**Univariate Analysis — Intro and Implementation**

**Univariate analysis using seaborn: statistical data visualization**



astronaut looking at data, by [DALL.E 2](https://openai.com/dall-e-2/)

As a data scientist, what is the first step you do when you receive a new and unfamiliar set of data? Well, we start familiarizing ourselves with the data. This post focuses on answering that question by analyzing only one variable at a time, which is called a univariate…

Similar to my other posts, learning will be achieved through practice questions and answers. I will include hints and explanations in the questions as needed to make the journey easier. Lastly, the notebook that I used to create this exercise is also linked in the bottom of the post, which you can download, run and follow along.

Let’s get started!

*(All images, unless otherwise noted, are by the author.)*

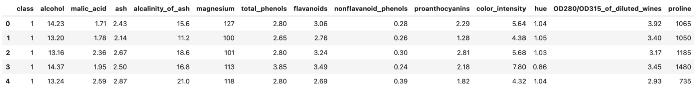
**Data Set**

In order to practice univariate analysis, we are going to use a data set about the chemical analysis of various wines from [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/datasets/Wine), which is based on “An Extendible Package for Data Exploration, Classification and Correlation” ([Forina, M. et al, 1998](https://archive.ics.uci.edu/ml/datasets/Wine)) and can be downloaded from [this link](https://gist.github.com/fmnobar/26904b27a6efbea70b1c96169a74c8db) (CC BY 4.0).

Let’s start with importing the libraries we will be using today, then read the data set into a dataframe and look at the top 5 rows of the dataframe to familiarize ourselves with the data.

# Import libraries  
import pandas as pd  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
%matplotlib inline  
  
# Read the data  
df = pd.read\_csv('wine.csv')  
  
# Return top 5 rows of the dataframe  
df.head()

Results:



As we see above, these are the chemical analysis of various wines. We will be mainly using a few columns, which I will explain briefly below:

* “class” — Refers to the cultivar where the wine comes from. There are three cultivars in this study (1, 2 and 3)
* “alcohol” — Demonstrates the alcohol content of the wine
* “malic\_acid” — Is the level of this specific acid, which is present in wines. Wines from cool climate areas have a higher malic acid level compared to wines from warmer climates

Now that we are familiar with the columns we will be using, let’s star the analysis.

**Frequency Analysis**

Frequency analysis is one of the fundamental concepts in descriptive analysis where count of instances that an event occurs is studied. For example, if we roll a dice 12 times and get the following results:

[1, 3, 6, 6, 4, 5, 2, 3, 3, 6, 5, 1]

Then frequency of occurrence for 1 is 2, since there are two times that 1 came in the tosses. Now let’s see how this concept can be implemented in Python. We will be using “value\_counts” method to see how many times each distinct value of a variable occurs in the dataframe. But since “value\_counts” does not include null values, let’s first see if there are any null values.

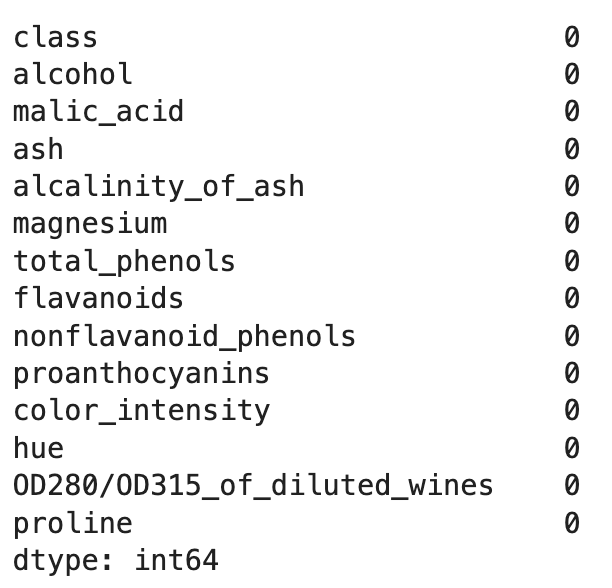
**Question 1:**

How many null values exist in the datframe and in what columns?

