In [215…
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import statsmodels.api as sm
```

In [223…
```python
file_path = "C:/Users/gundr/Downloads/Civil_Engineering_Regression_Dataset.csv"
df = pd.read_csv(file_path)
```

In [225…
```python
X = df[['Building_Height', 'Material_Quality_Index', 'Labor_Cost', 'Concrete_Strength', 'Foundation_Depth']]
y = df['Construction_Cost']
```

In [227…
```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [229…
```python
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[229…
```
▾ LinearRegression
LinearRegression()
```

In [231…
```python
intercept = model.intercept_
coefficients = model.coef_
feature_names = X.columns
```

In [233…
```python
print("Regression Equation:")
equation = f"Construction_Cost = {intercept:.2f}"
for feature, coef in zip(feature_names, coefficients):
    equation += f" + ({coef:.2f} * {feature})"
print(equation)
```

```
Regression Equation:
Construction_Cost = -9.64 + (49.81 * Building_Height) + (10.33 * Material_Quality_Index) + (0.53 * Labor_Cost) + (20.20 * Concrete_Strength) + (30.14 * Foundation_Depth)
```

In [235…
```python
max_coef_index = np.argmax(np.abs(coefficients))
highest_impact_variable = feature_names[max_coef_index]
print(f"The variable with the highest impact on Construction Cost is: {highest_impact_variable} ({coefficients[max_coef_index]:.2f})")
```

```
The variable with the highest impact on Construction Cost is: Building_Height (49.81)
```

In [237…
```python
y_pred = model.predict(X_test)
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
print(f"R-squared: {r2:.4f}")
print(f"Mean Squared Error: {mse:.4f}")
```

```
R-squared: 0.9998
Mean Squared Error: 113.5044
```

In [239…
```python
n = X_train.shape[0]  # Number of observations
p = X_train.shape[1]  # Number of predictors
adjusted_r2 = 1 - ((1 - r2) * (n - 1) / (n - p - 1))
print(f"Adjusted R-squared: {adjusted_r2:.4f}")
```

```
Adjusted R-squared: 0.9998
```

In [241…
```python
# Variance Inflation Factor (VIF) to check multicollinearity
X_with_const = sm.add_constant(X)
vif_data = pd.DataFrame()
vif_data["Feature"] = X_with_const.columns
vif_data["VIF"] = [variance_inflation_factor(X_with_const.values, i) for i in range(X_with_const.shape[1])]
print("Variance Inflation Factor (VIF) Values:")
print(vif_data)
```

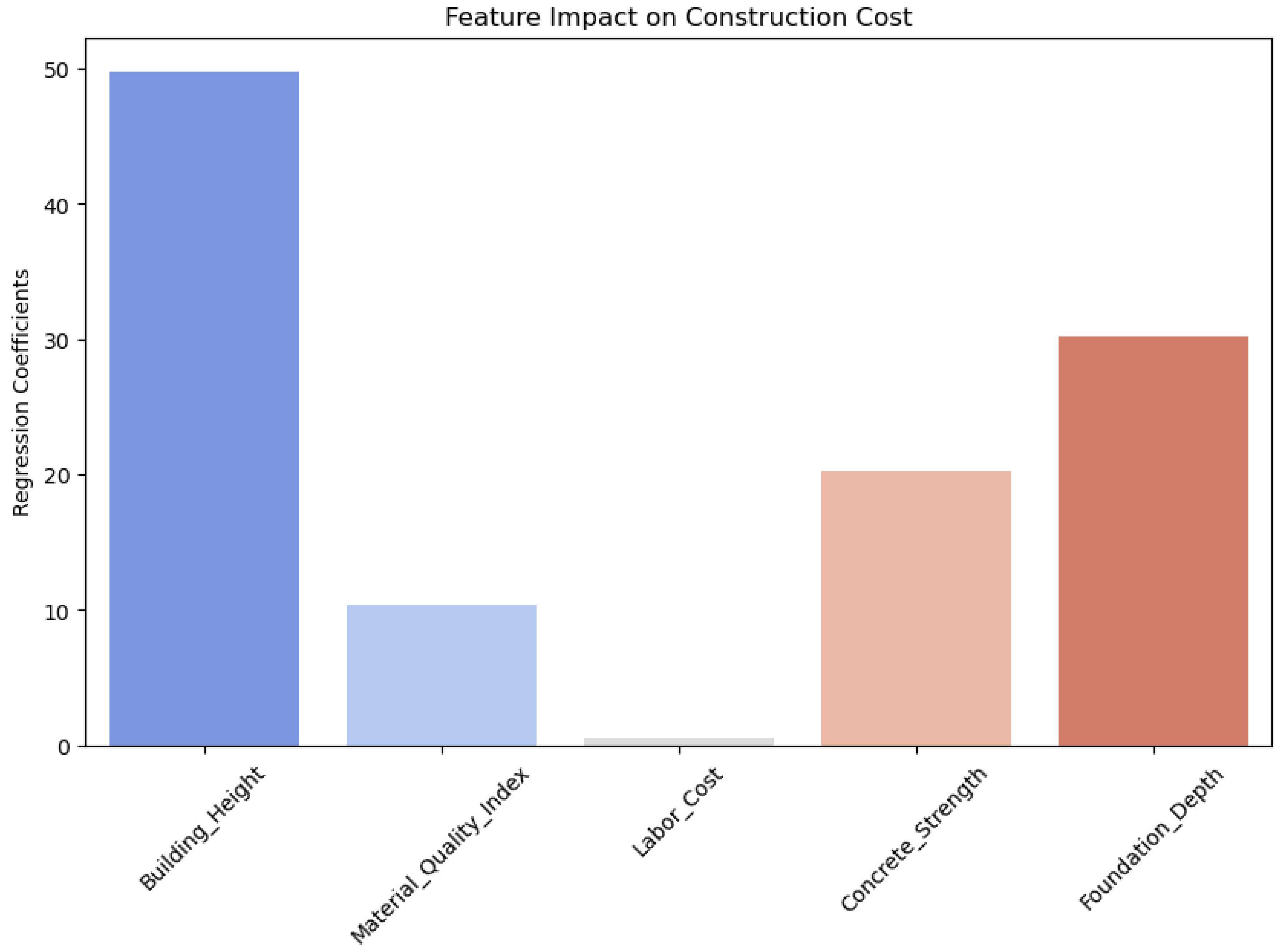
```
Variance Inflation Factor (VIF) Values:
                  Feature        VIF
0                   const  36.217244
1         Building_Height   1.047164
2  Material_Quality_Index   1.048067
3              Labor_Cost   1.054086
4       Concrete_Strength   1.019701
5        Foundation_Depth   1.040594
```

In [243…
```python
plt.figure(figsize=(10,6))
sns.barplot(x=feature_names, y=coefficients, palette='coolwarm')
plt.xlabel("Independent Variables")
plt.ylabel("Regression Coefficients")
plt.title("Feature Impact on Construction Cost")
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\gundr\AppData\Local\Temp\ipykernel_12256\3856101264.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=feature_names, y=coefficients, palette='coolwarm')
```



In [ ]:

In [ ]: