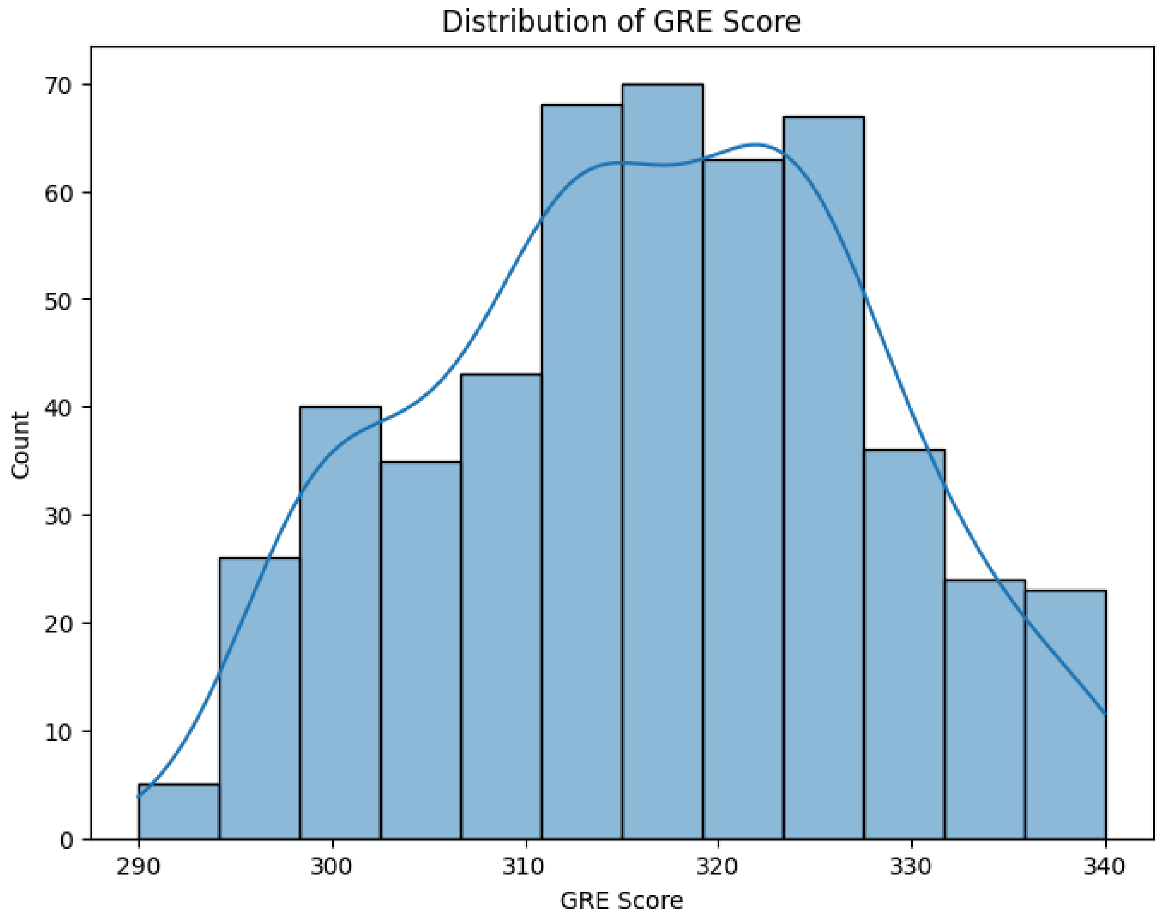
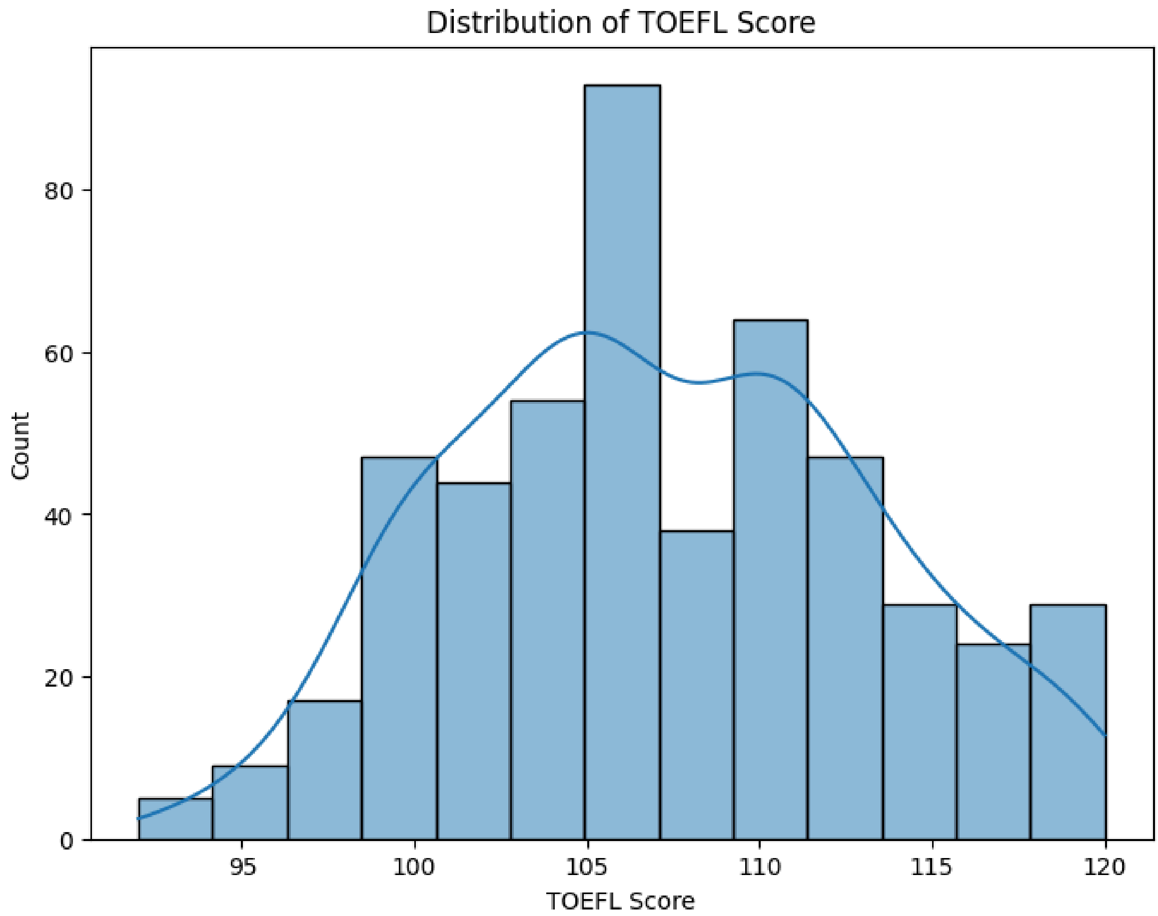
In [8]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.linear_model import Ridge, Lasso
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_s
from statsmodels.stats.outliers_influence import variance_inflation_facto

# Load the dataset
os.chdir("/Users/ayeshasiddiqha/Downloads")
df = pd.read_csv('jamboree_admission.csv')

# Step 1: Exploratory Data Analysis (EDA)

# Checking the basic structure of the data
print(df.info())
print(df.describe())

# Dropping the Serial No. column as it is not relevant
df = df.drop(columns=['Serial No.'])

# Checking for missing values
print(df.isnull().sum())

# Step 1.1: Univariate Analysis - Distribution Plots for Continuous Varia
continuous_vars = ['GRE Score', 'TOEFL Score', 'University Rating',
                   'SOP', 'CGPA', 'Chance of Admit ']

for var in continuous_vars:
    plt.figure(figsize=(8, 6))
    sns.histplot(df[var], kde=True)
    plt.title(f'Distribution of {var}')
    plt.show()

# Step 1.2: Bivariate Analysis - Correlation Matrix
corr_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidt
plt.title('Correlation Matrix')
plt.show()

# Step 2: Data Preprocessing

# 2.1: Duplicate value check
print(f'Duplicate rows: {df.duplicated().sum()}')

# 2.2: Missing value treatment (Impute with median for continuous variabl
df = df.fillna(df.median())

# 2.3: Outlier treatment - Checking outliers using IQR
Q1 = df[continuous_vars].quantile(0.25)
Q3 = df[continuous_vars].quantile(0.75)
IQR = Q3 - Q1

outlier_condition = ((df[continuous_vars] < (Q1 - 1.5 * IQR)) | (df[conti
```

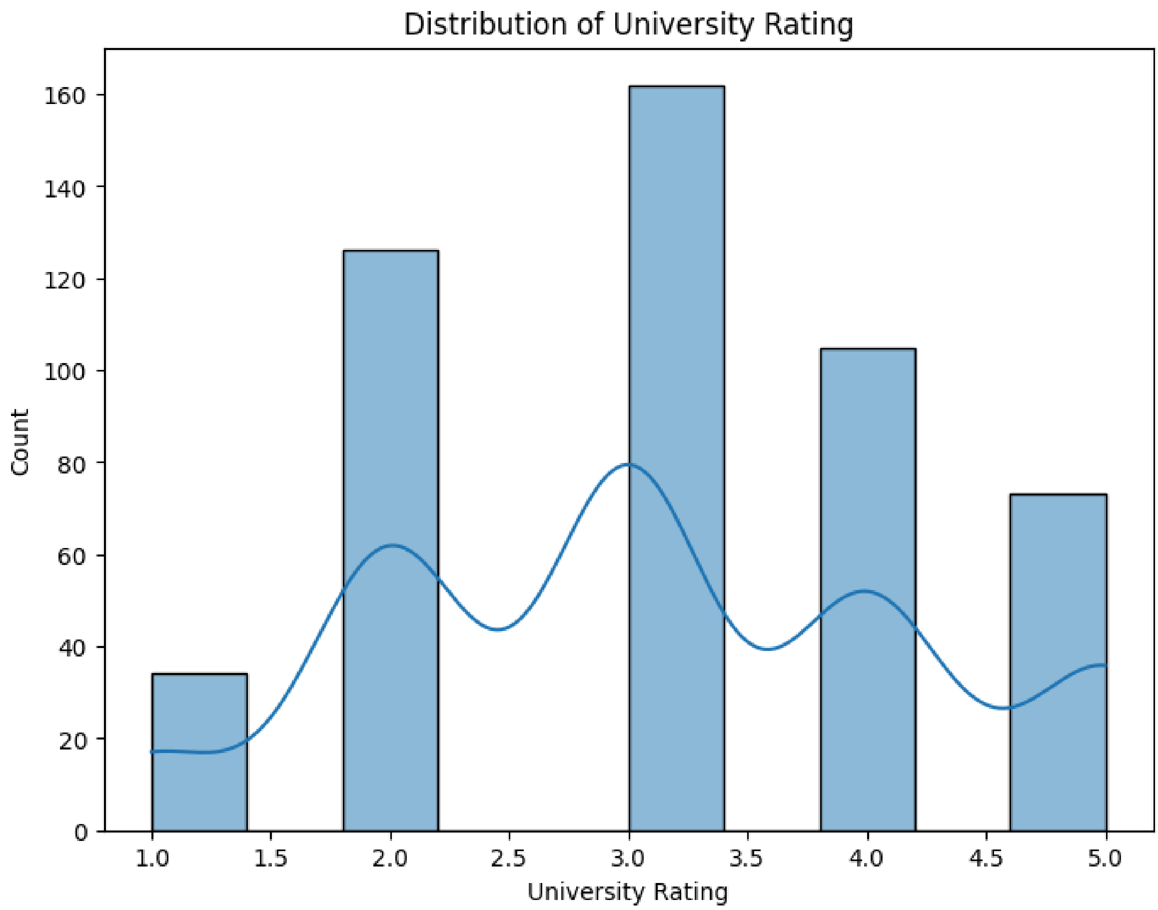
```
df[outlier_condition] = np.nan
df = df.fillna(df.median())

# 2.4: Feature Engineering - Standardization (Scaling continuous variable
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

df[['GRE Score', 'TOEFL Score', 'University Rating',
    'SOP', 'CGPA']] = scaler.fit_transform(df[['GRE Score', 'TOEFL Score'
                                                'University Rating',
                                                'SOP', 'CGPA']])

# Step 3: Model Building - Linear Regression
# Separate features and target variable
X = df.drop(columns=['Chance of Admit '])
y = df['Chance of Admit ']

# Adding a constant to the model (for intercept)
X = sm.add_constant(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,

# Linear Regression model
model = sm.OLS(y_train, X_train).fit()
print(model.summary())

# Step 4: Testing Assumptions of Linear Regression

# 4.1: Multicollinearity check (VIF)
vif_data = pd.DataFrame()
vif_data['Variable'] = X_train.columns
vif_data['VIF'] = [variance_inflation_factor(X_train.values, i) for i in

# Drop columns with VIF > 5
vif_data = vif_data[vif_data['VIF'] < 5]
print(vif_data)

# 4.2: Mean of residuals
residuals = y_train - model.fittedvalues
print(f'Mean of Residuals: {np.mean(residuals)}')

# 4.3: Linearity of variables (Residual Plot)
plt.scatter(model.fittedvalues, residuals)
plt.axhline(y=0, color='r', linestyle='--')
plt.title('Residuals vs Fitted Values')
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.show()

# 4.4: Homoscedasticity (Constant Variance of Errors)
sns.scatterplot(x=model.fittedvalues, y=residuals)
plt.axhline(y=0, color='r', linestyle='--')
plt.title('Homoscedasticity Check')
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.show()

# 4.5: Normality of Residuals (Histogram & Q-Q Plot)
sns.histplot(residuals, kde=True)
```

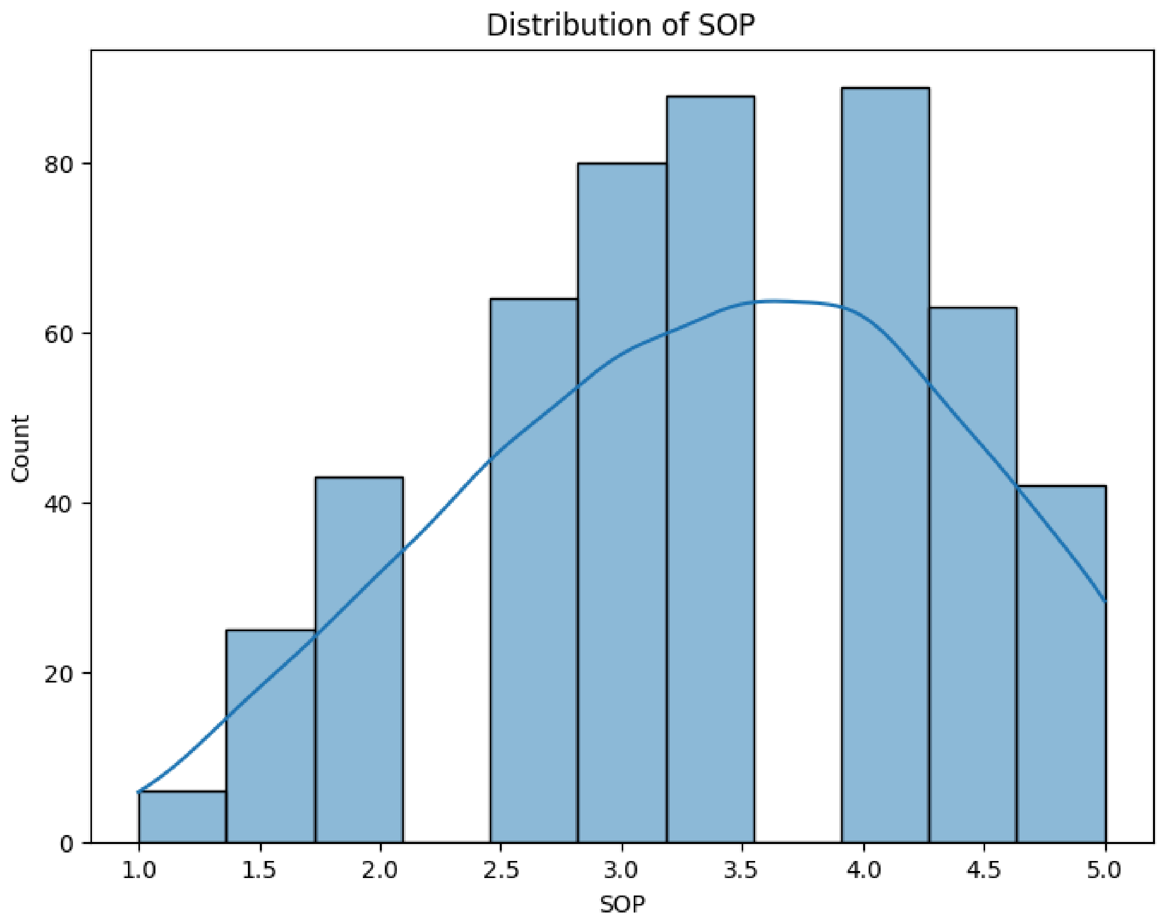
```
plt.title('Residuals Distribution')
plt.show()

import scipy.stats as stats
stats.probplot(residuals, dist="norm", plot=plt)
plt.title('Q-Q Plot for Normality Check')
plt.show()

# Step 5: Model Performance Evaluation
y_pred = model.predict(X_test)

# Metrics
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
adj_r2 = 1 - (1 - r2) * (len(y_test) - 1) / (len(y_test) - X_test.shape[1

print(f'Mean Absolute Error (MAE): {mae}')
print(f'Root Mean Squared Error (RMSE): {rmse}')
print(f'R2 Score: {r2}')
print(f'Adjusted R2 Score: {adj_r2}')

# Step 6: Ridge and Lasso Regression
# Ridge Regression
ridge = Ridge(alpha=1.0)
ridge.fit(X_train, y_train)
ridge_pred = ridge.predict(X_test)

# Lasso Regression
lasso = Lasso(alpha=0.1)
lasso.fit(X_train, y_train)
lasso_pred = lasso.predict(X_test)

# Evaluate Ridge and Lasso models
ridge_mae = mean_absolute_error(y_test, ridge_pred)
ridge_rmse = np.sqrt(mean_squared_error(y_test, ridge_pred))
ridge_r2 = r2_score(y_test, ridge_pred)

lasso_mae = mean_absolute_error(y_test, lasso_pred)
lasso_rmse = np.sqrt(mean_squared_error(y_test, lasso_pred))
lasso_r2 = r2_score(y_test, lasso_pred)

print("Ridge Regression Performance:")
print(f'MAE: {ridge_mae}, RMSE: {ridge_rmse}, R2: {ridge_r2}')

print("Lasso Regression Performance:")
print(f'MAE: {lasso_mae}, RMSE: {lasso_rmse}, R2: {lasso_r2}')
```

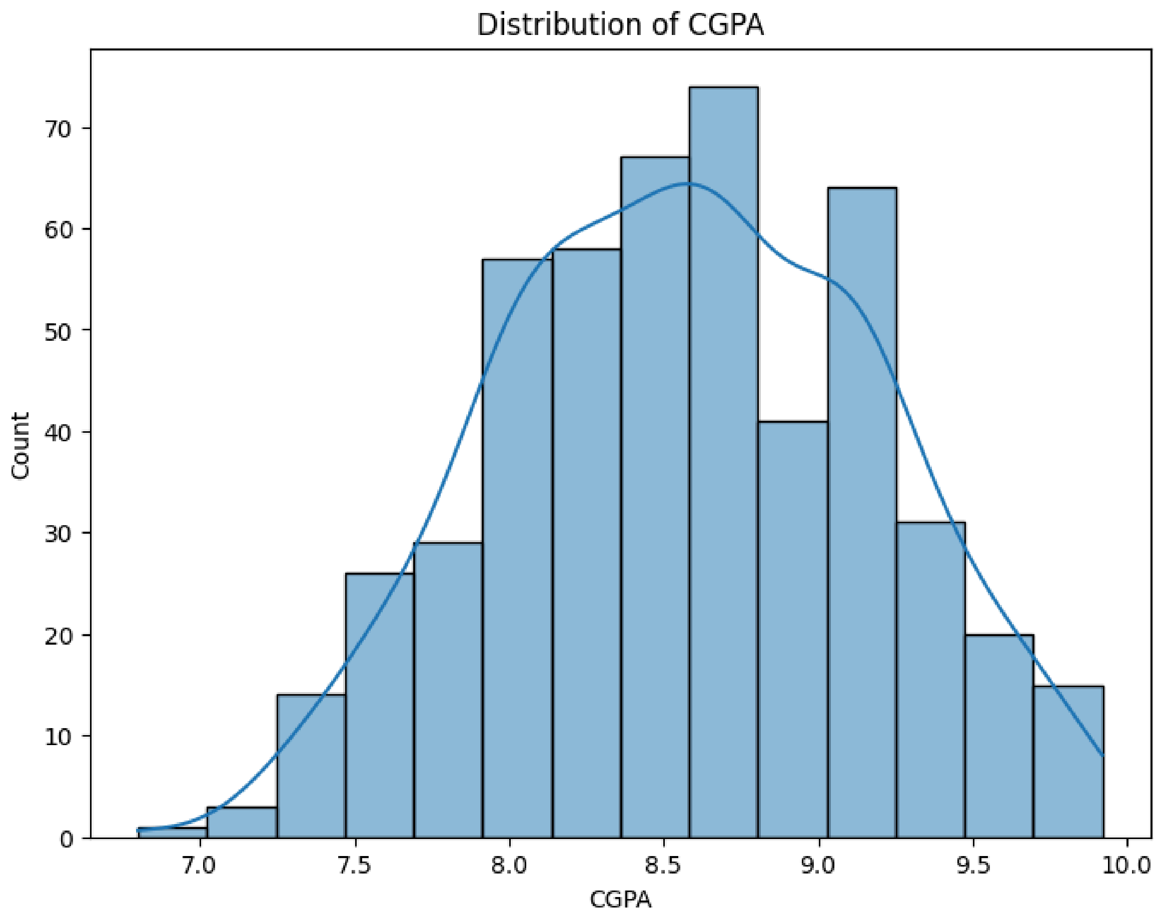
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Serial No.         500 non-null    int64  
 1   GRE Score          500 non-null    int64  
 2   TOEFL Score        500 non-null    int64  
 3   University Rating  500 non-null    int64  
 4   SOP                500 non-null    float64
 5   LOR                500 non-null    float64
 6   CGPA               500 non-null    float64
 7   Research           500 non-null    int64  
 8   Chance of Admit    500 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 35.3 KB
None
       Serial No.   GRE Score  TOEFL Score  University Rating         SOP  
\
count  500.000000  500.000000   500.000000         500.000000  500.000000   
mean   250.500000  316.472000   107.192000           3.114000    3.374000   
std    144.481833   11.295148     6.081868           1.143512    0.991004   
min      1.000000  290.000000    92.000000           1.000000    1.000000   
25%    125.750000  308.000000   103.000000           2.000000    2.500000   
50%    250.500000  317.000000   107.000000           3.000000    3.500000   
75%    375.250000  325.000000   112.000000           4.000000    4.000000   
max    500.000000  340.000000   120.000000           5.000000    5.000000   

            LOR        CGPA    Research  Chance of Admit   
count  500.00000  500.000000  500.000000        500.00000  
mean     3.48400    8.576440    0.560000          0.72174  
std      0.92545    0.604813    0.496884          0.14114  
min      1.00000    6.800000    0.000000          0.34000  
25%      3.00000    8.127500    0.000000          0.63000  
50%      3.50000    8.560000    1.000000          0.72000  
75%      4.00000    9.040000    1.000000          0.82000  
max      5.00000    9.920000    1.000000          0.97000  
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

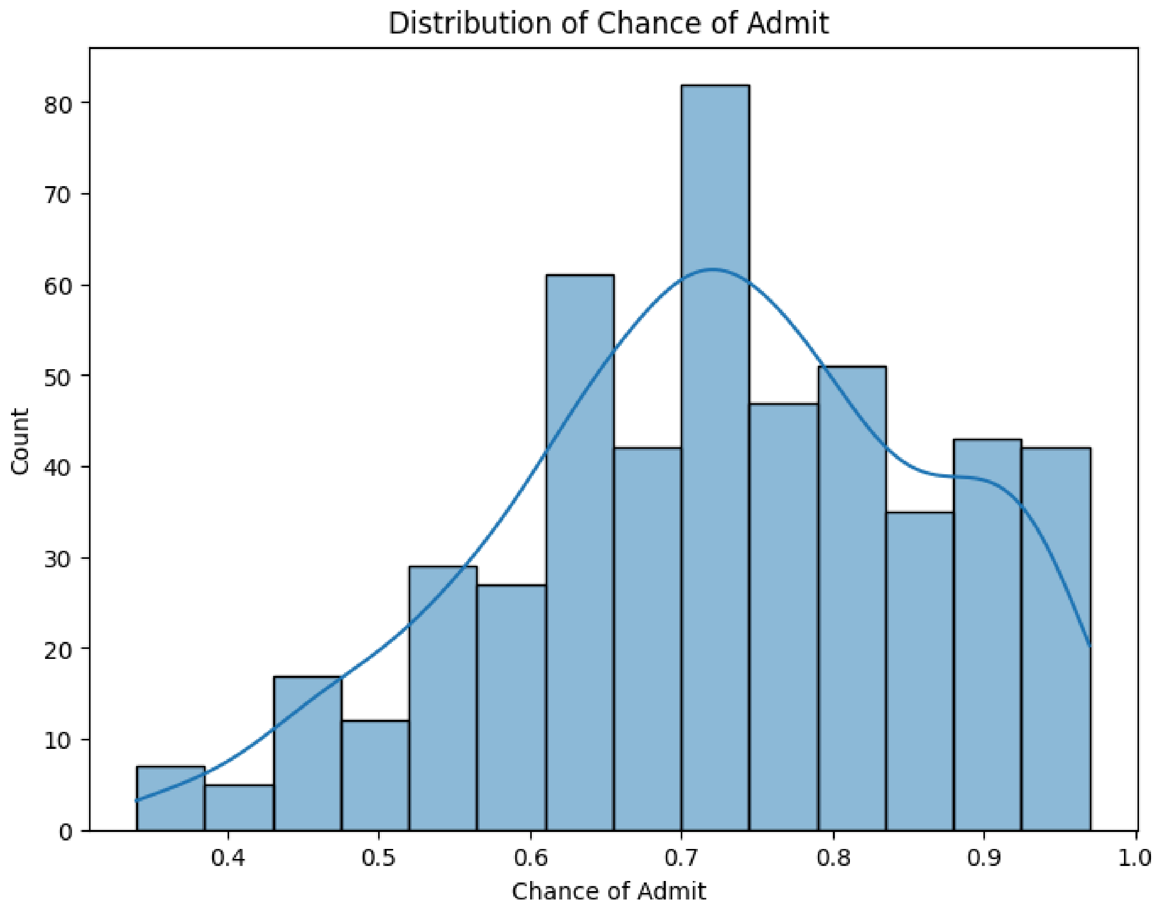

Distribution of GRE Score
Count
0
10
20
30
40
50
60
70
290
300
310
320
330
340
GRE Score

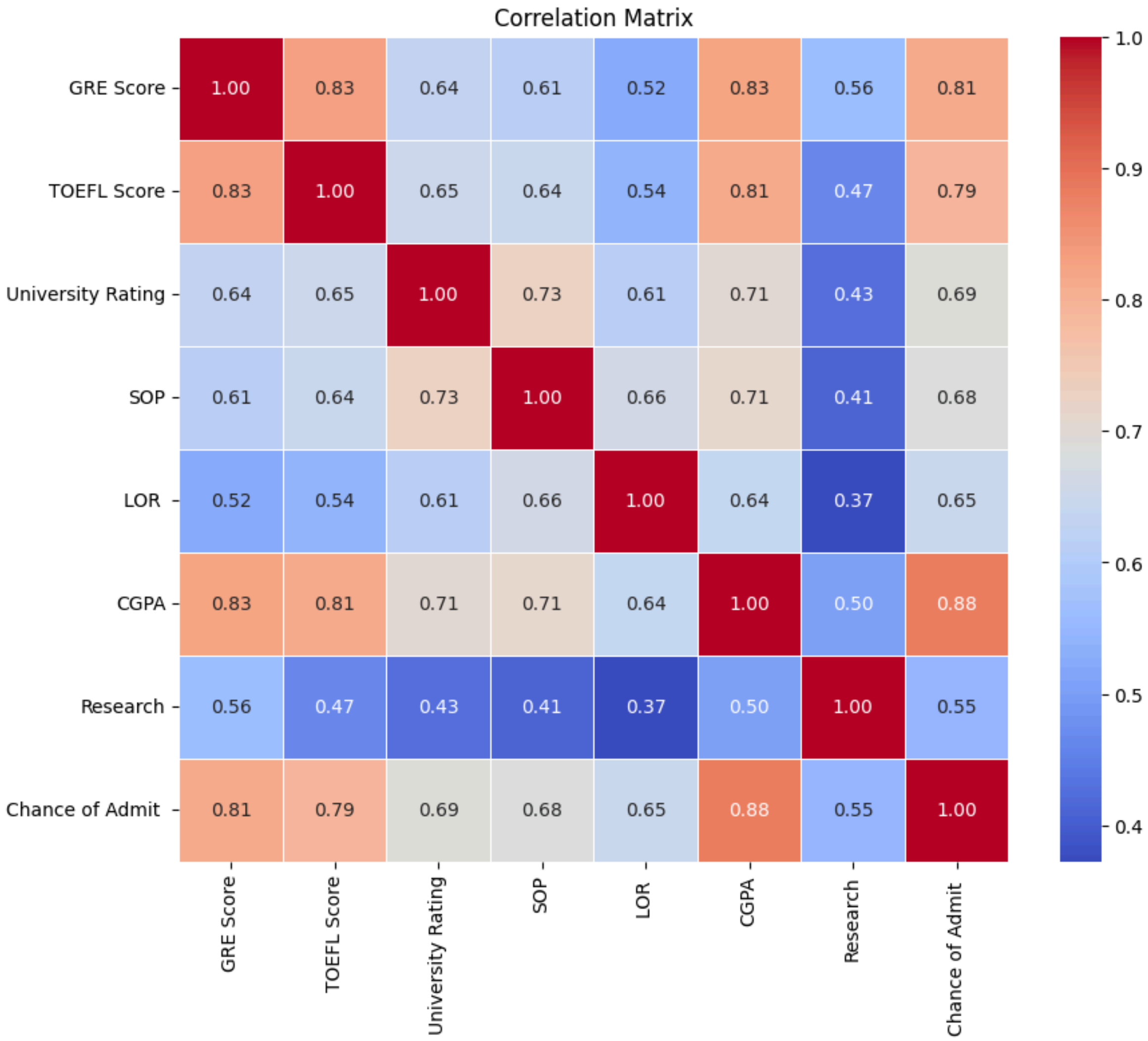

Distribution of TOEFL Score
Count
0
20
40
60
80
95
100
105
110
115
120
TOEFL Score


Distribution of University Rating
Count
University Rating
0
20
40
60
80
100
120
140
160
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Distribution of SOP
Count
SOP
0
20
40
60
80
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

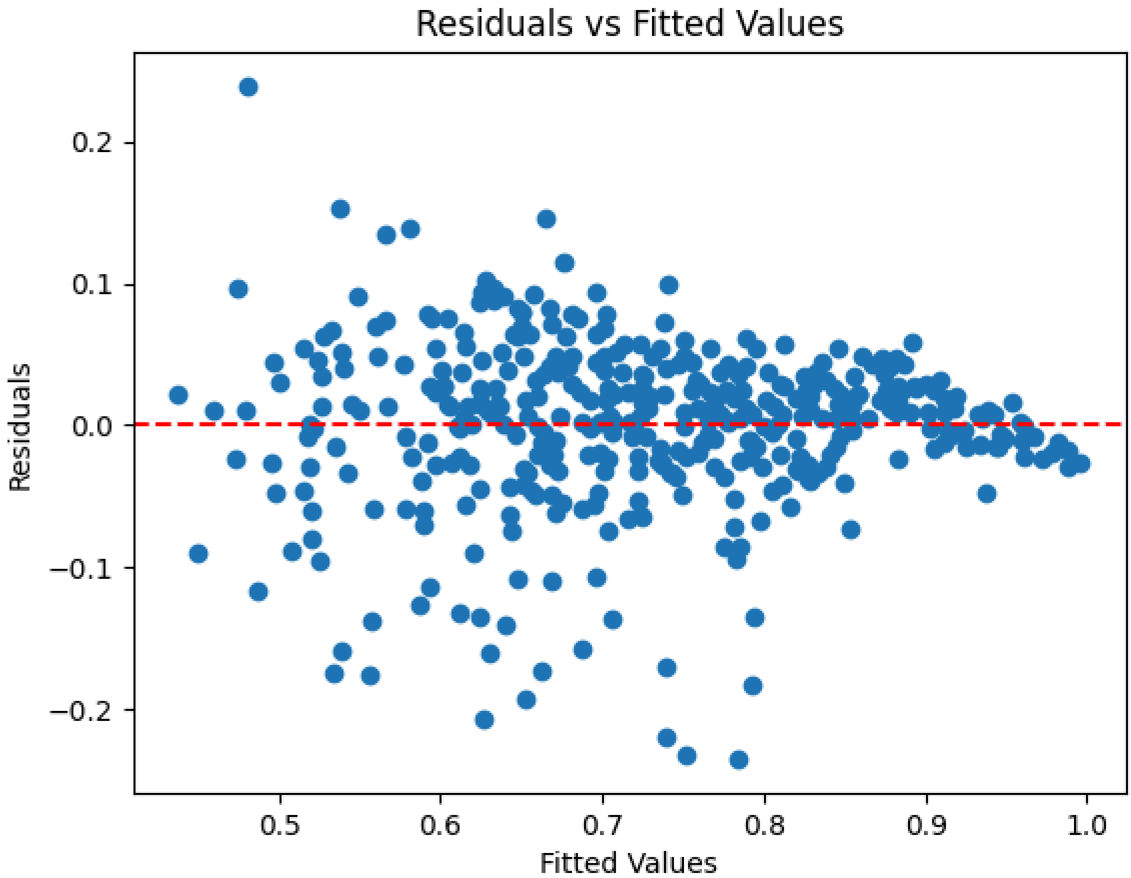

Distribution of CGPA
Count
CGPA
0
10
20
30
40
50
60
70
7.0
7.5
8.0
8.5
9.0
9.5
10.0

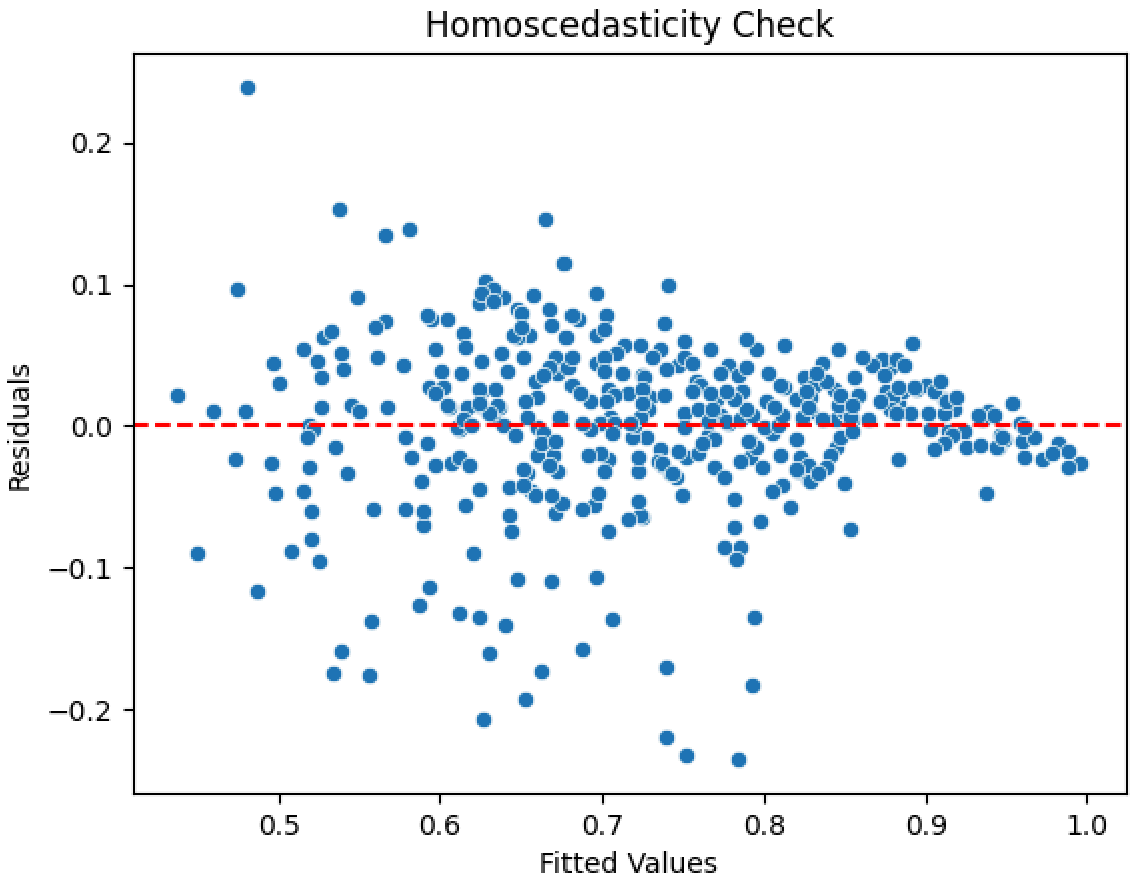

Distribution of Chance of Admit
Count
Chance of Admit
0
10
20
30
40
50
60
70
80
0.4
0.5
0.6
0.7
0.8
0.9
1.0

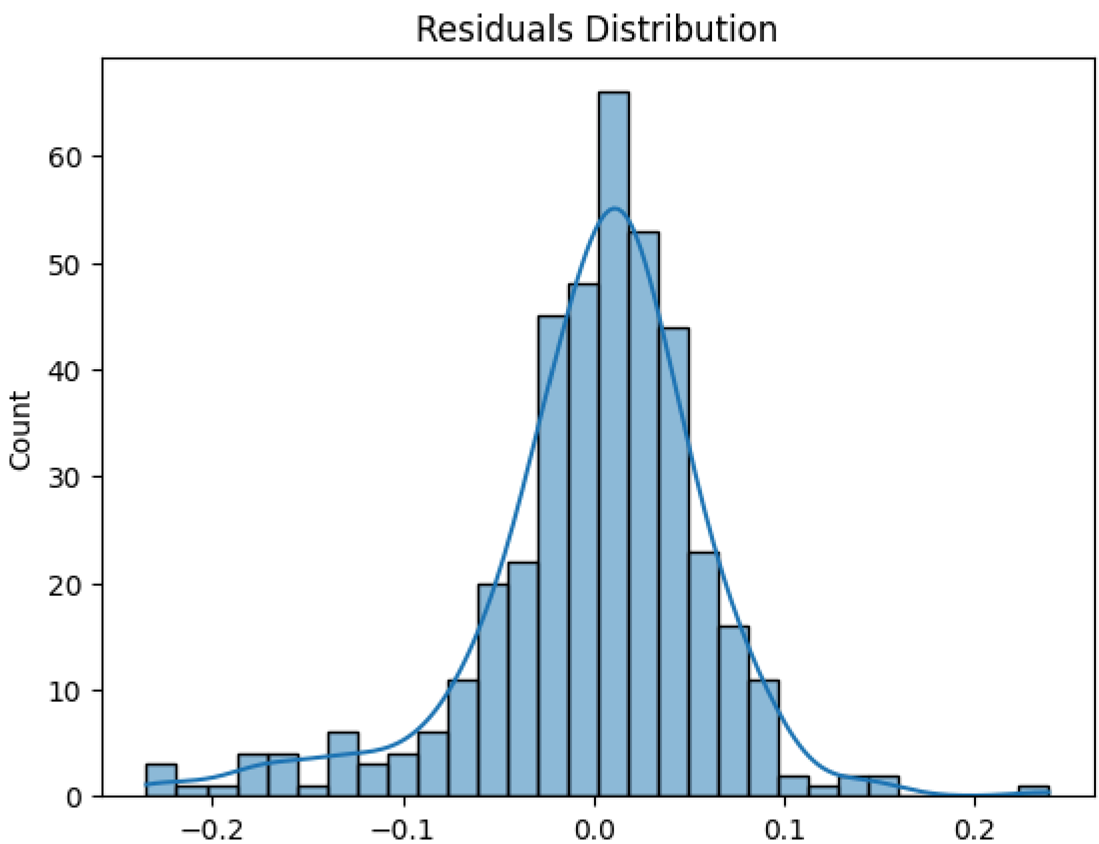

Correlation Matrix
GRE Score
TOEFL Score
University Rating
SOP
LOR
CGPA
Research
Chance of Admit
1.00 0.83 0.64 0.61 0.52 0.83 0.56 0.81
0.83 1.00 0.65 0.64 0.54 0.81 0.47 0.79
0.64 0.65 1.00 0.73 0.61 0.71 0.43 0.69
0.61 0.64 0.73 1.00 0.66 0.71 0.41 0.68
0.52 0.54 0.61 0.66 1.00 0.64 0.37 0.65
0.83 0.81 0.71 0.71 0.64 1.00 0.50 0.88
0.56 0.47 0.43 0.41 0.37 0.50 1.00 0.55
0.81 0.79 0.69 0.68 0.65 0.88 0.55 1.00
1.0
0.9
0.8
0.7
0.6
0.5
0.4

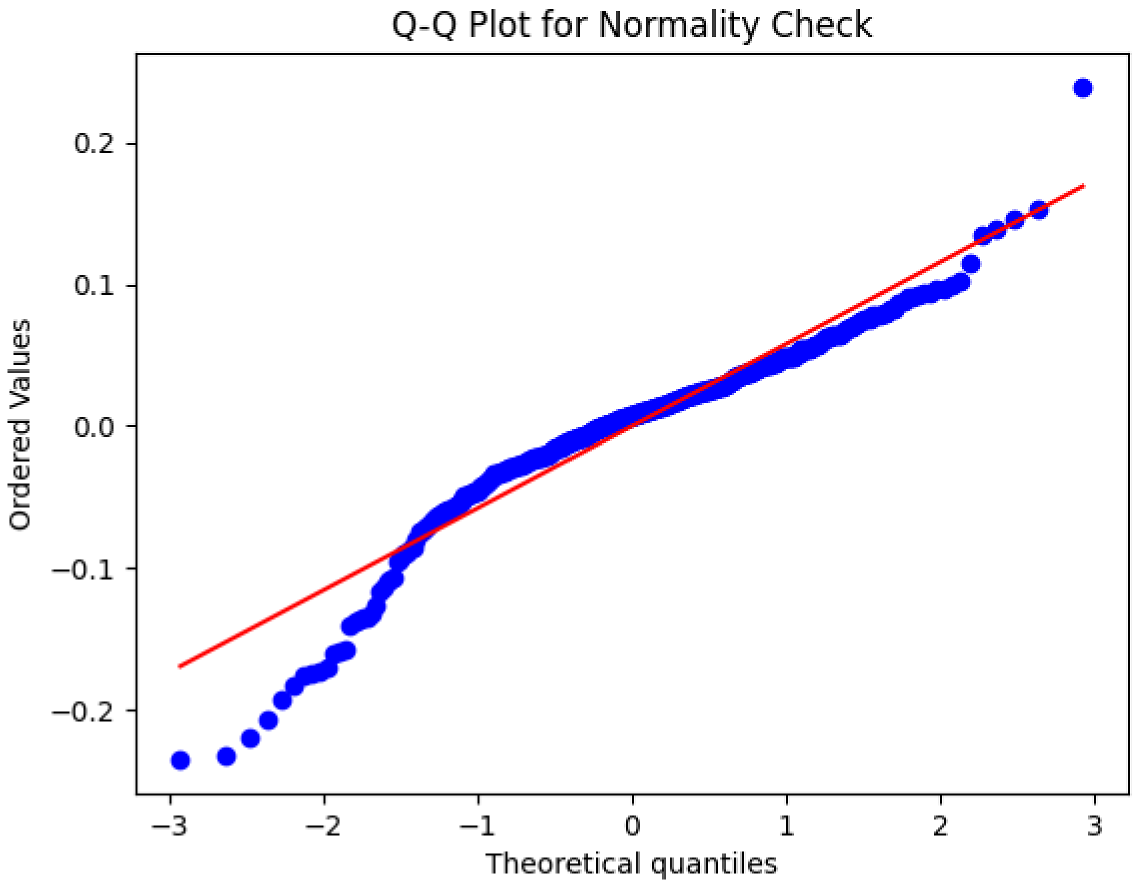
```
Duplicate rows: 0
                            OLS Regression Results
==========================================================================
====
Dep. Variable:      Chance of Admit   R-squared:
0.813
Model:                          OLS   Adj. R-squared:
0.810
Method:               Least Squares   F-statistic:                       2
43.3
Date:              Thu, 12 Dec 2024   Prob (F-statistic):            2.07e
-138
Time:                      12:03:39   Log-Likelihood:                   56
0.67
No. Observations:               400   AIC:                              -1
105.
Df Residuals:                   392   BIC:                              -1
073.
Df Model:                         7
Covariance Type:          nonrobust
==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------
-----------
const                 0.6598      0.017     39.357      0.000       0.627
0.693
GRE Score             0.0251      0.007      3.821      0.000       0.012
0.038
TOEFL Score           0.0155      0.006      2.698      0.007       0.004
0.027
University Rating     0.0023      0.005      0.484      0.628      -0.007
0.012
SOP                   0.0070      0.005      1.387      0.166      -0.003
0.017
LOR                   0.0150      0.005      3.253      0.001       0.006
0.024
CGPA                  0.0676      0.007     10.353      0.000       0.055
0.080
Research              0.0231      0.007      3.099      0.002       0.008
0.038
==========================================================================
====
Omnibus:                     72.597   Durbin-Watson:
1.974
Prob(Omnibus):                0.000   Jarque-Bera (JB):                18
0.042
Skew:                        -0.895   Prob(JB):                       8.02
e-40
Kurtosis:                     5.756   Cond. No.
22.2
==========================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
            Variable       VIF
1          GRE Score  4.489983
```

```
2         TOEFL Score  3.664298
3  University Rating  2.572110
4                SOP  2.785764
5                LOR  1.977698
6               CGPA  4.654540
7           Research  1.518065
Mean of Residuals: 3.5735303605122226e-16
```




Homoscedasticity Check
Residuals
Fitted Values
0.2
0.1
0.0
−0.1
−0.2
0.5
0.6
0.7
0.8
0.9
1.0


Residuals Distribution
Count
60
50
40
30
20
10
0
−0.2
−0.1
0.0
0.1
0.2




```
Mean Absolute Error (MAE): 0.04181871751792343
Root Mean Squared Error (RMSE): 0.06044901917513528
R2 Score: 0.821316189768417
Adjusted R2 Score: 0.8056077229348713
Ridge Regression Performance:
MAE: 0.04183775276281172, RMSE: 0.06046868388638309, R2: 0.821199915356903
4
Lasso Regression Performance:
MAE: 0.10078324542875508, RMSE: 0.1255411111457596, R2: 0.2293119517011262
8
```

In [ ]: