You want to show that algorithm A is better than algorithm B. You have a dataset \( D = (x_1, y_1, \ldots, x_n, y_n) \) to prove it.

1 Do not Use These Methods

Here are some “natural” ideas which, unfortunately, will not support the claim due to the stochastic fluctuation in the dataset \( D \):

- **Training set accuracy.** Train A on \( D \), test A on \( D \) again to get the training set accuracy \( a_A \). Repeat for B to get \( a_B \). Show \( a_A > a_B \). Problems: overfitting, stochastic fluctuation.

- **Test set accuracy.** Split \( D \) into \( D_{train} \) and \( D_{test} \). Train A, B on \( D_{train} \), get their accuracies on \( D_{test} \). Show \( a_A > a_B \). Problem: stochastic fluctuation.

- **CV accuracy.** Perform \( k \)-fold cross validation on \( D \) with A. Use exactly the same folds on B too. Show the CV accuracy \( a_A > a_B \). Problem: stochastic fluctuation. (OK, people actually use this quite often. But it is better to assess the statistical significance. Read on...)

- **Dataset selection.** Select and only report experiments on certain datasets \( D \) that “worked”. Problem: Hmm...

2 Statistical Tests

An accepted method is to perform a statistical significance test. The idea is simple. Let us assume that A and B indeed have the same generalization accuracy. Their CV accuracies \( a_A \) and \( a_B \) will still exhibit all kinds of fluctuations (i.e., be different). If we were to be certain that we do not call A and B different, we will need to tolerate all possible differences in \( a_A \) and \( a_B \), including very large ones. This is useless, because if A and another algorithm C is truly different we will not be able to detect that.

However, we expect most of the time \( a_A \) and \( a_B \) are “fairly close”. Only rarely do they differ a lot. In fact, we can find a threshold such that \( a_A > a_B \) differ by that much in only 5% of the times we do the test. We will call two algorithms different if their CV accuracies differ more than the threshold.

More formally, we entertain two hypotheses:

- **\( H_0 \):** The null hypothesis that A and B have the same generalization performance.

- **\( H_a \):** The alternative hypothesis that A and B have different generalization performance.

If the empirical results \( a_A \) and \( a_B \) differ more than the threshold, we reject \( H_0 \) and adopt \( H_a \). Otherwise, we retain \( H_0 \); this does not mean that we believe in \( H_0 \), but simply that we do not have enough evidence to say otherwise. Some immediate observations:

- Statistical test does not really test whether \( H_a \) is true, i.e., two algorithms have different performance. It is only concerned with how often (5% in the above) we will call two algorithms with the same underlying performance different.
Paired \textit{t}-test

- Being able to say two algorithms are different is a \textit{by-product}.
- We will make mistakes 5\% of the time by calling A and B different, when they in fact have the same performance. This is known as Type I error.
- We do not know how often we call A and C the same because they fall within the threshold, when they are truly different. This is Type II error and is not addressed by statistical test (but is important in practice!).
- One can adjust the 5\% figure by changing the threshold. When the threshold is close to zero, it is easier to say that A and C are different. But A and B will be called different more often too – the 5\% figure will increase to, say, 10\%. This is \textit{less significant} (for the difference in A and C). When the threshold is far from zero, it is very hard for A and C to be called different (therefore harder to publish...). A and B will be called different much less frequently, say 1\%. This is \textit{significant} (for A and C). We of course prefer significant results. The default is 5\%.

3 \textbf{Paired \textit{t}-Test}

There are many different tests. In this case, we use a specific test called a paired \textit{t}-test. Let $X_1, \ldots, X_k \sim N(\mu, \sigma^2)$ where both $\mu$ and $\sigma^2$ are unknown, and $k$ is relatively small. We want to test $H_0 : \mu = \mu_0$. Let the sample mean be

$$\bar{X}_k = \frac{1}{k} \sum_{i=1}^{k} X_i,$$

(1)

and the sample variance be

$$S_k^2 = \frac{1}{(k-1)} \sum_{i=1}^{k} (X_i - \bar{X}_k)^2.$$

(2)

The random variable

$$T = \frac{\sqrt{k}(\bar{X}_k - \mu_0)}{S_k}$$

(3)

follows a \textit{t-distribution with $k-1$ degree of freedom} under $H_0$. When $k$ is somewhat large, $T \rightarrow N(0,1)$.

How is this related to our goal? Recall we perform $k$-fold CV. Let the accuracy in each fold be $a_{A1}, \ldots, a_{Ak}$ for algorithm A, and $a_{B1}, \ldots, a_{Bk}$ for algorithm B. We assume that the pairwise differences $x_i = a_{Ai} - a_{Bi}, i = 1 \ldots k$ follow $N(0, \sigma^2)$ under $H_0$. Therefore $T$ has a \textit{t-distribution with $k-1$ degree of freedom}. We can look up the 5\% threshold (2-sided) from a table. When $T$ is outside the threshold we reject $H_0$, and claim that A and B are truly different.

Keep in mind that this procedure has 5\% Type I error. That roughly translates to “every 1 in 20 papers claims an advance that is really not there!”