Week 01 Example Statistics in Python 3

This Notebook introduces basic statistical analysis from week 01 using Python 3.x.

Jupyter notebooks blend Markdown (rich formatted text) with software code and output to ease the learning process.

Version: 01

Author: Chris Kennedy

In [1]:
```
import pandas as pd
from scipy import stats
```

In [2]:
```
dfWine = pd.read_excel(r'W1 - Wine Quality.xlsx')
```

In [3]:
```
dfWine.describe()
```

Out[3]:

	Wine #	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfu dioxid
count	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.00000
mean	3249.000000	7.215307	0.339666	0.318633	5.443235	0.056034	30.525319	115.74457
std	1875.666681	1.296434	0.164636	0.145318	4.757804	0.035034	17.749400	56.52185
min	1.000000	3.800000	0.080000	0.000000	0.600000	0.009000	1.000000	6.00000
25%	1625.000000	6.400000	0.230000	0.250000	1.800000	0.038000	17.000000	77.00000
50%	3249.000000	7.000000	0.290000	0.310000	3.000000	0.047000	29.000000	118.00000
75%	4873.000000	7.700000	0.400000	0.390000	8.100000	0.065000	41.000000	156.00000
max	6497.000000	15.900000	1.580000	1.660000	65.800000	0.611000	289.000000	440.00000

Practice Question 1

What is the 99% confidence interval for the average alcohol level of a bottle of wine?

In [4]:
```
n = dfWine['alcohol'].count()
print("# Wines: %6.2f" % n)
```

```
# Wines: 6497.00
```

In [5]:
```
avg = dfWine['alcohol'].mean()
print("Average Alcohol: %6.4f" % avg)
```

```
Average Alcohol: 10.4918
```

In [6]:
```
stderr = dfWine['alcohol'].sem()
print("Standard Error: %6.4f" % stderr)
```

```
Standard Error: 0.0148
```

In [7]:
```
t_CI = stats.t.ppf(0.995,df = n - 1)
print("T-statistic: %6.4f" % t_CI)
```

```
T-statistic: 2.5766
```

In [8]:
```
moe = t_CI * stderr
print("Margin of error: %6.4f" % moe)
```

```
Margin of error: 0.0381
```

In [9]:
```
print("Lower: %6.4f" % (avg - moe))
print("Upper: %6.4f" % (avg + moe))
```

```
Lower: 10.4537
Upper: 10.5299
```

In [10]:
```
# Concise using stats package directly:
stats.t.interval(0.99, loc=avg, scale=stderr, df = n-1)
```

Out[10]:
```
(10.453674609436044, 10.529927052869633)
```

Practice Question 2

What is the 90% confidence interval around the proportion of white wines that are rated very good quality (7 or higher)?

In [11]:
```
filteredDF = dfWine[dfWine['type'] == 'white']
n = filteredDF['type'].count()
```

In [12]:
```
qfilteredDF = filteredDF[filteredDF['quality'] >= 7]
nq = qfilteredDF['type'].count()
```

In [13]:
```
proportion  = nq / n
print("Proportion: %5.2f%%" % (proportion*100))
```

```
Proportion: 21.64%
```

In [14]:
```
stderr = (proportion * (1 - proportion) / n)**0.50
print("Standard error:  %5.2f%%" % (stderr*100))
```

```
Standard error:  0.59%
```

In [15]:
```
z_CI = stats.norm.ppf(0.95)
```

In [16]:
```
print ("Z for 90%%: %6.4f" % z_CI)
```

```
Z for 90%: 1.6449
```

In [17]:
```
stats.norm.interval(0.90, loc=proportion*100, scale=stderr*100)
```

Out[17]:
```
(20.67364423441555, 22.6093284074791)
```

Question 3

Can you conclude (at the 5% significance level) that the average fixed acid level for all wines is above 7.2?

Creating a simple helper function:

In [18]:
```
def getStatistic(x, H0, stderr):
    return (x - H0) / stderr
```

$H_0: \mu \leq 7.2$

$H_a: \mu > 7.2$

In [19]:
```
H0 = 7.20
avg = dfWine['fixed acidity'].mean()
stderr = dfWine['fixed acidity'].sem()
tStatistic = getStatistic(avg, H0, stderr)
nWines = dfWine['fixed acidity'].count()
print("T-stat: %7.4f" %tStatistic)
dfWine['fixed acidity'].describe()
```

```
T-stat:  0.9517
```

Out[19]:
```
count    6497.000000
mean        7.215307
std         1.296434
min         3.800000
25%         6.400000
50%         7.000000
75%         7.700000
max        15.900000
Name: fixed acidity, dtype: float64
```

Right tail test due to alternative hypothesis.

In [20]:
```
prob_value = 1 - stats.t.cdf(tStatistic, df=nWines-1)
print(prob_value)
```

```
0.17064341607837663
```

Reject the null hypothesis if probability value is less than significance level.

In [24]:
```
print("Reject Null") if prob_value < 0.05 else print("Cannot Reject Null")
```

```
Cannot Reject Null
```

End of Notebook!