

Using Bootstrapping to Evaluate Your Experiment

Introduction to Bootstrapping

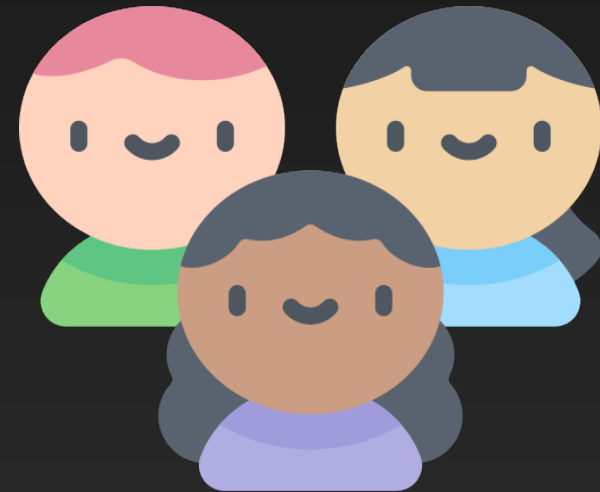
Szu-Min Yu, Data Scientist at CVS Health

Agenda

- Imagine you are doing an experiment
- Hypothesis testing
- Introduction to bootstrapping
- Apply bootstrapping to evaluate your experiment
- Code example

Imagine You're Doing Experiment for the Company...

