

# Risk Analysis of the Space Shuttle: Pre-Challenger

## Prediction of Failure

In this document we reperform some of the analysis provided in *Risk Analysis of the Space Shuttle: Pre-Challenger Prediction of Failure* by Siddhartha R. Dalal, Edward B. Fowlkes, Bruce Hoadley published in Journal of the American Statistical Association, Vol. 84, No. 408 (Dec., 1989), pp. 945-957 and available at <http://www.jstor.org/stable/2290069>.

On the fourth page of this article, they indicate that the maximum likelihood estimates of the logistic regression using only temperature are:  $\hat{\alpha} = 5.085$  and  $\hat{\beta} = -0.1156$  and their asymptotic standard errors are  $s_{\hat{\alpha}} = 3.052$  and  $s_{\hat{\beta}} = 0.047$ . The Goodness of fit indicated for this model was  $G^2 = 18.086$  with 21 degrees of freedom. Our goal is to reproduce the computation behind these values and the Figure 4 of this article, possibly in a nicer looking way.

## 1 - Technical information on the computer on which the analysis is run

We will be using the [Julia](#) language:

```
using InteractiveUtils
versioninfo()

Julia Version 1.4.0
Commit b8e9a9ecc6 (2020-03-21 16:36 UTC)
Platform Info:
  OS: Linux (x86_64-pc-linux-gnu)
  CPU: Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-8.0.1 (ORCJIT, skylake)
Environment:
  JULIA_PROJECT = @.

# Setup environment
using Pkg
Pkg.activate(@_DIR_)
Pkg.instantiate()

# Load dependencies
using HTTP, CSV
using Plots, plotly()
using GLM
using DataFrames
using Printf
include("utils.jl")

# Summary
Pkg.status()

Project Challenger v0.1.0
Status `~/tmp/MOOC-RR/module4/Project.toml`
 [336ed68f] CSV v0.6.1
 [a93c6f00] DataFrames v0.20.2
 [82cc6244] DataInterpolations v2.0.0
 [38e38edf] GLM v1.3.9
 [cd3eb016] HTTP v0.8.14
 [9b87118b] PackageCompiler v1.1.1
 [91a5bcd] Plots v0.29.9
 [44d3d7a6] Weave v0.9.4
```

## 2 - Loading and inspecting data

Let's start by reading data.

```
res = HTTP.request(:GET, "https://app-learninglab.inria.fr/moocrr/gitlab/moocrr-session3/moocrr-r")
data = CSV.read(res.body)
```

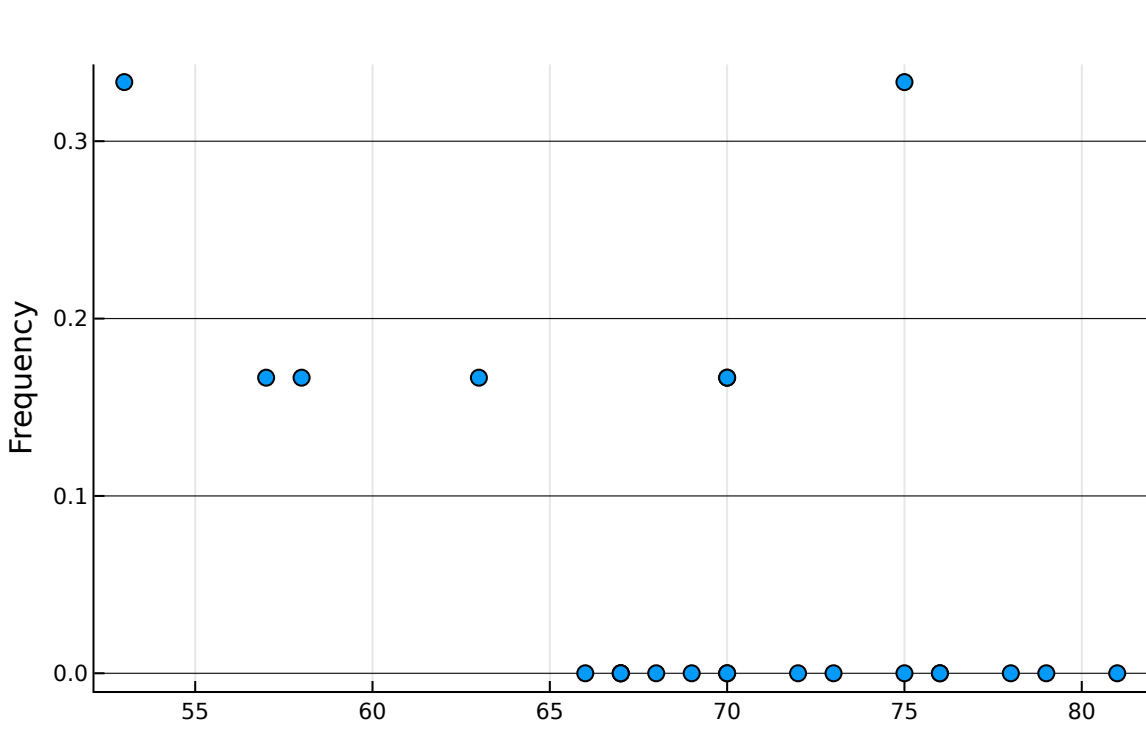
23 rows × 5 columns

	Date	Count	Temperature	Pressure	Malfunction
	String	Int64	Int64	Int64	Int64
1	4/12/81	6	66	50	0
2	11/12/81	6	70	50	1
3	3/22/82	6	69	50	0
4	11/11/82	6	68	50	0
5	4/04/83	6	67	50	0
6	6/18/82	6	72	50	0
7	8/30/83	6	73	100	0
8	11/28/83	6	70	100	0
9	2/03/84	6	57	200	1
10	4/06/84	6	63	200	1
11	8/30/84	6	70	200	1
12	10/05/84	6	78	200	0
13	11/08/84	6	67	200	0
14	1/24/85	6	53	200	2
15	4/12/85	6	67	200	0
16	4/29/85	6	75	200	0
17	6/17/85	6	70	200	0
18	7/2903/85	6	81	200	0
19	8/27/85	6	76	200	0
20	10/03/85	6	79	200	0
21	10/30/85	6	75	200	2
22	11/26/85	6	76	200	0
23	1/12/86	6	58	200	1

We know from our previous experience on this data set that filtering data is a really bad idea. We will therefore process it as such.

```
data.Frequency = data.Malfunction ./ data.Count

plot(xlabel="Temperature [F]", ylabel="Frequency")
plot!(data.Temperature, data.Frequency, seriestype=:scatter, label=nothing)
disp()
```



## 3 - Logistic regression

Let's assume O-rings independently fail with the same probability which solely depends on temperature. A logistic regression should allow us to estimate the influence of temperature.

```
model = glm(@formula(Frequency ~ Temperature), data,
            Binomial(), LogitLink())

α, β = coef(model)
σα, σβ = stderror(model)

G2 = deviance(model)
nDOF = Int(dof_residual(model))

model

StatsModels.TableRegressionModel{GLM.GeneralizedLinearModel{GLM.GlmResp{Array{Float64,1},Distributions.Multinomial{1}},GLM.Logistic{1}}}

Frequency ~ 1 + Temperature

Coefficients:

      Estimate Std. Error z value Pr(>|z|) Lower 95% Upper 95%
(Intercept)  5.08498    7.47703  0.68008  0.4965  -9.56973  19.7397
Temperature -0.115601    0.115184 -1.00362  0.3156  -0.341358  0.110156
```

The maximum likelihood estimator of the intercept and of Temperature are thus  $\hat{\alpha} = 5.085$  and  $\hat{\beta} = -0.116$ .

This corresponds to the values from the article of Dalal et al. The standard errors are  $s_{\hat{\alpha}} = 7.477$  and  $s_{\hat{\beta}} = 0.115$ , which is different from the 3.052 and 0.047 reported by Dallal et al.

The deviance is  $G^2 = 3.014$  with 22 degrees of freedom. I cannot find any value similar to the Goodness of fit reported by Dalal et al. ( $G^2 = 18.086$ ). However, the number of degrees of freedom is different but at least similar to theirs (21).

