Testing the Prediction Profiler with Disallowed Combinations

Jeremy Ash, Caleb King, Laura Lancaster, Ryan Lekivetz, Joseph Morgan, Yeng Saanchi

JMP Statistical Discovery LLC

14 August 2024

Table of Contents

What is software testing?

"Testing is the process of executing a program with the intent of finding errors."

G. Myers, The Art of Software Testing, Wiley, 1979

Where are the bugs?

“Bugs lurk in corners and congregate at boundaries.”

B. Beizer, Software Testing Techniques, Van Nostrand Reinhold, 1983

The Software Testing Challenge

- Selection problem:
  - How do you select test cases from the input space of the system so that the chance of finding faults, while staying within budget, is maximized?
- Quality problem:
  - How do you assert that software is fit for use during the course of testing?
- Oracle problem:
  - How do you determine the appropriate “oracle” for comparing and evaluating test results

Prediction Profiler

- Visually explore the relationships between multiple factors and responses
- Set of univariate plots for each factor that shows predicted response(s) for settings of the factors
- Interact with plots to change factor values
- Often used in conjunction with desirability function

Microsoft Teams

Profiler Recording

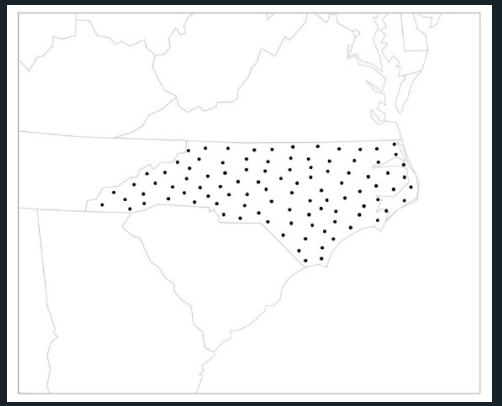
2024-07-23 19:07 UTC

Recorded by
Yeng Saanchi

Organized by
Yeng Saanchi

Disallowed Combinations

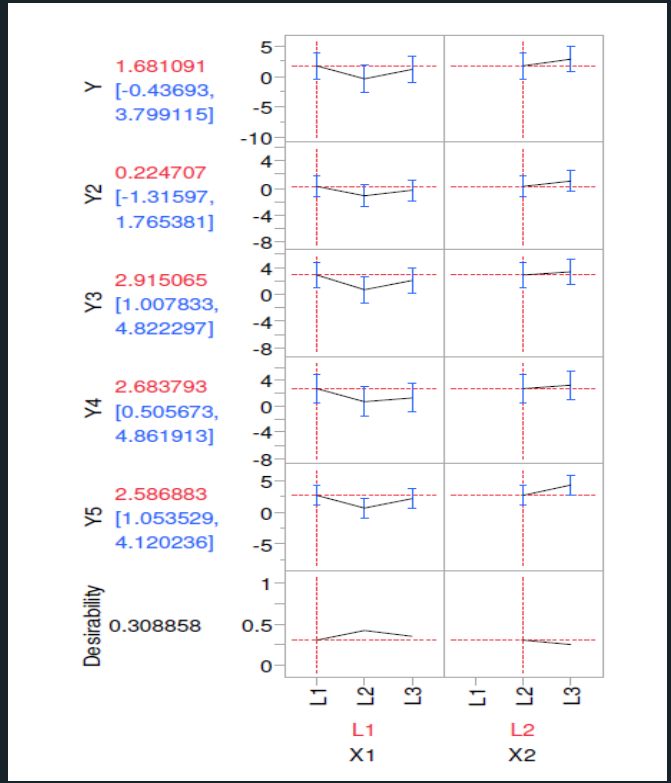
- Boolean expression that evaluates to true if a given design point is not in the design space and false for a design point that is in the design space.
- Typically appear in JMP in the DOE suite and are used to create designs which consist of only points in the allowable region
- Examples:
  - A * B > 0.5 | A * B < -0.5
  - A = “old machine” & B = “new part”
  - outside of a polygon



Disallowed Combinations

- 5 responses, Y1-Y5
- 3-level Categorical X1 & X2
- Disallowed Combination

  X1 = “L1” & X2 = “L1”



Microsoft Teams

Profiler Recording for Stat Speaking

2024-07-20 00:57 UTC

Recorded by
Yeng Saanchi

Organized by
Yeng Saanchi

The Task

Test the prediction profiler with disallowed combinations

Some Aspects of the Profiler to Test

- Does the profiler display the constrained region correctly?
- Does the profiler only profile allowable regions?
- Does maximizing desirability find the optimum in the constrained space?