Test



Converted 20 people out of total 1000
Conversion: 2%

Control



Converted 15 people out of total 1000
conversion: 1.5%

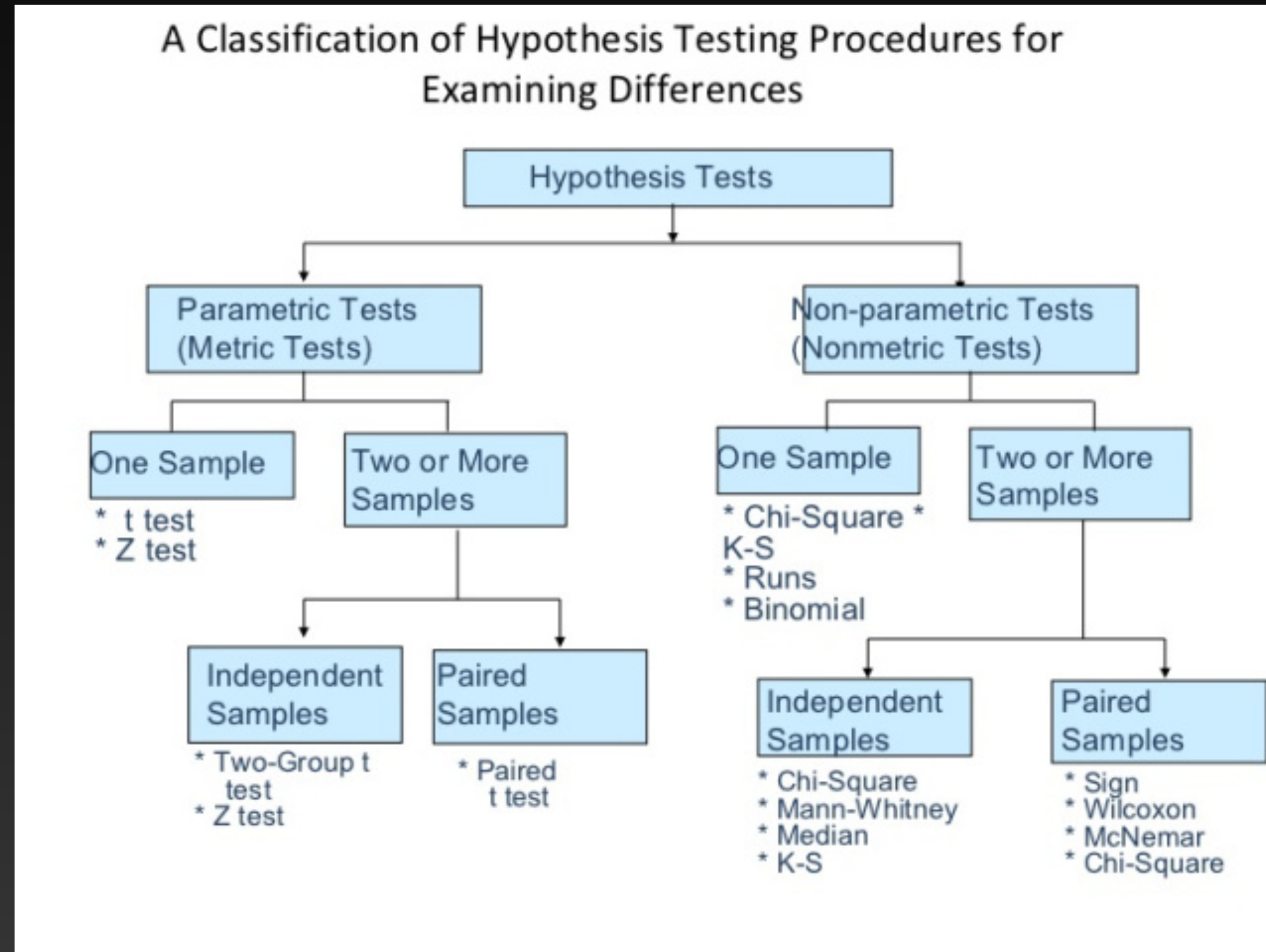
Hypothesis Testing

Test group looks better than control group - but does it really perform better?

- You want to test whether test group works better, a.k.a. the difference between two groups are statistically significant.
- Parametric test - make assumptions about the parameters of the population distribution from which the sample is drawn. This is often the assumption that the population data are **normally distributed**.
- Non-parametric tests are “distribution-free” and, as such, can be used for non-Normal variables.

