Confidence interval using boootstrapping

- https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.bootstrap.html
- https://towardsdatascience.com/calculating-confidence-interval-with-bootstrapping-872c657c058d
- https://machinelearningmastery.com/calculate-bootstrap-confidence-intervals-machine-learning-results-python/
- https://medium.com/@wenjun.sarah.sun/bootstrap-confidence-interval-in-python-3fe8d5a6fd56

CI using bootstrapping for two groups

- https://github.com/SUN-Wenjun/bootstrapping

```
import numpy as np

def bootstrap_ci_two_groups(df, variable, classes, repetitions = 1000,
alpha = 0.05, random_state=None):
    # df: a data frame that includes observations of the two sample
    # variable: the column name of the column that includes observations
    # classes: the column name of the column that includes group
assignment (This column should contain two different group names)
    # repetitions: number of times you want the bootstrapping to repeat.
Default is 1000.
    # alpha: likelihood that the true population parameter lies outside
the confidence interval. Default is 0.05.
    # random_stata: enable users to set their own random_state, default is
None.

    df = df[[variable, classes]]
    bootstrap_sample_size = len(df)

    mean_diffs = []

    for i in range(repetitions):
        bootstrap_sample = df.sample(n = bootstrap_sample_size, replace =
True, random_state = random_state)
        mean_diff = bootstrap_sample.groupby(classes).mean().iloc[1,0] -
bootstrap_sample.groupby(classes).mean().iloc[0,0]
        mean_diffs.append(mean_diff)

    # confidence interval
    left = np.percentile(mean_diffs, alpha/2*100)
    right = np.percentile(mean_diffs, 100-alpha/2*100)

    # point estimate
    point_est = df.groupby(classes).mean().iloc[1,0] -
df.groupby(classes).mean().iloc[0,0]

    print('Point estimate of difference between means:',
```

```
round(point_est,2))
    print((1-alpha)*100,'%','confidence interval for the difference
between means:', (round(left,2), round(right,2)))
```