**Answer:**

# Return null values  
df.isnull().sum()

Results:



Based on the results, none of the columns include any null values therefore, we can go ahead and use “value\_counts”. Let’s continue with our frequency analysis.

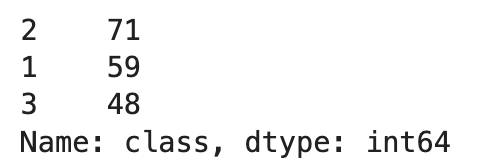
**Question 2:**

The data set includes wine information from three different cultivars, as indicated in column “class”. How many rows per class are there in the data set?

**Answer:**

# Apply value\_counts to the df['class'] column  
df['class'].value\_counts()

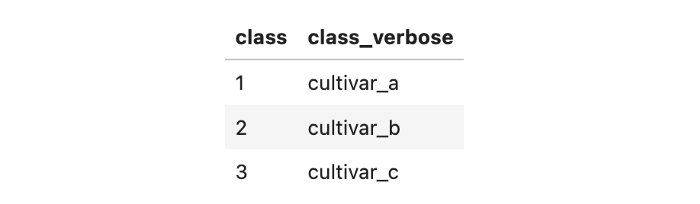
Results:



As we see, there are three classes (as stated in the question), there are 71 instances from cultivar 2, 59 from cultivar 1 and 48 from cultivar 3.

**Question 3:**

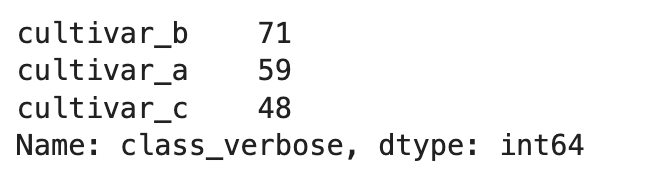
Create a new column named “class\_verbose” that replaces values of the “class” column as defined in the table below. Then determine how many instances of each of the new class exists, which should match the results from Question 2.



**Answer:**

# Replace according to the mapping table provided above  
df['class\_verbose'] = df['class'].replace({1 : 'cultivar\_a', 2 : 'cultivar\_b', 3 : 'cultivar\_c'})  
  
# Compare results  
df.class\_verbose.value\_counts()

Results:



As expected, the number of instances per class remained the same as the results of Question 2.

**Numerical Summarization**

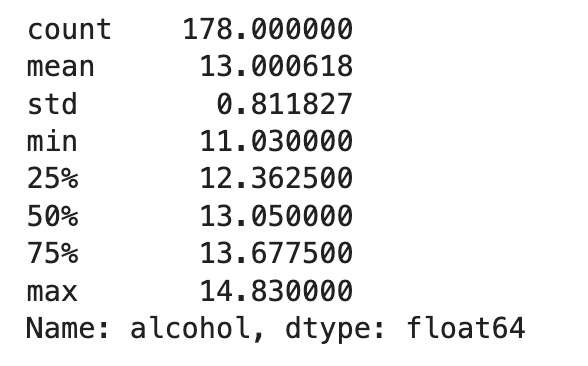
In this section we are going to focus more on the quantitative variables and explore ways to summarize such columns. One easy approach is using the “describe” method. Let’s see how it works in an example.

**Question 4:**

Create a numerical summary of the “alcohol” column of the data set using the “describe” method.

**Answer:**

# Use describe method  
df['alcohol'].describe()



Descriptions are self-explanatory and as you can see it is a very convenient method to take an overview of the distribution of the data, instead of manually generating these values. Let’s manually generate some of them in the next question for practice.

