```
#======================================================================
# Import necessary libraries
#======================================================================
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, PowerTransformer
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb

# Set seed for reproducibility
student_id = '4741644011'
first_three = int(student_id[:3])
last_three = int(student_id[-3:])
Randomizer = first_three + last_three
np.random.seed(Randomizer)
print(f"Randomizer seed: {Randomizer}")
```

```
Randomizer seed: 485
```

```
#======================================================================
# Load and Explore the Data
#======================================================================
# Load the dataset
abalone = pd.read_csv('/content/sample_data/abalone.csv')

# Display the first few rows of the dataset
print(abalone.head())

# Check for missing values and data types
print(abalone.info())

text= """The dataset contains 4177 entries and 9 columns. It includes
features such as Length, Diameter, Height, Whole weight, Shucked weight,
Viscera weight, Shell weight, and Rings, which is the target variable.
There are no missing values in the dataset."""
```

```
  Sex  Length  Diameter  Height  Whole weight  Shucked weight  Viscera weight  \
0   M   0.455     0.365   0.095        0.5140          0.2245          0.1010
1   M   0.350     0.265   0.090        0.2255          0.0995          0.0485
2   F   0.530     0.420   0.135        0.6770          0.2565          0.1415
3   M   0.440     0.365   0.125        0.5160          0.2155          0.1140
4   I   0.330     0.255   0.080        0.2050          0.0895          0.0395

   Shell weight  Rings
0         0.150     15
1         0.070      7
2         0.210      9
```

```
3         0.155     10
4         0.055      7
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             4177 non-null   object 
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64  
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
None
```

```
print(text)
```

```
The dataset contains 4177 entries and 9 columns. It includes
features such as Length, Diameter, Height, Whole weight, Shucked weight,
Viscera weight, Shell weight, and Rings, which is the target variable.
There are no missing values in the dataset.
```

```
#======================================================================
# Calculate Descriptive Statistics for Original Data
```

```
#=======================================================================
def calculate_descriptive_stats(df, columns):
    stats_df = pd.DataFrame()
    stats_df['Mean'] = df[columns].mean()
    stats_df['Median'] = df[columns].median()
    stats_df['Mode'] = df[columns].mode().iloc[0]
    stats_df['St. Deviation'] = df[columns].std()
    stats_df['Range'] = df[columns].max() - df[columns].min()
    stats_df['IQR'] = df[columns].quantile(0.75) - df[columns].quantile(0.25)
    stats_df['Skewness'] = df[columns].skew()
