```
from typing_extensions import dataclass_transform
#Introduction
# In the exercise, We will perform the following tasks
# Load and study data
# View the distributions of the various features in the data set and calculate their central tendency
# Create a new pandas series that contain the details of the representative factor for quality



# Task-1: Load and study the data
# Load the data and study its features such as:
   # fixed acidity
   # velocity acidity
   # citric acid etc



# Load "numpy" and "pandas" for manipulating numbers and dataframes
# Load "matplotlib" and "seaborn" for data visualiazation


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


data=pd.read_csv("Wine Quality Dataset.csv")
data.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulph
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	

Next steps: Generate code with `data` View recommended plots

```
data.shape
```

```
(4898, 12)
```

```
data.index
```

```
RangeIndex(start=0, stop=4898, step=1)
```

```
data.columns
```

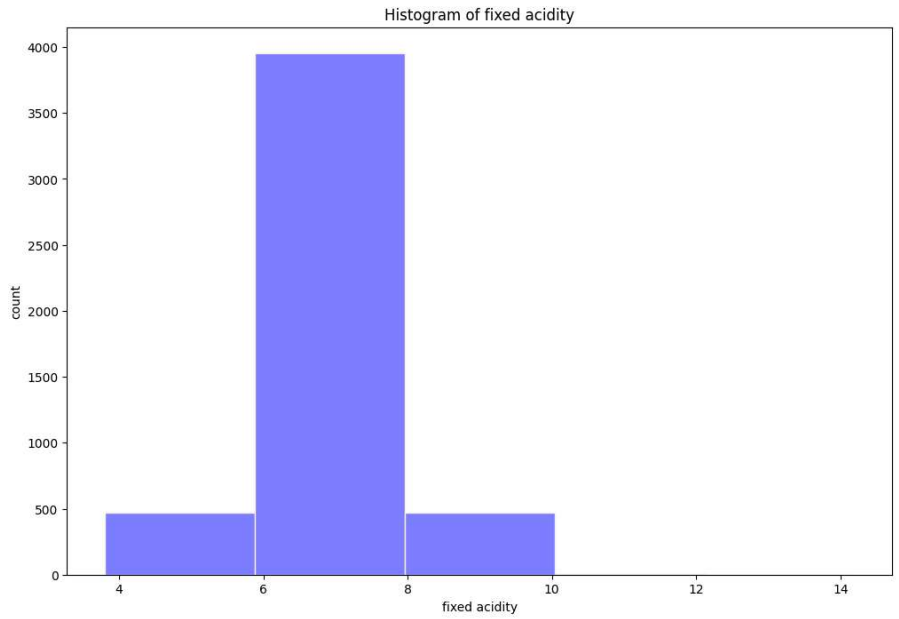
```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         4898 non-null   float64
 1   volatile acidity      4898 non-null   float64
 2   citric acid           4898 non-null   float64
 3   residual sugar        4898 non-null   float64
 4   chlorides             4898 non-null   float64
 5   free sulfur dioxide   4898 non-null   float64
 6   total sulfur dioxide  4898 non-null   float64
 7   density               4898 non-null   float64
 8   pH                    4898 non-null   float64
```

```
 9   sulphates             4898 non-null   float64
 10  alcohol               4898 non-null   float64
 11  quality               4898 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 459.3 KB
```

```
# Create a histogram "fixed acidity" feature
plt.figure(figsize=(12,8))
sns.histplot(data=data, x="fixed acidity", color="blue", edgecolor="linen", alpha=0.5, bins=5)
plt.title("Histogram of fixed acidity")
plt.xlabel("fixed acidity")
plt.ylabel("count")
plt.show()
```



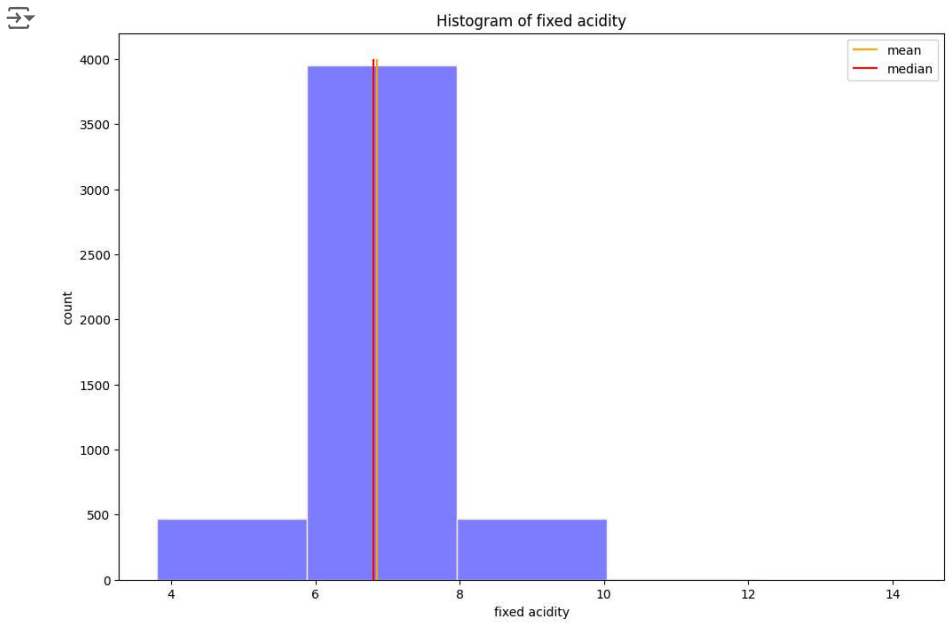

```
# Observation
# we observe that the histogram is normally distributed.
# The maximum count of values for fixed acidity lines in between 6 to 8

# Lets see the measures of central tendency in working
  # Mean
  # Median
  # Mode


# Calculate the mean "fixed acidity" feature
data["fixed acidity"].mean()
```

```
6.854787668436097
```

```
# Calculate the median "fixed acidity" feature
data["fixed acidity"].median()
```

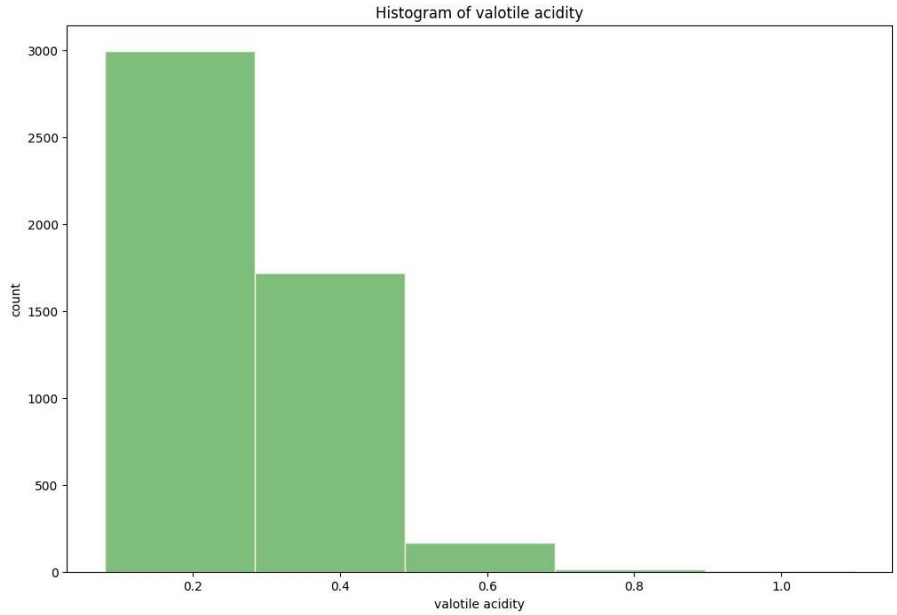
```
6.8
```

```
# Create a histogram of the "fixed acidity" feature and also show the mean and median
plt.figure(figsize=(12,8))
sns.histplot(data=data, x="fixed acidity", color="blue", edgecolor="linen", alpha=0.5, bins=5)
plt.title("Histogram of fixed acidity")
plt.xlabel("fixed acidity")
plt.ylabel("count")
plt.vlines(data["fixed acidity"].mean(),ymin=0,ymax=4000,color="orange",label="mean")
plt.vlines(data["fixed acidity"].median(),ymin=0,ymax=4000,color="red",label="median")
plt.legend()
plt.show()
```