The Approach

- Objective:
  - Develop test cases that are effective at causing failures (due to combinations of inputs as well as individual components)
    - Find the oracle for each test case
    - Remain within budget
- Proposed Strategy: Consider the construction of the test suite as a DOE problem

The Solution

Combinatorial Testing via Covering Arrays

- ***Covering Arrays*: For a system with *m* inputs, a strength *t* covering array ensures that all possible combinations for any set of *t* inputs will occur at least once in the suite of test cases**

N	4	5	6	7	8	9	10	11	12	13	14
m	3	4	10	15	35	56	126	210	462	792	1716

Why Covering Arrays?

- Cost-efficient
- Selection problem – What to test
  - all combinations involving up to *t* inputs - “pseudo-exhaustive”
- Quality Problem - Enable finding of failures due to interactions between multiple factors
  - Moves beyond one-factor-at-a-time (OFAT) testing
- Disciplined approach to testing vs. “let’s just test more”

The Data

- What do you do with limited data sets?
- Need to anticipate how they might be used
- Our Solution:
  - Combine data set generation and test case selection using combinatorial testing
  - Consider both test case selection and data set generation factors
    - Data set generation – 15 factors
    - Test case selection – 3 factors

Inputs and Levels

Table 1. Factors and selected levels.

Factor	Levels			
X1	L1 = 3-level discrete numeric	L2 = 3-level categorical		
X2	L1 = 3-level discrete numeric	L2 = 3-level categorical		
X3	L1 = 3-level discrete numeric	L2 = 3-level categorical		
X4	L1 = 3-level discrete numeric	L2 = 3-level categorical		
X5-10	L1 = categorical	L2 = continuous	L3 = mix	
Number of Responses	L1 = 1	L2 = 3	L3 = 5	
Response Goal	L1 = Match Target	L2 = Maximize	L3 = Minimize	L4 = Random
Model	L1 = Main Effects	L2 = If Possible Interactions		
Constraint 1	L1 = {X1 = 1 ∧ X2 =1}	L2 = { X1 = 1 ∧ X2 =1 ∧ X3=1}		
Constraint 2	L1 = {X3 = 1 ∧ X4 =3}	L2 = none		
Constraint 3	L1 = {X4 = 1 ∧ X11 > 0.5}	L2 = none		
Continuous Constraint	L1 = {X11 * X12 > 0.8}	L2 = {X11 + X12 > 0.5}		
Run Size	L1 = 32	L2 = 64		
Augment Design	L1 = Yes	L2 = No		
Simulated Model	L1 = Main Effects	L2 = Few Interactions		
Maximize & Remember	L1 = Yes	L2 = No		
Maximize Desirability	L1 = 1X	L2 = 3X		
Sensitivity Indicator	L1 = Yes	L2 = No		

Test Suite

- $2^4 * 3^2 * 4 * 2^8 * 2^3$ = 1,179,648 possible test cases
- Strength 2 covering array has all pairwise combinations covered in 13 runs 😮
- All 8 combinations of the 3 profiler options are covered 🎆

Sequential Nature of Testing

- Nightly regression test stream
- Revisit oracles
- Augmentation to improve coverage

Summary

- Challenges of testing statistical software
  - Deriving oracles is difficult
  - Lack of data sets
- Combinatorial testing - effective and efficient
  - Combined both data set generation and profiler features as factors

Want to Know More?

Testing the prediction profiler with disallowed combinations—A statistical engineering case study

Jeremy Ash, Caleb King, Laura Lancaster, **Ryan Lekivetz**, Joseph Morgan & Yeng Saanchi

Pages 507-521 | Published online: 11 Nov 2022

Cite this article https://doi.org/10.1080/08982112.2022.2135445 Check for updates

Ash, Jeremy, et al. "Testing the prediction profiler with disallowed combinations—A statistical engineering case study." *Quality Engineering* 34.4 (2022): 507-521.

jmp STATISTICAL DISCOVERY

Thank you!

jmp.com