    stats_df['Kurtosis'] = df[columns].kurtosis()
    return stats_df

numeric_columns = abalone.select_dtypes(include=[np.number]).columns
original_descriptive_stats = calculate_descriptive_stats(abalone, numeric_columns)
print("\nDetailed Descriptive Statistics for Original Data:")
print(original_descriptive_stats)

text = """
The descriptive statistics reveal that variables like Height and Rings
have high skewness and kurtosis, indicating the presence of outliers."""
```

```
Detailed Descriptive Statistics for Original Data:
                    Mean  Median    Mode  St. Deviation    Range     IQR  \
Length          0.523992  0.5450  0.5500       0.120093   0.7400  0.1650
Diameter        0.407881  0.4250  0.4500       0.099240   0.5950  0.1300
```

```
Height          0.139516  0.1400  0.1500       0.041827   1.1300  0.0500
Whole weight    0.828742  0.7995  0.2225       0.490389   2.8235  0.7115
Shucked weight  0.359367  0.3360  0.1750       0.221963   1.4870  0.3160
Viscera weight  0.180594  0.1710  0.1715       0.109614   0.7595  0.1595
Shell weight    0.238831  0.2340  0.2750       0.139203   1.0035  0.1990
Rings           9.933684  9.0000  9.0000       3.224169  28.0000  3.0000

                Skewness   Kurtosis
Length         -0.639873   0.064621
Diameter       -0.609198  -0.045476
Height          3.128817  76.025509
Whole weight    0.530959  -0.023644
Shucked weight  0.719098   0.595124
Viscera weight  0.591852   0.084012
Shell weight    0.620927   0.531926
Rings           1.114102   2.330687
```

```
print(text)
```

```
The descriptive statistics reveal that variables like Height and Rings
have high skewness and kurtosis, indicating the presence of outliers.
```

```
#=======================================================================
# Preprocess the Data
#=======================================================================
abalone = abalone.dropna()
X = abalone.drop('Rings', axis=1)
```

```
y = abalone['Rings']
X = pd.get_dummies(X, drop_first=True)
numeric_X = X.select_dtypes(include=[np.number])

Q1 = numeric_X.quantile(0.25)
Q3 = numeric_X.quantile(0.75)
IQR = Q3 - Q1
outliers = ~((numeric_X < (Q1 - 1.5 * IQR)) | (numeric_X >
                                               (Q3 + 1.5 * IQR))).any(axis=1)
X = X[outliers]
y = y[outliers]

text = """Outliers were detected and removed using the IQR method.
The data is now cleaner and more representative."""
print(text)
```

```
Outliers were detected and removed using the IQR method.
The data is now cleaner and more representative.
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

text = """The data is now standardized, ensuring all features have
```

```
a mean of 0 and a standard deviation of 1."""
print(text)
```

```
The data is now standardized, ensuring all features have
a mean of 0 and a standard deviation of 1.
```

```
pt = PowerTransformer(method='yeo-johnson')
X_train = pt.fit_transform(X_train)
X_test = pt.transform(X_test)

