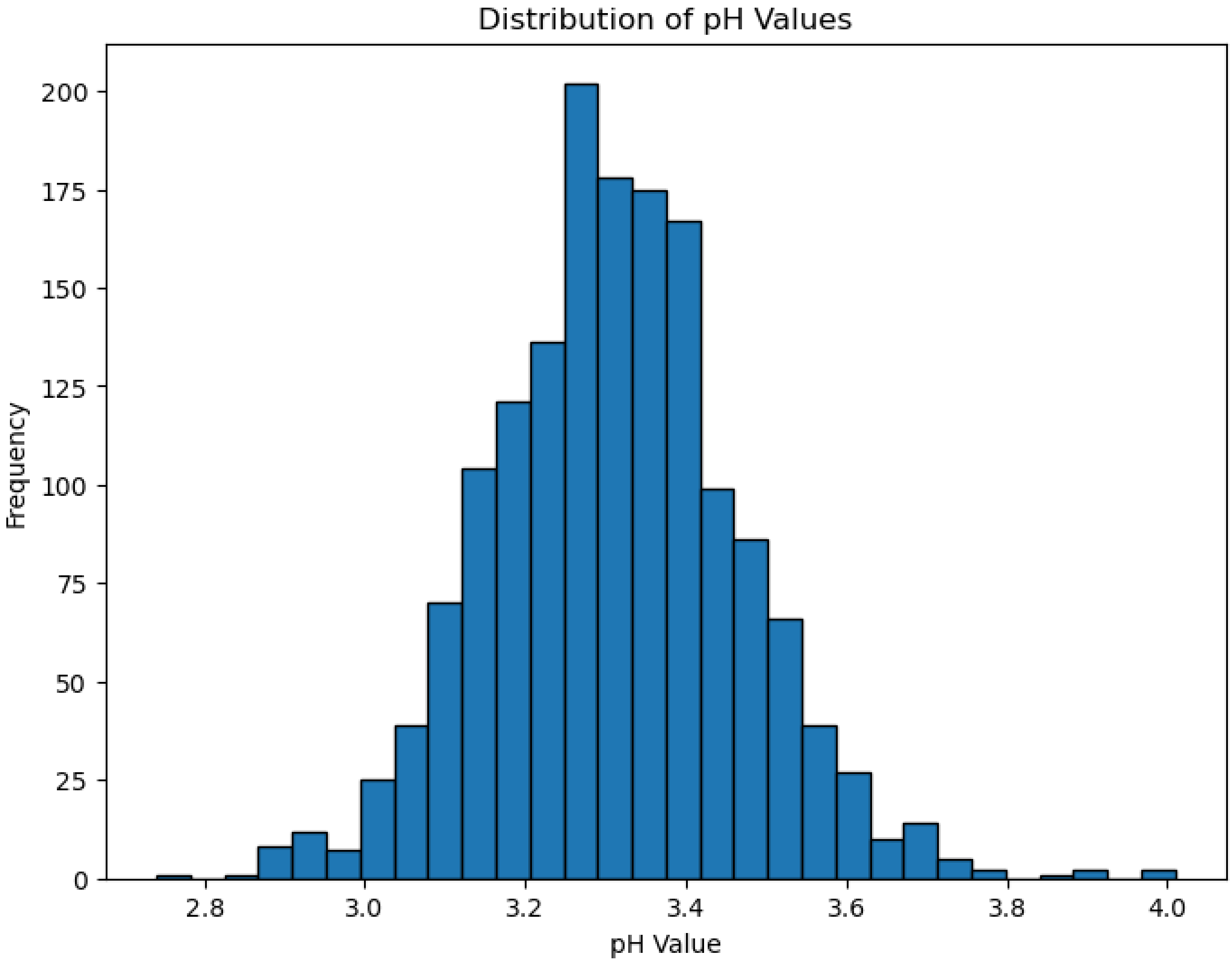
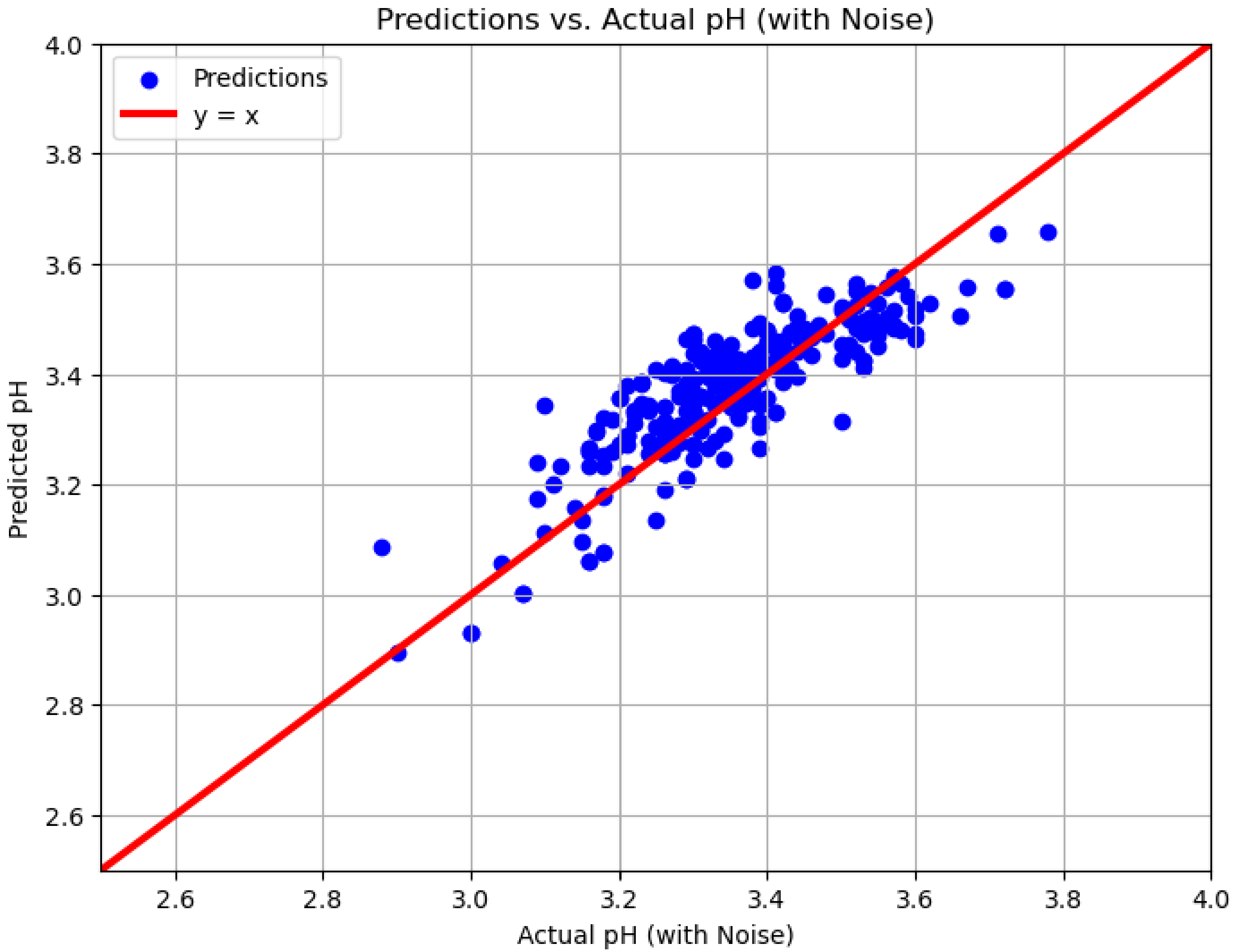
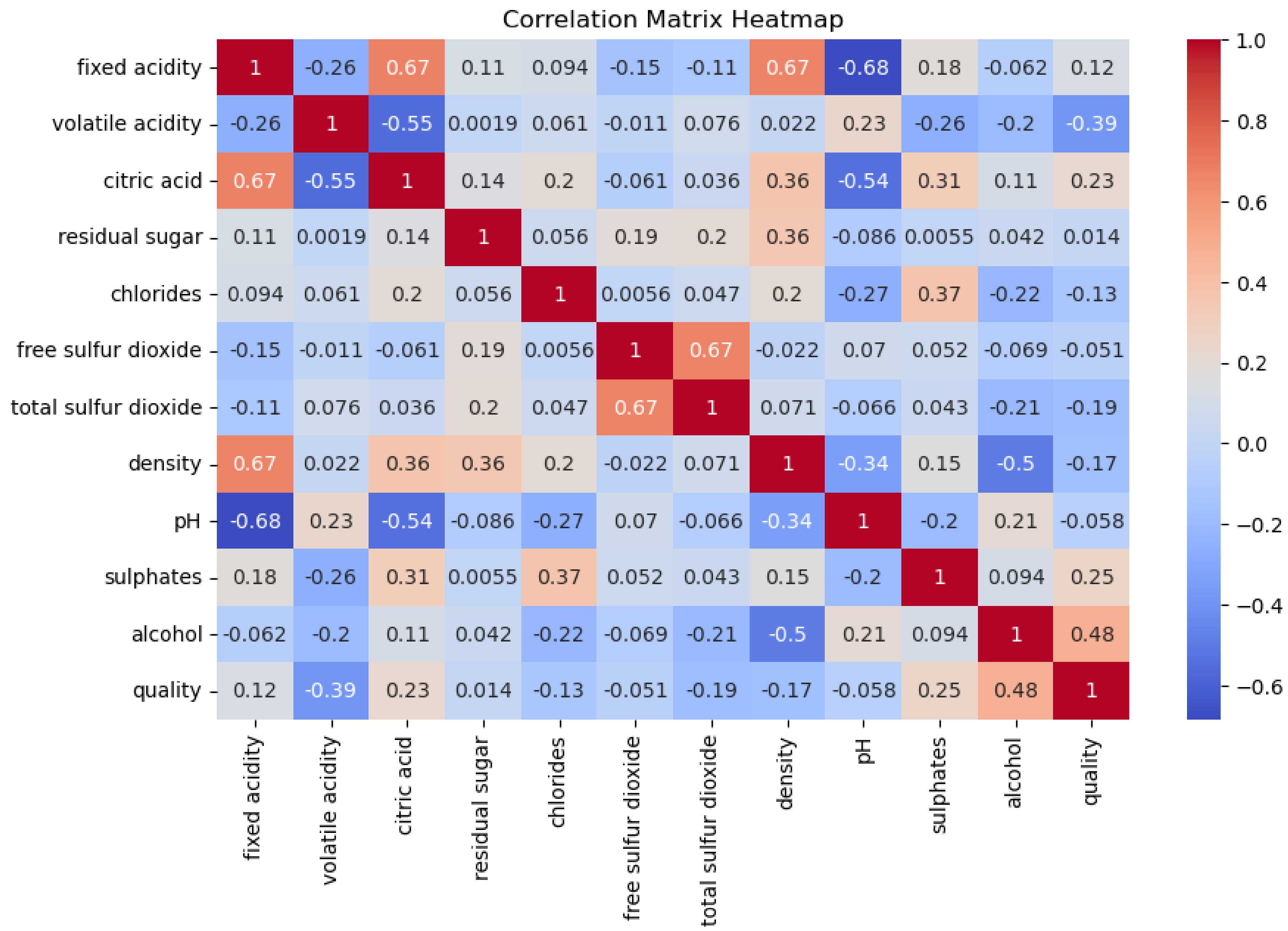
In [9]:

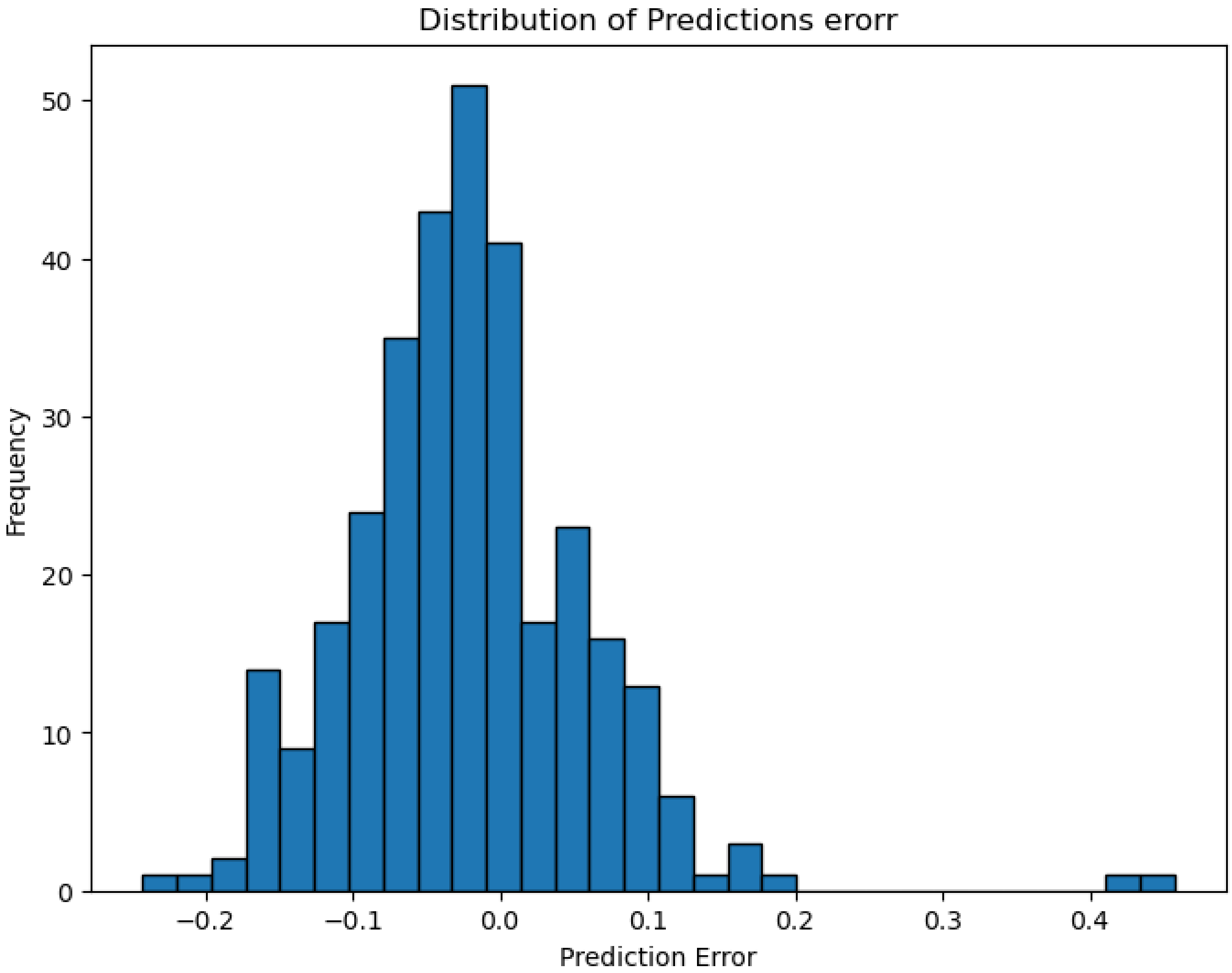
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the wine quality dataset
wine_data = pd.read_csv("winequality-red.csv")

# Data Preprocessing
# Step 1: Handling missing values
wine_data = wine_data.dropna()

# Step 2: Feature selection (choose relevant features)
# In this example, we use all columns except 'pH' as features
target_column = 'pH'
features = wine_data.drop(columns=[target_column])

# Step 3: Data split
# We'll use 80% of the data for training and 20% for testing
total_data_points = len(wine_data)
train_ratio = 0.8
train_data_size = int(total_data_points * train_ratio)
test_data_size = total_data_points - train_data_size

# Split the data into training and test sets
X_train = features.iloc[:train_data_size].values
y_train = wine_data[target_column].iloc[:train_data_size].values
X_test = features.iloc[train_data_size:].values
y_test = wine_data[target_column].iloc[train_data_size:].values


# Calculate the weights for the linear regression model using the closed-form solution
X = np.concatenate((X_train, np.ones((train_data_size, 1))), axis=1)  # Add a column of ones for bias
y = y_train
X_transpose = X.transpose()
weights = np.linalg.inv(X_transpose @ X) @ X_transpose @ y

# Calculate the mean squared error (MSE) on the test data with noise
X_test_with_bias = np.concatenate((X_test, np.ones((test_data_size, 1))), axis=1)
predictions = X_test_with_bias @ weights
errors = (y_test - predictions) ** 2
```

```
mse = np.mean(errors)

# Visualizations (can be retained for analysis)

# Visualization 1: Distribution of Wine pH
plt.figure(figsize=(8, 6))
plt.hist(wine_data['pH'], bins=30, edgecolor='black')
plt.xlabel("pH Value")
plt.ylabel("Frequency")
plt.title("Distribution of pH Values")
plt.show()

# Visualization 2: Predicted vs. Actual pH
plt.figure(figsize=(8, 6))
plt.scatter(y_test, predictions, c='blue', label='Predictions')
plt.plot([2.5, 4.0], [2.5, 4.0], 'r', linewidth=3, label='y = x')
plt.xlabel("Actual pH ")
plt.ylabel("Predicted pH")
plt.legend()
plt.grid()
plt.title("Predictions vs. Actual pH")
plt.xlim([2.5, 4.0])
plt.ylim([2.5, 4.0])
plt.show()

# Visualization 3: Correlation Matrix Heatmap
correlation_matrix = wine_data.corr()
plt.figure(figsize=(10, 6))
plt.title("Correlation Matrix Heatmap")
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm")
plt.show()

# Visualization 4: Distribution of Prediction Errors
plt.figure(figsize=(8, 6))
plt.hist(y_test - predictions, bins=30, edgecolor='black')
plt.xlabel("Prediction Error")
plt.ylabel("Frequency")
plt.title("Distribution of Predictions erorr")
plt.show()

print("Mean Squared Error :", mse)

```


Distribution of pH Values
Frequency
pH Value
0
25
50
75
100
125
150
175
200
2.8
3.0
3.2
3.4
3.6
3.8
4.0


Predictions vs. Actual pH (with Noise)
Predictions
y = x
Predicted pH
Actual pH (with Noise)
4.0
3.8
3.6
3.4
3.2
3.0
2.8
2.6
2.6
2.8
3.0
3.2
3.4
3.6
3.8
4.0


Correlation Matrix Heatmap
fixed acidity 1 -0.26 0.67 0.11 0.094 -0.15 -0.11 0.67 -0.68 0.18 -0.062 0.12
volatile acidity -0.26 1 -0.55 0.0019 0.061 -0.011 0.076 0.022 0.23 -0.26 -0.2 -0.39
citric acid 0.67 -0.55 1 0.14 0.2 -0.061 0.036 0.36 -0.54 0.31 0.11 0.23
residual sugar 0.11 0.0019 0.14 1 0.056 0.19 0.2 0.36 -0.086 0.0055 0.042 0.014
chlorides 0.094 0.061 0.2 0.056 1 0.0056 0.047 0.2 -0.27 0.37 -0.22 -0.13
free sulfur dioxide -0.15 -0.011 -0.061 0.19 0.0056 1 0.67 -0.022 0.07 0.052 -0.069 -0.051
total sulfur dioxide -0.11 0.076 0.036 0.2 0.047 0.67 1 0.071 -0.066 0.043 -0.21 -0.19
density 0.67 0.022 0.36 0.36 0.2 -0.022 0.071 1 -0.34 0.15 -0.5 -0.17
pH -0.68 0.23 -0.54 -0.086 -0.27 0.07 -0.066 -0.34 1 -0.2 0.21 -0.058
sulphates 0.18 -0.26 0.31 0.0055 0.37 0.052 0.043 0.15 -0.2 1 0.094 0.25
alcohol -0.062 -0.2 0.11 0.042 -0.22 -0.069 -0.21 -0.5 0.21 0.094 1 0.48
quality 0.12 -0.39 0.23 0.014 -0.13 -0.051 -0.19 -0.17 -0.058 0.25 0.48 1
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6




```
Mean Squared Error : 0.00710389269362751
```

In [ ]:

```
1
```