**2311cs02068-day25**

**Using the same data set of Civil\_Engineering\_Regression\_Dataset.csv**

**Part 5: Model Interpretation & Conclusion**

1. Summarize the key takeaways from your regression models.
2. How can construction companies use regression analysis to estimate costs more effectively?
3. What limitations did you encounter in this analysis?
4. If you were to improve this model, what additional variables might you consider?
5. How does regression analysis in civil engineering contribute to cost-effective planning?
6. Provide a conclusion on the **role of data science in optimizing construction project costs**.

Code:

import pandas as pd

import numpy as np

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import r2\_score, mean\_squared\_error

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

# Load dataset

try:

df = pd.read\_csv("Civil\_Engineering\_Regression\_Dataset.csv")

print("CSV file loaded successfully.\n")

except FileNotFoundError:

print("Error: CSV file not found. Check the file path.")

exit()

# Ensure correct column names

required\_columns = ["Building Height", "Material Quality", "Labor Cost", "Concrete Strength", "Foundation Depth", "Construction Cost"]

for col in required\_columns:

if col not in df.columns:

print(f"Error: Column '{col}' is missing. Check your dataset headers.")

exit()

# Define independent (X) and dependent (y) variables

X\_multi = df[["Building Height", "Material Quality", "Labor Cost", "Concrete Strength", "Foundation Depth"]]

y = df["Construction Cost"]

# Handle missing or non-numeric values

X\_multi = X\_multi.apply(pd.to\_numeric, errors="coerce")

y = pd.to\_numeric(y, errors="coerce")

df.dropna(inplace=True)

# Fit Simple Linear Regression (Building Height only)

X\_simple = df[["Building Height"]]

simple\_model = LinearRegression()

simple\_model.fit(X\_simple, y)

r2\_simple = r2\_score(y, simple\_model.predict(X\_simple))

# Fit Multiple Linear Regression

multi\_model = LinearRegression()

multi\_model.fit(X\_multi, y)

r2\_multi = r2\_score(y, multi\_model.predict(X\_multi))

# Adjusted R-squared Calculation

n = len(y) # Number of observations

p = X\_multi.shape[1] # Number of predictors

adjusted\_r2\_multi = 1 - ((1 - r2\_multi) \* (n - 1) / (n - p - 1))

# Calculate VIF for multicollinearity detection

vif\_data = pd.DataFrame()

vif\_data["Feature"] = X\_multi.columns

vif\_data["VIF"] = [variance\_inflation\_factor(X\_multi.values, i) for i in range(X\_multi.shape[1])]

# Identify most influential variable

coefficients = multi\_model.coef\_

most\_influential\_var = X\_multi.columns[np.argmax(abs(coefficients))]

# Print Summary Report

print("\nRegression Model Summary:")

print(f"Simple Linear Regression R-squared: {r2\_simple:.4f}")

print(f"Multiple Linear Regression R-squared: {r2\_multi:.4f}")

print(f"Multiple Linear Regression Adjusted R-squared: {adjusted\_r2\_multi:.4f}")

print("\nKey Takeaways:")

if r2\_multi > r2\_simple:

print("Multiple Linear Regression explains more variance in construction cost.")

else:

print("Additional variables do not significantly improve the model.")

print(f"Most Influential Variable: {most\_influential\_var} (Coefficient: {multi\_model.coef\_[np.argmax(abs(coefficients))]:.2f})")

# VIF Interpretation

print("\nMulticollinearity Analysis (VIF):")

print(vif\_data)

high\_vif\_features = vif\_data[vif\_data["VIF"] > 10]["Feature"].tolist()

if high\_vif\_features:

print(f"High multicollinearity detected in {high\_vif\_features}. Consider removing or transforming variables.")

else:

print("No significant multicollinearity detected.")

print("\nBusiness Applications of Regression Analysis:")

print("1. Estimate project costs based on key factors.")

print("2. Optimize material selection and labor costs.")

print("3. Reduce cost overruns by predicting expenses early.")

print("\nLimitations of This Analysis:")

print("1. Assumes a linear relationship between variables.")

print("2. Missing key factors like location, land cost, and economic trends.")

print("3. Potential multicollinearity issues affecting coefficient interpretations.")

print("\nWays to Improve the Model:")

print("1. Include variables such as land cost, inflation, and project complexity.")

print("2. Consider advanced regression techniques like Ridge/Lasso regression.")

print("3. Collect more data to enhance model accuracy.")

print("\nConclusion:")

print("Regression analysis helps construction firms make data-driven cost estimates.")

print("Adding more predictors improves accuracy, but multicollinearity must be managed.")

print("Data science optimizes budgeting, forecasting, and cost-effective planning.")