**Nome do arquivo sugerido:** Cyclistic\_BikeShare\_Analysis.ipynb

**Estrutura do Notebook:**

Python

# Importando as bibliotecas necessárias no início do notebook

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.express as px # Para visualizações interativas

from datetime import datetime, timedelta

# Configurações para melhor visualização dos DataFrames no Jupyter

pd.set\_option('display.max\_columns', None)

pd.set\_option('display.width', 1000)

pd.set\_option('display.max\_rows', 100)

# Configurações visuais para gráficos

sns.set\_style("whitegrid")

plt.rcParams['figure.figsize'] = (12, 7) # Tamanho padrão para figuras

plt.rcParams['font.size'] = 12

plt.rcParams['axes.labelsize'] = 14

plt.rcParams['axes.titlesize'] = 16

plt.rcParams['xtick.labelsize'] = 12

plt.rcParams['ytick.labelsize'] = 12

plt.rcParams['legend.fontsize'] = 12

**Título: Análise de Dados de Compartilhamento de Bicicletas Cyclistic: Convertendo Ciclistas Casuais em Membros Anuais**

**Introdução e Contexto do Negócio**

A Cyclistic é um programa de compartilhamento de bicicletas em Chicago, com mais de 5.800 bicicletas e 600 estações. A empresa se destaca por oferecer opções de bicicletas inclusivas (reclinadas, triciclos manuais, bicicletas de carga), embora a maioria dos usuários opte por bicicletas tradicionais. Cerca de 30% dos usuários utilizam as bicicletas para o trajeto casa-trabalho, enquanto a maioria as usa para lazer.

O diretor de marketing da Cyclistic, Lily Moreno, identificou que maximizar o número de assinaturas anuais é crucial para o sucesso futuro da empresa. A análise financeira da Cyclistic já demonstrou que membros anuais são mais lucrativos do que ciclistas casuais (que utilizam passes de uma única viagem ou de dia inteiro). O objetivo central desta análise é entender como os ciclistas casuais e os membros anuais usam as bicicletas da Cyclistic de forma diferente, a fim de desenvolver uma estratégia de marketing para converter ciclistas casuais em membros.

Este estudo de caso seguirá o processo de análise de dados: Perguntar (Ask), Preparar (Prepare), Processar (Process), Analisar (Analyze), Compartilhar (Share) e Agir (Act).

**Etapa 1: Perguntar (Ask)**

Nesta fase, definimos a tarefa de negócios e as perguntas que guiarão nossa análise.

**Tarefa de Negócios:** A tarefa principal é entender como os membros anuais e os ciclistas casuais usam as bicicletas da Cyclistic de forma diferente. A partir desses insights, a equipe de marketing desenvolverá uma nova estratégia para converter ciclistas casuais em membros anuais. As recomendações devem ser apoiadas por insights de dados convincentes e visualizações profissionais.

**Perguntas-chave para a análise:**

1. Como os membros anuais e os ciclistas casuais usam as bicicletas da Cyclistic de forma diferente?
2. (Perguntas adicionais para insights mais profundos, embora o foco principal seja a primeira pergunta)
   * Existe diferença na duração média das viagens entre os dois tipos de usuários?
   * Existem padrões de uso diferentes por dia da semana ou por hora do dia?
   * Existem estações de partida ou chegada populares para cada tipo de usuário?
   * Quais tipos de bicicletas (tradicionais vs. assistência) são usados por cada grupo?

**Etapa 2: Preparar (Prepare)**

Nesta etapa, focamos na aquisição e primeira inspeção dos dados.

**Fonte de Dados:** Os dados utilizados são os dados históricos de viagens da Cyclistic dos 12 meses anteriores. Embora a Cyclistic seja fictícia, os dados são provenientes do programa Divvy (que é a versão real dos dados) e foram disponibilizados pela Motivate International Inc.. É importante notar que esses dados são públicos e não contêm informações de identificação pessoal dos ciclistas devido a questões de privacidade.

Para este projeto, assumimos que os arquivos CSV dos últimos 12 meses (por exemplo, de maio de 2023 a abril de 2024, ou o período mais recente disponível do Divvy) foram baixados e estão na pasta data/. Você precisará concatenar esses arquivos em um único DataFrame para facilitar a análise.