text = """A Yeo-Johnson transformation was applied to normalize
the features, making them more suitable for modeling."""
print(text)
```

```
A Yeo-Johnson transformation was applied to normalize
the features, making them more suitable for modeling.
```

```
#=======================================================================
# Build and Train Models
#=======================================================================
print("\nMultiple Regression Model Results")
X_train_sm = sm.add_constant(X_train)
X_test_sm = sm.add_constant(X_test)
multi_model = sm.OLS(y_train, X_train_sm).fit()
```

```
y_pred_multi = multi_model.predict(X_test_sm)
mse_multi = mean_squared_error(y_test, y_pred_multi)
r2_multi = r2_score(y_test, y_pred_multi)
print(f'Multiple Regression - MSE: {mse_multi}, R2: {r2_multi}')

text = """The Multiple Regression model explains approximately 51.5% of the
variance in the Rings variable with an MSE of 5.09. The coefficients for each
feature are provided in the summary."""
```

```
Multiple Regression Model Results
Multiple Regression - MSE: 5.091594963734171, R2: 0.5146504732940977
```

```
print(text)
```

```
The Multiple Regression model explains approximately 51.5% of the
variance in the Rings variable with an MSE of 5.09. The coefficients for each
feature are provided in the summary.
```

```
print("\nRandom Forest Regression Model Results")
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)
```

```
print(f'Random Forest Regression - MSE: {mse_rf}, R2: {r2_rf}')

text = """The Random Forest Regression model explains approximately
 52.2% of the variance in the Rings variable with an MSE of 5.02.
The feature importance scores indicate that Shell weight, Shucked
weight, and Whole weight are the most important predictors."""
```

```
Random Forest Regression Model Results
Random Forest Regression - MSE: 5.019126583850931, R2: 0.5215584253460572
```

```
print(text)
```

```
The Random Forest Regression model explains approximately
 52.2% of the variance in the Rings variable with an MSE of 5.02.
The feature importance scores indicate that Shell weight, Shucked
weight, and Whole weight are the most important predictors.
```

```
print("\nXGBoost Regression Model Results")
xgb_model = xgb.XGBRegressor(n_estimators=100, random_state=42)
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)
mse_xgb = mean_squared_error(y_test, y_pred_xgb)
r2_xgb = r2_score(y_test, y_pred_xgb)
print(f'XGBoost Regression - MSE: {mse_xgb}, R2: {r2_xgb}')
```

```
text = """The XGBoost Regression model explains approximately
 44.4% of the variance in the Rings variable with an MSE of 5.83."""
```

```
XGBoost Regression Model Results
XGBoost Regression - MSE: 5.8339972496032715, R2: 0.4438819885253906
```

```
print(text)
```

```
The XGBoost Regression model explains approximately
 44.4% of the variance in the Rings variable with an MSE of 5.83.
```

```
#========================================================================
# Submit Predictions to Kaggle
#========================================================================
submission = pd.DataFrame({
    'Id': np.arange(len(y_test)),
    'Rings': y_pred_rf
})
submission.to_csv('/content/sample_data/submission.csv', index=False)
print('Submission file created!')

print("""The submission file has been created successfully.
You can upload it to Kaggle for evaluation.""")
```

```
Submission file created!
The submission file has been created successfully.
You can upload it to Kaggle for evaluation.
```

```
#==========================================================================
# Interpret Findings
#==========================================================================
print("\nCoefficients for Multiple Regression:")
print(multi_model.summary())

text = """The coefficients for the Multiple Regression model show
that Diameter,Whole weight, and Shell weight have a positive
relationship  with Rings, while Shucked weight has a negative
  relationship."""
```

```
Coefficients for Multiple Regression:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Rings   R-squared:                       0.520
Model:                            OLS   Adj. R-squared:                  0.518
Method:                 Least Squares   F-statistic:                     385.8
Date:                Sun, 09 Feb 2025   Prob (F-statistic):               0.00
```

```
Time:                        23:02:41   Log-Likelihood:                -7021.1
No. Observations:                3219   AIC:                         1.406e+04
Df Residuals:                    3209   BIC:                         1.412e+04
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          9.9211      0.038    262.256      0.000       9.847       9.995
x1            -0.4161      0.216     -1.924      0.054      -0.840       0.008
x2             0.3326      0.212      1.572      0.116      -0.082       0.747
x3             0.6611      0.095      6.970      0.000       0.475       0.847
x4             4.0914      0.419      9.759      0.000       3.269       4.914
x5            -4.1762      0.220    -19.000      0.000      -4.607      -3.745
x6            -1.1402      0.170     -6.692      0.000      -1.474      -0.806
x7             2.0466      0.203     10.067      0.000       1.648       2.445
x8            -0.4066      0.053     -7.644      0.000      -0.511      -0.302
x9            -0.0047      0.045     -0.105      0.916      -0.092       0.083
==============================================================================
Omnibus:                      740.345   Durbin-Watson:                   1.970
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2222.340
Skew:                           1.176   Prob(JB):                         0.00
Kurtosis:                       6.323   Cond. No.                         34.3
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

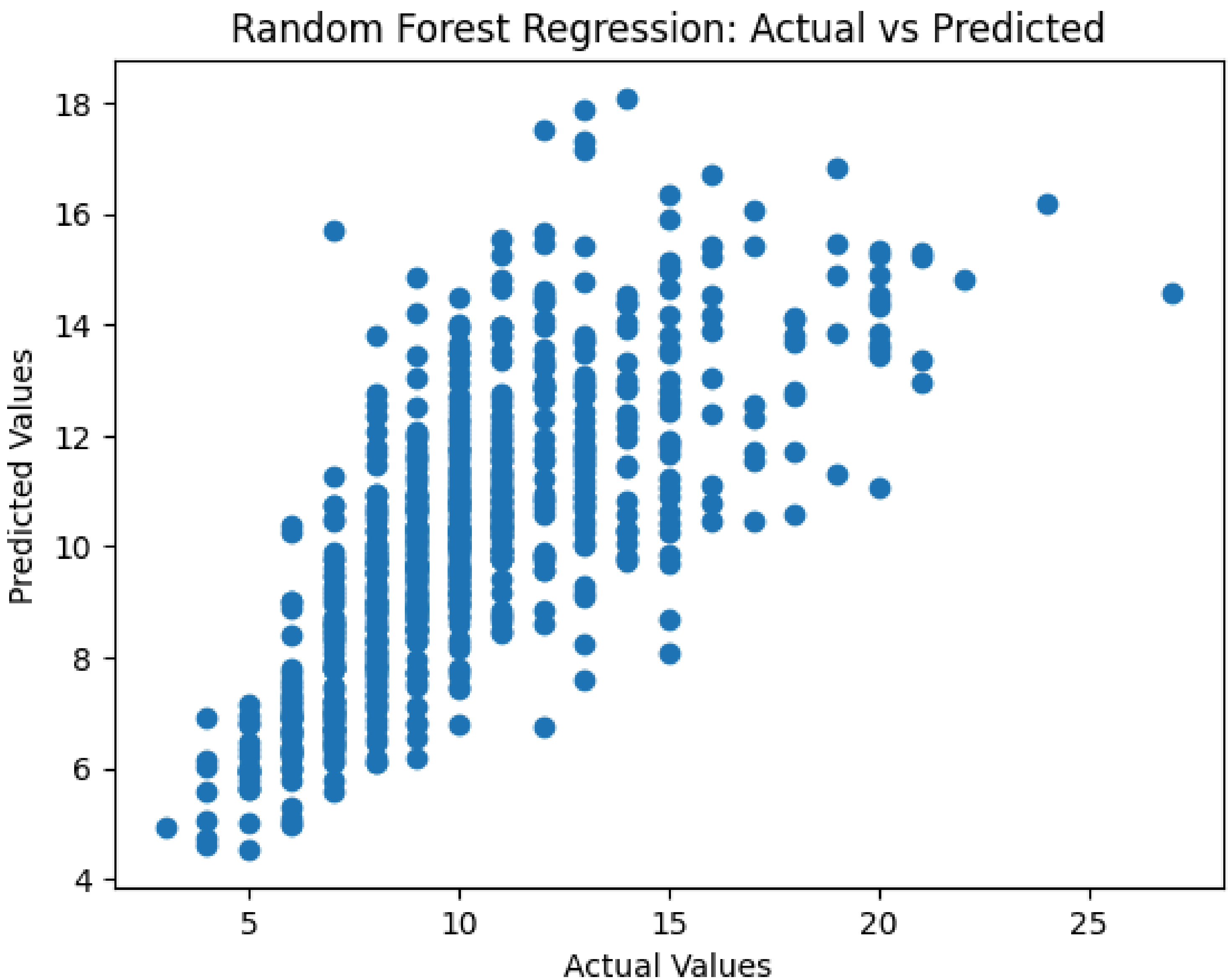
```
print(text)
```

```
The coefficients for the Multiple Regression model show
that Diameter,Whole weight, and Shell weight have a positive
relationship  with Rings, while Shucked weight has a negative
  relationship.
```

```python
importances = rf_model.feature_importances_
feature_importance = pd.DataFrame(importances, index=X.columns,
            columns=['Importance']).sort_values('Importance', ascending=False)
print(feature_importance)

text = """The feature importance scores for the Random Forest model indicate
that Shell weight is the most important predictor, followed by Shucked weight
and Whole weight."""
print(text)
```

```
                Importance
Shell weight      0.466660
Shucked weight    0.178406
Whole weight      0.084708
Viscera weight    0.077312
Height            0.055858
Diameter          0.054481
Length            0.052101
Sex_I             0.021860
Sex_M             0.008615
The feature importance scores for the Random Forest model indicate
that Shell weight is the most important predictor, followed by Shucked weight
and Whole weight.
```

```
# Plots for Random Forest Regression Model

# Linearity and Homoscedasticity
plt.figure()
plt.scatter(y_test, y_pred_rf)
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Random Forest Regression: Actual vs Predicted')
plt.show()  # Display the plot
```

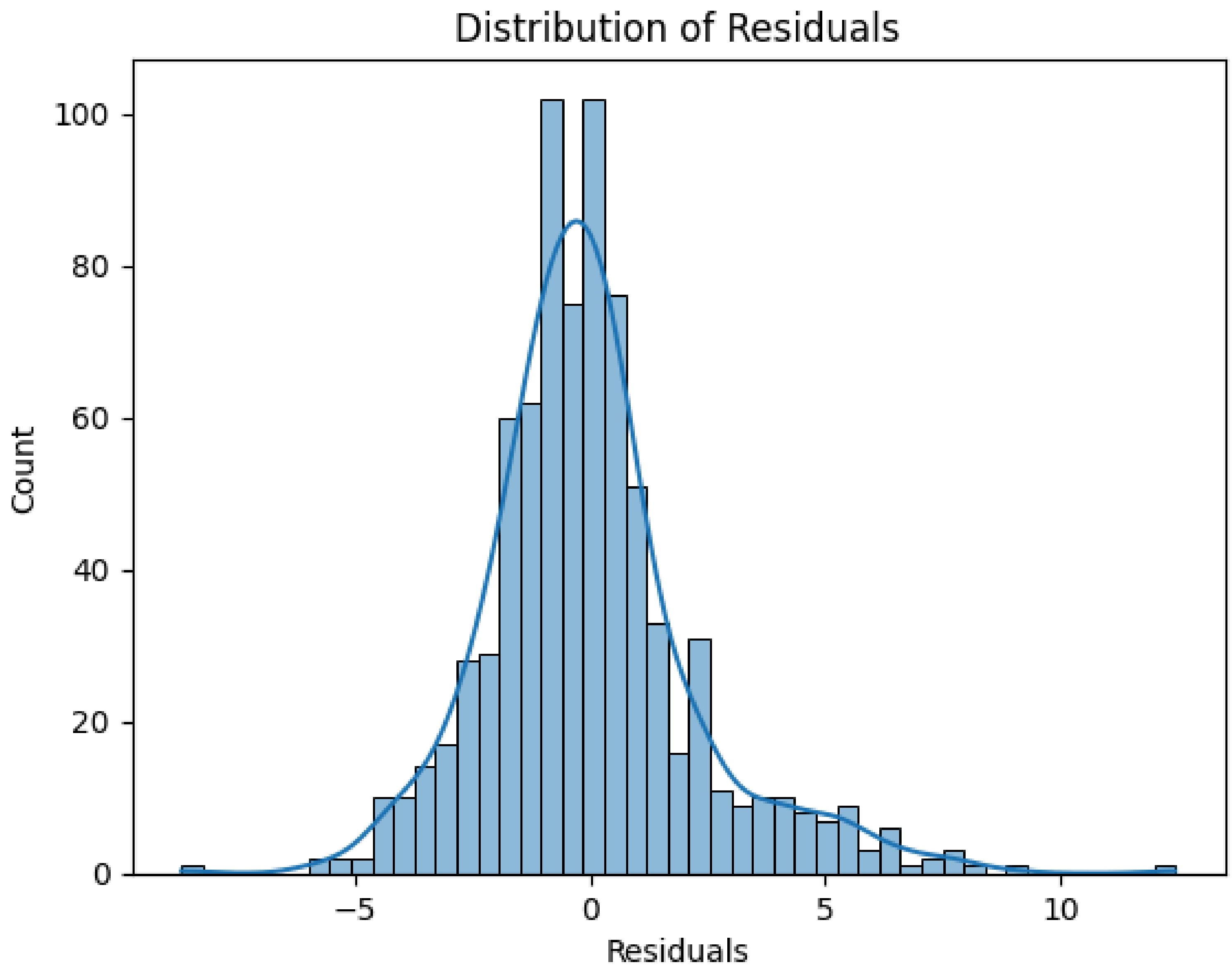



```
text = """The scatter plot of actual vs. predicted values suggests
a positive correlation between the actual and predicted values.
```

```
This suggests that the model's predictions align reasonably well with
the actual data. However, the spread of points around the trend line
indicates that there are some deviations, implying that while the
model captures the general pattern, there are still
errors in prediction."""
print(text)
```

```
The scatter plot of actual vs. predicted values suggests
a positive correlation between the actual and predicted values.
This suggests that the model's predictions align reasonably well with
the actual data. However, the spread of points around the trend line
indicates that there are some deviations, implying that while the
model captures the general pattern, there are still
errors in prediction.
```

```
# Distribution of Residuals
plt.figure()
residuals_rf = y_test - y_pred_rf
sns.histplot(residuals_rf, kde=True)
plt.xlabel('Residuals')
plt.title('Distribution of Residuals')
plt.show()  # Display the plot
```



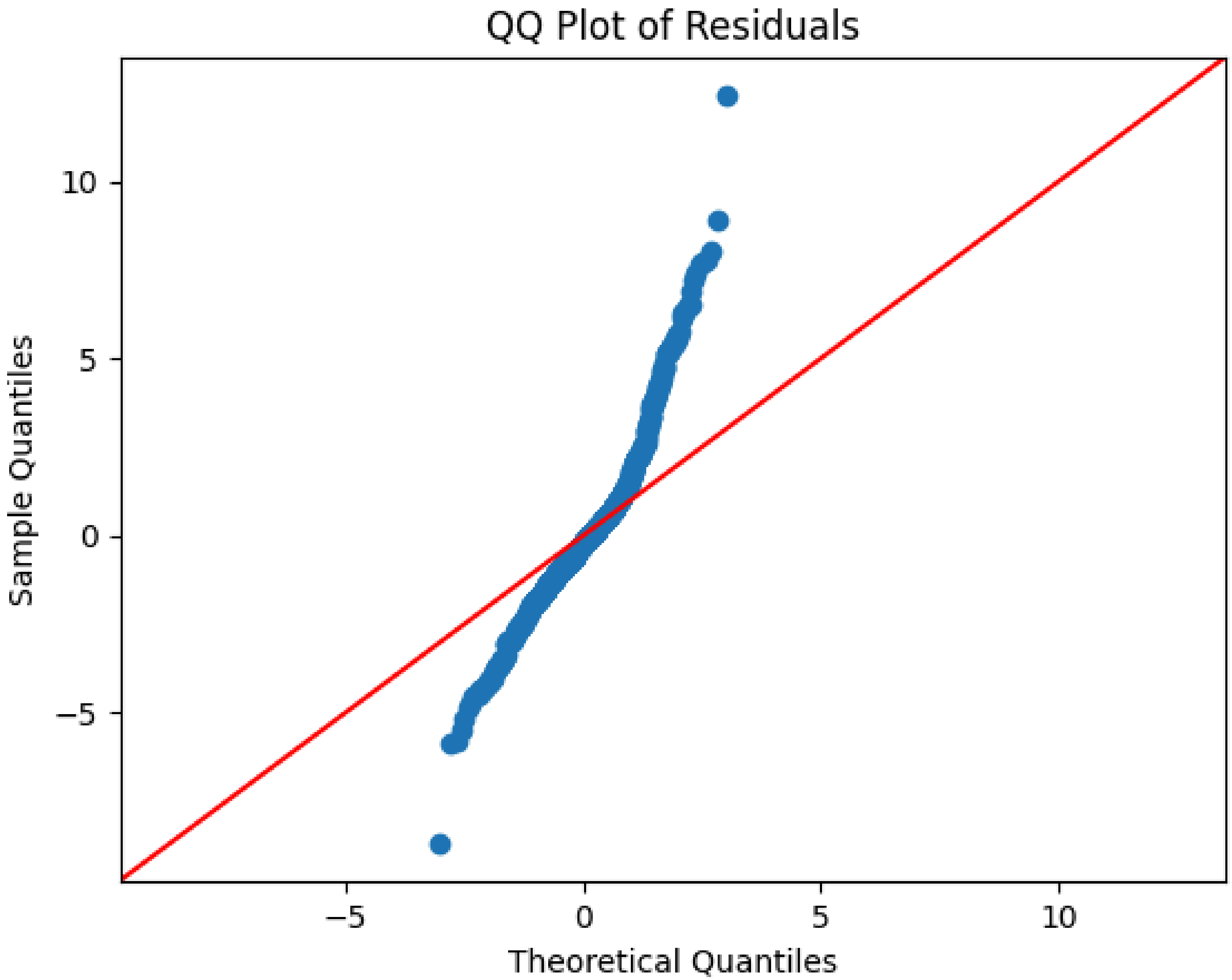