**Question 5:**

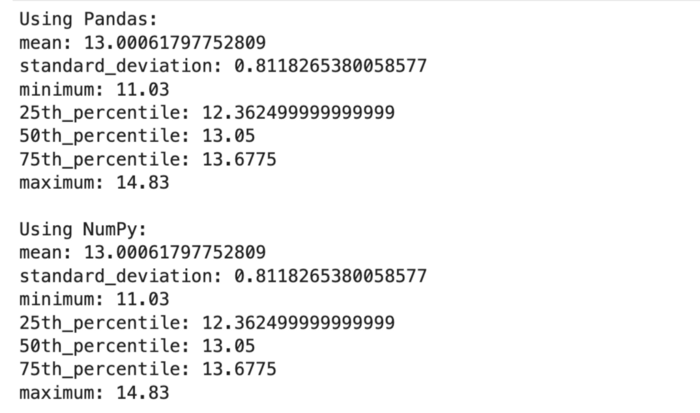
Return the following values of the “alcohol” column of the data set: mean, standard deviation, minimum, 25th, 50th and 75th percentile, and maximum.

**Answer:**

These can be calculated using Pandas and/or NumPy (among others). I have provided both approaches here for reference.

# Approach 1 - Using Pandas  
print(f"Using Pandas:")  
print(f"mean: {df.alcohol.mean()}")  
print(f"standard\_deviation: {df.alcohol.std()}")  
print(f"minimum: {df.alcohol.min()}")  
print(f"25th\_percentile: {df.alcohol.quantile(0.25)}")  
print(f"50th\_percentile: {df.alcohol.quantile(0.50)}")  
print(f"75th\_percentile: {df.alcohol.quantile(0.75)}")  
print(f"maximum: {df.alcohol.max()}\n")  
  
# Approach 2 - Using NumPy  
print(f"Using NumPy:")  
print(f"mean: {np.mean(df.alcohol)}")  
print(f"standard\_deviation: {np.std(df.alcohol, ddof = 1)}")  
print(f"minimum: {np.min(df.alcohol)}")  
print(f"25th\_percentile: {np.percentile(df.alcohol, 25)}")  
print(f"50th\_percentile: {np.percentile(df.alcohol, 50)}")  
print(f"75th\_percentile: {np.percentile(df.alcohol, 75)}")  
print(f"maximum: {np.max(df.alcohol)}\n")

Results:



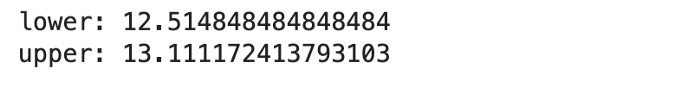
**Question 6:**

How does the mean of the alcohol content of wines with “malic\_acid” smaller than 1.5 compare to that of the wines with “malic\_acid” greater than or equal to 1.5?

**Answer:**

lower\_bound = np.mean(df['alcohol'][df.malic\_acid < 1.5])  
upper\_bound = np.mean(df['alcohol'][df.malic\_acid >= 1.5])  
  
print(f"lower: {lower\_bound}")  
print(f"upper: {upper\_bound}")

Results:



**Graphical Summarization**

In this section we will be looking at visualizing quantitative variables. We will be using histograms and boxplots, which I will introduce before starting the questions.

**Histograms**

Histogram is a visulization tool representing the distribution of one or more variables by counting the number of instances (or observations) within each bin. In this post we will focus on univariate histograms, using seaborn’s “histplot” class. Let’s look at an example.

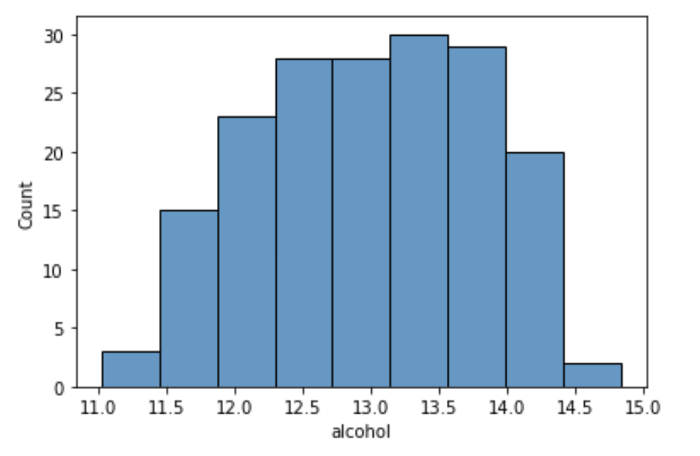
**Question 7:**

Create a histogram of the alcohol levels in the data set.