Hypothesis Testing II

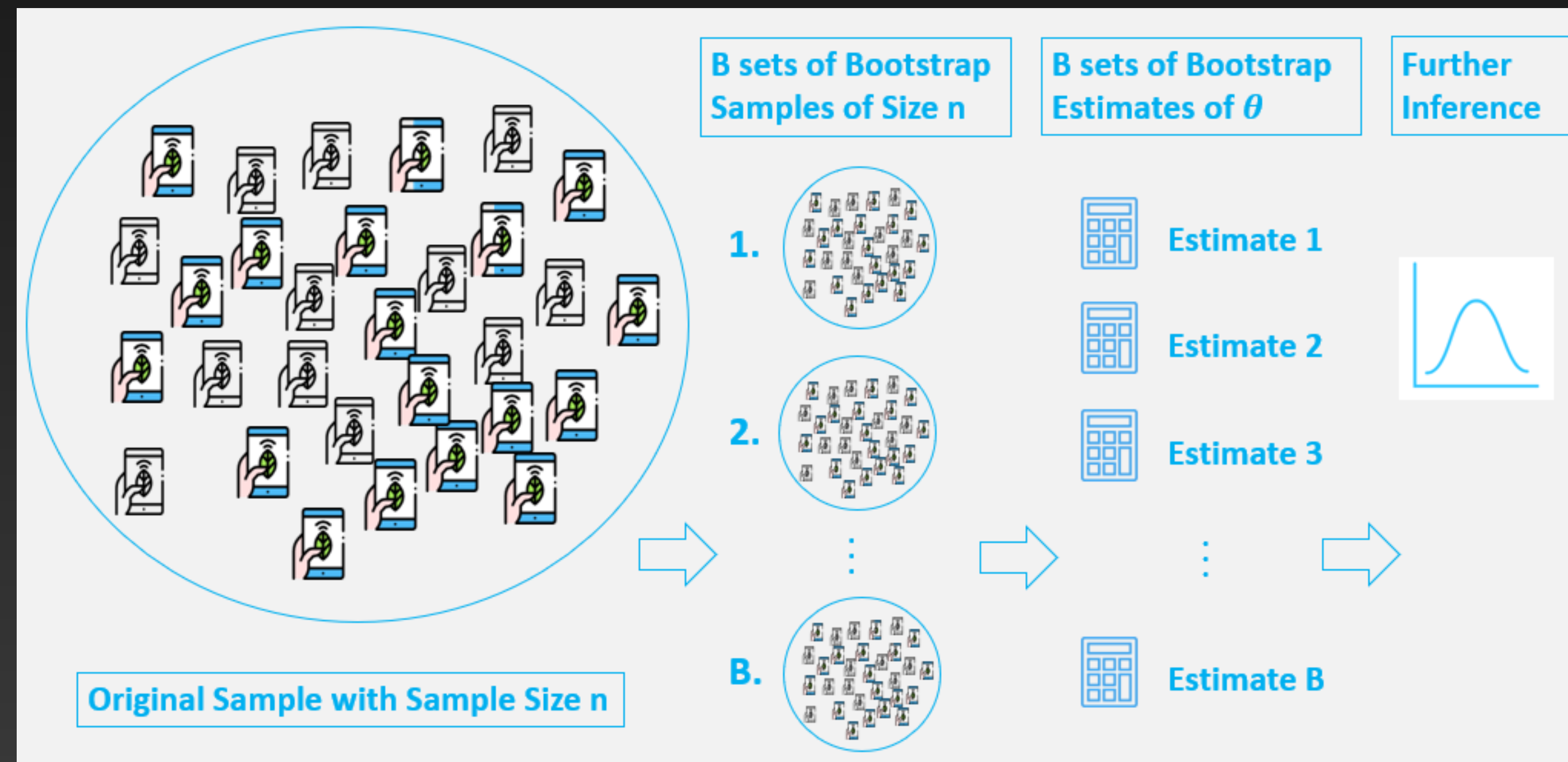
Statistical Tests



Bootstrapping Method

“Bootstrapping is any test or metric that uses **random sampling with replacement**, and falls under the broader class of resampling methods.”

“Bootstrapping is making inference about an estimate (such as sample mean) for a population parameter θ (such as population mean).”



Bootstrapping Method

Process

1. A sample from population with sample size n : your experiment data or generated data
2. Resample your data with replacement with size n
Each resample is called a Bootstrapped sample
3. Repeat 1 & 2 for multiple times
4. Evaluate the statistic of θ for each bootstrap sample
5. Construct a sampling distribution of bootstrap statistics and make further statistical inference (e.g. estimating standard error and confidence interval for θ)

Bootstrapping Method

- This resampling method is widely used in statistics and machine learning.
- For instance, Bagging (bootstrap aggregating) will create bootstrapped subsamples from the population and calculate the average.
- Random Forest tweaks Bagging a bit to reduce the variance by learning from only a handful of features while growing trees.
- Central Limit Theorem:
if you have a population with mean μ and standard deviation σ and take sufficiently large random samples from the population with replacement, then the distribution of the sample means will be approximately normally distributed
- Law of Large Numbers: the mean from bootstrapped samples will converge to the true mean of population as iteration $\rightarrow \infty$
- Use sampled standard error to obtain the estimated standard error of the population (to construct the confidence intervals)

Apply Bootstrapping to the Evaluation

- Non-parametric: no assumption about the population distribution
- Tackle small sample problems
- Can easily get p-value and confidence intervals
- Code is reused for different experiments
- Can also be applied to design the experiment & conduct power analysis; however, I will only focus on the evaluation today.
- Disadvantage: bootstrapping can be time-consuming if there are too many iterations (at least 30 to fulfill the Central Limit Theorem)

Apply Bootstrapping to the Evaluation II

Example

- If we are interested in knowing whether the test group convert more than control group, we test whether the difference in conversion rate is statistically significant.
 - Outcome variable: 0 for non-converted, 1 for converted (the mean of all conversion data is the group conversion rate)
1. Resample the data
 2. Get conversion rates in both groups
 3. Calculate the difference in conversion rate
 4. Repeat 1 - 3 for x times (normally I will do 5000)
 5. Construct the distribution and do further statistical inference (e.g. calculate confidence interval and/or p-value from the bootstrapped data)

	test	cntrl
0	0	0
1	0	0
2	0	0
3	0	0
4	1	0

Code Example

Resampling

```
def resampling(dataframe, i):  
    """  
    This function is used for resampling.  
  
    data: your dataset  
    i: the number of iteration  
    """  
  
    # 1.resample your data, replace will be True because bootstrapping allows replacement  
  
    df = dataframe.sample(frac = 1, replace = True)  
  
    # 2. get the difference in conversion rate and the number of iteration  
    df1 = df.mean().reset_index()  
    df1.columns = ['group', 'value']  
    df1 = df1.pivot_table(columns = ['group'])  
    df1['diff'] = df1['test'] - df1['cntrl']  
    df1['delta'] = df1['diff'] > 0  
  
    return df1
```

Build bootstrapped data with 5000 iterations

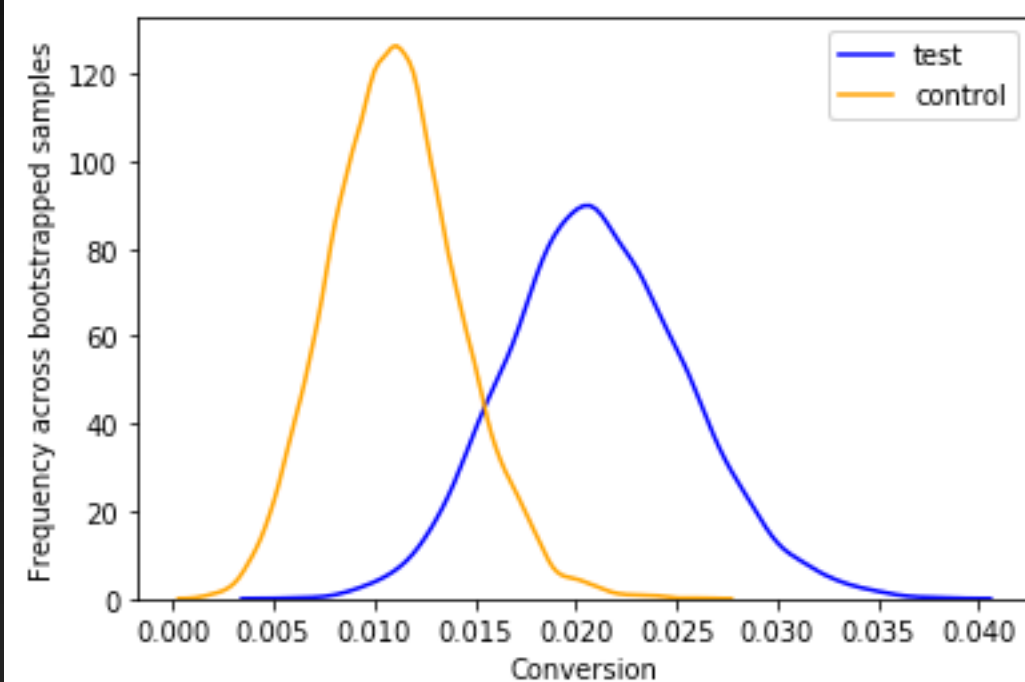
```
# get all bootstrapped data  
  
bootstrapped = pd.concat([resampling(dataframe = data, i = i) for i in range(0,5000)])  
bootstrapped.index = np.arange(0, bootstrapped.shape[0])  
  
bootstrapped.head()
```

	group	cntrl	test	diff	delta
0	0	0.014	0.020	0.006	True
1	1	0.019	0.019	0.000	False
2	2	0.013	0.022	0.009	True
3	3	0.016	0.029	0.013	True
4	4	0.012	0.028	0.016	True