**Guia de Dados (Exemplo):** Os arquivos CSV geralmente contêm colunas como:

* ride\_id (ID da viagem)
* rideable\_type (tipo de bicicleta: "classic\_bike", "electric\_bike", "docked\_bike")
* started\_at (data e hora de início da viagem)
* ended\_at (data e hora de término da viagem)
* start\_station\_name (nome da estação de início)
* start\_station\_id (ID da estação de início)
* end\_station\_name (nome da estação de término)
* end\_station\_id (ID da estação de término)
* start\_lat (latitude da estação de início)
* start\_lng (longitude da estação de início)
* end\_lat (latitude da estação de término)
* end\_lng (longitude da estação de término)
* member\_casual (tipo de usuário: "member" ou "casual")

Python

# 1. Carregar os Dados

# A. Listar todos os arquivos CSV na pasta 'data'

import os

# Certifique-se de que seus arquivos CSV estejam em uma pasta chamada 'data' no mesmo diretório do seu notebook.

# Exemplo: data/202305-divvy-tripdata.csv, data/202306-divvy-tripdata.csv, etc.

data\_path = 'data/' # Altere se seus arquivos estiverem em outro lugar

csv\_files = [os.path.join(data\_path, f) for f in os.listdir(data\_path) if f.endswith('.csv')]

if not csv\_files:

print(f"Nenhum arquivo CSV encontrado na pasta '{data\_path}'. Por favor, baixe os dados do Divvy e coloque-os nesta pasta.")

print("Para continuar com a estrutura, criaremos um DataFrame de exemplo.")

# Criando um DataFrame de exemplo para demonstração se nenhum arquivo for encontrado

data = {

'ride\_id': [f'R{i:05}' for i in range(1, 101)],

'rideable\_type': np.random.choice(['classic\_bike', 'electric\_bike', 'docked\_bike'], 100),

'started\_at': pd.to\_datetime([f'2023-05-{np.random.randint(1,31):02d} {np.random.randint(0,24):02d}:{np.random.randint(0,60):02d}:{np.random.randint(0,60):02d}' for \_ in range(100)]),

'ended\_at': pd.to\_datetime([f'2023-05-{np.random.randint(1,31):02d} {np.random.randint(0,24):02d}:{np.random.randint(0,60):02d}:{np.random.randint(0,60):02d}' for \_ in range(100)]),

'start\_station\_name': np.random.choice(['Station A', 'Station B', 'Station C', 'Station D'], 100),

'start\_station\_id': np.random.choice(['STA1', 'STA2', 'STA3', 'STA4'], 100),

'end\_station\_name': np.random.choice(['Station A', 'Station B', 'Station C', 'Station D', np.nan], 100),

'end\_station\_id': np.random.choice(['STA1', 'STA2', 'STA3', 'STA4', np.nan], 100),

'start\_lat': np.random.uniform(41.8, 41.9, 100),

'start\_lng': np.random.uniform(-87.7, -87.6, 100),

'end\_lat': np.random.uniform(41.8, 41.9, 100),

'end\_lng': np.random.uniform(-87.7, -87.6, 100),

'member\_casual': np.random.choice(['member', 'casual'], 100, p=[0.6, 0.4]) # 60% members, 40% casual

}

df = pd.DataFrame(data)

# Ajustar ended\_at para ser sempre depois de started\_at no exemplo

df['ended\_at'] = df.apply(lambda row: row['started\_at'] + timedelta(minutes=np.random.randint(5, 60)), axis=1)

else:

# B. Concatenar todos os arquivos em um único DataFrame

list\_df = []

print(f"Carregando {len(csv\_files)} arquivos CSV...")

for file in csv\_files:

try:

temp\_df = pd.read\_csv(file)

list\_df.append(temp\_df)

print(f"Carregado: {os.path.basename(file)}")

except Exception as e:

print(f"Erro ao carregar {os.path.basename(file)}: {e}")

print("Pode ser um arquivo corrompido ou formato inesperado. Ignorando este arquivo.")

if list\_df:

df = pd.concat(list\_df, ignore\_index=True)

print("\nTodos os arquivos CSV foram concatenados com sucesso!")

else:

print("Nenhum DataFrame foi carregado com sucesso. Verifique os arquivos CSV.")

# Fallback para DataFrame de exemplo se a concatenação falhar

print("Criando DataFrame de exemplo para demonstração...")

data = {

'ride\_id': [f'R{i:05}' for i in range(1, 101)],

'rideable\_type': np.random.choice(['classic\_bike', 'electric\_bike', 'docked\_bike'], 100),

'started\_at': pd.to\_datetime([f'2023-05-{np.random.randint(1,31):02d} {np.random.randint(0,24):02d}:{np.random.randint(0,60):02d}:{np.random.randint(0,60):02d}' for \_ in range(100)]),

'ended\_at': pd.to\_datetime([f'2023-05-{np.random.randint(1,31):02d} {np.random.randint(0,60):02d}:{np.random.randint(0,60):02d}' for \_ in range(100)]), # Hora de término pode ser antes da de início, será tratado no processamento

'start\_station\_name': np.random.choice(['Station A', 'Station B', 'Station C', 'Station D'], 100),

'start\_station\_id': np.random.choice(['STA1', 'STA2', 'STA3', 'STA4'], 100),

'end\_station\_name': np.random.choice(['Station A', 'Station B', 'Station C', 'Station D', np.nan], 100),

'end\_station\_id': np.random.choice(['STA1', 'STA2', 'STA3', 'STA4', np.nan], 100),

'start\_lat': np.random.uniform(41.8, 41.9, 100),

'start\_lng': np.random.uniform(-87.7, -87.6, 100),

'end\_lat': np.random.uniform(41.8, 41.9, 100),

'end\_lng': np.random.uniform(-87.7, -87.6, 100),

'member\_casual': np.random.choice(['member', 'casual'], 100, p=[0.6, 0.4]) # 60% members, 40% casual

}

df = pd.DataFrame(data)

df['ended\_at'] = df.apply(lambda row: row['started\_at'] + timedelta(minutes=np.random.randint(5, 60)), axis=1) # Garante que ended\_at > started\_at

# 2. Inspeção Inicial do DataFrame

print("\n--- Primeiras 5 linhas do DataFrame ---")

print(df.head())

print("\n--- Informações do DataFrame (tipos de dados e valores não-nulos) ---")

print(df.info())

print("\n--- Estatísticas Descritivas do DataFrame ---")

print(df.describe(include='all'))

print("\n--- Contagem de valores únicos para 'member\_casual' ---")

print(df['member\_casual'].value\_counts())

**Etapa 3: Processar (Process)**

Esta é a fase de limpeza, transformação e preparação dos dados para análise. A integridade dos dados é crucial.

Python

# Criar uma cópia do DataFrame original para manter o estado inicial

df\_processed = df.copy()

# 1. Padronizar Nomes de Colunas (se necessário)

# Os nomes de coluna padrão do Divvy são geralmente snake\_case e bons.

# Se seus dados tivessem nomes como 'Start Station Name', você faria:

# df\_processed.rename(columns={'Start Station Name': 'start\_station\_name'}, inplace=True)

# 2. Corrigir Tipos de Dados

# Converter colunas de data e hora para o tipo datetime

df\_processed['started\_at'] = pd.to\_datetime(df\_processed['started\_at'])

df\_processed['ended\_at'] = pd.to\_datetime(df\_processed['ended\_at'])

print("\n--- Tipos de Dados Após Conversão ---")

print(df\_processed.info())

# 3. Criar Novas Colunas Essenciais para Análise

# Duração da Viagem em Minutos (arredondado para o inteiro mais próximo)

df\_processed['ride\_length\_minutes'] = (df\_processed['ended\_at'] - df\_processed['started\_at']).dt.total\_seconds() / 60

df\_processed['ride\_length\_minutes'] = df\_processed['ride\_length\_minutes'].round().astype(int)

# Dia da Semana (nome e número)

df\_processed['day\_of\_week'] = df\_processed['started\_at'].dt.day\_name()

df\_processed['weekday\_num'] = df\_processed['started\_at'].dt.dayofweek # 0=Monday, 6=Sunday

# Mês da Viagem

df\_processed['month'] = df\_processed['started\_at'].dt.month\_name()

df\_processed['month\_num'] = df\_processed['started\_at'].dt.month

# Hora do Dia

df\_processed['hour\_of\_day'] = df\_processed['started\_at'].dt.hour

# 4. Tratar Valores Ausentes (NaNs)

print("\n--- Valores Ausentes Antes do Tratamento ---")

print(df\_processed.isnull().sum())

# Para nomes de estações e IDs, NaNs indicam viagens que não começaram/terminaram em uma estação

# Para Lat/Lng, NaNs são mais problemáticos para visualizações de mapa. Podemos remover ou imputar.

# Para esta análise, removeremos linhas com NaNs em colunas cruciais para a análise de estações/geográfica.

# Para o objetivo de comparar membros vs. casuais, o foco é mais na duração e tempo de uso.

# Vamos focar em NaNs que afetam a duração da viagem ou o tipo de usuário.

# Remover linhas onde 'member\_casual' é nulo (improvável, mas boa prática)

df\_processed.dropna(subset=['member\_casual'], inplace=True)

# Remover linhas onde a estação de início ou fim, ou as coordenadas, são nulas

# Isso é crucial para análises espaciais ou por estação.

df\_processed.dropna(subset=['start\_station\_name', 'end\_station\_name', 'start\_lat', 'start\_lng', 'end\_lat', 'end\_lng'], inplace=True)

print("\n--- Valores Ausentes Depois do Tratamento ---")

print(df\_processed.isnull().sum())

# 5. Tratar Duplicatas

print(f"\nDuplicatas antes do tratamento: {df\_processed.duplicated().sum()}")

df\_processed.drop\_duplicates(inplace=True)

print(f"Duplicatas depois do tratamento: {df\_processed.duplicated().sum()}")

# 6. Tratar Valores Inconsistentes/Inválidos

# Remover viagens com duração <= 0 minutos (viagens inválidas ou de teste)

print(f"\nViagens com duração <= 0 minutos antes: {df\_processed[df\_processed['ride\_length\_minutes'] <= 0].shape[0]}")

df\_processed = df\_processed[df\_processed['ride\_length\_minutes'] > 0]

print(f"Viagens com duração <= 0 minutos depois: {df\_processed[df\_processed['ride\_length\_minutes'] <= 0].shape[0]}")

# Remover viagens com duração muito longa (outliers extremos, como 24h+ que podem ser bikes perdidas ou dados incorretos)

# Vamos considerar que uma viagem de 24 horas (1440 minutos) é um outlier extremo para análise de uso diário.

# O limite pode ser ajustado com base na compreensão do negócio.

MAX\_RIDE\_LENGTH\_MINUTES = 1440 # 24 horas

print(f"Viagens com duração > {MAX\_RIDE\_LENGTH\_MINUTES} minutos antes: {df\_processed[df\_processed['ride\_length\_minutes'] > MAX\_RIDE\_LENGTH\_MINUTES].shape[0]}")

df\_processed = df\_processed[df\_processed['ride\_length\_minutes'] <= MAX\_RIDE\_LENGTH\_MINUTES]

print(f"Viagens com duração > {MAX\_RIDE\_LENGTH\_MINUTES} minutos depois: {df\_processed[df\_processed['ride\_length\_minutes'] > MAX\_RIDE\_LENGTH\_MINUTES].shape[0]}")

# 7. Verificar a Integridade dos Dados (Após Processamento)

print("\n--- Amostra do DataFrame Processado ---")

print(df\_processed.head())

print("\n--- Informações Finais do DataFrame Processado ---")

print(df\_processed.info())

print("\n--- Estatísticas Descritivas Finais (Duração da Viagem) ---")

print(df\_processed['ride\_length\_minutes'].describe())

**Etapa 4: Analisar (Analyze)**

Agora, vamos mergulhar nos dados para responder às perguntas-chave, focando nas diferenças entre membros e ciclistas casuais.

Python

# 1. Comparação da Duração Média da Viagem

# Agrupar por tipo de usuário e calcular a média da duração da viagem

avg\_ride\_length = df\_processed.groupby('member\_casual')['ride\_length\_minutes'].mean().reset\_index()

print("\n--- Duração Média da Viagem por Tipo de Usuário (minutos) ---")

print(avg\_ride\_length)

# 2. Distribuição da Duração da Viagem

# Para visualizar a distribuição, podemos usar um boxplot ou histograma para cada grupo.

# Boxplot é bom para comparar medianas, quartis e outliers.

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

sns.boxplot(x='member\_casual', y='ride\_length\_minutes', data=df\_processed)

plt.title('Distribuição da Duração da Viagem por Tipo de Usuário')

plt.xlabel('Tipo de Usuário')

plt.ylabel('Duração da Viagem (minutos)')

plt.ylim(0, 60) # Limitar para melhor visualização, pois há outliers longos

plt.show()

# 3. Contagem de Viagens por Tipo de Usuário

ride\_counts = df\_processed['member\_casual'].value\_counts(normalize=True) \* 100

print("\n--- Porcentagem de Viagens por Tipo de Usuário ---")

print(ride\_counts)

# 4. Padrões de Uso por Dia da Semana

# Ordenar os dias da semana corretamente

day\_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

df\_processed['day\_of\_week'] = pd.Categorical(df\_processed['day\_of\_week'], categories=day\_order, ordered=True)

# Duração média da viagem por dia da semana e tipo de usuário

avg\_ride\_length\_dow = df\_processed.groupby(['day\_of\_week', 'member\_casual'])['ride\_length\_minutes'].mean().unstack().reindex(day\_order)

print("\n--- Duração Média da Viagem por Dia da Semana e Tipo de Usuário ---")

print(avg\_ride\_length\_dow)

# Contagem de viagens por dia da semana e tipo de usuário

ride\_counts\_dow = df\_processed.groupby(['day\_of\_week', 'member\_casual']).size().unstack().reindex(day\_order)

print("\n--- Contagem de Viagens por Dia da Semana e Tipo de Usuário ---")

print(ride\_counts\_dow)

# 5. Padrões de Uso por Hora do Dia

# Duração média da viagem por hora do dia e tipo de usuário

avg\_ride\_length\_hour = df\_processed.groupby(['hour\_of\_day', 'member\_casual'])['ride\_length\_minutes'].mean().unstack()

print("\n--- Duração Média da Viagem por Hora do Dia e Tipo de Usuário ---")

print(avg\_ride\_length\_hour.head()) # Mostra as primeiras horas

# Contagem de viagens por hora do dia e tipo de usuário

ride\_counts\_hour = df\_processed.groupby(['hour\_of\_day', 'member\_casual']).size().unstack()

print("\n--- Contagem de Viagens por Hora do Dia e Tipo de Usuário ---")

print(ride\_counts\_hour.head()) # Mostra as primeiras horas

# 6. Estações Mais Populares (Top N)

# Top 10 estações de início para membros

top\_start\_members = df\_processed[df\_processed['member\_casual'] == 'member']['start\_station\_name'].value\_counts().head(10)

print("\n--- Top 10 Estações de Início - Membros ---")

print(top\_start\_members)

# Top 10 estações de início para casuais

top\_start\_casuals = df\_processed[df\_processed['member\_casual'] == 'casual']['start\_station\_name'].value\_counts().head(10)

print("\n--- Top 10 Estações de Início - Casuais ---")

print(top\_start\_casuals)

# Você pode repetir para estações de término ('end\_station\_name')

# 7. Tipos de Bicicleta Usados

bike\_type\_usage = df\_processed.groupby(['rideable\_type', 'member\_casual']).size().unstack().fillna(0)

print("\n--- Uso de Tipos de Bicicleta por Tipo de Usuário ---")

print(bike\_type\_usage)

# Calcular a porcentagem de uso de cada tipo de bicicleta dentro de cada categoria de usuário

bike\_type\_usage\_percent = bike\_type\_usage.apply(lambda x: x / x.sum() \* 100, axis=0)

print("\n--- Porcentagem de Uso de Tipos de Bicicleta por Tipo de Usuário ---")

print(bike\_type\_usage\_percent)

**Etapa 5: Compartilhar (Share) - Visualizações e Insights**

Aqui, criamos visualizações profissionais e as acompanhamos com insights claros. O objetivo é contar uma história com os dados que seja compreensível para os executivos da Cyclistic.

Python

# 1. Comparação da Duração Média da Viagem (Bar Plot)

plt.figure(figsize=(8, 5))

sns.barplot(x='member\_casual', y='ride\_length\_minutes', data=avg\_ride\_length, palette='viridis')

plt.title('Duração Média da Viagem: Membros vs. Ciclistas Casuais')

plt.xlabel('Tipo de Usuário')

plt.ylabel('Duração Média da Viagem (minutos)')

plt.text(0, avg\_ride\_length.loc[avg\_ride\_length['member\_casual'] == 'member', 'ride\_length\_minutes'].values[0] + 1,

f"{avg\_ride\_length.loc[avg\_ride\_length['member\_casual'] == 'member', 'ride\_length\_minutes'].values[0]:.1f} min",

ha='center', va='bottom', color='black')

plt.text(1, avg\_ride\_length.loc[avg\_ride\_length['member\_casual'] == 'casual', 'ride\_length\_minutes'].values[0] + 1,

f"{avg\_ride\_length.loc[avg\_ride\_length['member\_casual'] == 'casual', 'ride\_length\_minutes'].values[0]:.1f} min",

ha='center', va='bottom', color='black')

plt.show()

print("\nInsight 1: Ciclistas casuais tendem a ter durações de viagem significativamente mais longas do que os membros anuais. Isso sugere que casuais podem usar as bicicletas para lazer ou turismo, enquanto membros as usam para deslocamentos mais rápidos e eficientes.")

# 2. Número de Viagens por Dia da Semana e Tipo de Usuário (Line Plot)

plt.figure(figsize=(14, 8))

sns.lineplot(x='day\_of\_week', y='count', hue='member\_casual', data=ride\_counts\_dow.stack().reset\_index(name='count'), marker='o', lw=2, palette='coolwarm')

plt.title('Número de Viagens por Dia da Semana e Tipo de Usuário')

plt.xlabel('Dia da Semana')

plt.ylabel('Número de Viagens')

plt.legend(title='Tipo de Usuário')

plt.grid(True, linestyle='--', alpha=0.7)

plt.show()

print("\nInsight 2: Membros utilizam as bicicletas de forma consistente durante a semana, com um leve aumento nos dias úteis, indicando uso para deslocamento. Ciclistas casuais, por outro lado, têm um pico de uso nos fins de semana, reforçando a ideia de uso recreativo.")

# 3. Duração Média da Viagem por Hora do Dia (Line Plot)

plt.figure(figsize=(14, 8))

sns.lineplot(x='hour\_of\_day', y='ride\_length\_minutes', hue='member\_casual', data=avg\_ride\_length\_hour.stack().reset\_index(name='ride\_length\_minutes'), marker='o', lw=2, palette='plasma')

plt.title('Duração Média da Viagem por Hora do Dia e Tipo de Usuário')

plt.xlabel('Hora do Dia')

plt.ylabel('Duração Média da Viagem (minutos)')

plt.xticks(range(0, 24))

plt.grid(True, linestyle='--', alpha=0.7)

plt.show()

print("\nInsight 3: Ciclistas casuais tendem a fazer viagens mais longas em todas as horas do dia, mas especialmente fora dos horários de pico de deslocamento (manhã e fim de tarde), o que reforça o uso de lazer. Membros mantêm durações de viagem mais curtas e consistentes.")

# 4. Uso de Tipos de Bicicleta (Stacked Bar Plot)

bike\_type\_usage\_percent.T.plot(kind='bar', stacked=True, figsize=(10, 6), colormap='tab20')

plt.title('Distribuição do Tipo de Bicicleta por Usuário')

plt.xlabel('Tipo de Usuário')

plt.ylabel('Porcentagem (%)')

plt.xticks(rotation=0)

plt.legend(title='Tipo de Bicicleta')

plt.tight\_layout()

plt.show()

print("\nInsight 4: Ambos os grupos usam principalmente bicicletas clássicas e elétricas. A proporção de 'docked\_bike' (bicicletas fixas em docas) pode ser mínima ou inexistente nos dados mais recentes, já que as bicicletas elétricas e clássicas são mais prevalentes. Se houver 'docked\_bike', pode indicar um uso mais tradicional por membros.")

# 5. Top 5 Estações de Início (Bar Plot Comparativo)

# Unindo os top 5 para ambos os grupos para uma visualização combinada

top\_stations\_combined = pd.concat([

top\_start\_members.head(5).rename('Membros'),

top\_start\_casuals.head(5).rename('Casuais')

], axis=1).fillna(0) # Preenche NaN se uma estação não estiver no top 5 do outro grupo

top\_stations\_combined.plot(kind='bar', figsize=(14, 7), colormap='Set2')

plt.title('Top 5 Estações de Início Populares: Membros vs. Casuais')

plt.xlabel('Estação de Início')

plt.ylabel('Número de Viagens')

plt.xticks(rotation=45, ha='right')

plt.legend(title='Tipo de Usuário')

plt.tight\_layout()

plt.show()

print("\nInsight 5: Observar as estações mais populares de cada grupo pode revelar se membros e casuais têm pontos de partida preferenciais que se alinham com deslocamento (membros) ou lazer/turismo (casuais). Por exemplo, estações perto de escritórios para membros e perto de parques/atrações turísticas para casuais.")

# Visualização Interativa (Opcional, mas altamente recomendada para impressionar)

# Scatter plot de duração da viagem vs. hora do dia, com cor por tipo de usuário (amostra para desempenho)

df\_sample = df\_processed.sample(min(10000, len(df\_processed)), random\_state=42) # Amostra para gráficos grandes

fig = px.scatter(df\_sample, x='hour\_of\_day', y='ride\_length\_minutes', color='member\_casual',

title='Duração da Viagem vs. Hora do Dia por Tipo de Usuário',

labels={'hour\_of\_day': 'Hora do Dia', 'ride\_length\_minutes': 'Duração da Viagem (minutos)'},

hover\_data=['day\_of\_week', 'start\_station\_name', 'end\_station\_name'],

opacity=0.6, marginal\_y='histogram', marginal\_x='histogram',

color\_discrete\_map={'member': 'blue', 'casual': 'red'}) # Cores para membros e casuais

fig.update\_layout(xaxis\_title="Hora do Dia", yaxis\_title="Duração da Viagem (Minutos)",

xaxis\_tickmode='linear', xaxis\_dtick=1, yaxis\_range=[0, 100]) # Ajusta limites para visibilidade

fig.show()

print("\nInsight Adicional (Interativo): A visualização interativa permite explorar a densidade de viagens e os outliers, confirmando que as viagens mais longas são predominantemente de ciclistas casuais e ocorrem em horários de menor pico de deslocamento.")

**Etapa 6: Agir (Act) - Recomendações e Próximos Passos**

Com base nos insights obtidos, vamos formular recomendações acionáveis para a equipe de marketing da Cyclistic.

**Resumo da Análise:**

A análise dos dados de viagens da Cyclistic revelou diferenças claras no comportamento de uso entre membros anuais e ciclistas casuais.

* **Duração da Viagem:** Ciclistas casuais realizam viagens significativamente mais longas (média de X minutos) em comparação com os membros anuais (média de Y minutos). Isso sugere que casuais usam as bicicletas para fins recreativos ou turísticos, enquanto os membros as utilizam para deslocamentos mais curtos e eficientes.
* **Padrões de Uso Semanal:** Membros utilizam as bicicletas de forma consistente durante a semana, com picos nos dias úteis (provavelmente para deslocamento). Casuais, por outro lado, mostram um aumento notável de uso nos fins de semana.
* **Padrões de Uso Diário:** Ciclistas casuais tendem a fazer viagens mais longas em todas as horas do dia, com destaque para horários fora do pico de deslocamento, reforçando o uso de lazer.
* **Tipos de Bicicleta e Estações:** (Dependendo dos insights específicos das suas visualizações de tipos de bicicleta e estações, você os resumiria aqui. Por exemplo, se membros usam mais classic\_bike para deslocamento e casuais usam mais electric\_bike para passeios mais longos, ou se estações perto de parques são populares para casuais, etc.)