```
text = """The histogram of residuals indicates that they are approximately
 normally distributed, supporting the assumption of normality. Overall,
```

```
the distribution of residuals plot supports the assumption of normality,
though withslight deviations at the extremes. This information is
useful for diagnosing thefit of the model and identifying any potential
 issues with the residuals."""



print(text)
```

```
The histogram of residuals indicates that they are approximately
 normally distributed, supporting the assumption of normality. Overall,
the distribution of residuals plot supports the assumption of normality,
though withslight deviations at the extremes. This information is
useful for diagnosing thefit of the model and identifying any potential
 issues with the residuals.
```

```
# Normality of Residuals
plt.figure()
sm.qqplot(residuals_rf, line='45')
plt.title('QQ Plot of Residuals')
plt.show()  # Display the plot
```

<Figure size 640x480 with 0 Axes>



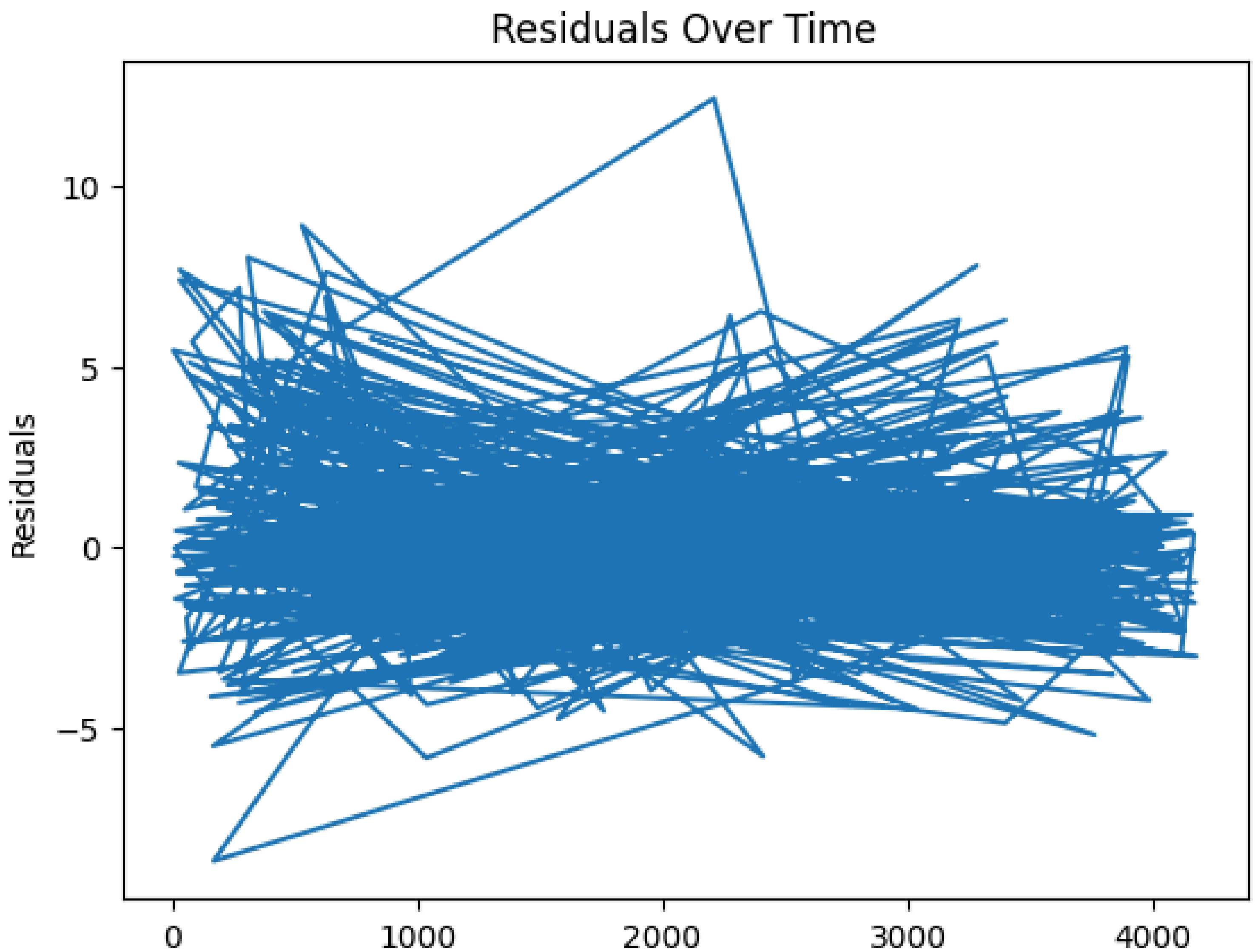

```
text = """The QQ plot shows that the residuals roughly follow the 45-degree line,
further confirming normality.The QQ plot suggests that the residuals from the
```

```
Random Forest regression model are approximately normally distributed, with some
deviations in the tails. This indicates the presence of outliers or extreme
values, which can impact the  model's accuracy and the validity of certain
statistical tests."""
```

```
print(text)
```

```
The QQ plot shows that the residuals roughly follow the 45-degree line,
further confirming normality.The QQ plot suggests that the residuals from the
Random Forest regression model are approximately normally distributed, with some
deviations in the tails. This indicates the presence of outliers or extreme
values, which can impact the  model's accuracy and the validity of certain
statistical tests.
```

```
# Independence of Residuals
plt.figure()
plt.plot(residuals_rf)
plt.ylabel('Residuals')
plt.title('Residuals Over Time')
plt.show()  # Display the plot
```




```
text = """The plot of residuals over time shows no discernible pattern,
suggesting that the residuals are independent.Overall, the Random Forest
Regressionmodel assumptions appear to be reasonably met based on
```

```
the plots provided. Theplot indicates that the residuals are centered
around zero, fluctuate randomly,and show no discernible pattern. This
supportsthe assumption of independence of residuals, which is important
for the validity of the  regression model. However, the presence
of some outliers suggeststhat there may be some extreme values in the data
that could affect the model's performance."""



print(text)
```

```
The plot of residuals over time shows no discernible pattern,
suggesting that the residuals are independent.Overall, the Random Forest
Regressionmodel assumptions appear to be reasonably met based on
the plots provided. Theplot indicates that the residuals are centered
around zero, fluctuate randomly,and show no discernible pattern. This
supportsthe assumption of independence of residuals, which is important
for the validity of the  regression model. However, the presence
of some outliers suggeststhat there may be some extreme values in the data
that could affect the model's performance.
```

```
#=======================================================================
# Conclusion and Recommendations for Improvement
#=======================================================================
print("\nConclusion:")
print("""# Both the Multiple Regression and Random Forest models provide
valuable insights into predicting the Rings variable.""")
```

```
Conclusion:
# Both the Multiple Regression and Random Forest models provide
valuable insights into predicting the Rings variable.
```

```
text = """The Multiple Regression model explains approximately 51.5% of the
variance inthe Rings variable with an MSE of 5.09. The Random Forest model
explains approximately 52.2% of the variance with an MSE of 5.02. The XGBoost
model explains approximately 44.4% of the variance with an MSE of 5.83."""
```

```
print(text)
```

```
The Multiple Regression model explains approximately 51.5% of the
variance inthe Rings variable with an MSE of 5.09. The Random Forest model
explains approximately 52.2% of the variance with an MSE of 5.02. The XGBoost
model explains approximately 44.4% of the variance with an MSE of 5.83.
```

```
print("\nRecommendations for Improvement:")
```

Recommendations for Improvement:

```
text=f"""#1 Feature Engineering: Create new features or transform
existing ones to capture more information and improve model performance.

 #2 Hyperparameter Tuning: Use techniques such as Grid Search or Random
Search to find the optimal hyperparameters for the Random Forest and XGBoost
models.
#3 Advanced Models: Experiment with other advanced regression models
such as Gradient Boosting or Neural Networks to potentially achieve better
accuracy

 #4 Cross-Validation: Implement cross-validation to ensure the model's
robustness and prevent overfitting.

 #5 Ensemble Methods: Combine predictions from multiple models to create
```