Descriptive statistics using `statistics` module

Disclaimer: These notes and examples are an adaptation of the references listed at the end. They are compiled to fit the scope of this specific course.

Introduction

Python offers a comprehensive set of tools that will help you work with different kinds of data. This handout will list some of the most popular statistical analysis packages and will focus on using the `statistics` module available via python's `standard library`.

Libraries for Statistical Analysis/ Data Science

The following table summarizes some of the most popular and widely used libraries used in the broad area of statistical analysis/data science. Visualization libraries will be introduced later.

Library/module	Description
statistics	This module is a part of python's standard library. Used mainly for descriptive statistics. Always evolving. Details at: https://docs.python.org/3/library/statistics.html and this handout!
stattools	Contains a comprehensive set of statistical learning and inference algorithms. Ultimate python package providing descriptive statistics, estimation, and inference for statistical models. Details at https://pypi.org/project/stattools/ and https://github.com/artemmavrin/StatTools

Library/module	Description
statsmodels	This comprehensive library provide tools for statistical data exploration, statistical testing and estimation. Details at: https://www.statsmodels.org/stable/index.html
numpy	A 3rd party library. Widely used for numerical computations. Optimized for vector calculations. Contains many statistical analysis capabilities. Details at: https://numpy.org/doc/stable/reference/routines.statistics.html
scipy	A 3rd party library, based on numpy. Offers features on top of numpy's capabilities. Details at: https://docs.scipy.org/doc/scipy/reference/stats.html
pingouin	A new data analysis library. This is my favorite! This library offers most common statistical tests and analysis in a very user friendly manner. It integrates very well with pandas. Details at: https://pingouin-stats.org/index.html
pandas	A 3rd party library. Based on numpy. Most widely used data manipulation library. Details at: https://pandas.pydata.org/docs/reference/
sklearn	Machine learning library in python. Developed by google. Features ML algorithms. Details at: https://scikit-learn.org/stable/index.html
tensorflow	An Artificial Intelligence (AI) library. Used for developing large-scale neural networks. Used for building deep learning models as well. Details at: https://www.tensorflow.org/
keras	Used for building and training Deep Neural Networks. Makes statistical modeling, working with images and text a lot easier. Details at: https://keras.io/

A quick (descriptive) statistics review

The following are the three types of commonly used measures for describing data:

Measure	Description	Common measures
Central tendency	Tells you about the data centers.	Arithmetic mean (average) Geometric mean Median Mode

Measure	Description	Common measures
Variability (Spread)	Tells you about the spread of the data.	Range Variance Inter-quantile Standard deviation
Correlation (or joint variability)	Tells you about the relation between a pair of variables	Covariance Correlation coefficient

Python's `statistics` module

This module provides basic descriptive measures. If you have already installed python's standard library, no separate installation is needed. Import it using

```
import statistics
```

`statistics` methods for central tendency calculation

These are the most common methods (functions) available for calculating central tendency measures:

Method	Description	Example

statistics.mean(data)
&
statistics.fmean(data)
	Return the sample arithmetic mean of data (iterable or a sequence).
	
	fmean( ) turns the data to float first, then calculates the mean.
	fmean( ) always return a float. fmean( ) is faster than mean( )
		statistics.mean([2,5,8])
		
		statistics.fmean([2,5,8])

statistics.geometric_mean(data)	Return the central tendency or typical value of the data using the product of the values (as opposed to the arithmetic mean which uses their sum)	statistics.geometric_mean([3,10,52])

Method	Description	Example

statistics.median(data)	This method will return median (middle value).
	If the number of values is even, the **mean of the middle two values** is returned
	If the number of values is odd, the **middle value** is returned
		statistics.median([2,9,5])
		
		statistics.median([2,9,5,13])

statistics.median_low(data)¶	If the number of values is odd, the middle value will be returned
	If the number of values is even, the smaller of the middle two will be returned
		statistics.median_low([2,9,5])
		
		statistics.median_low([2,9,5,13])

statistics.median_high(data)	If the number of values is odd, the middle value will be returned
	If the number of values is even, the larger of the middle two will be returned
		statistics.median_high([2,9,5])
		
		statistics.median_high([2,9,5,13])

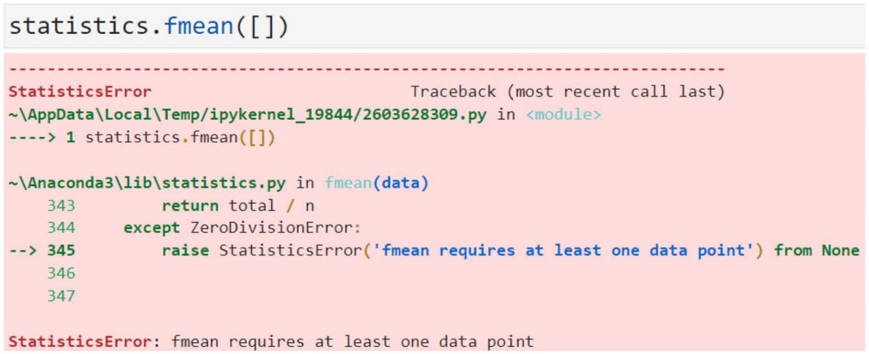
statistics.mode(data)	It will return the most frequent data point (most typical value).
	If there is more than one mode, it will return the first mode encountered.
	mode supports numerical and non-numerical data types
		statistics.mode([3,3,4,1])
		
		statistics.mode(['a', 'a', 'b', 'a'])

statistics.multimode(data)	If there is more than one mode, this method will return all of them in a list	statistics.multimode([3,3,4,'a','a'])

Note 1: These functions accept a `sequence` or an `iterable`. Hence, `range`, `list`, `tuple`, `set`, and `dictionary` are all legitimate input types to these methods.

Question: A dictionary is typically compromised of pairs of `key: value`. If you input a dictionary to these methods, will the method consider `key` values, `value` values, or both?

Note 2: If `data` passed to these functions is empty, `statisticsError` will be raised. Here is an example of passing an empty list to the `fmean` method:



```
statistics.fmean([])
```

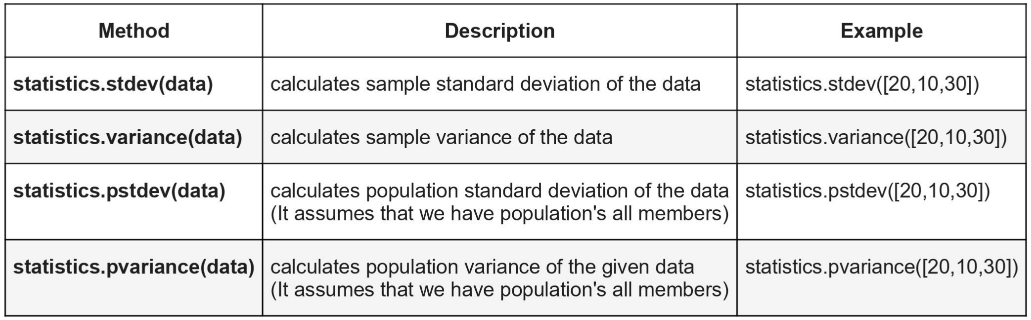
```
---------------------------------------------------------------------------
StatisticsError                           Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_19844/2603628309.py in <module>
----> 1 statistics.fmean([])

~\Anaconda3\lib\statistics.py in fmean(data)
    343         return total / n
    344     except ZeroDivisionError:
--> 345         raise StatisticsError('fmean requires at least one data point') from None
    346 
    347 

StatisticsError: fmean requires at least one data point
```

`statistics` methods for variability calculation

These are the most common functions available for calculating variability (spread).



Method	Description	Example
statistics.stdev(data)	calculates sample standard deviation of the data	statistics.stdev([20,10,30])
statistics.variance(data)	calculates sample variance of the data	statistics.variance([20,10,30])
statistics.pstdev(data)	calculates population standard deviation of the data (It assumes that we have population's all members)	statistics.pstdev([20,10,30])
statistics.pvariance(data)	calculates population variance of the given data (It assumes that we have population's all members)	statistics.pvariance([20,10,30])

`statistics` methods for joint variability

These are the most common functions to describe the relationship between two inputs.

Method	Description	Example
statistics.covariance(data_1, data_2)	calculates sample covariance of **data_1** and **data_2** both **data_1** and **data_2** must be of the same length both **data_1** and **data_2** must have length of more than 1	statistics.covariance(range(4), range(2,15,3))
statistics.correlation(data_1, data_2)	calculates pearson correlation coefficient of **data_1** and **data_2** both **data_1** and **data_2** must be of the same length both **data_1** and **data_2** must have length no less than two	statistics.correlation([1,2,3], [7,2,4])
statistics.linear_regression(data_1, data_2)	calculates the slope and intercept of simple linear regression of **data_2 = slope * data_1 + intercept** both **data_1** and **data_2** must be of the same length both **data_1** and **data_2** must have length no less than two	statistics.linear_regression([2,3, 5], [7,2,8])

Note: `covariance`, `correlation`, and `linear_regression` methods all are new in python 3.10.

Other methods

With each new version, `statistics` module offers more capabilities. This module has other functions, such as:

- `quantiles` for dividing *data* into continuous intervals with equal probability
- `fabs` (float abs) for calculating absolute value and returning a float number
- `fsum` (float sum) for calculating the sum of the data and returning a float number

Last feature of `statistics` library that this handout will cover is `statistics.NormalDist` class (type). The following snippet shows a few straightforward calculations using this class.

```python
import statistics

# create two normally distributed objects
```

```
my_dist_1 = statistics.NormalDist(mu=10, sigma = 2)
my_dist_2 = statistics.NormalDist(mu=13, sigma = 5)

print(f'cdf of N(mu = 10, sigma = 2) at 7 is {my_dist_1.cdf(7)}')
print(f'zscore of N(mu = 10, sigma = 2) at 13 is {my_dist_1.zscore(13)}')
print(f'overlap between two normals is {my_dist_1.overlap(my_dist_2)}')
```

yields

```
cdf of N(mu = 10, sigma = 2) at 7 is 0.06680720126885809
zscore of N(mu = 10, sigma = 2) at 13 is 1.5
overlap between two normals is 0.5063542366912017
```

References

This document is an adaption from the following references:

1- *Python online documentation*; available at https://docs.python.org/3/

2- *A Byte of Python*; available at https://Python.swaroopch.com/

3- *Introduction to programming in Python*; available at https://introcs.cs.princeton.edu/Python/home

4- *Introduction to Computation and Programming Using Python: With Application to Understanding Data*, John Guttag, The MIT Press, 2nd edition, 2016

5- *Introduction to Python for Computer Science and Data Science*, Paul J. Deitel, Harvey Deitel, Pearson, 1st edition, 2019

6- *Data Mining for Business Analytics*, Galit Shmueli, Peter C. Bruce, Peter Gedeck, Nitin R. Patel, Wiley, 2020

7- *Introduction to Programming Using Java*; available at http://math.hws.edu/javanotes/

8- https://betterprogramming.pub/

9- https://www.learnpython.org/

10- *Business Statistics and Analytics in Practice*, Bowerman et al. Ninth edition, McGraw Hill, 2019

11- https://realpython.com/