**Answer:**

# Create the histogram  
sns.histplot(df.alcohol)  
plt.show()

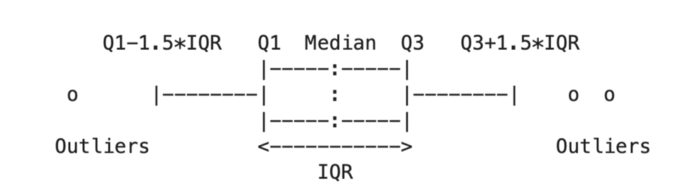
Results:



This shows how many instances are within each of the alcohol content bins. For example, it looks like that the bin containing 13.5 alcohol level has the highest number of instances.

**Boxplots**

Boxplots demonstrate the distribution of quantitative data. The box shows the quartiles of the data (i.e. 25th percentile or Q1, 50th percentile or median and 75th percentile or Q3), while the whiskers show the rest of the distribution, except for what is determined as outliers, defined as extending beyond 1.5 times the Inter-Quartile Range (IQR) below Q1 or above Q3. IQR is the distance between Q1 and Q3, as demonstrated below.



Let’s look at examples.

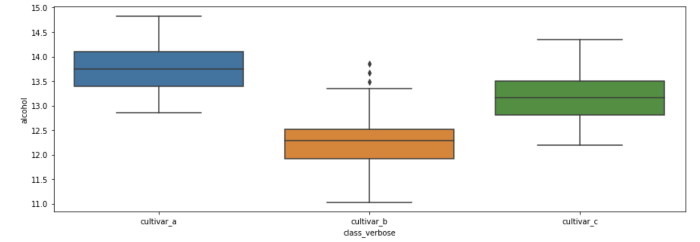
**Question 8:**

Create a boxplot comparing alcohol distribution across three cultivars.

**Answer:**

# Assign a figure size  
plt.figure(figsize = (15, 5))  
  
# Create the box plots  
sns.boxplot(data = df, x = 'class\_verbose', y = 'alcohol')  
plt.show()

Results:



**Stratification**

One of the ways to find patterns in the data is to break it down into smaller subsets or strata and analyze those strata separately. There might be new findings for each startum. In order to demonstrate this technique, we are going to look at some examples.

**Question 9:**

Create a new column named “malic\_acid\_level”, which breaks down the values of the “malic\_acid” column into three segments as described below:

1. Minimum to 33rd percentile
2. 33rd percentile to 66th percentile
3. 66 percentile to maximum

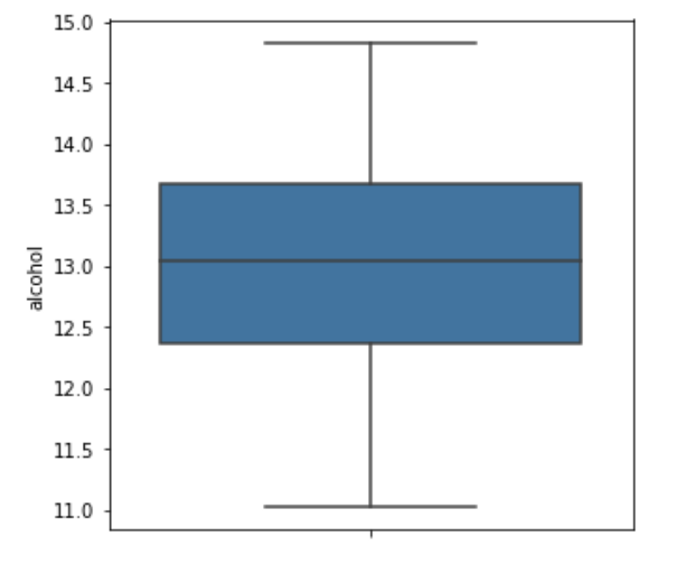
Then create a set of boxplots for alcoholo distribution at each of the strata. Do you see any new patterns as result?