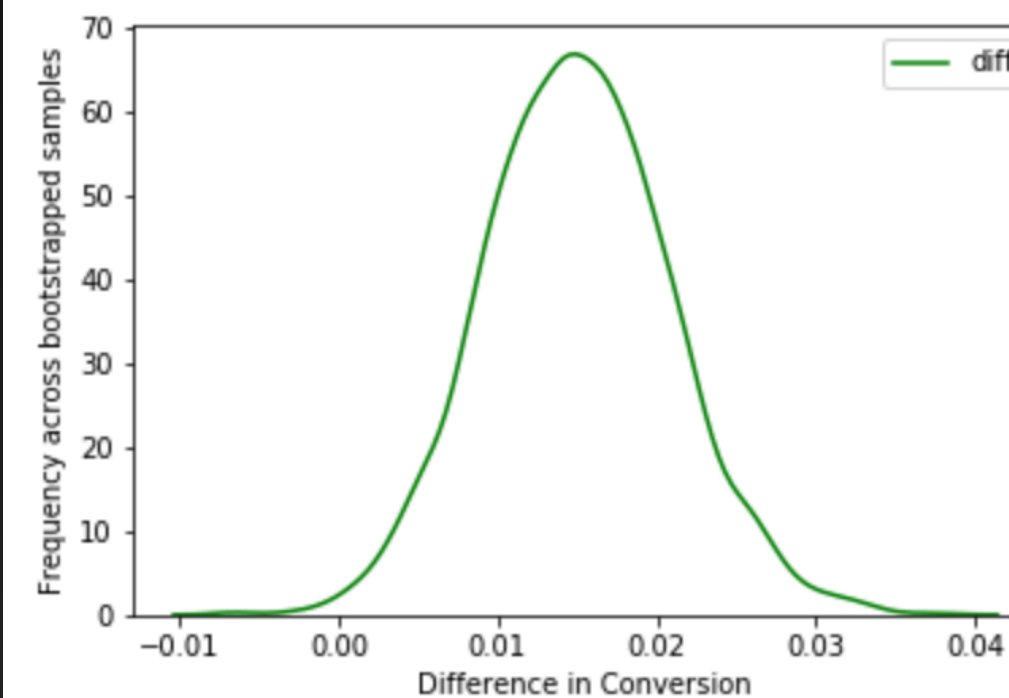
Code Example

Plotting the distribution

```
plt.figure()
ax = sns.distplot(bootstrapped['test'], label = 'test', hist = False, color = 'blue')
ax = sns.distplot(bootstrapped['cntrl'], label = 'control', hist = False, color = 'orange')
ax.set_ylabel('Frequency across bootstrapped samples')
ax.set_xlabel('Conversion')
plt.legend()
plt.show()
```



```
plt.figure()
ax = sns.distplot(bootstrapped['diff'], label = 'diff', hist = False, color = 'green')
ax.set_ylabel('Frequency across bootstrapped samples')
ax.set_xlabel('Difference in Conversion')
plt.legend()
plt.show()
```



Calculate p-value & confidence interval

```
# calculate p-value and confidence interval

p_value = 1 - bootstrapped['delta'].mean()

print('p-value: ', '{:0.5f}'.format(p_value))
if p_value < 0.05:
    print('Mean difference in conversion is significant at 0.05 level')
else: print('Mean difference in conversion is NOT significant at 0.05 level')

lower, upper = sm.stats.DescrStatsW(bootstrapped['diff']).tconfint_mean()
print('95% confidence interval', '{:0.5f}, {:0.5f}'.format(lower, upper))

p-value: 0.00500
Mean difference in conversion is significant at 0.05 level
95% confidence interval [0.01482, 0.01515]
```

Code Example

Full Code

```
class bootstrapping:

    def resampling(self, dataframe, i):

        """
        This function is used for resampling.

        dataframe: your dataset
        i: the number of iteration
        """

        # 1.resample your data, replace will be True because bootstrapping allows replacement

        df = dataframe.sample(frac = 1, replace = True)

        # 2. get the difference in conversion rate and the number of iteration
        df1 = df.mean().reset_index()
        df1.columns = ['group', 'value']
        df1 = df1.pivot_table(columns = ['group'])
        df1['diff'] = df1['test'] - df1['cntrl']
        df1['delta'] = df1['diff'] > 0

        return df1

    def measuring(self, data, iteration):

        # 3. get all bootstrapped data

        bootstrapped = pd.concat([self.resampling(dataframe = data, i = i) for i in range(0, iteration)])
        bootstrapped.index = np.arange(0, bootstrapped.shape[0])

        # plot test v. control
        plt.figure()
        ax = sns.distplot(bootstrapped['test'], label = 'test', hist = False, color = 'blue')
        ax = sns.distplot(bootstrapped['cntrl'], label = 'control', hist = False, color = 'orange')
        ax.set_ylabel('Frequency across bootstrapped samples')
        ax.set_xlabel('Conversion')
        plt.legend()
        plt.show()

        # plot the difference in conversion
        plt.figure()
        ax = sns.distplot(bootstrapped['diff'], label = 'diff', hist = False, color = 'green')
        ax.set_ylabel('Frequency across bootstrapped samples')
        ax.set_xlabel('Difference in Conversion')
        plt.legend()
        plt.show()

        # 4. calculate p-value and confidence interval

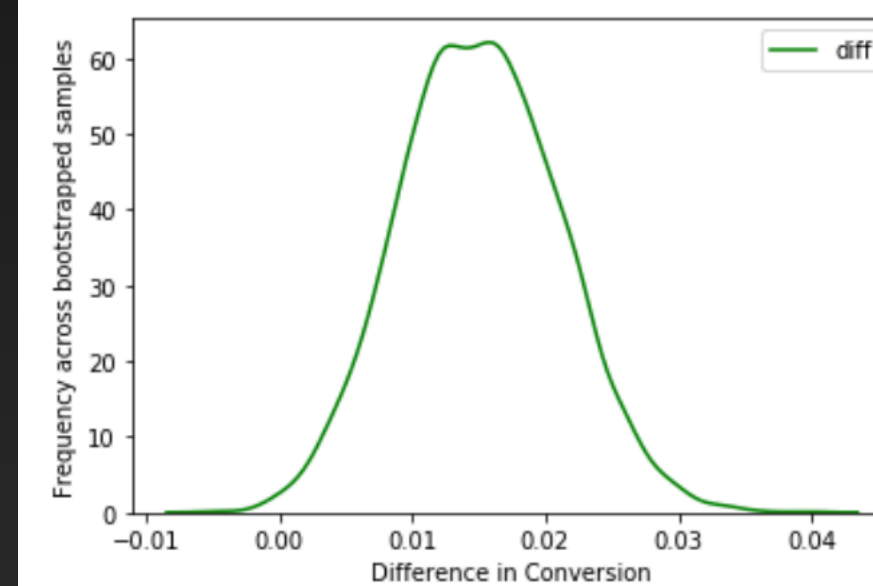
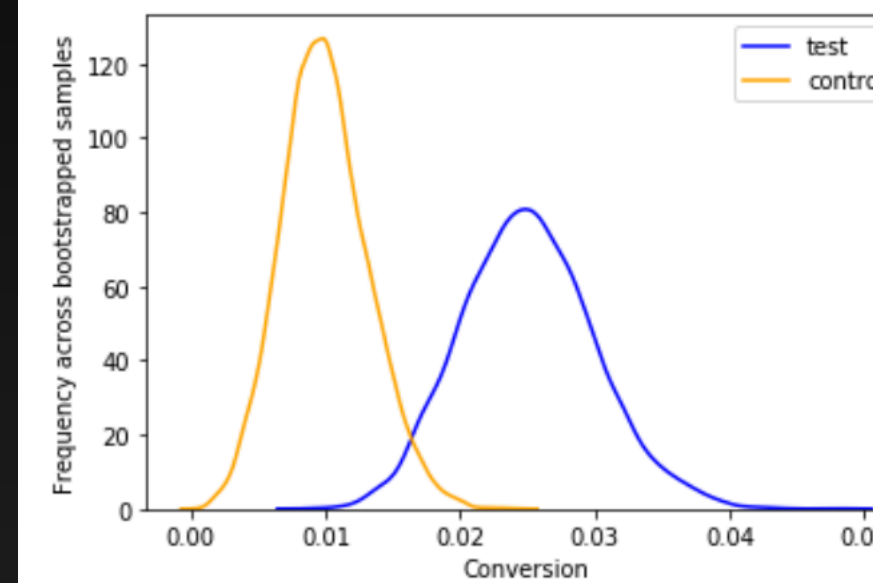
        p_value = 1 - bootstrapped['delta'].mean()

        print('p-value: ', '{:0.5f}'.format(p_value))
        if p_value < 0.05:
            print('Mean difference in conversion is significant at 0.05 level')
        else: print('Mean difference in conversion is NOT significant at 0.05 level')

        lower, upper = sm.stats.DescrStatsW(bootstrapped['diff']).tconfint_mean()
        print('95% confidence interval', '{:0.5f}, {:0.5f}'.format(lower, upper))
```

Result

```
b = bootstrapping()
b.measuring(data, 5000)
```



p-value: 0.00480
Mean difference in conversion is significant at 0.05 level
95% confidence interval [0.01492, 0.01525]

Thank You!

Questions/Comments?

- Code on GitHub: [szuminyu/bootstrap_TDP_monthly_2021](https://github.com/szuminyu/bootstrap_TDP_monthly_2021)
- Connect me on LinkedIn: szuminyu
- Follow TDP on Facebook & join slack channel
- <https://www.facebook.com/twdatany>
- We are looking for volunteers!