Exercise 1: the t-test for non-Gaussian distributions

In this first exercise we will look at how the 1-sample t-test (which tests whether a Gaussian distribution has mean 0) behaves when the Gaussian assumption is not valid.

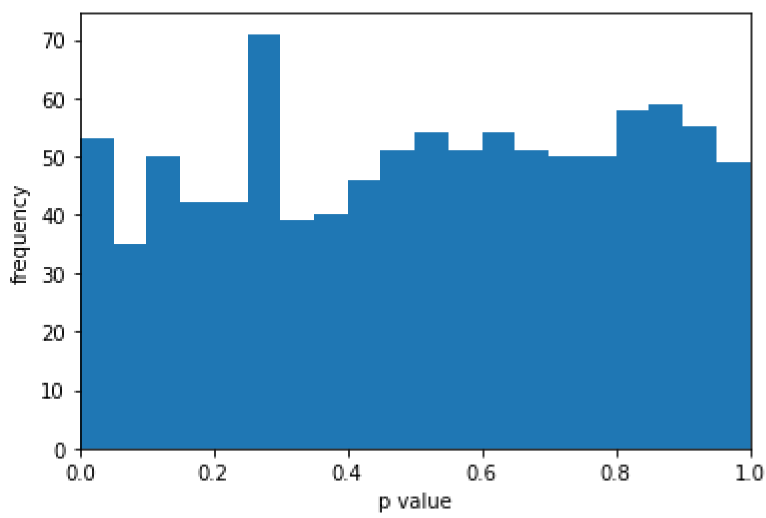
First let's look at when the assumption is valid.

A. Generate 10,000 random numbers from a Gaussian distribution with mean 0 and variance 1. Use a t-test to test the null hypothesis that the sample comes from a distribution with mean 0. Do this 1000 times. Make a histogram of the p-values you get. How often is the null hypothesis rejected at p<.05?

Result should look like this:

Show code

```
53 out of 1000 tests with p<.05
```

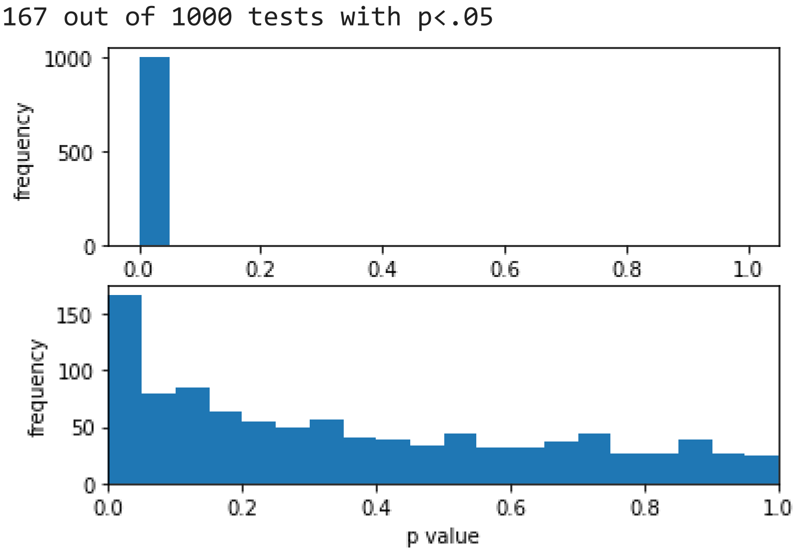


B. Now do the same but with a Gaussian of mean 0.1 and variance 1. Then with mean 0.01 and variance 1.

Result should look like this:

Show code

```
1000 out of 1000 tests with p<.05
167 out of 1000 tests with p<.05
```



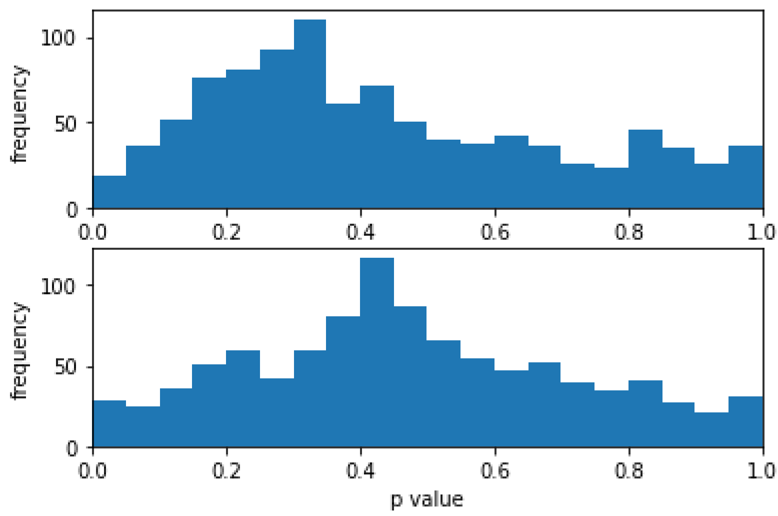
C. Now let's a non-Gaussian distribution. First, a symmetrical distribution with large kurtosis, i.e. heavy tails. The [Cauchy distribution] is symmetrical about 0, it's median and mode are 0. It has such a heavy tail that the mean is not even defined! What happens if you use a t-test to ask whether 10,000 values generated from a standard Cauchy have mean 0? Now do it again, but with a sample size of 3 rather than 10,000.

You should find a deviation from 5% signifiance rate. Was it in the direction you expected to see, or the opposite? Why do you think it is this way round?

Result should look like this:

Show code

```
19 out of 1000 tests with p<.05
28 out of 1000 tests with p<.05
```



D. Now, a distribution with light tails. The [Rademacher distribution] takes the value +1 with probability 0.5 and -1 with probability 0.5. It has very low kurtosis: there is a much smaller chance of getting an occasional extreme value, compared to a Gaussian distribution.

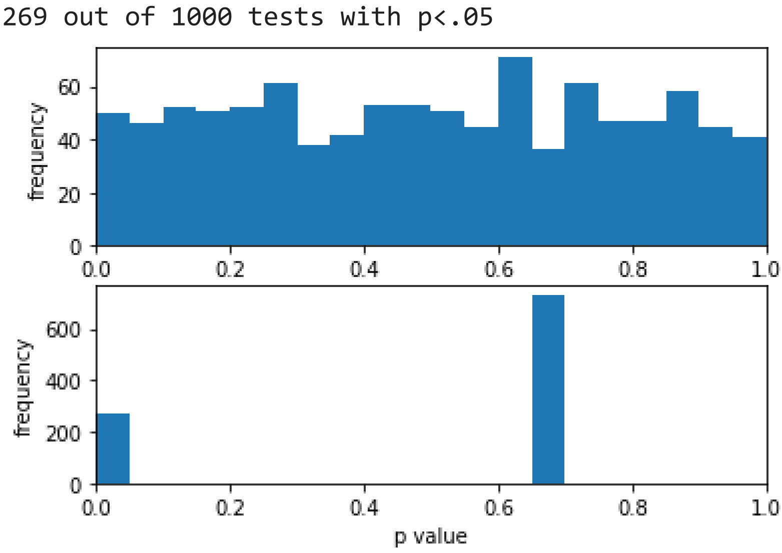
Use a t-test to ask whether 10,000 values from a Rademacher distribution have mean 0. Now do it again, but with a sample size of 3 rather than 10,000.

There is a deviation from 5% signifiance rate. Is it in the direction you expected to see, or the opposite? Why do you think it is this way round? What is the effect of sample size and why does it have this effect?

Result should look like this:

Show code

```
50 out of 1000 tests with p<.05
269 out of 1000 tests with p<.05
```



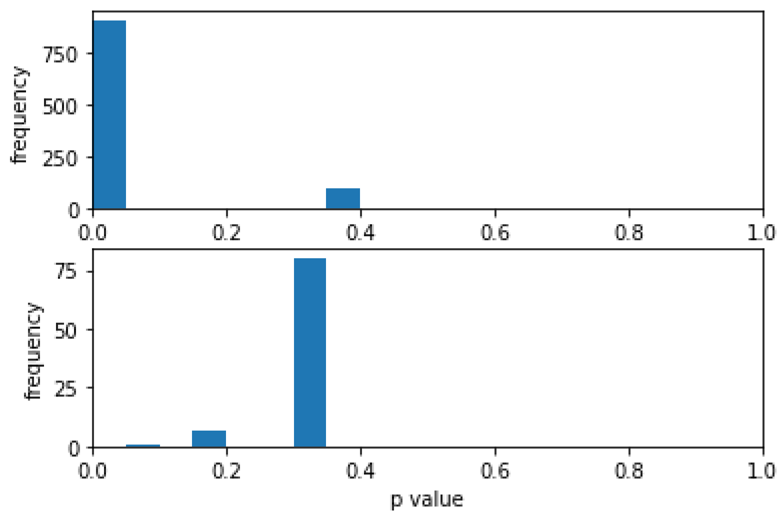
D. Finally, let's do the same for a very skewed distribution. Consider a distribution that has the value 1 one time in 1000, and the value 0 every other time. What is this distribution's mean? Repeatedly generate samples of size 100 from this distribution. How often does a t-test reject the null that the mean has the correct value? And how often does a t-test reject the incorrect null hypothesis that the mean is 0?

Is this the result you expected? Why do you think it is happening?

Result should look like this:

Show code

```
903 out of 1000 tests with p<.05
0 out of 1000 tests with p<.05
```



Exercise 2: randomization test

The t-test asks if the mean is 0, assuming a Gaussian distribution. We have seen how it can fail for a non-Gaussian distribution, and fail catastrophically for a highly skewed distribution.

Now let's use a randomization test, where we compare the mean of the distribution to a null ensemble computed by randomly flipping the sign of each observation. To get a p-value, do this flipping 1000 times, and see where the actual test statistic falls in the null distribution. You can do this with `scipy.stats.percentileofscore`. (Note there is a subtlety here about what to do when the actual test statistic ties with some values in the null distribution. For now, just use the option `kind='mean'` in `scipy.stats.percentileofscore`).

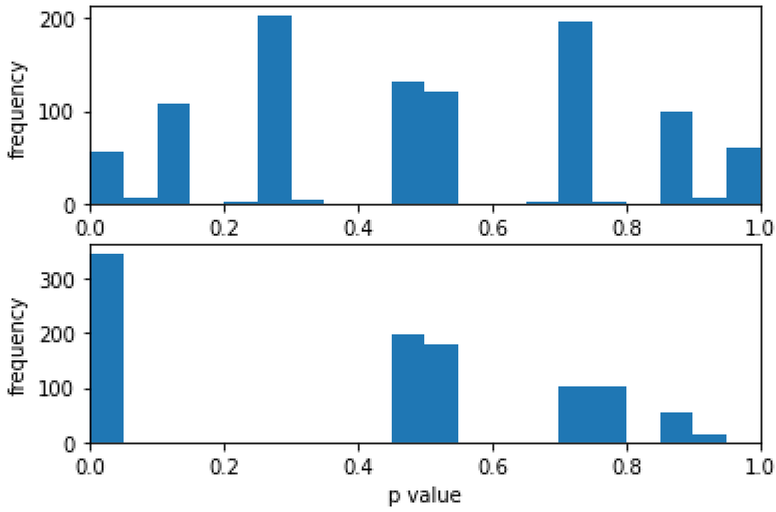
What null hypothesis does this test?

Try this for two cases where the t-test failed: a Rademacher distribution, and a skewed distribution with mean 0, which which takes the value -.1 with 90% probability, and the value .9 with 10% probability. Use a sample size of 10. Which one does it reject more often by chance level? Is this what the test is supposed to do?

Result should look like this:

Show code

```
57 out of 1000 tests with p<.05
346 out of 1000 tests with p<.05
```



41s completed at 11:34 AM