```
# Observations
# We can see that mean and median clear representative of the data
# Mean and meadian are very close to each other
# We can choose either of the parameter say mean as the measure of central trendecy
```

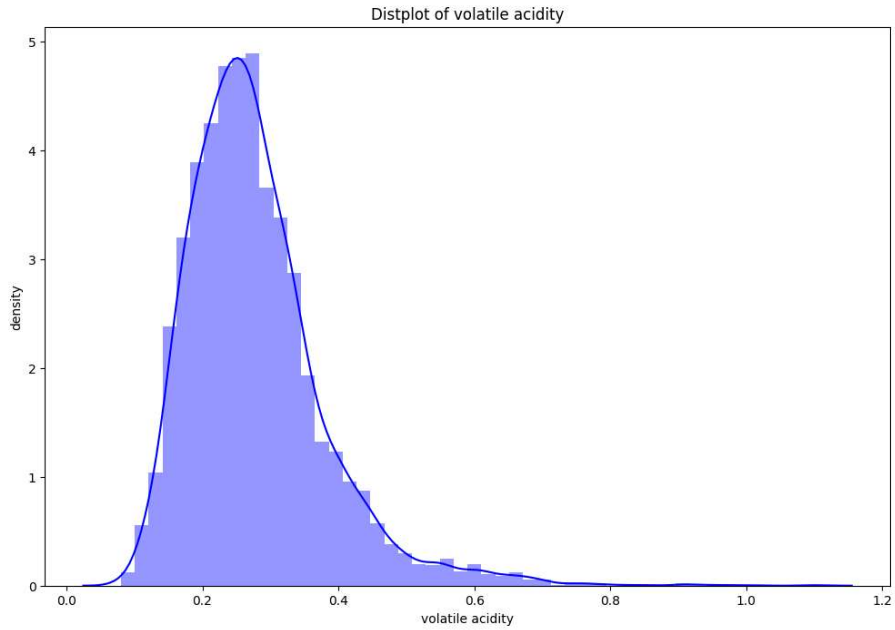
```
# Create a histogram of the "velocity acidity" feature
plt.figure(figsize=(12,8))
sns.histplot(data=data,x="volatile acidity",color="green",edgecolor="linen",alpha=0.5,bins=5)
plt.title("Histogram of valotile acidity")
plt.xlabel("valotile acidity")
plt.ylabel("count")
plt.show()
```




```
# Observations
# We observe that this histogram is not well distributed,it is skewed a little towards the right
# As we have been skewness, therefore we can check the distribution using distplot function


# Do you guys skewness?
# No need to worry, we are here!Let's tackle the skewness together



# Plot distplot using "volatile acidity" feature

plt.figure(figsize=(12,8))
sns.distplot(data["volatile acidity"],color="blue")
plt.title("Distplot of volatile acidity")
plt.xlabel("volatile acidity")
plt.ylabel("density")
plt.show()
```

```
<ipython-input-19-1824f45a3d74>:4: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```

```
sns.distplot(data["volatile acidity"],color="blue")
```




```
# Observations:
# The above plot shows the normal distribution

# The normal distribution is described by the mean and satndard deviation
# The normal distribution is often refered to as a "bell curve" because of it's shape:
    # The median and mean are equal
    # It has only one mode
    # It is symmetric, meaning it drecreses the same amount on the left and the right of the center


# Calculate skewness of volatile acidity
data["volatile acidity"].skew()
```

```
1.5769795029952025
```

