|  |  |  |
| --- | --- | --- |
| In [1]: | | **import** numpy **as** np  **import** pandas **as** pd  **import** matplotlib.pyplot **as** plt |
|  |  | **from** scipy.stats **import** norm  **import** seaborn **as** sns |
|  |  |  |
| In | [2]: | *# 1. Generate synthetic wine data using Gaussian distributions*  np**.**random**.**seed(42) num\_samples **=** 1000 |
|  |  | *# Synthetic features based on Gaussian assumptions*  data **=** { |
|  | | "fixed\_acidity": np**.**random**.**normal(7.0, 0.7, num\_samples),  "volatile\_acidity": np**.**random**.**normal(0.5, 0.1, num\_samples), "citric\_acid": np**.**random**.**normal(0.3, 0.1, num\_samples),  "residual\_sugar": np**.**random**.**normal(6.0, 1.5, num\_samples), "chlorides": np**.**random**.**normal(0.05, 0.01, num\_samples),  "alcohol": np**.**random**.**normal(10.0, 1.0, num\_samples)  }  *# Generate wine quality based on a combination of other features # Add randomness to simulate real-world data*  wine\_quality **=** (  0.3 **\*** data["alcohol"] **-**  1.5 **\*** data["volatile\_acidity"] **+**  0.8 **\*** data["citric\_acid"] **+**  np**.**random**.**normal(0, 0.5, num\_samples) |
|  |  | )**.**round()**.**astype(int) |
|  |  |  |
| In | [3]: | *# Clamp wine quality between 3 and 8*  wine\_quality **=** np**.**clip(wine\_quality, 3, 8) |
|  |  | *# Add wine\_quality to the dataset*  data["wine\_quality"] **=** wine\_quality |
|  |  |  |
| In | [4]: | *# Convert to DataFrame*  df **=** pd**.**DataFrame(data) |
|  |  | *# 2. Fit Gaussian models and 3. Compute statistical measures*  print("Statistical Summary:") print(df**.**describe()) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Statistical Summary:  fixed\_acidity | | volatile\_acidity | | citric\_acid | residual\_sugar \ |
| count 1000.000000 | | 1000.000000 | | 1000.000000 | 1000.000000 |
| mean 7.013532 | | 0.507084 | | 0.300583 | 5.971921 |
| std 0.685451 | | 0.099745 | | 0.098345 | 1.540699 |
| min 4.731113 | | 0.205961 | | -0.001951 | 1.605827 |
| 25% 6.546687 | | 0.439376 | | 0.235200 | 4.893869 |
| 50% 7.017710 | | 0.506308 | | 0.299975 | 6.000277 |
| 75% 7.453561 | | 0.572888 | | 0.366092 | 7.000418 |
| max 9.696912 | | 0.819311 | | 0.692624 | 10.864639 |
|  | chlorides | alcohol | wine\_quality | | |
| count | 1000.000000 | 1000.000000 | 1000.00000 | | |
| mean | 0.049507 | 9.953262 | 3.04600 | | |
| std | 0.009924 | 1.007389 | 0.20959 | | |
| min | 0.018233 | 7.100486 | 3.00000 | | |
| 25% | 0.043174 | 9.306307 | 3.00000 | | |
| 50% | 0.049818 | 9.957173 | 3.00000 | | |
| 75% | 0.056391 | 10.612447 | 3.00000 | | |
| max | 0.081129 | 13.098299 | 4.00000 | | |

In [9]:

*#Visualizations - Histogram & PDF*

features **=** ["alcohol", "volatile\_acidity", "citric\_acid", "wine\_quality"]

**for** feature **in** features:

mean **=** np**.**mean(df[feature]) std **=** np**.**std(df[feature])

x **=** np**.**linspace(df[feature]**.**min(), df[feature]**.**max(), 100) pdf **=** norm**.**pdf(x, mean, std)

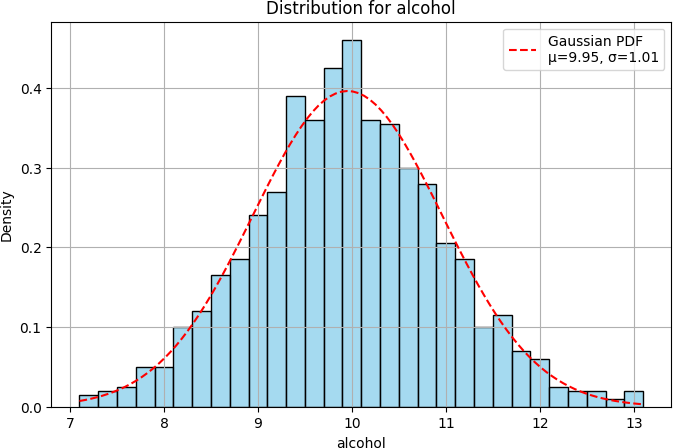
plt**.**figure(figsize**=**(8, 5))

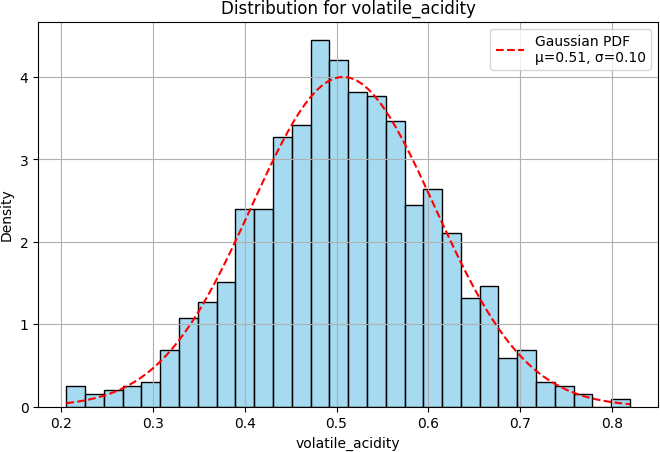
sns**.**histplot(df[feature], bins**=**30, kde**=False**, color**=**'skyblue', stat**=**"density plt**.**plot(x, pdf, 'r--', label**=**f'Gaussian PDF\nμ={mean:.2f}, σ={std:.2f}')

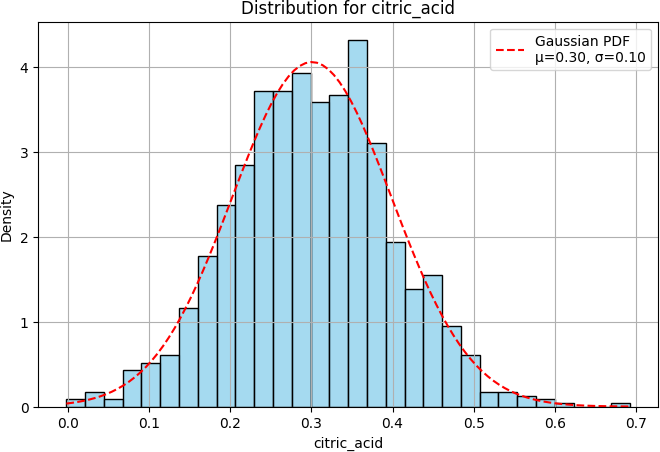
plt**.**title(f'Distribution for {feature}') plt**.**xlabel(feature)

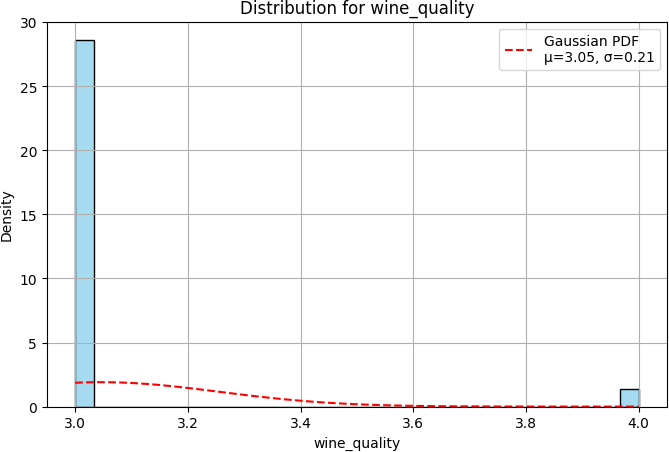
plt**.**ylabel("Density") plt**.**legend()

plt**.**grid(**True**) plt**.**show()









In [7]:

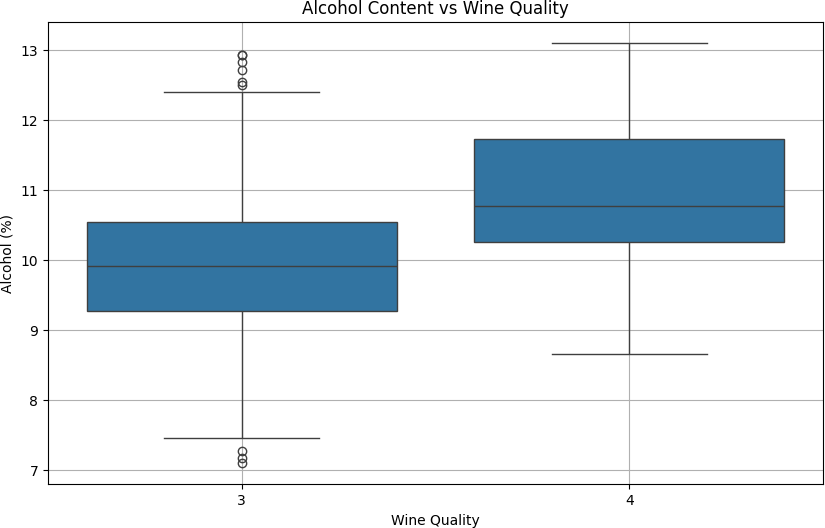
*# Analyze feature vs wine\_quality*

plt**.**figure(figsize**=**(10, 6))

sns**.**boxplot(x**=**"wine\_quality", y**=**"alcohol", data**=**df) plt**.**title("Alcohol Content vs Wine Quality")

plt**.**xlabel("Wine Quality") plt**.**ylabel("Alcohol (%)") plt**.**grid(**True**)

plt**.**show()



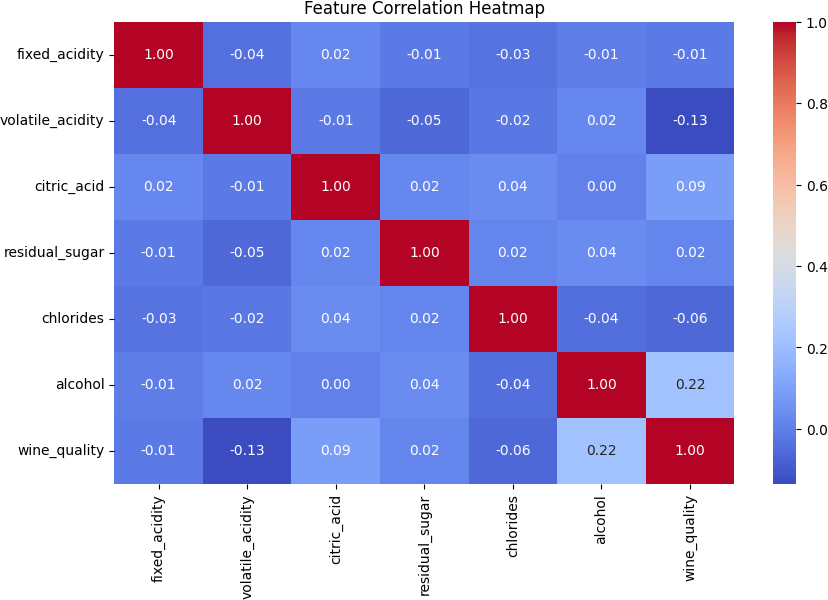
In [8]:

*# Correlation heatmap*

plt**.**figure(figsize**=**(10, 6))

sns**.**heatmap(df**.**corr(), annot**=True**, cmap**=**"coolwarm", fmt**=**".2f") plt**.**title("Feature Correlation Heatmap")

plt**.**show()



In [ ]: