Hands On Modelos Matemáticos

1. **Criando uma regressão linear multivariada**
    a. **Com o dataset disponibilizado, tratamos algumas variáveis para torná-las utilizáveis no modelo.**
        i. O primeiro tratamento feito foi na variável de data de compra. Esta foi transformada na variável “Dias de referência”, que nada mais é do que a diferença em dias entre da menor data de compra existente no dataset e a data em questão.

```
#Convertendo a coluna 'Data_da_compra' de string para datetime
result_df['Data da compra'] = pd.to_datetime(result_df['Data da compra'])

#Encontrando a data de referência
data_referencia = result_df['Data da compra'].min()

#Calculando o número de dias desde a data de referência para cada data na coluna 'Data_da_compra'
result_df['Dias desde referencia'] = (result_df['Data da compra'] - data_referencia).dt.days
```

        ii. O segundo tratamento feito foi na variável “Valor”, que estava como string. Precisávamos transformá-la em float.

```
#Removendo o $ da coluna de preço a partir da criação de uma nova coluna:
split_currency = df['Valor'].str.split('$', expand=True)

# Renomeando coluna nova
split_currency.columns = ['Moeda','Valor_float']

#Derrubando coluna 'Moeda'
df_drop = split_currency.drop(columns=['Moeda'])

# Concatenando a coluna nova 'Valor_float' com dataframe antigo
result_df = pd.concat([df, df_drop], axis=1)
result_df['Valor_float'] = result_df['Valor_float'].astype(float)
```

        iii. Outra adaptação necessária foi com a variável de localização e a variável de sexo do cliente. Fizemos transformações para criar variáveis “dummy” porque não seria possível incluir num modelo matemático uma variável de string. Então ficamos com várias variáveis booleanas em forma de integer.

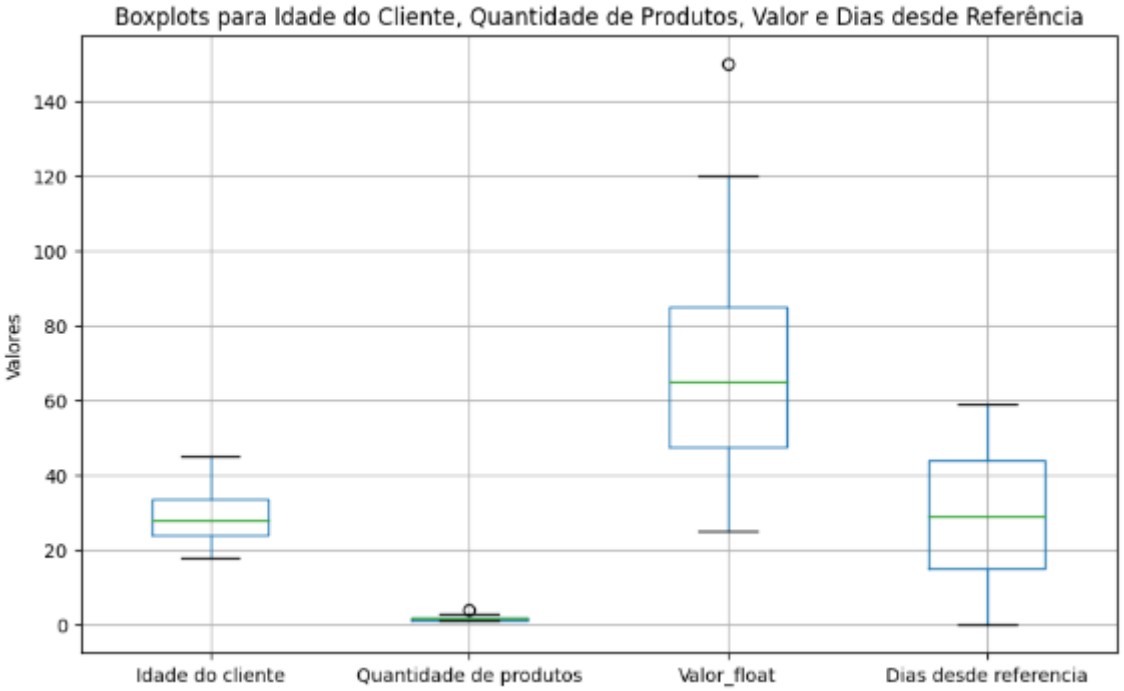
```
[24] #Criando variáveis dummy para 'Localizacao do cliente' e 'Sexo do cliente'
     result_df = pd.get_dummies(result_df, columns=['Localização do cliente', 'Sexo do cliente'], drop_first=True)
```

    b. **Avaliamos também se o dataset tinha valores nulos e/ou duplicados, e no caso não tinha.**

```
#Verificando a existência de valores nulos e linhas duplicadas, respectivamente
print(df.info())
print(df.duplicated().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 115 entries, 0 to 114
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   Sequência               115 non-null    int64 
 1   Valor                   115 non-null    object
 2   Idade do cliente        115 non-null    int64 
 3   Localização do cliente  115 non-null    object
 4   Sexo do cliente         115 non-null    object
 5   Data da compra          115 non-null    object
 6   Quantidade de produtos  115 non-null    int64 
dtypes: int64(3), object(4)
memory usage: 6.4+ KB
None
0
```

c. E então pudemos analisar se teríamos que tratar outliers.



No caso, vimos que para as variáveis numéricas que realmente teriam variações nos números, haviam poucos outliers, então não seriam realmente uma preocupação para o modelo.

d. Construindo os modelos

i. Modelo 1 (modelo sem uso de variáveis de localização):

Variáveis incluídas no modelo 1: 'Dias desde referencia', 'Quantidade de produtos', 'Idade do cliente', 'Sexo do cliente_M'

```python
#Crie um objeto de regressão linear e ajuste o modelo aos dados:
X = result_df[['Dias desde referencia', 'Quantidade de produtos', 'Idade do cliente', 'Sexo do cliente_M']]
y = result_df['Valor_float']
modelo = LinearRegression()
modelo.fit(X, y)
# Coeficientes da regressão:
coeficientes = modelo.coef_
intercepto = modelo.intercept_
# Fazer previsões:
previsoes = modelo.predict(X)
#Avaliando o desempenho do seu modelo, coeficiente de determinação (R²) e erro quadrático médio (MSE):
r2 = r2_score(y, previsoes)
mse = mean_squared_error(y, previsoes)
rmse = np.sqrt(mse)

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

mape = mean_absolute_percentage_error(y, previsoes)


print(f"Coeficiente de Determinação (R²): {r2}")
print(f"Erro Quadrático Médio (MSE): {mse}")
print(f"Erro Quadrático Médio (RMSE): {rmse}")
print(f"Erro Percentual Absoluto Médio (MAPE): {mape:.2f}%")
```

```
Coeficiente de Determinação (R²): 0.15595608876707967
Erro Quadrático Médio (MSE): 624.3531623904613
Erro Quadrático Médio (RMSE): 24.98705989888489
Erro Percentual Absoluto Médio (MAPE): 34.80%
```

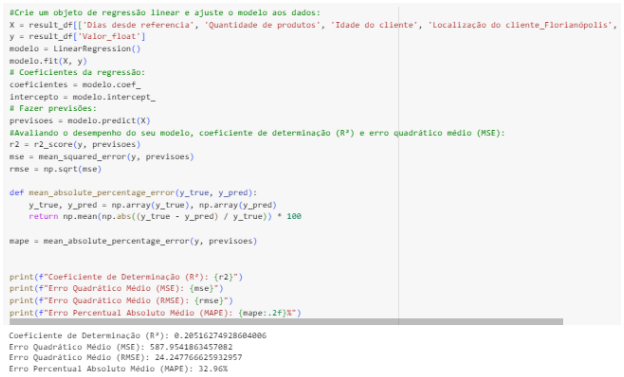
Métricas de avaliação do modelo 1:

Coeficiente de Determinação (R²): 0.15595608876707967 Erro Quadrático Médio (MSE): 624.3531623904613 Erro Quadrático Médio (RMSE): 24.98705989888489 Erro Percentual Absoluto Médio (MAPE): 34.80%

ii. Construindo o modelo 2:

Variáveis incluídas no modelo 2:

'Dias desde referencia', 'Quantidade de produtos', 'Idade do cliente', 'Localização do cliente_Florianópolis', 'Localização do cliente_Fortaleza', 'Localização do cliente_Brasília', 'Localização do cliente_Curitiba', 'Localização do cliente_Belém', 'Localização do cliente_Manaus', 'Localização do cliente_Goiânia', 'Localização do cliente_São Paulo', 'Localização do cliente_Salvador', 'Localização do cliente_Rio de Janeiro', 'Localização do cliente_Porto Alegre', 'Localização do cliente_Recife', 'Sexo do cliente_M'



Percebe-se o resultado do modelo 2:

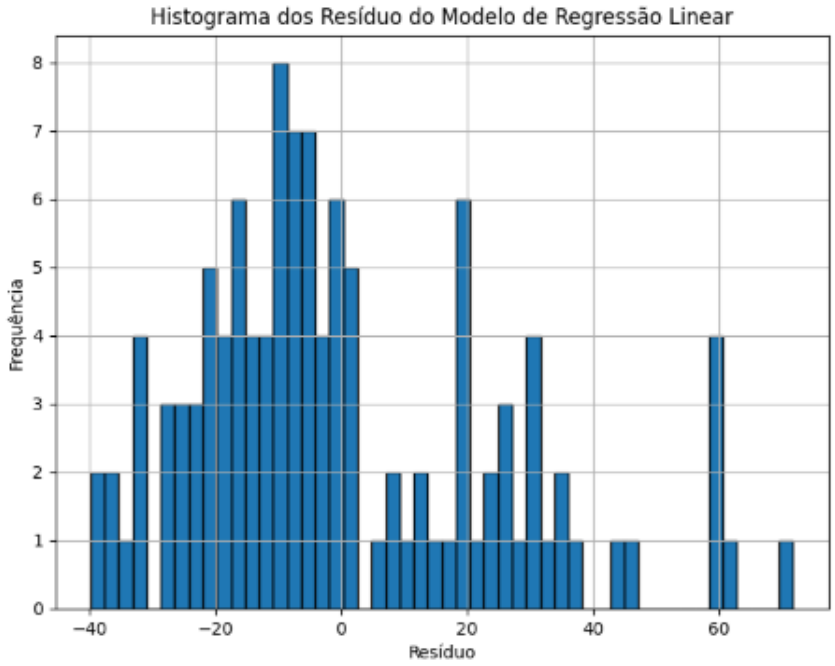
Coeficiente de Determinação (R^2): 0.20516274928604006

Erro Quadrático Médio (MSE): 587.9541863457082

Erro Quadrático Médio (RMSE): 24.247766625932957

Erro Percentual Absoluto Médio (MAPE): 32.96%

Como esse foi o modelo que ficou com um erro menor, apesar de ainda ser muito falho para explicar a variação do valor de venda, construímos um histograma dos valores residuais:



iii. Construindo o modelo 3:

Variáveis incluídas no modelo 3: 'Quantidade de produtos', 'Idade do cliente', 'Sexo do cliente_M', 'Localização do cliente_Curitiba','Localização do cliente_Rio de Janeiro'

```
#Crie um objeto de regressão linear e ajuste o modelo aos dados:
X = result_df[['Quantidade de produtos', 'Idade do cliente', 'Sexo do cliente_M', 'Localização do cliente_Curitiba','Localização do cliente_Rio de Janeiro']]
y = result_df['Valor_float']
modelo = LinearRegression()
modelo.fit(X, y)
# Coeficientes da regressão:
coeficientes = modelo.coef_
intercepto = modelo.intercept_
# Fazer previsões:
previsoes = modelo.predict(X)
#Avaliando o desempenho do seu modelo, coeficiente de determinação (R²) e erro quadrático médio (MSE):
r2 = r2_score(y, previsoes)
mse = mean_squared_error(y, previsoes)
rmse = np.sqrt(mse)

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

mape = mean_absolute_percentage_error(y, previsoes)


print(f"Coeficiente de Determinação (R²): {r2}")
print(f"Erro Quadrático Médio (MSE): {mse}")
print(f"Erro Quadrático Médio (RMSE): {rmse}")
print(f"Erro Percentual Absoluto Médio (MAPE): {mape:.2f}%")
```

```
Coeficiente de Determinação (R²): 0.1626623004320361
Erro Quadrático Médio (MSE): 619.3924673306993
Erro Quadrático Médio (RMSE): 24.88759665638085
Erro Percentual Absoluto Médio (MAPE): 34.13%
```

Resultado do modelo 3:

Coeficiente de Determinação (R^2): 0.1626623004320361
Erro Quadrático Médio (MSE): 619.3924673306993
Erro Quadrático Médio (RMSE): 24.88759665638085
Erro Percentual Absoluto Médio (MAPE): 34.13%

e. Conclusão a respeito dos modelos construídos:

Após o teste de uma série de combinações das variáveis disponibilizadas para tentar prever o valor da venda, percebe-se que, mesmo no melhor dos casos, as informações disponibilizadas não são capazes de dar uma estimativa que se aproxime do real valor de venda.

Ao analisar as métricas de avaliação do 2º modelo criado pelo grupo, a partir do coeficiente de determinação, vemos que apenas 20% da variação dos dados pode ser explicada pelo modelo. E ao analisar o RMSE, é possível ver que, em média, os valores de venda previstos pelo modelo estão desviando aproximadamente 24 reais do valor de venda real.

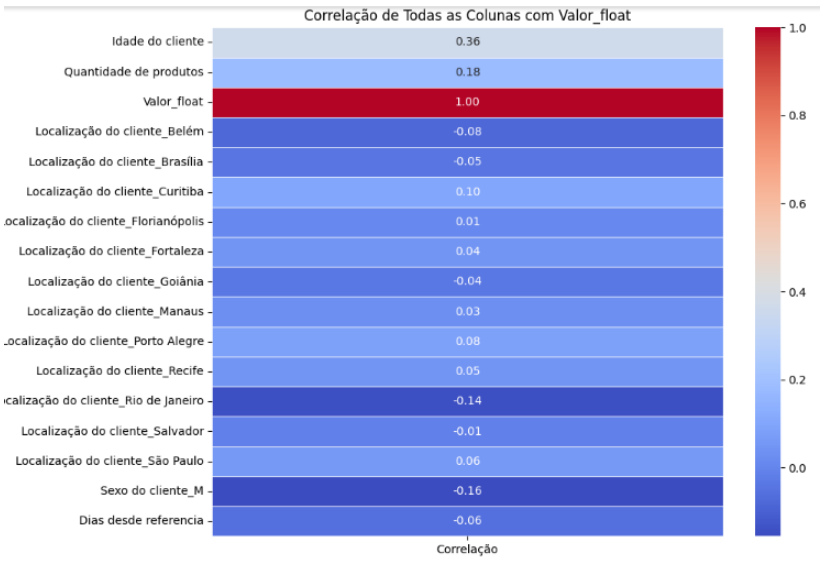
Já usando o MAPE, podemos inferir que, em média, as previsões do modelo desviam cerca de 32.96% dos valores reais. Com isso, cabe dizer que o modelo é pouco capaz de explicar a variabilidade da variável valor de venda.

Analisando também o histograma dos resíduos, seu formato indica que o modelo comete erros aleatórios e não prevê sistematicamente para mais ou para menos nenhum intervalo de valores de destino.

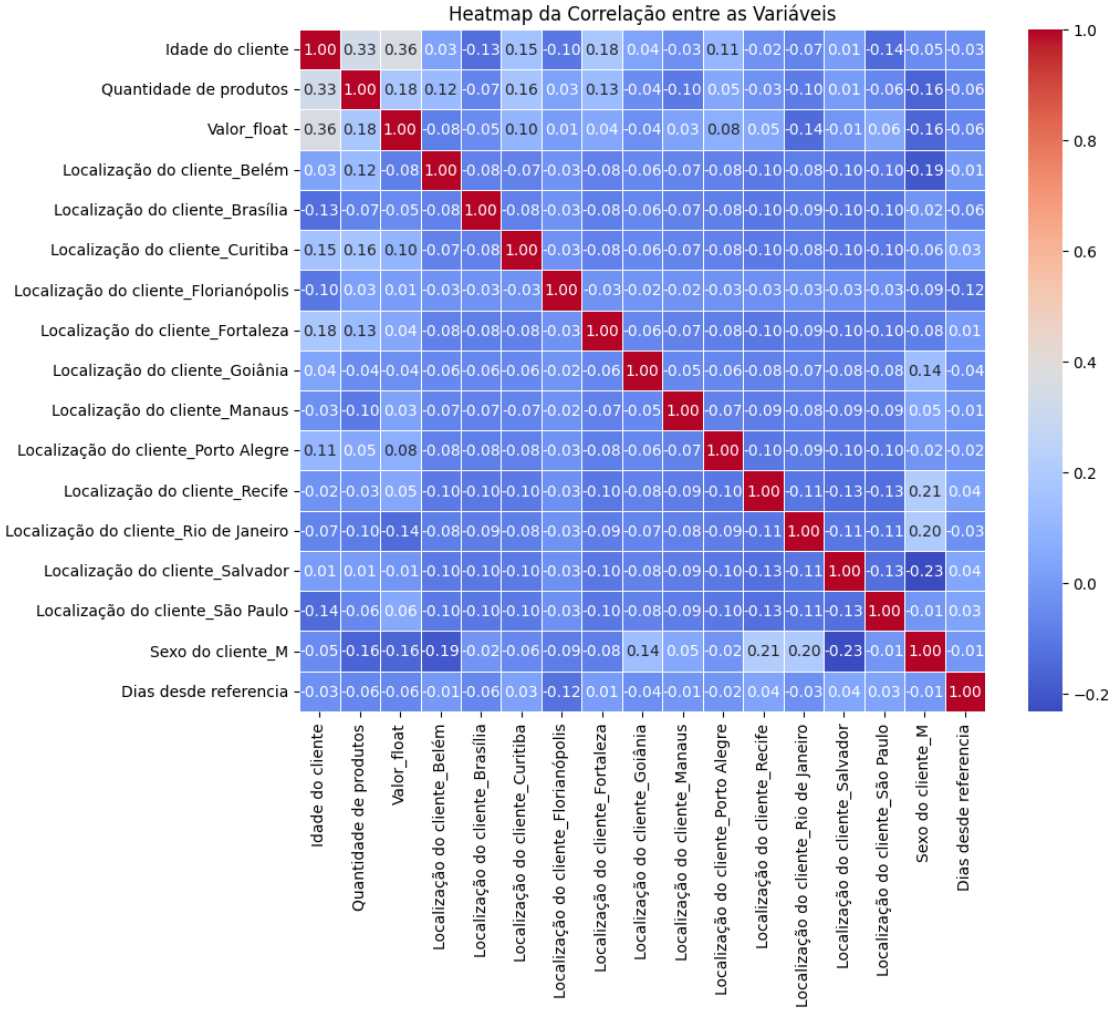
Sendo assim, conclui-se que essa amostra de dados não foi suficiente para tornar possível a construção de um modelo com uma precisão adequada para a previsão do valor de venda.

2. Matriz de correlação de variáveis

Correlação de pearson entre o valor de venda e as outras variáveis:



Matriz de correlação de pearson de todas as variáveis:



3. Descobertas

Vemos que a variável que tem maior correlação positiva com o valor da venda é “idade do cliente”, ou seja, quanto maior a idade do cliente, maior o valor da venda. Entretanto, a correlação é de 0.33, que ainda é um pouco fraca.

Também cabe citar que, pela matriz de correlação, vemos que a quantidade de produtos também tem uma correlação um pouco mais chamativa que a maioria das variáveis, sendo de 0.18. Isso significa que um maior número de itens numa venda pode ser um indício de que o valor de venda será maior. Essa variável precisa ser analisada com cuidado porque os tipos e preços de produtos vendidos no e-commerce influenciam em como a quantidade de produtos irá se relacionar com o valor de venda. Caso sejam disponibilizados produtos com menor preço, por exemplo, produtos com valores abaixo de R$10,00, e produtos com preços maiores, como televisões, com valores acima de R$1.000,00, a correlação de quantidade de itens com valor de venda realmente deve ser mais baixa. Entretanto, se a Melhores Compras tivesse apenas produtos com menor variação de valor, observariamos uma correlação positiva mais forte entre essas variáveis.

Outra descoberta é a de que a correlação entre valor de venda e o sexo do comprador ser masculino é negativa e fraca. Sendo assim, há um pequeno indício de que, na Melhores Compras, as mulheres fazem compras de maior valor.

Ainda da matriz de correlação, vemos que a variável localização Rio de Janeiro tem correlação negativa de -0.14, significando que compras de clientes do Rio de Janeiro tendem a ter valores menores. O contrário se observa com clientes de Curitiba, cuja correlação com o valor de venda é +0.10. Nesse caso, clientes de Curitiba tendem a ter compras com valores um pouco mais elevados.

4. Sugestões para os próximos passos

Como sugestão para os próximos passos para a modelagem na Melhores Compras, seria interessante estudar o comportamento de novas variáveis em relação ao valor de venda, por exemplo, se o tempo que o cliente ficou na página antes da compra tem relação com uma compra de valor maior, ou clientes com maior tempo de cadastro fazem compras de valores maiores - pensando numa hipótese de que talvez depois de algumas compras bem sucedidas o cliente passa a confiar mais na plataforma.

Vejo que há também oportunidade para o uso de outras modelagens matemáticas que poderiam beneficiar a Melhores Compras, como o uso de modelos de regressão logística para entender o que influencia uma compra, pensando em avaliar se, por exemplo, produtos com

mais fotos ou mais avaliações são mais comprados do que outros, buscando melhorar a maneira como os produtos são ofertados.