**Answer:**

First, let’s create a boxplot for the alcobol level, before dividing “malic\_acid” into the starata described in the question. Then we will apply the stratification and compare the results visually.

# Assign a figure size  
plt.figure(figsize = (5, 5))  
  
# Create the box plots  
sns.boxplot(data = df, y = 'alcohol')  
plt.show()

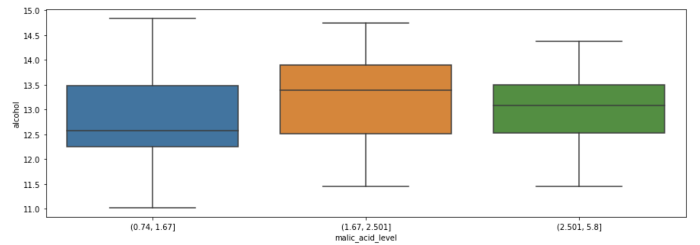
Results:



As we see above, Q1, median and Q3 are around 12.4, 13 and 13.7, respectively. Let’s see how these values vary across “malic\_acid” starta.

# Calculate the cut levels  
minimum = np.min(df.malic\_acid)  
p33 = np.percentile(df.malic\_acid, 33)  
p66 = np.percentile(df.malic\_acid, 66)  
maximum = np.max(df.malic\_acid)  
  
# Create the new column  
df['malic\_acid\_level'] = pd.cut(df.malic\_acid, [minimum, p33, p66, maximum])  
  
# Assign a figure size  
plt.figure(figsize = (15, 5))  
  
# Create the box plots  
sns.boxplot(data = df, x = 'malic\_acid\_level', y = 'alcohol')  
plt.show()

Results:



This is quite interesting. Recall that the median alcohol level was around 13? Now we see some variation of medians across “malic\_acid” levels. For example, we see there is a relatively large difference between the medians of “malic\_acid” of the blue and orange boxplots, which correspond to two different starata, representing low and mid range “malic\_acid” levels, respectively. Another observation is that the blue boxplot has a much larger range (from ~11 to ~14.8), while the green one, with larger “malic\_acid” levels, has a smaller range (from ~11.5 to ~14.4).

Let’s stratify this one futher layer down as an exercise.

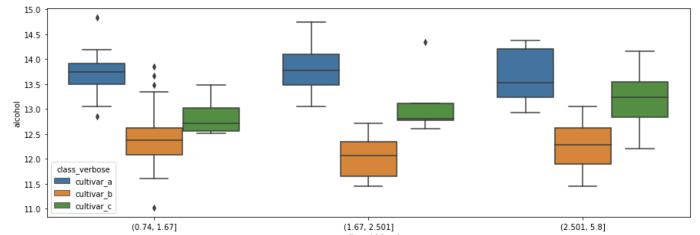
**Question 10:**

Create similar box plots as the previous question but for each of the cultivars.

**Answer:**

# Assign a figure size  
plt.figure(figsize = (15, 5))  
  
# Create the box plots  
sns.boxplot(data = df, x = 'malic\_acid\_level', y = 'alcohol', hue = 'class\_verbose')  
plt.show()

Results:



Next, let’s try to summarize these in a tabular fashion.

**Pivot Tables**

Pivot tables are tabular representation of grouped values that aggregate data within certain discrete categories. Let’s look at examples to understand pivot tables in practice.

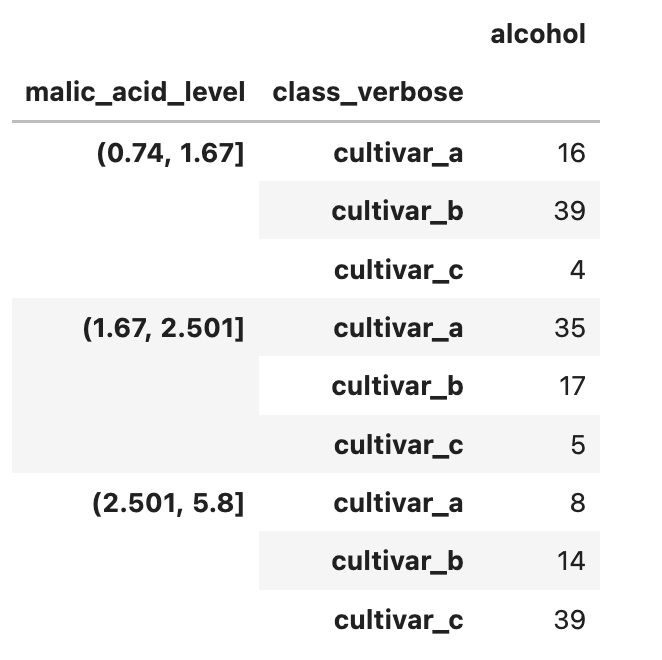
**Question 11:**

Create a pivot table indicating how many instances of alcohol content are available for each cultivar within each malic acid level.

**Answer:**

# Create the pivot table  
pd.pivot\_table(df[['malic\_acid\_level', 'class\_verbose', 'alcohol']], index = ['malic\_acid\_level', 'class\_verbose'], aggfunc = 'count')

Results:



Let’s read one of the rows to understand the results. The first row tells us that there are 16 instances of “cultivar\_a” within the “malic\_acid\_level” of (0.74, 1.67]. As you can see in the script above, we are using “count” as the aggregate function in this pivot table since the question asked how many instances are within those discrete classes. There are other aggregate functions that can be used. Let’s try one of them in the next example.

**Question 12:**

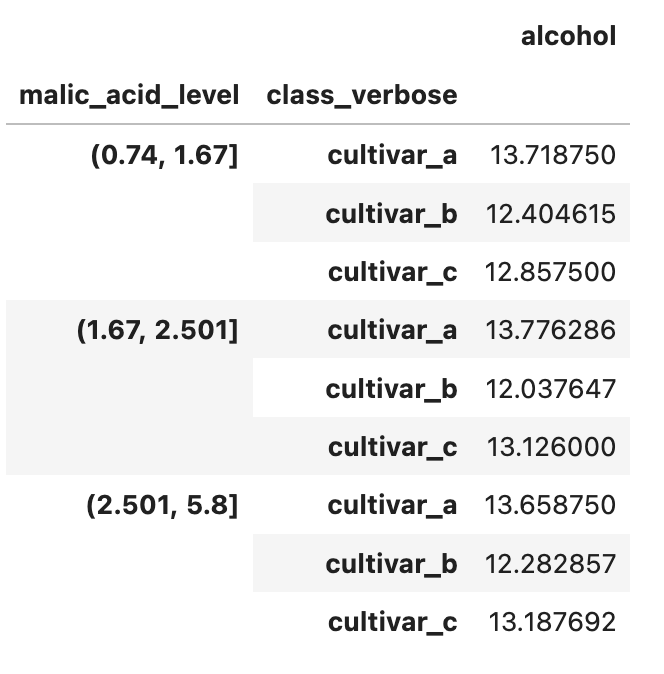
Create a pivot table demonstrating the average alcohol level for each of the cultivars within each of the malic acid levels.

**Answer:**

Note this time we want to implement an aggregate function to calculate the average.

# Create the pivot table  
pd.pivot\_table(df[['malic\_acid\_level', 'class\_verbose', 'alcohol']], index = ['malic\_acid\_level', 'class\_verbose'], aggfunc = 'mean')

Results:



**Notebook with Practice Questions**

Below is the notebook with both questions and answers that you can download and practice.

**Conclusion**

In this post, we talked about how we can leverage univariate analysis as the very first step in getting to know a new space through data. Before starting to make any inferences about the data, we would want to learn what the data is about and univariate analysis equips us with a tool to get to know each of the variables, one at a time. As part of the univariate analysis, we learned how to implement frequency analysis and how to summarize the data into various subsets / strata and how to leverage visualization tools such as histograms and boxplots to better understand the distribution of the data.