**Top Três Recomendações para Converter Ciclistas Casuais em Membros Anuais:**

1. **Ofertas de Assinatura com Foco no Lazer e Valor a Longo Prazo:**
   * **Justificativa:** Ciclistas casuais valorizam viagens mais longas e recreativas, especialmente nos fins de semana. Uma assinatura anual pode parecer menos atraente se eles só a virem como "deslocamento".
   * **Ação:** Criar campanhas de marketing que destaquem os benefícios de uma assinatura anual para o **uso de lazer e turismo**. Por exemplo: "Explore Chicago sem limites", "Viagens de fim de semana ilimitadas por um preço fixo". Promover a economia a longo prazo para quem faz múltiplas viagens casuais.
   * **Mídia Digital:** Segmentar anúncios em mídias sociais (Instagram, Facebook) com imagens de pessoas desfrutando de passeios longos e relaxantes, perto de atrações turísticas ou parques, e oferecendo um "Teste de Fim de Semana Grátis" que, após um número X de viagens, sugere a assinatura anual.
2. **Programas de Fidelidade e Incentivo Progressivo:**
   * **Justificativa:** Ciclistas casuais já estão cientes do programa e o utilizam ocasionalmente. Eles precisam de um "empurrão" para a transição.
   * **Ação:** Implementar um programa de pontos ou descontos progressivos que recompense ciclistas casuais por cada viagem. Após um certo número de viagens ou gastos, oferecer um desconto significativo ou um mês grátis na primeira assinatura anual. Mensagens personalizadas para ciclistas que atingem esses marcos.
   * **Mídia Digital:** E-mails automatizados ou notificações push no aplicativo Cyclistic (se houver) que mostrem aos ciclistas casuais quanto eles teriam economizado com uma assinatura, baseando-se em seu histórico de uso, e apresentando a oferta de transição.
3. **Comunicação Direcionada por Caso de Uso e Benefício de Conveniência:**
   * **Justificativa:** Diferentes tipos de usuários têm diferentes necessidades. Para converter, é preciso apelar para o que eles valorizam. Casuais podem não perceber a conveniência de não ter que comprar um passe a cada vez.
   * **Ação:** Criar segmentos de comunicação baseados nos padrões de uso observados. Para casuais que usam muito nos fins de semana, focar em "liberdade ilimitada". Para aqueles que fazem viagens mais curtas, mas frequentes, focar na "conveniência e agilidade" da assinatura. Destacar a facilidade de simplesmente pegar e devolver uma bicicleta sem se preocupar com pagamentos por viagem.
   * **Mídia Digital:** Anúncios contextuais em sites de turismo, blogs de lazer ou aplicativos de clima no fim de semana. Usar retargeting para exibir anúncios para ciclistas casuais que acabaram de finalizar uma viagem, reforçando a mensagem de "Pronto para a próxima viagem? Seja um membro!".

**Limitações e Próximos Passos:**

* **Limitações:** Não temos acesso a dados demográficos ou informações pessoais identificáveis dos usuários, o que limita a segmentação mais profunda. Também não temos dados sobre a motivação direta para a compra da assinatura ou feedback dos usuários. A análise é baseada apenas no comportamento de uso das bicicletas.
* **Próximos Passos:**
  1. **Pesquisa de Usuários:** Realizar pesquisas com ciclistas casuais para entender suas razões para não terem uma assinatura e o que os motivaria a se converterem.
  2. **Teste A/B de Campanhas:** Lançar as campanhas de marketing recomendadas e monitorar seu impacto nas taxas de conversão de casuais para membros.
  3. **Análise de Estações:** Investigar a densidade e o tipo de estações próximas a áreas residenciais vs. turísticas para otimizar a disponibilidade de bicicletas para cada tipo de usuário.
  4. **Análise de Bicicletas Elétricas:** Aprofundar a análise do uso de bicicletas elétricas, que podem ser mais atraentes para viagens mais longas e lazer.