Design of Experiments

Monday, October 21, 2024 1:29 PM

Design of Experiments Introduction

Sometimes data is difficult to get; it's either impossible to get or would take too long to gather, so we have to design a way to get the best subset of data we can quickly and ensure that the data is sufficient for our needs. This process is called Design of Experiments. Two important concepts to consider when using DOE are:

- Comparison and Control: we want to compare two items to see which is "better" for our needs, for example if red used cars sell for higher prices than blue used cars. But we need to control for other factors that may be impacting the price: age of the car, make and model, etc.
- Blocking Factors: a factor that creates variation, such as the type of car: sports or family. This factor may have a significant impact on the selling price of a car, but if we analyze them within groups (red family cars vs blue family cars), there should be less variation than if all types were analyzed together.

AB Testing

Suppose we have two designs for something and we need to decide which is the better one to put into production. We can conduct an experiment in which both options are put into use and we collect a certain amount of data, which we can then use hypothesis testing to determine which option performed better.

The process of designing an experiment to choose between two alternatives is called AB Testing and it requires three things:

1. We need to be able to gather a lot of data quickly
2. The data must be from a representative sample of the whole population we want to use the design on
3. The amount of data to be collected must be small compared to the size of the whole population that we want to use the design on

Factorial Designs

Suppose we want to compare multiple alternatives and before we collect the data, we want to determine the effects of factors.

We can do a Full Factorial Design with an ANOVA test to determine the determine the importance of each factor in every combination of factors.

In cases where there are too many variations of factors and combinations to test them each, we can use a Fractional Factorial Design to test a subset of combinations. The goal is to test fewer combinations while still covering a good range of values for each factor. A balanced design will test each choice and each pair of choices the same number of times.

If we believe the factors we can change are independent, then we can test a subset of combinations and use regression to estimate the effects of each choice. Each factor would become a categorical variable (such as the background color of a banner (red, gold, blue) and font size (10, 12, 14 pt)) and then the response variable to predict is the number of clicks each combination gets. In cases where the factors aren't really independent (background color and font color likely aren't), interaction variables would need to be included.

Multi-Armed Bandits

Say we have 10 alternatives and we test each alternative a thousand times. We discover alternative #1 performed the best, then we got 1000 tests of the best value and 9000 tests of lost value. Thus every time we show an alternative, we have to balance the tradeoff between exploration vs exploitation; more information vs immediate value.

Multi-Armed Bandits: we start with k alternatives and start with no information (i.e., equal probability of selecting each alternative). After performing some tests on each alternative, we have more information and can update the probability of selecting each alternative. We can continue testing, testing those that have the higher probabilities of being "best". This process is continued until we are sufficiently confident that we have chosen the best alternative. There are several parameters we can change during the iterative process:

- The number of tests between recalculating the probabilities
- How to update the probabilities, such as Bayesian updates or estimating probability distributions
- How to pick an alternative to test based on probabilities and/or expected values

There isn't a good rule of thumb for the "best way" to run a Multi-Armed Bandit, but it's better than running a fixed large number of tests. Multi-Armed Bandits mean you learn faster on-the-fly and can create more value.