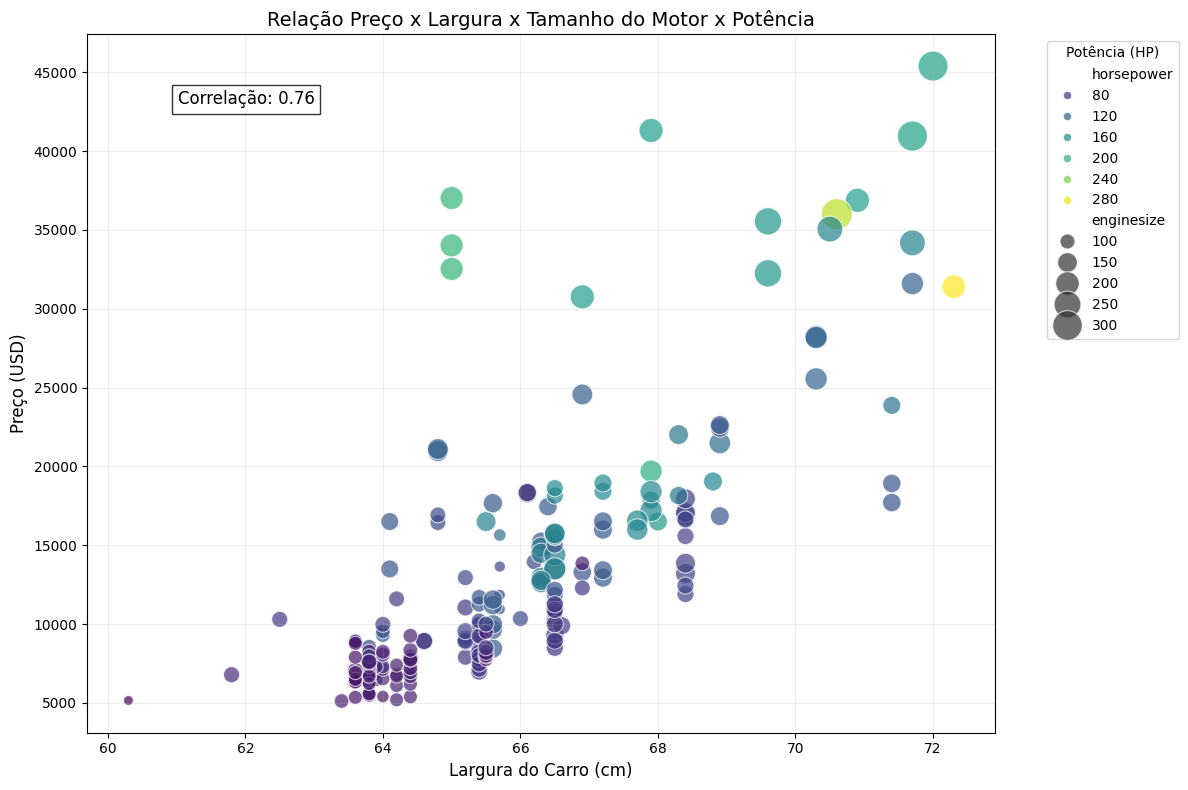
plt.text(0.1, 0.9, f'Correlação: {corr:.2f}', transform=plt.gca().transAxes,

fontsize=12, bbox=dict(facecolor='white', alpha=0.8))

plt.grid(True, alpha=0.2)

plt.tight\_layout()

plt.show()



Como Interpretar:

Correlação Positiva:

- Pontos formam uma "linha" ascendente da esquerda inferior para direita superior

- Exemplo: carros mais largos tendem a ser mais caros

Correlação Negativa:

- Pontos formam uma "linha" descendente da esquerda superior para direita inferior

- Exemplo: carros mais pesados tendem a ter menor consumo (mpg)

Sem Correlação:

- Pontos dispersos sem padrão aparente