Verificar qual seria o impacto de retorno financeiro maior a empresa, investir no site ou app? Para que o investimento desejado, traga o maior retorno possível, onde tais resultados irão auxiliar na tomada de decisão dos líderes da equipe.

In [1]:
```
# Imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score, explained_variance_score
sns.set_style('whitegrid')
%matplotlib inline
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Carga de dados

In [2]:
```
# Carrega o dataset
dados = pd.read_csv('dados/dataset.csv')
dados.shape
```

Out[2]: (500, 5)

In [3]:
```
dados.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 5 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   tempo_cadastro_cliente          500 non-null    float64
 1   numero_medio_cliques_por_sessao  500 non-null    float64
 2   tempo_total_logado_app          500 non-null    float64
 3   tempo_total_logado_website      500 non-null    float64
 4   valor_total_gasto               500 non-null    float64
dtypes: float64(5)
memory usage: 19.7 KB
```

In [4]:
```
dados.sample(7)
```

Out[4]:

	tempo_cadastro_cliente	numero_medio_cliques_por_sessao	tempo_total_logado_app	tempo_total_logado_website	valor_total_gasto
446	3.982462	31.673915	12.329147	37.074371	475.725068
36	3.273434	33.987773	13.386235	37.534497	570.200409
228	3.261325	32.484260	10.933252	36.545506	425.745092
432	3.531402	32.278443	12.527472	36.688367	488.270298
297	2.907095	33.136655	13.891313	39.220713	524.797628
42	2.494544	32.387976	13.148726	36.619957	470.452733
347	5.566385	33.302672	13.459222	36.339521	689.787604

Análise Exploratória

Vamos verificar a correlação entre as variáveis, a fim de verificar as variáveis preditoras com a variável alvo (valor_total_gasto)

In [5]:
```
dados.columns
```

Out[5]:
```
Index(['tempo_cadastro_cliente', 'numero_medio_cliques_por_sessao',
       'tempo_total_logado_app', 'tempo_total_logado_website',
       'valor_total_gasto'],
      dtype='object')
```

In [6]:
```
# Correlação
dados.corr()
```

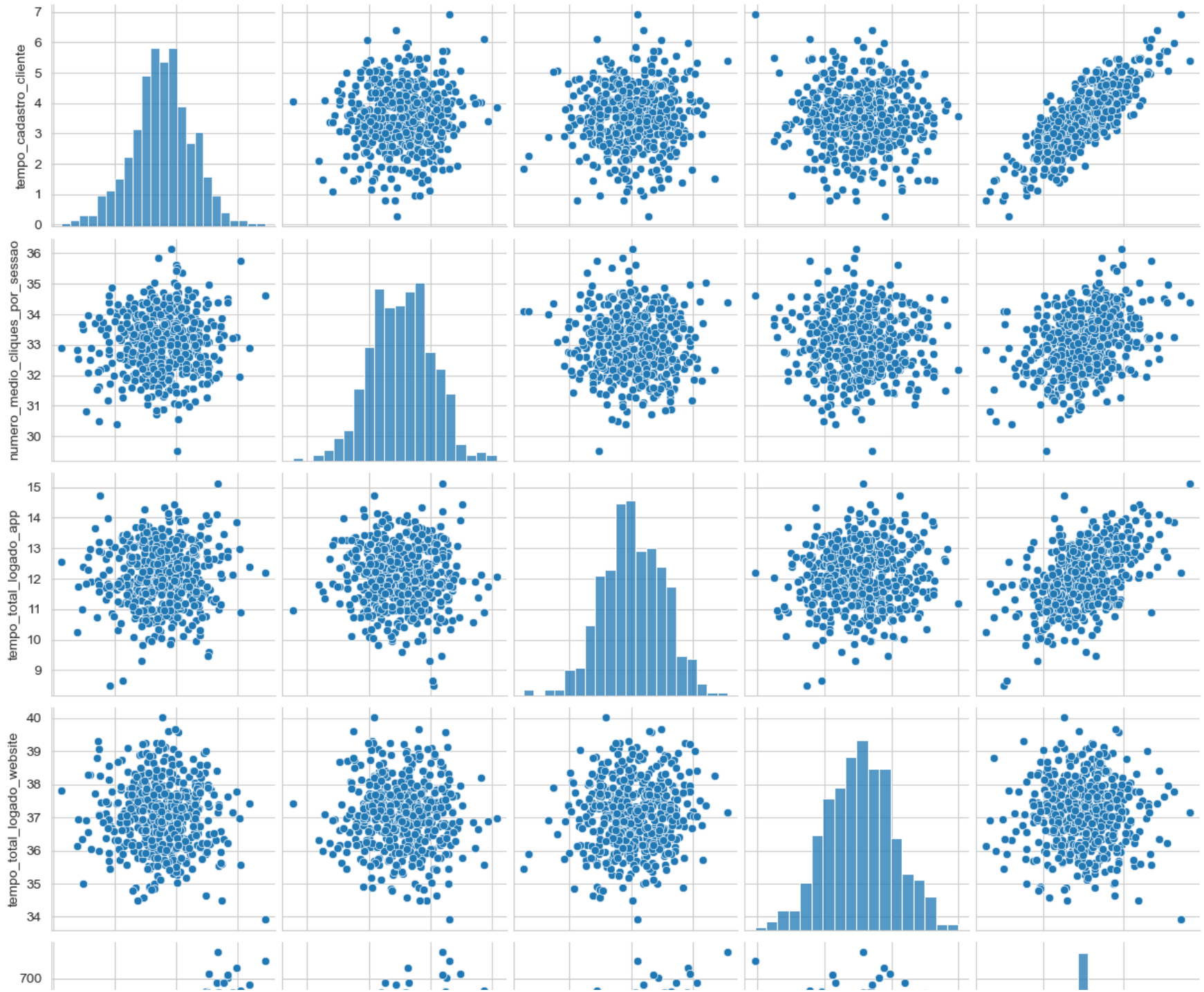
Out[6]:

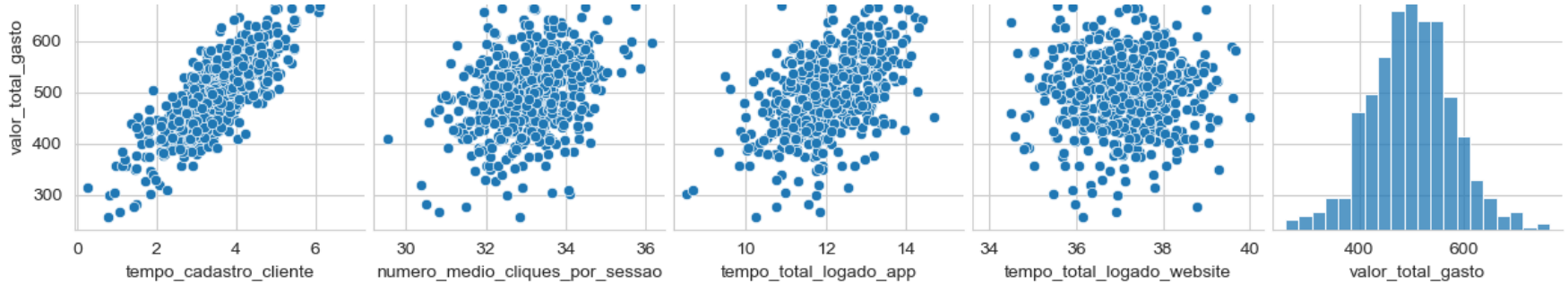
	tempo_cadastro_cliente	numero_medio_cliques_por_sessao	tempo_total_logado_app	tempo_total_logado_website	va
tempo_cadastro_cliente	1.000000	0.060247	0.029143	-0.047582	
ro_medio_cliques_por_sessao	0.060247	1.000000	-0.027826	-0.034987	
tempo_total_logado_app	0.029143	-0.027826	1.000000	0.082388	
tempo_total_logado_website	-0.047582	-0.034987	0.082388	1.000000	
valor_total_gasto	0.809084	0.355088	0.499328	-0.002641	

In [7]:
```
sns.pairplot(dados)
```

```
/Users/emerson/anaconda3/lib/python3.11/site-packages/seaborn/axisgrid.py:118: UserWarning: The figure l
ayout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x12bb44150>
```


tempo_cadastro_cliente
numero_medio_cliques_por_sessao
tempo_total_logado_app
tempo_total_logado_website
0
1
2
3
4
5
6
7
30
31
32
33
34
35
36
9
10
11
12
13
14
15
34
35
36
37
38
39
40
700



Relação entre o tempo no web site e valor gasto

In [8]:
```
# Correlação
dados[['tempo_total_logado_website', 'valor_total_gasto']].corr()
```

Out[8]:

	tempo_total_logado_website	valor_total_gasto
tempo_total_logado_website	1.000000	-0.002641
valor_total_gasto	-0.002641	1.000000

```
   Correlação muito baixa entre as variáveis tempo_total_logado_website e valor_total_gasto,
aparentando não haver alguma relação entre o tempo logado no website e o valor total gasto.
```

Relação entre o tempo na App e o valor total gasto

In [9]:
```
dados[['tempo_total_logado_app', 'valor_total_gasto']].corr()
```

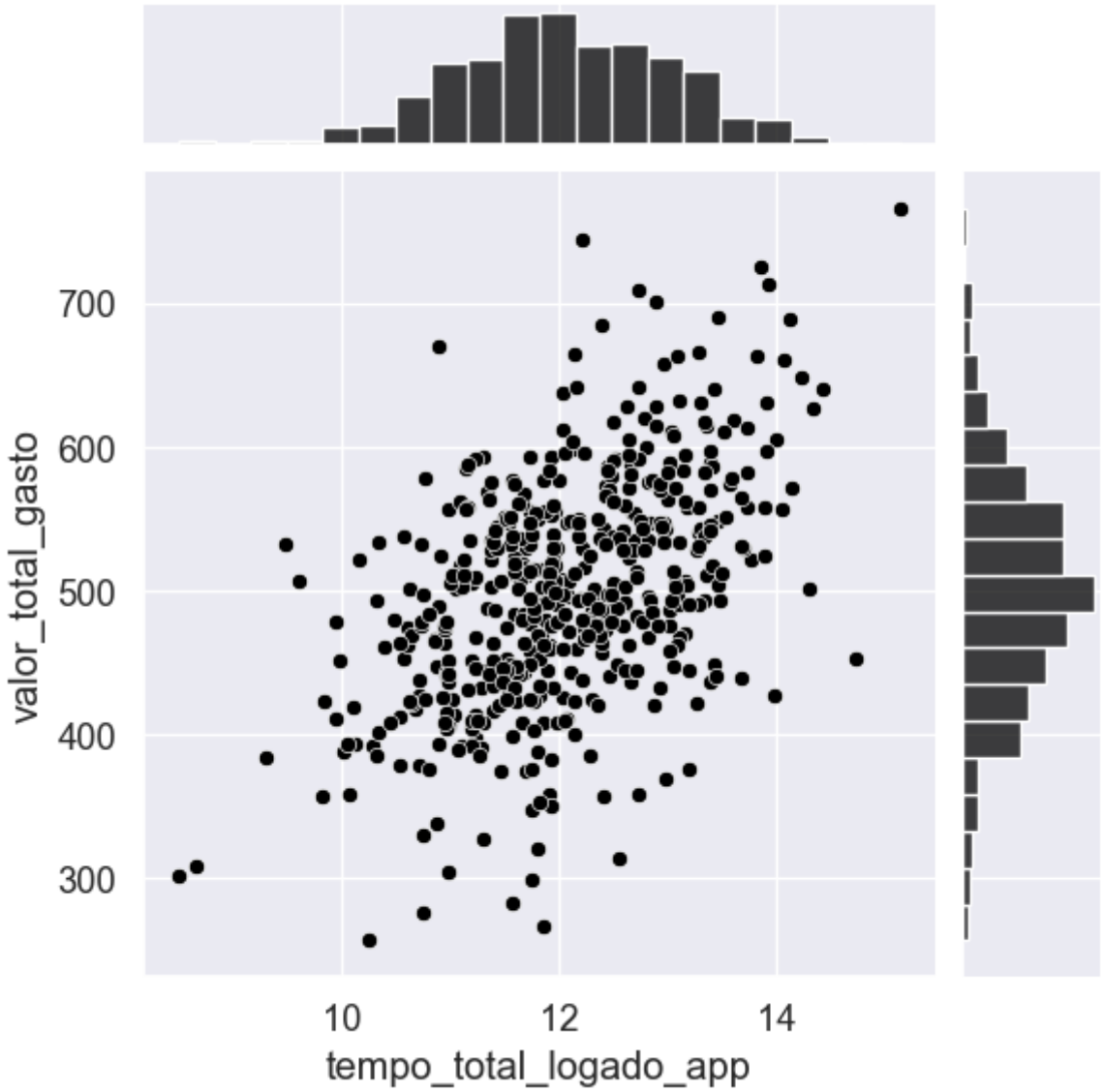
Out[9]:

	tempo_total_logado_app	valor_total_gasto
tempo_total_logado_app	1.000000	0.499328
valor_total_gasto	0.499328	1.000000

In [10]:
```
plt.figure(figsize = (18, 12))
sns.set(font_scale = 1.2)
sns.jointplot(data = dados,
              x = 'tempo_total_logado_app',
              y = 'valor_total_gasto',
              color = 'black')
```

Out[10]:
```
<seaborn.axisgrid.JointGrid at 0x12c131450>

<Figure size 1800x1200 with 0 Axes>
```



Os dados apresentam uma possível alta correlação positiva entre o tempo logado no App e o valor total gasto, ou seja, a medida que aumenta o tempo logado no App, aumenta valor total gasto.

Relação entre o Tempo de Cadastro e o Valor Total Gasto

In [11]:
```
dados[['tempo_cadastro_cliente', 'valor_total_gasto']].corr()
```

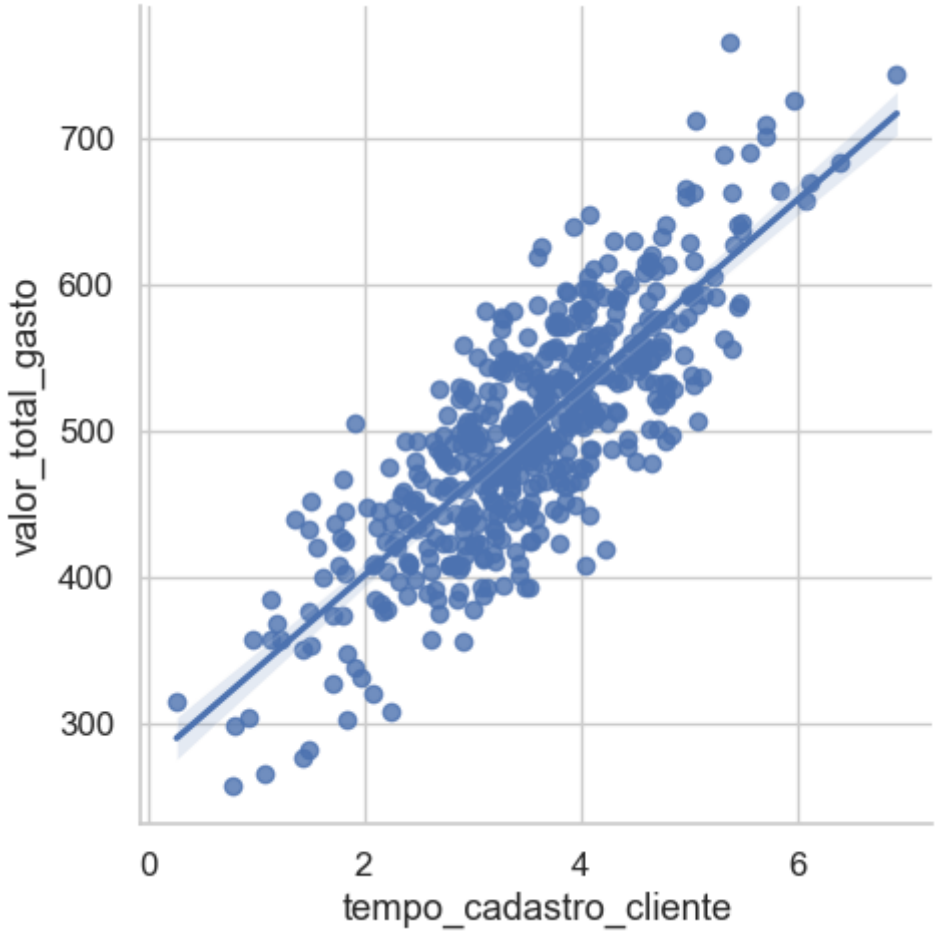
Out[11]:

	tempo_cadastro_cliente	valor_total_gasto
tempo_cadastro_cliente	1.000000	0.809084
valor_total_gasto	0.809084	1.000000

In [12]:
```
sns.set(font_scale = 1.1)
sns.set_style('whitegrid')
sns.lmplot(y = "valor_total_gasto", x = "tempo_cadastro_cliente", data = dados)
```

```
/Users/emerson/anaconda3/lib/python3.11/site-packages/seaborn/axisgrid.py:118: UserWarning: The figure l
ayout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[12]:
```
<seaborn.axisgrid.FacetGrid at 0x12c8c6450>
```



Os dados sugerem que, quanto mais tempo de cadastro tem o cliente, maior é o valor total gasto pelo cliente, ou seja, clientes com mais tempo de fidelização tendem a gastar mais.

Pré-processamento dos dados

Vamos pré-processar os dados separando as variáveis de entrada e variável alvo (valor_total_gasto), e então dividir em 70% para treinar e 30% para testar o modelo. Após, como temos dados em diferentes grandezas, vamos colocar os dados na mesma escala para não haver overfitting nas variáveis compostas por dados de maior grandeza, aplicando uma função que modifica o dado sem modificar a informação contida nele, a partir de cálculos matemáticos, onde será deslocado a média para 0 e o desvio padrão para 1 em todos os dados.

In [13]:
```
dados.columns
```

Out[13]:
```
Index(['tempo_cadastro_cliente', 'numero_medio_cliques_por_sessao',
       'tempo_total_logado_app', 'tempo_total_logado_website',
       'valor_total_gasto'],
      dtype='object')
```

In [14]:
```
# Var. de entrada
X = dados[['tempo_cadastro_cliente',
           'numero_medio_cliques_por_sessao',
           'tempo_total_logado_app',
           'tempo_total_logado_website']]
```

In [15]:
```
# Var. de saída
y = dados['valor_total_gasto']
```

In [16]:
```
# Divisão em dados de treino e teste
X_treino, X_teste, y_treino, y_teste = train_test_split(X, y, test_size = 0.3, random_state = 101)
```

In [17]:
```
len(X_treino)
```

Out[17]:
```
350
```

In [18]:
```
len(X_teste)
```

Out[18]:
```
150
```

Padronização

In [19]:
```
scaler = StandardScaler()
```

In [20]:
```
scaler.fit(X_treino)
```

Out[20]:
```
StandardScaler
StandardScaler()
```

In [21]:
```
X_treino = scaler.transform(X_treino)
```

In [22]:
```
X_teste = scaler.transform(X_teste)
```

In [23]:
```
X_treino[:5]
```

Out[23]:
```
array([[-0.21902935, -0.23735512,  0.33914084,  0.92765292],
       [ 1.8073082 ,  0.09393489,  1.05266311,  0.2388907 ],
       [-0.00962736, -0.47064535, -0.26005737,  0.81461639],
       [-0.06171807, -0.23157636, -0.19229742, -0.26198867],
       [-2.03669802, -1.54671013, -1.27813419,  1.65800995]])
```

In [24]:
```
X_teste[:5]
```

Out[24]:
```
array([[-0.74134552,  0.35042401,  1.32428694,  0.85663193],
       [ 0.51634261,  0.33850011,  1.36472848,  0.1071885 ],
       [ 1.22462847, -0.84278168,  1.70184401,  0.88678948],
       [ 0.48199668, -0.30436824, -0.65228214, -1.76956776],
       [ 1.88519232, -0.14493923,  0.70658414, -1.44006612]])
```

Serão criados 3 modelos de Regressão, o primeiro será o benchmark, utilizando a Regressão Linear na sua forma mais simples, o segundo será utilizado a Regressão Ridge e o terceiro usando a Regressão Lasso. A Regressão Ridge quanto a Regressão Lassosão técnicas de regressão regularizada usadas para lidar com problemas de overfitting e multicolinearidade em modelos de regressão linear, logo escolhemos estes modelos para testar e verificar se para este conjunto de dados e problema de negócio, podem ser mais eficiente.

Modelo 1 - Regressão Linear

In [25]:
```
# Modelo
modelo_v1 = LinearRegression()
```

In [26]:
```
# Treinamento
modelo_v1.fit(X_treino, y_treino)
```

Out[26]:
```
▾ LinearRegression

LinearRegression()
```

In [27]:
```
print('Coeficientes: \n', modelo_v1.coef_)
```

```
Coeficientes:
 [63.74220716 26.23901606 38.57185551  0.6847366 ]
```

In [28]:
```
# Coeficientes das variáveis preditoras
df_coef = pd.DataFrame(modelo_v1.coef_, X.columns, columns = ['Coeficiente'])
df_coef
```

Out[28]:

	Coeficiente
tempo_cadastro_cliente	63.742207
numero_medio_cliques_por_sessao	26.239016
tempo_total_logado_app	38.571856
tempo_total_logado_website	0.684737

Avaliação do Modelo

In [29]:
```
# Previsões com dados de teste
pred_v1 = modelo_v1.predict(X_teste)
```

In [30]:
```
# 10 primeiras prev.
pred_v1[:10]
```

Out[30]:
```
array([513.06429807, 593.96597774, 621.6550031 , 495.82353395,
       642.08919639, 615.61800045, 592.1273355 , 493.61084354,
       457.58835597, 532.03644608])
```

In [31]:
```
# Valor médio gasto pelos clientes
print('Valor máximo gasto pelos clientes é de:', dados['valor_total_gasto'].max())
print('Valor médio gasto pelos clientes é de:', dados['valor_total_gasto'].mean())
print('Valor mínimo gasto pelos clientes é de:', dados['valor_total_gasto'].min())
print('O MAE - Erro Médio Absoluto é de:', mean_absolute_error(y_teste, pred_v1))
```

```
Valor máximo gasto pelos clientes é de: 765.5184619
Valor médio gasto pelos clientes é de: 499.31403826080003
Valor mínimo gasto pelos clientes é de: 256.6705823
O MAE - Erro Médio Absoluto é de: 7.76241864577898
```

In [32]:
```
# MSE - Erro quadrático médio
mean_squared_error(y_teste, pred_v1)
```

Out[32]:
```
94.9565430843867
```

In [33]:
```
# RMSE - Raiz quadrada do erro quadrático médio
np.sqrt(mean_squared_error(y_teste, pred_v1))
```

Out[33]:
```
9.74456479707466
```

In [34]:
```
# Coeficiente R2
r2_score(y_teste, pred_v1)
```

Out[34]:
```
0.9813622791776302
```

In [35]:
```
# Variância Explicada
explained_variance_score(y_teste, pred_v1)
```

Out[35]:
```
0.9817449183428639
```

Modelo 2 com Regressão Ridge

In [37]:
```
# Modelo
modelo_v2 = Ridge(alpha = 1.0)
```

In [38]:
```
# Treinamento
modelo_v2.fit(X_treino, y_treino)
```

Out[38]:
```
Ridge
Ridge()
```

In [39]:
```
print('Coeficientes: \n', modelo_v2.coef_)
```

```
Coeficientes:
 [63.57245999 26.17198131 38.46758178  0.68013543]
```

In [40]:
```
# Coeficientes das variáveis preditoras
df_coef = pd.DataFrame(modelo_v2.coef_, X.columns, columns = ['Coeficiente'])
df_coef
```

Out[40]:

	Coeficiente
tempo_cadastro_cliente	63.572460
numero_medio_cliques_por_sessao	26.171981
tempo_total_logado_app	38.467582
tempo_total_logado_website	0.680135

In [41]:
```
# Previsões com dados de teste
pred_v2 = modelo_v2.predict(X_teste)
pred_v2[:10]
```

Out[41]:
```
array([513.02461894, 593.71284026, 621.32208364, 495.83827761,
       641.71185412, 615.30802007, 591.87884788, 493.61867235,
       457.70805605, 531.9426755 ])
```

In [42]:
```
# MAE
mean_absolute_error(y_teste, pred_v2)
```

Out[42]:
```
7.764151148981964
```

In [43]:
```
# MSE
mean_squared_error(y_teste, pred_v2)
```

Out[43]:
```
94.93731602493466
```

In [44]:
```
# RMSE
np.sqrt(mean_squared_error(y_teste, pred_v2))
```

Out[44]:
```
9.7435781941202
```

In [45]:
```
# Coeficiente R2
r2_score(y_teste, pred_v2)
```

Out[45]:
```
0.981366052994101
```

In [46]:
```
# Variância Explicada
explained_variance_score(y_teste, pred_v2)
```

Out[46]:
```
0.9817472659016085
```

Modelo 3 - Regressão LASSO

In [48]:
```
# Cria o modelo
modelo_v3 = Lasso(alpha = 1.0)
```

In [49]:
```
# Treinamento
modelo_v3.fit(X_treino, y_treino)
```

Out[49]:
```
Lasso()
```

In [50]:
```
print('Coeficientes: \n', modelo_v3.coef_)
```

```
Coeficientes:
 [62.86344076 25.18747244 37.62149243  0.        ]
```

In [51]:
```
# Coeficientes
df_coef = pd.DataFrame(modelo_v3.coef_, X.columns, columns = ['Coeficiente'])
df_coef
```

Out[51]:

	Coeficiente
tempo_cadastro_cliente	62.863441
numero_medio_cliques_por_sessao	25.187472
tempo_total_logado_app	37.621492
tempo_total_logado_website	0.000000

In [52]:
```
# Previsões com dados de teste
pred_v3 = modelo_v3.predict(X_teste)
pred_v3[:10]
```

Out[52]:
```
array([511.50216083, 591.78590214, 619.24047552, 497.55162062,
       640.89951717, 614.42803424, 590.05764493, 494.76617949,
       459.30498489, 529.64197449])
```

In [53]:
```
# MAE
mean_absolute_error(y_teste, pred_v3)
```

Out[53]:
```
7.7885046969510645
```

In [54]:
```
# MSE
mean_squared_error(y_teste, pred_v3)
```

Out[54]:
```
96.05606348970672
```

In [55]:
```
# RMSE
np.sqrt(mean_squared_error(y_teste, pred_v3))
```

Out[55]:
```
9.800819531534428
```

In [56]:
```
# Coeficiente R2
r2_score(y_teste, pred_v2)
```

Out[56]:
```
0.981366052994101
```

In [57]:
```
# Variância Explicada
explained_variance_score(y_teste, pred_v3)
```

Out[57]:
```
0.9815600649101045
```

In [59]:
```
# Coeficientes
df_coef_final = pd.DataFrame(modelo_v1.coef_, X.columns, columns = ['Coeficiente'])
df_coef_final
```

Out[59]:

	Coeficiente
tempo_cadastro_cliente	63.742207
numero_medio_cliques_por_sessao	26.239016
tempo_total_logado_app	38.571856
tempo_total_logado_website	0.684737

Considerações

A partir da construção e teste dos 3 modelos, em consideração a taxa de erro (RMSE), foi escolhido seguir para produção o modelo BenchMark (primeiro modelo), visto que apresenta uma alta performance e uma boa explicação sobre a variação entre os dados, além de ser o modelo mais simples, o que torna sua execução mais rápida e exige menor poder computacional e menores custos a empresa.

A partir do MAE podemos ver que nosso modelo escolhido preve em média R$7.76 de erro, o que é um valor muito baixo se compararmos a média de R$499,31.
Com o RMSE podemos ver que nosso modelo preve em média R$9.74 reais de erro, o que também é um valor muito baixo se compararmos com o valor médio gasto por cliente de R$499,31.
O R2 de 98% e RMSE de 9.74 demonstram uma alta eficiência do modelo, pois obteve uma alta precisão, conseguindo explicar 98% da variância entre os dados.

A partir das previsões feitas com o modelo podemos identificar que:

O aumento de 1 unidade ao tempo total logado no web site está associado ao aumento de R$ 0.68 no valor total gasto por cliente por mês. O que indica que possivelmente, não será eficiente e com grandes retornos o investimento de mais recursos no web site.

O aumento de 1 unidade ao tempo total logado na app está associado ao aumento de R$ 38.57 no valor total gasto por cliente no mês. Logo, sobre a dúvida principal entre investir no App ou Site, fica demonstrado a partir das previsões que a margem de retorno em um investimento será maior investindo no App perante ao site.

O aumento de 1 unidade ao número médio de cliques por sessão está associado ao aumento de R$ 26.24 no valor total total gasto por cliente no mês. O que indica que o cliente ao interagir em mais clicks com App, possui uma maior probabilidade de aumentar seu valor total gasto, logo investir na melhoria do App para que o cliente tenha mais opções de interação, pode aumentar o seu valor total gasto.

Porém, o maior insight gerado pelo nosso modelo, foi que o aumento de 1 unidade no tempo de cadastro do cliente está associado ao aumento de R$ 63.74 no valor total gasto por cliente no mês. Logo, clientes que possuem mais tempo fidelizados, são os que possuem um maior gasto mensal. Isso demonstra que segundo as probabilidades geradas pelo nosso modelo, dedica recursos e politicas para maior retenção e fidelização a médio e longo prazo dos clientes, tende a aumentar o valor gasto mensal por tais, logo aumentando a receita da empresa e diminuindo gastos desnecessários ou que atualmente não trariam significativos avanços as vendas, como por exemplo a baixa assosciação do uso do web site com o valor total gasto por cliente.