There seems to be something wrong. Oh I know, I haven't indicated that my observations are actually the result of 6 observations for each rocket launch. Let's indicate these weights (since the weights are always the same throughout all experiments, it does not change the estimates of the fit but it does influence the variance estimate).

```
model = glm(@formula(Frequency ~ Temperature), data,
            Binomial(), LogitLink();
            wts=data.Count)

α, β = coef(model)
σα, σβ = stderror(model)

G2 = deviance(model)
nDOF = Int(dof_residual(model))

model

StatsModels.TableRegressionModel{GLM.GeneralizedLinearModel{GLM.GlmResp{Array{Float64,1},Distributions.Multinomial{6}},GLM.Logistic{1}}}

Frequency ~ 1 + Temperature

Coefficients:

      Estimate Std. Error z value Pr(>|z|) Lower 95% Upper 95%
(Intercept)  5.08498    3.05247  1.66585  0.0957  -0.897762  11.0677
Temperature -0.115601    0.0470236 -2.45836  0.0140  -0.207766  -0.0234366
```

Good, now I have recovered the asymptotic standard errors  $s_{\hat{\alpha}} = 3.052$  and  $s_{\hat{\beta}} = 0.047$ .

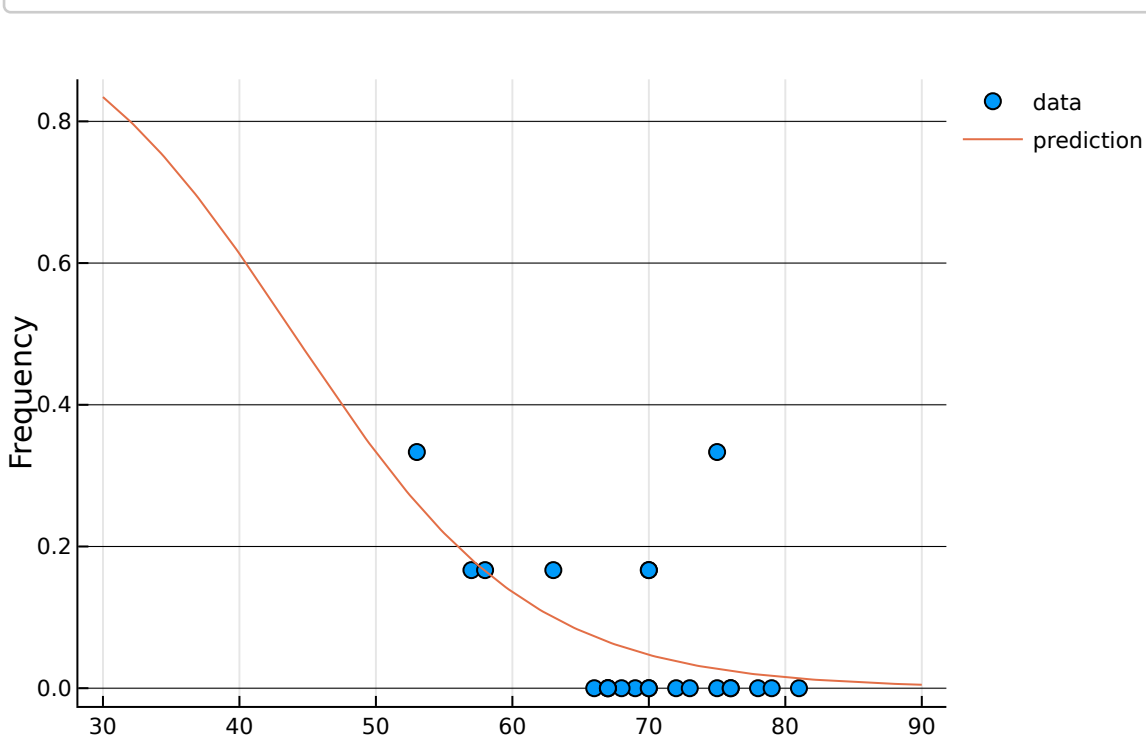
The Goodness of fit (Deviance) indicated for this model is  $G^2 = 18.086$  with 137 degrees of freedom. Now  $G^2$  is in good accordance to the results of the Dalal et al. article, but the number of degrees of freedom is approximately 6 times larger than that of Dalal et al. Note that, even removing this factor (which is probably due to the way the number of residual degrees of freedom are defined in both libraries in the presence of weights), the values are similar but still differ by 9%.

## 4 - Predicting failure probability

The temperature when launching the shuttle was 31°F. Let's try to estimate the failure probability for such temperature using our model:

```
prediction = DataFrame(Temperature=30:0.25:90)
plot!(prediction.Frequency = predict(model, prediction))

plot(xlabel="Temperature [F]", ylabel="Frequency")
plot!(data.Temperature, data.Frequency, seriestype=:scatter, label="data")
plot!(prediction.Temperature, prediction.Frequency, label="prediction")
disp()
```



This figure is very similar to the Figure 4 of Dalal et al.