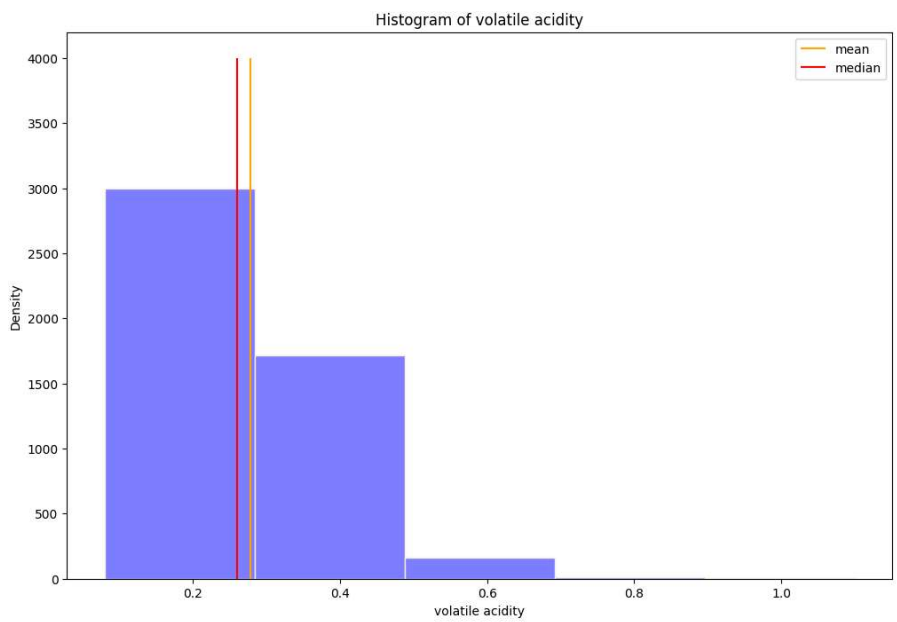
```
# Observations

# We can clearly see that the skewness value is greater than 1, hence it is positively skewed


data["volatile acidity"].mean()
```

```
0.27824111882400976
```

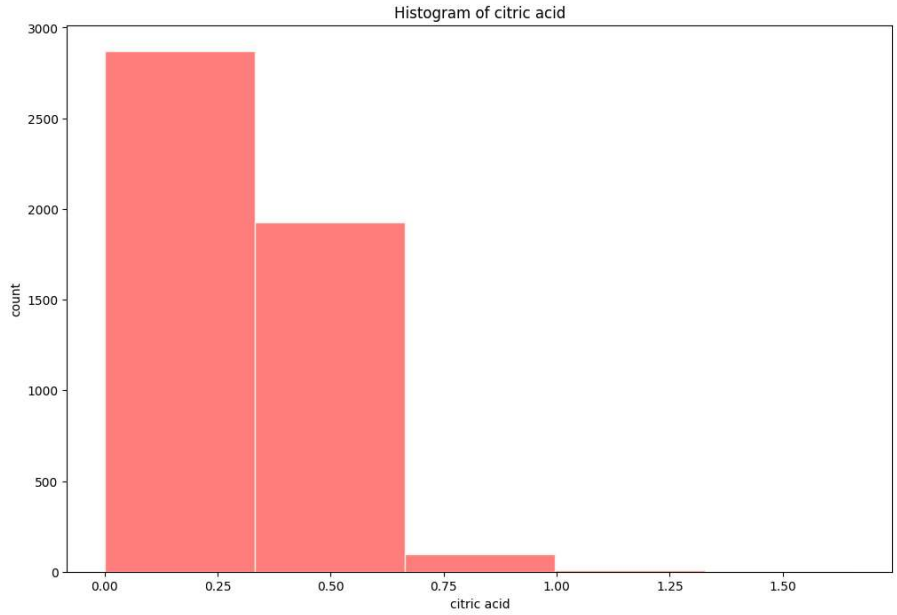
```
data["volatile acidity"].median()
```

```
0.26
```

```
# Create a histogram of the "volatile acidity" feature and also show the mean and median
plt.figure(figsize=(12,8))
sns.histplot(data=data, x="volatile acidity", color="blue", edgecolor="linen", alpha=0.5, bins=5)
plt.title("Histogram of volatile acidity")
plt.xlabel("volatile acidity")
plt.ylabel("Density")
plt.vlines(data["volatile acidity"].mean(),ymin=0,ymax=4000,color="orange",label="mean")
plt.vlines(data["volatile acidity"].median(),ymin=0,ymax=4000,color="red",label="median")
plt.legend()
plt.show()
```




```
# Create a histogram of the "citric acid" feature
plt.figure(figsize=(12,8))
sns.histplot(data=data,x="citric acid",color="red",edgecolor="linen",alpha=0.5,bins=5)
plt.title("Histogram of citric acid")
plt.xlabel("citric acid")
plt.ylabel("count")
plt.show()
```



