T2_single_multi-linear-regression_VHA

June 3, 2024

1 Introduction to Linear Regression

Linear regression is a fundamental statistical technique used to model the relationship between a dependent variable and one or more independent variables. It aims to predict the value of the dependent variable based on the values of the independent variables by fitting a linear equation to observed data.

1.0.1 What is Linear Regression?

Linear regression models the relationship between variables by fitting a linear equation to the observed data. The simplest form is:

[y = _0 + _1 x]

Where: - (y) is the dependent variable. - (x) is the independent variable. - (_0) is the y-intercept (constant term). - (_1) is the slope (coefficient) of the independent variable.

The goal is to find the values of (_0) and (_1) that minimize the difference between the predicted values and the actual values of the dependent variable.

1.0.2 Types of Linear Regression

Linear regression can be categorized into two main types:

1. **Simple Linear Regression**:
   - Involves a single independent variable.
   - The relationship between the dependent variable and the independent variable is modeled as a straight line.
2. **Multiple Linear Regression**:
   - Involves two or more independent variables.
   - The relationship between the dependent variable and the independent variables is modeled as a linear combination of the predictors.

1.0.3 Simple Linear Regression

Simple linear regression models the relationship between a single independent variable and the dependent variable. The model equation is:

[y = _0 + _1 x]

Example: Predicting the price of a house based on its size.

1.0.4 Multiple Linear Regression

Multiple linear regression models the relationship between two or more independent variables and the dependent variable. The model equation is:

[y = _0 + _1 x_1 + _2 x_2 + ...+ _n x_n]

Example: Predicting the price of a house based on its size, number of bedrooms, and age.

2 Simple Linear Regression

[30]:
```
df = pd.read_csv('placement.csv')
```

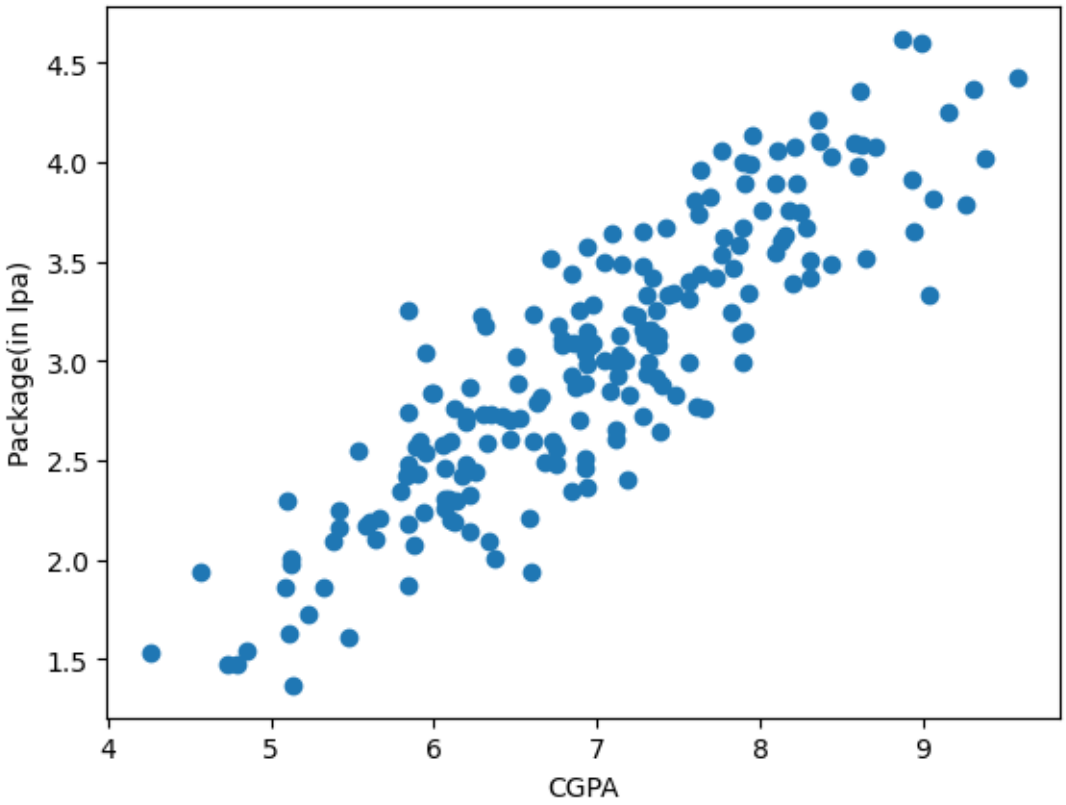
[31]:
```
df.head()
```

[31]:

	cgpa	package
0	6.89	3.26
1	5.12	1.98
2	7.82	3.25
3	7.42	3.67
4	6.94	3.57

[32]:
```
plt.scatter(df['cgpa'],df['package'])
plt.xlabel('CGPA')
plt.ylabel('Package(in lpa)')
```

[32]:
```
Text(0, 0.5, 'Package(in lpa)')
```



[33]:
```
X = df.iloc[:,0:1]
y = df.iloc[:,-1]
```

[34]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.
 ↪2,random_state=2)
```

[35]:
```
from sklearn.linear_model import LinearRegression
```

[36]:
```
lr = LinearRegression()
```

[37]:
```
lr.fit(X_train,y_train)
```

[37]:
```
LinearRegression()
```

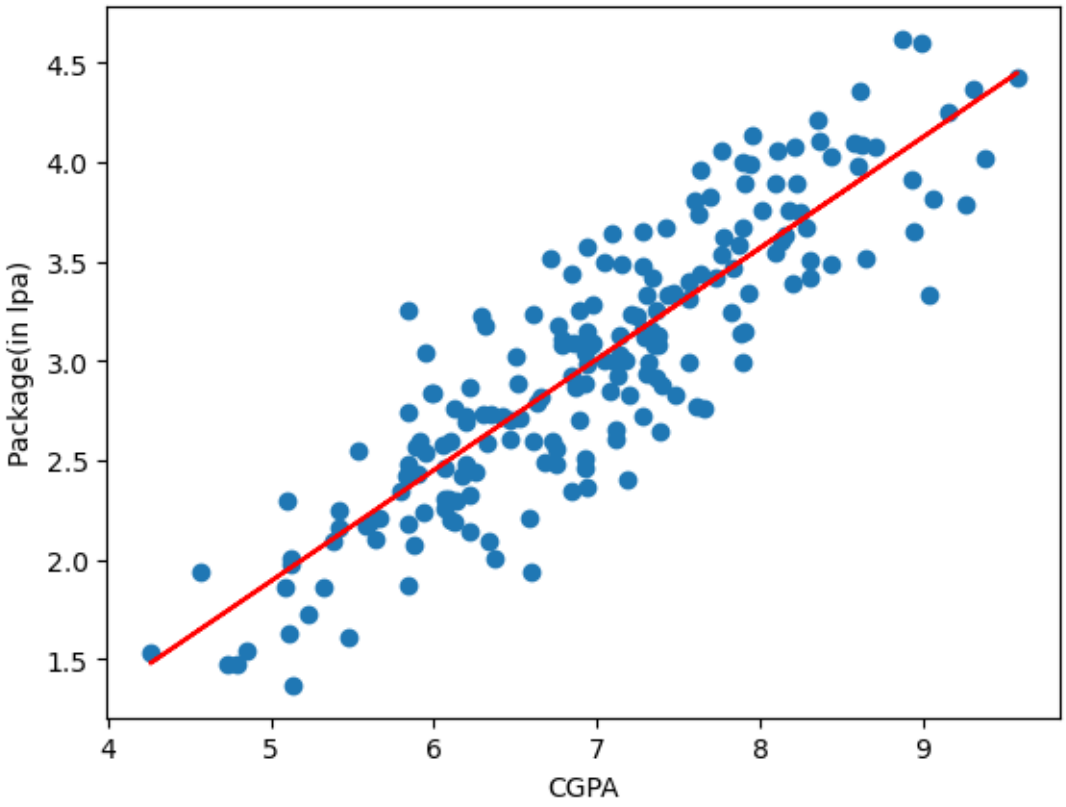
[38]:
```
lr.predict(X_test.iloc[0].values.reshape(1,1))
```

```
C:\Users\VISHAL\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
does not have valid feature names, but LinearRegression was fitted with feature
names
  warnings.warn(
```

[38]:
```
array([3.89111601])
```

[39]:
```
plt.scatter(df['cgpa'],df['package'])
plt.plot(X_train,lr.predict(X_train),color='red')
plt.xlabel('CGPA')
plt.ylabel('Package(in lpa)')
```

[39]:
```
Text(0, 0.5, 'Package(in lpa)')
```



[40]:
```
m = lr.coef_
```

[41]:
```
b = lr.intercept_
```

[42]:
```
# y = mx + b

m * 8.58 + b
```

[42]:
```
array([3.89111601])
```

3 Multiple Linear Regression¶

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
df = pd.read_csv('multiple_linear_regression_dataset.csv')
df
```

[2]:

	age	experience	income
0	25	1	30450
1	30	3	35670
2	47	2	31580
3	32	5	40130
4	43	10	47830
5	51	7	41630
6	28	5	41340
7	33	4	37650
8	37	5	40250
9	39	8	45150
10	29	1	27840
11	47	9	46110
12	54	5	36720
13	51	4	34800
14	44	12	51300
15	41	6	38900
16	58	17	63600
17	23	1	30870
18	44	9	44190
19	37	10	48700

[3]:
```
df.shape
```

[3]:
```
(20, 3)
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   age         20 non-null     int64
 1   experience  20 non-null     int64
 2   income      20 non-null     int64
dtypes: int64(3)
memory usage: 612.0 bytes
```

[5]:
```
df.head()
```

[5]:

	age	experience	income
0	25	1	30450
1	30	3	35670
2	47	2	31580
3	32	5	40130
4	43	10	47830

[6]:
```
df.describe()
```

[6]:

	age	experience	income
count	20.000000	20.000000	20.000000
mean	39.650000	6.200000	40735.500000
std	10.027725	4.124382	8439.797625
min	23.000000	1.000000	27840.000000
25%	31.500000	3.750000	35452.500000
50%	40.000000	5.000000	40190.000000
75%	47.000000	9.000000	45390.000000
max	58.000000	17.000000	63600.000000

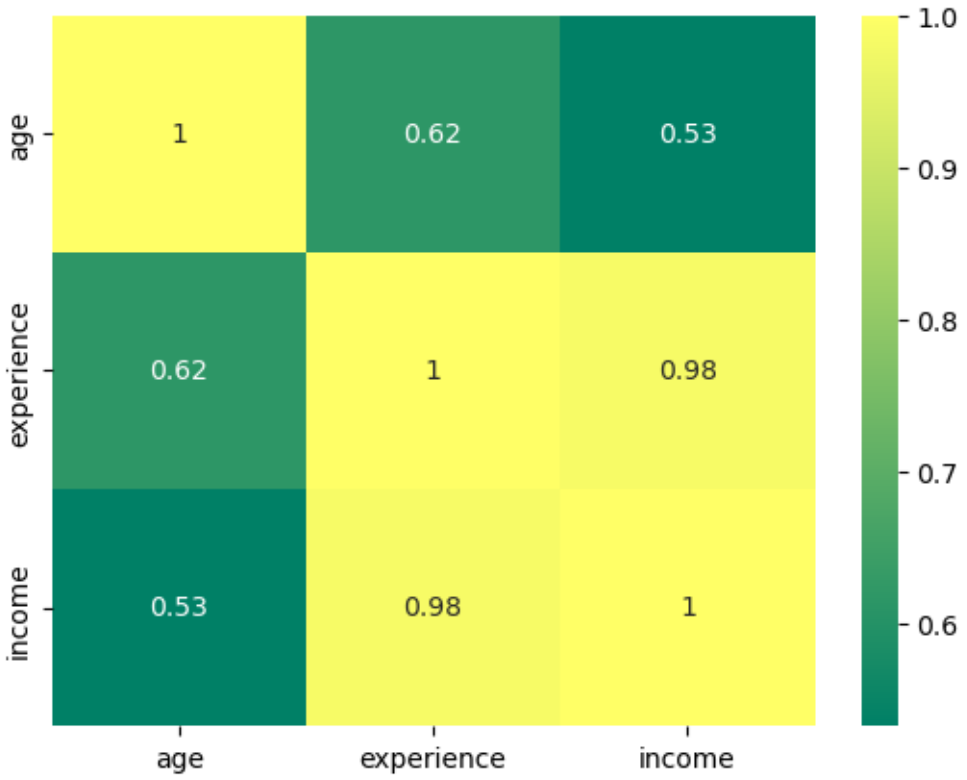
[7]:
```
df.corr()
```

[7]:

	age	experience	income
age	1.000000	0.615165	0.532204
experience	0.615165	1.000000	0.984227
income	0.532204	0.984227	1.000000

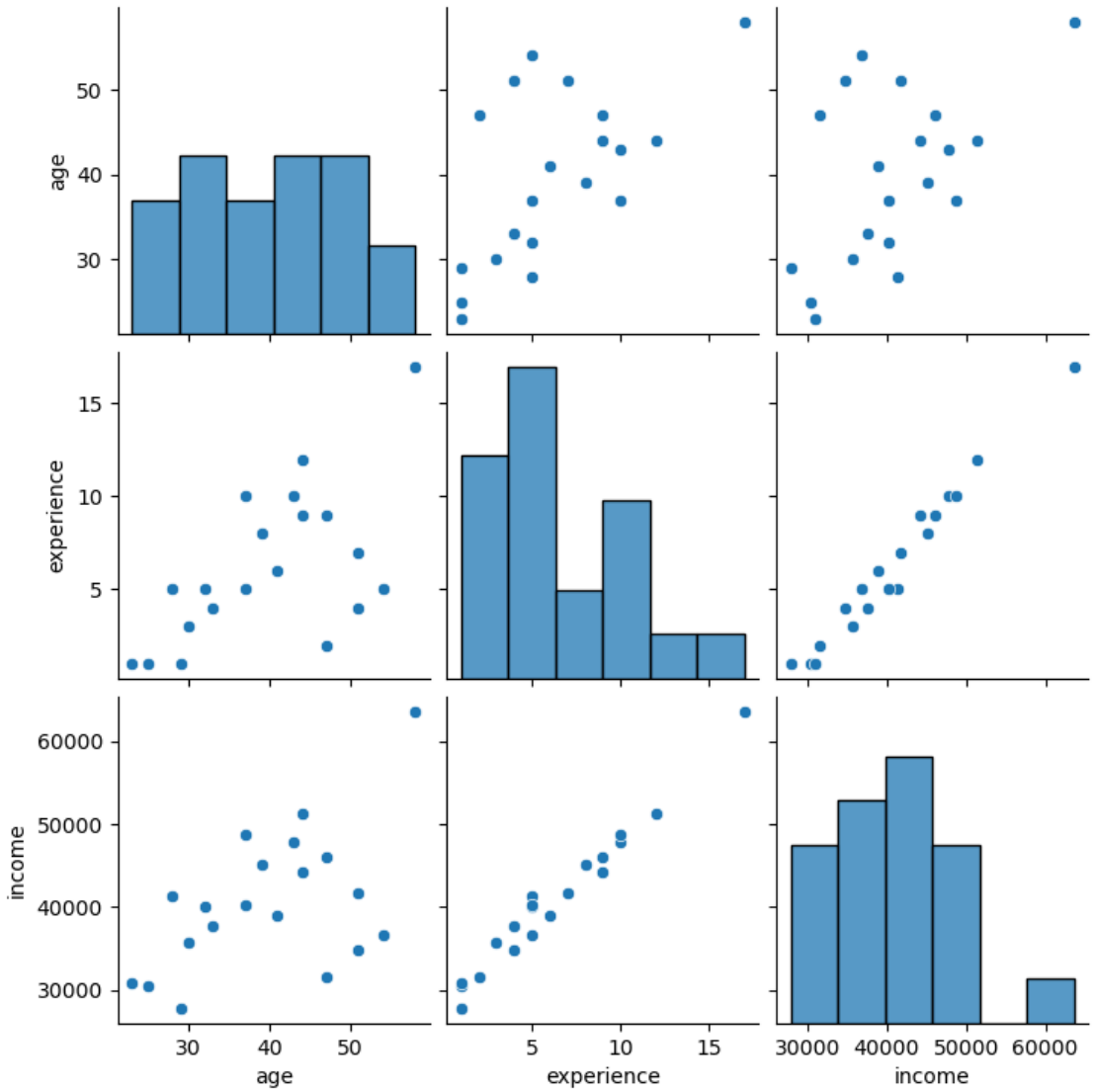
[8]:
```
sns.heatmap(df.corr(), annot=True,cmap='summer')
```

[8]:
```
<Axes: >
```



[9]:
```
sns.pairplot(df)
```

[9]: <seaborn.axisgrid.PairGrid at 0x265495dcbd0>



Defining X and y

[10]:
```
X = df.drop('income',axis=1)
y = df['income']
```

[11]:
```
X.shape
```

[11]: (20, 2)

[12]:
```
y.shape
```

[12]: (20,)

training and testing the data

[13]:
```
from sklearn.model_selection import train_test_split
```

[14]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, random_state=11,␣
↪test_size=0.2)
```

[15]:
```
print(X_train.shape)
print(X_test.shape)
```

```
(16, 2)
(4, 2)
```

applying linear regression model

[16]:
```
from sklearn.linear_model import LinearRegression
```

[17]:
```
lr = LinearRegression()
```

[18]:
```
lr.fit(X_train,y_train)
```

[18]:
```
LinearRegression()
```

[19]:
```
m = lr.coef_
m
```

[19]:
```
array([ -82.58668974, 2105.59814421])
```

[20]:
```
c = lr.intercept_
c
```

[20]:
```
30906.606569790496
```

[21]:
```
y_pred = lr.predict(X_test)
```

[22]:
```
y_pred
```

[22]:
```
array([38791.82321903, 41433.87240231, 30617.19071143, 39122.169978  ])
```

Evaluating the scores

[23]:
```
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
```

[24]:
```
r2_score(y_test,y_pred)
```

[24]:
```
0.8902944268799456
```

[25]:
```
mean_absolute_error(y_test,y_pred)
```

[25]:
```
1632.3312780201686
```

[26]:
```
mean_squared_error(y_test,y_pred)
```

[26]: 3615185.3464510385

Calculating the income when the age is 25 and experience is 2 years

[27]:
```
y = m * 23 + m * 2 + c
y
```

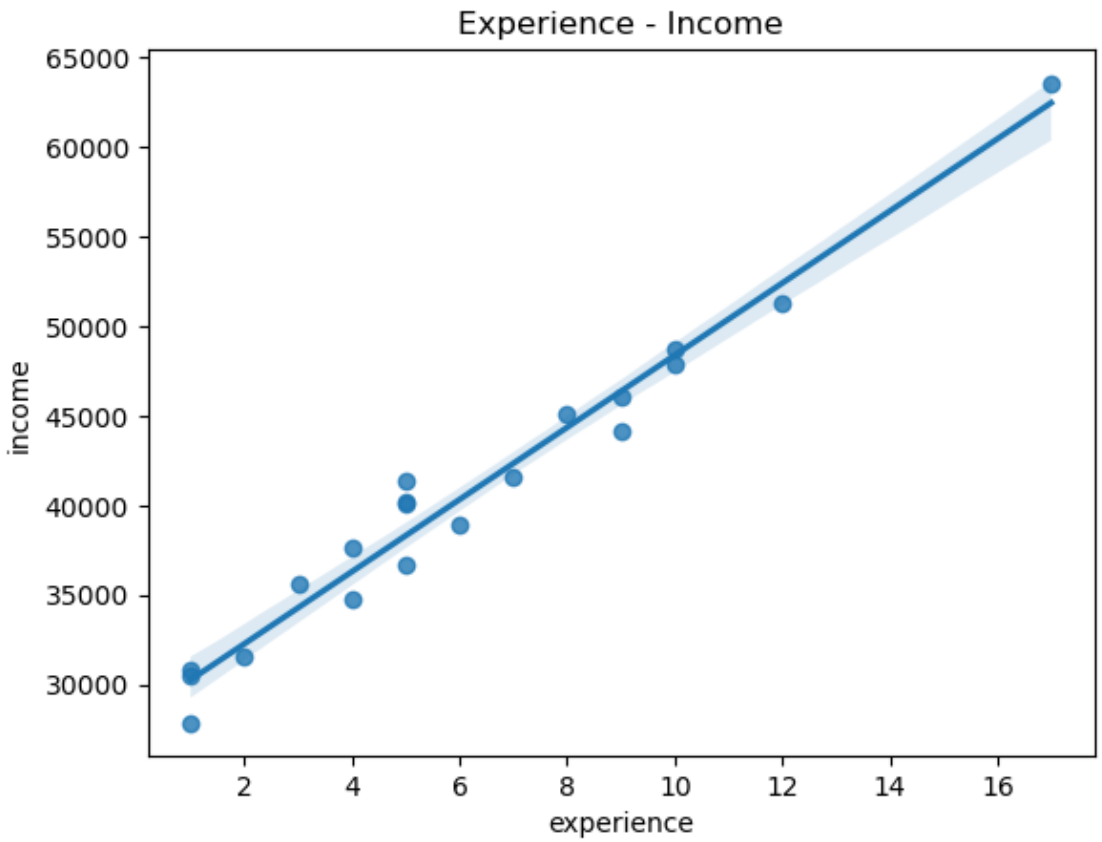
[27]: array([28841.93932619, 83546.560175])

[28]:
```
lr.predict([[23,2]])
```

```
C:\Users\VISHAL\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
does not have valid feature names, but LinearRegression was fitted with feature
names
  warnings.warn(
```

[28]: array([33218.3089941])

[29]:
```
plt.title("Experience - Income")
sns.regplot(x=df.experience,y=df.income)
plt.show()
```



4 Intuition

Linear regression tries to fit a straight line (in simple linear regression) or a hyperplane (in multiple linear regression) that best describes the relationship between the independent variables and the dependent variable. The best fit is achieved by minimizing the sum of squared residuals, which are the differences between the observed and predicted values.

4.0.1 Overfitting and Underfitting

- **Overfitting**: Occurs when the model is too complex (e.g., includes too many predictors), capturing the noise in the training data rather than the underlying trend. This leads to poor generalization to new data.
- **Underfitting**: Occurs when the model is too simple (e.g., ignores relevant predictors), failing to capture the underlying trend in the data. This results in poor performance on both training and new data.

4.0.2 Limitations of Linear Regression

1. **Linearity Assumption**: Assumes a linear relationship between the independent and dependent variables, which may not always hold true.
2. **Outliers**: Sensitive to outliers, which can disproportionately affect the model.
3. **Multicollinearity**: In multiple linear regression, high correlation between independent variables can destabilize the model and make it difficult to interpret.
4. **Homoscedasticity**: Assumes constant variance of errors across all levels of the independent variables, which may not be true.
5. **Independence**: Assumes that the observations are independent of each other.

4.0.3 Outro

Linear regression is a powerful and widely used statistical technique for modeling relationships between variables. It is simple to implement and interpret, making it a valuable tool for many applications. However, it comes with several assumptions and limitations that must be carefully considered. By understanding and addressing these limitations, one can effectively apply linear regression to a variety of predictive modeling problems.

5 Linear regression, while a powerful and widely-used tool, relies on several key assumptions to ensure the validity and reliability of its results. These assumptions are crucial for the model to provide unbiased and accurate predictions. Here are the primary assumptions of linear regression:

5.0.1 1. Linearity

The relationship between the independent variables (predictors) and the dependent variable (outcome) is linear. This means that the change in the dependent variable is proportional to the change in the independent variables. Mathematically, this is expressed as:

[y = _0 + _1 x_1 + _2 x_2 + ...+ _n x_n +]

where ( ) represents the error term.

5.0.2 2. Independence

The observations are independent of each other. This means that the value of the dependent variable for one observation is not influenced by the value of the dependent variable for another observation.

5.0.3 3. Homoscedasticity

The variance of the error terms (residuals) is constant across all levels of the independent variables. This implies that the spread of the residuals should be roughly the same throughout the range of the predictors.

5.0.4 4. Normality of Residuals

The error terms are normally distributed. This assumption is particularly important for hypothesis testing (e.g., t-tests for coefficients) and constructing confidence intervals. If the residuals are not normally distributed, the estimates of the coefficients may not be reliable.

5.0.5 5. No Perfect Multicollinearity

There should be no perfect multicollinearity among the independent variables. Multicollinearity occurs when two or more predictors are highly correlated, making it difficult to isolate the individual effect of each predictor on the dependent variable.

5.0.6 6. No Autocorrelation

The residuals (errors) should not be correlated with each other. This is especially relevant for time series data where residuals at one time point might be correlated with residuals at another time point. Autocorrelation can lead to underestimated standard errors and overconfident statistical tests.

5.0.7 Checking and Addressing Assumptions

1. **Linearity**: Scatter plots and residual plots can help check for linearity. If the relationship is not linear, transformations of the variables (e.g., logarithmic, polynomial) might be necessary.
2. **Independence**: Study design and data collection methods should ensure independence. In some cases, time series analysis or hierarchical models might be more appropriate.
3. **Homoscedasticity**: Residual plots (residuals vs. predicted values) can be used to check for homoscedasticity. If heteroscedasticity is present, weighted least squares regression or transforming the dependent variable might help.
4. **Normality of Residuals**: Q-Q plots (quantile-quantile plots) and statistical tests (e.g., Shapiro-Wilk test) can check for normality. If residuals are not normally distributed, transformations or robust regression techniques might be applied.
5. **Multicollinearity**: Variance Inflation Factor (VIF) and correlation matrices can detect multicollinearity. If multicollinearity is high, removing one of the correlated predictors or combining them into a single predictor might be necessary.

6. **No Autocorrelation**: The Durbin-Watson test can detect autocorrelation. If autocorrelation is present, time series models or adding lagged variables might be appropriate.

5.0.8 Conclusion

Understanding and validating these assumptions is crucial for the proper application of linear regression. Violations of these assumptions can lead to biased, inefficient, and inconsistent estimates, undermining the validity of the model's conclusions. By carefully checking and addressing these assumptions, one can ensure more reliable and accurate results from linear regression analysis.