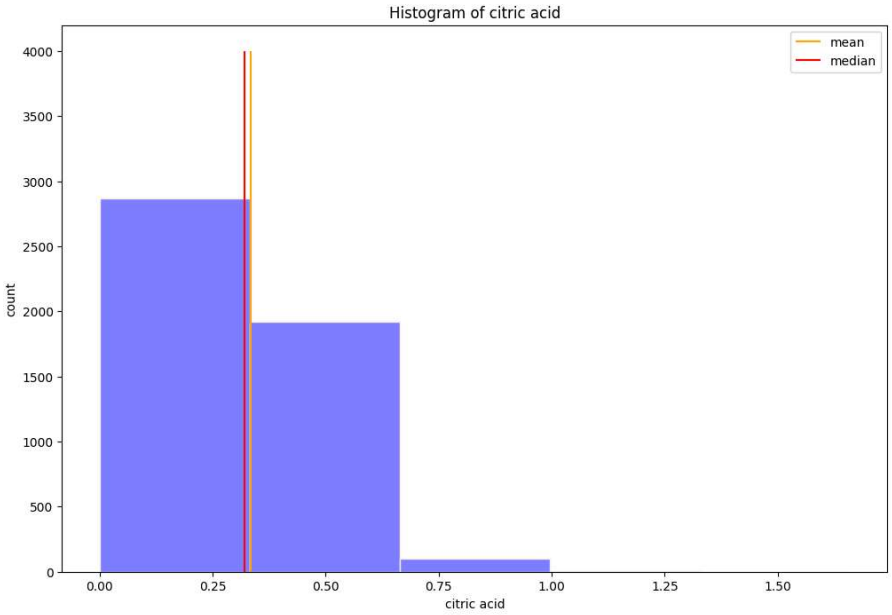

```
# Calculate the mean "citric acid" feature
data["citric acid"].mean()
```

```
0.33419150673744386
```

```
# Calculate the median "citric acid" feature
data["citric acid"].median()
```

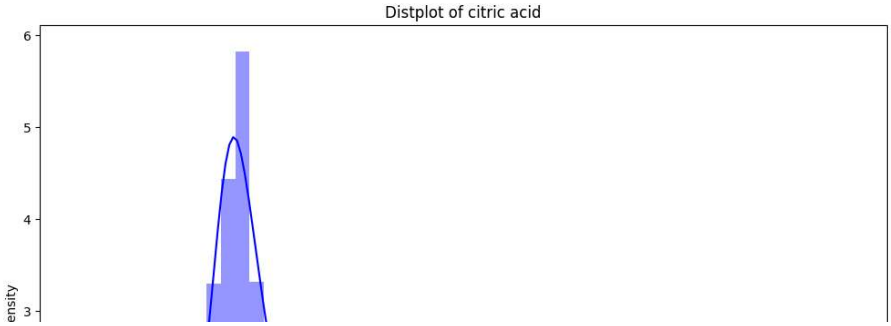
```
0.32
```

```
# Create a histogram of the "citric acid" feature and also show the mean and median
plt.figure(figsize=(12,8))
sns.histplot(data=data, x="citric acid", color="blue", edgecolor="linen", alpha=0.5, bins=5)
plt.title("Histogram of citric acid")
plt.xlabel("citric acid")
plt.ylabel("count")
plt.vlines(data["citric acid"].mean(),ymin=0,ymax=4000,color="orange",label="mean")
plt.vlines(data["citric acid"].median(),ymin=0,ymax=4000,color="red",label="median")
plt.legend()
plt.show()
```




```
# Observations:
# The mean and median is close to each other and the difference between them is very well
# We can safely choose the mean as the safely central tendency here


# Calculate citric acidity feature
plt.figure(figsize=(12,8))
sns.distplot(data["citric acid"],color="blue")
plt.title("Distplot of citric acid")
plt.xlabel("citric acid")
plt.ylabel("density")
plt.show()
```

```
<ipython-input-32-3bb90c3f5be9>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data["citric acid"],color="blue")
```




```
# Create a count plot of the quality feature

plt.figure(figsize=(12,8))

sns.countplot(data["quality"])

plt.title("Count plot of quality")
plt.xlabel("quality")
plt.ylabel("count")
plt.show()
```