ANOVA(Final)

October 9, 2024

[1]:
```julia
using Pkg
Pkg.add("MultivariateStats")
Pkg.add("CSV")
Pkg.add("DataFrames")
```

```
  Resolving package versions…
 No Changes to `~/.julia/environments/v1.10/Project.toml`
 No Changes to `~/.julia/environments/v1.10/Manifest.toml`
  Resolving package versions…
 No Changes to `~/.julia/environments/v1.10/Project.toml`
 No Changes to `~/.julia/environments/v1.10/Manifest.toml`
  Resolving package versions…
 No Changes to `~/.julia/environments/v1.10/Project.toml`
 No Changes to `~/.julia/environments/v1.10/Manifest.toml`
```

[24]:
```julia
using CSV, DataFrames,GLM
```

0.1 Weight Gain

[19]:
```julia
weightgain = CSV.read("/Users/VSR/Desktop/Capstone/ANOVA/weightgain.csv",␣
↪DataFrame)
```

[19]:

	source	type	weightgain
	String7	String7	Int64
1	Beef	Low	90
2	Beef	Low	76
3	Beef	Low	90
4	Beef	Low	64
5	Beef	Low	86
6	Beef	Low	51
7	Beef	Low	72
8	Beef	Low	90
9	Beef	Low	95
10	Beef	Low	78
11	Beef	High	73
12	Beef	High	102
13	Beef	High	118
14	Beef	High	104
15	Beef	High	81
16	Beef	High	107
17	Beef	High	100
18	Beef	High	87
19	Beef	High	117
20	Beef	High	111
21	Cereal	Low	107
22	Cereal	Low	95
23	Cereal	Low	97
24	Cereal	Low	80
25	Cereal	Low	98
26	Cereal	Low	74
27	Cereal	Low	74
28	Cereal	Low	67
29	Cereal	Low	89
30	Cereal	Low	58
...	...	...	...

[20]:

```
using Statistics

# Calculate mean
mean_stats = combine(groupby(weightgain, [:source, :type]), :weightgain => mean␣
↪=> :Mean)

# Calculate standard deviation
std_stats = combine(groupby(weightgain, [:source, :type]), :weightgain => std␣
↪=> :SD)

# Display results
println(mean_stats)
println(std_stats)
```

```
4×3 DataFrame
 Row   source    type      Mean
       String7   String7   Float64

   1   Beef      Low          79.2
   2   Beef      High        100.0
   3   Cereal    Low          83.9
   4   Cereal    High         85.9
4×3 DataFrame
 Row   source    type      SD
       String7   String7   Float64

   1   Beef      Low       13.8868
   2   Beef      High      15.1364
   3   Cereal    Low       15.7088
   4   Cereal    High      15.0218
```

[21]:

```
using SimpleANOVA

# Perform two-way ANOVA with SimpleANOVA
anova_results = anova(weightgain, :weightgain, [:source, :type])

# Display the results
println(anova_results)
```

```
Analysis of Variance Results

       Effect        SS  DF         MS          F           p              ²
-----------------------------------------------------------------------
        Total  10453.5   39
         type   1299.6    1   1299.6    5.81231   0.0211449    0.107388
       source    220.9    1    220.9    0.987949  0.326878    -0.000301355
type × source    883.6    1    883.6    3.9518    0.0544676    0.0687235
        Error   8049.4   36    223.594
```

0.2 Foster

[22]:

```
# Load the foster dataset (assuming it's saved as a CSV file)
foster = CSV.read("/Users/VSR/Desktop/Capstone/ANOVA/foster.csv", DataFrame)
```

[22]:

	litgen	motgen	weight
	String1	String1	Float64
1	A	A	61.5
2	A	A	68.2
3	A	A	64.0
4	A	A	65.0
5	A	A	59.7
6	A	B	55.0
7	A	B	42.0
8	A	B	60.2
9	A	I	52.5
10	A	I	61.8
11	A	I	49.5
12	A	I	52.7
13	A	J	42.0
14	A	J	54.0
15	A	J	61.0
16	A	J	48.2
17	A	J	39.6
18	B	A	60.3
19	B	A	51.7
20	B	A	49.3
21	B	A	48.0
22	B	B	50.8
23	B	B	64.7
24	B	B	61.7
25	B	B	64.0
26	B	B	62.0
27	B	I	56.5
28	B	I	59.0
29	B	I	47.2
30	B	I	53.0
...	...	...	...

[27]:
```
using GLM

using GLM

# Fit the two-way ANOVA model with interaction between litgen and motgen
foster_model = fit(LinearModel, @formula(weight ~ litgen * motgen), foster)
```

[27]:
```
StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}},
GLM.DensePredChol{Float64, CholeskyPivoted{Float64, Matrix{Float64},
Vector{Int64}}}}, Matrix{Float64}}

weight ~ 1 + litgen + motgen + litgen & motgen
```

```
Coefficients:

                            Coef.  Std. Error      t  Pr(>|t|)   Lower 95%  Upper
95%

(Intercept)              63.68        3.29364  19.33    <1e-22   57.0463
70.3137
litgen: B               -11.355       4.94046  -2.30    0.0262  -21.3056
-1.4044
litgen: I               -16.58        5.37849  -3.08    0.0035  -27.4128
-5.74716
litgen: J                -9.33        4.94046  -1.89    0.0654  -19.2806
0.620601
motgen: B               -11.28        5.37849  -2.10    0.0416  -22.1128
-0.447157
motgen: I                -9.555       4.94046  -1.93    0.0594  -19.5056
0.395601
motgen: J               -14.72        4.65791  -3.16    0.0028  -24.1015
-5.33848
litgen: B & motgen: B    19.595       7.30317   2.68    0.0102    4.88565
34.3043
litgen: I & motgen: B    28.5467      8.06774   3.54    0.0009   12.2974
44.7959
litgen: J & motgen: B    13.03        7.78257   1.67    0.1010   -2.6449
28.7049
litgen: B & motgen: I    11.155       7.17832   1.55    0.1272   -3.30289
25.6129
litgen: I & motgen: I    14.055       7.30317   1.92    0.0606   -0.654349
28.7643
litgen: J & motgen: I     9.73833     7.48655   1.30    0.2000   -5.34035
24.817
litgen: B & motgen: J     8.295       7.89787   1.05    0.2992   -7.61214
24.2021
litgen: I & motgen: J    17.0533      7.60634   2.24    0.0299    1.73338
32.3733
litgen: J & motgen: J     9.43        6.79002   1.39    0.1717   -4.24579
23.1058
```

[29]:
```
# Compute the ANOVA table
foster_anova = anova(foster_model)
```

```
MethodError: no method matching anova(::StatsModels.
↪TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.
↪DensePredChol{Float64, CholeskyPivoted{Float64, Matrix{Float64},␣
↪Vector{Int64}}}}, Matrix{Float64}})

Closest candidates are:
  anova(::DataFrame, ::Symbol, ::Vector{Symbol}, ::Vector{FactorType};␣
↪factornames)
   @ SimpleANOVA ~/.julia/packages/SimpleANOVA/1j0VA/src/anova_dataframes.jl:3
  anova(::DataFrame, ::Symbol, ::Vector{Symbol}; …)
   @ SimpleANOVA ~/.julia/packages/SimpleANOVA/1j0VA/src/anova_dataframes.jl:3
  anova(::AbstractArray{T}, ::Vector{FactorType}; factornames, hasreplicates)␣
↪where T<:Union{Number, AbstractVector{<:Number}}
   @ SimpleANOVA ~/.julia/packages/SimpleANOVA/1j0VA/src/anova.jl:98
  …


Stacktrace:
 [1] top-level scope
   @ In[29]:2
```

[ ]:

[ ]:

0.3 WATER

[1]:
```
# Load the water dataset (assuming you saved it as a CSV file)
water = CSV.read("/Users/VSR/Desktop/Capstone/ANOVA/water.csv", DataFrame)
```

[1]:

	location	town	mortality	hardness
	String7	String15	Int64	Int64
1	South	Bath	1247	105
2	North	Birkenhead	1668	17
3	South	Birmingham	1466	5
4	North	Blackburn	1800	14
5	North	Blackpool	1609	18
6	North	Bolton	1558	10
7	North	Bootle	1807	15
8	South	Bournemouth	1299	78
9	North	Bradford	1637	10
10	South	Brighton	1359	84
11	South	Bristol	1392	73
12	North	Burnley	1755	12
13	South	Cardiff	1519	21
14	South	Coventry	1307	78
15	South	Croydon	1254	96
16	North	Darlington	1491	20
17	North	Derby	1555	39
18	North	Doncaster	1428	39
19	South	East Ham	1318	122
20	South	Exeter	1260	21
21	North	Gateshead	1723	44
22	North	Grimsby	1379	94
23	North	Halifax	1742	8
24	North	Huddersfield	1574	9
25	North	Hull	1569	91
26	South	Ipswich	1096	138
27	North	Leeds	1591	16
28	South	Leicester	1402	37
29	North	Liverpool	1772	15
30	North	Manchester	1828	8
...	...	...	...	...

[3]:
```
using MultivariateStats, LinearAlgebra, CategoricalArrays

using Statistics

# Response matrix Y (dependent variables: hardness and mortality)
Y = Matrix(water[:, [:hardness, :mortality]])

# Encode location as dummy variables
water.location = categorical(water.location)
X = hcat(ones(nrow(water)), Matrix(water[:, [:location]]))
```

[3]:
```
61×2 Matrix{Any}:
 1.0  String7("South")
```

```
 1.0  String7("North")
 1.0  String7("South")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("South")
 1.0  String7("North")
 1.0  String7("South")
 1.0  String7("South")
 1.0  String7("North")
 1.0  String7("South")

 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("North")
 1.0  String7("South")
 1.0  String7("North")
 1.0  String7("South")
 1.0  String7("South")
 1.0  String7("South")
 1.0  String7("South")
 1.0  String7("North")
```

[6]:

```julia
# Mean vector of Y
Y_bar = mean(Y, dims=1)

# Compute the total SSCP matrix
T = (Y .- Y_bar)' * (Y .- Y_bar)

# Compute the within-group SSCP matrix
groups = groupby(water, :location)
W = zeros(2, 2)
for g in groups
    Y_g = Matrix(g[:, [:hardness, :mortality]])
    Y_g_bar = mean(Y_g, dims=1)
    W += (Y_g .- Y_g_bar)' * (Y_g .- Y_g_bar)
end

# Between-group SSCP matrix
B = T - W
```

[6]:

```
2×2 Matrix{Float64}:
 23122.0        -1.50817e5
    -1.50817e5   9.83729e5
```

[8]:
```
using LinearAlgebra

# Ensure to use LinearAlgebra's eigvals function
eigvals_ = LinearAlgebra.eigvals

# Compute the Hotelling-Lawley trace
W_inv = inv(W)
eigvals_result = eigvals_(W_inv * B)  # Use the alias eigvals_
hotelling_lawley_trace = sum(eigvals_result)

println("Hotelling-Lawley Trace: ", hotelling_lawley_trace)
```

```
Hotelling-Lawley Trace: 0.9002148133569229
```

0.4 Skulls

[9]:
```
using CSV, DataFrames

# Load the skulls dataset
skulls = CSV.read("/Users/VSR/Desktop/Capstone/ANOVA/skulls.csv", DataFrame)
```

[9]:

	epoch	mb	bh	bl	nh
	String7	Int64	Int64	Int64	Int64
1	c4000BC	131	138	89	49
2	c4000BC	125	131	92	48
3	c4000BC	131	132	99	50
4	c4000BC	119	132	96	44
5	c4000BC	136	143	100	54
6	c4000BC	138	137	89	56
7	c4000BC	139	130	108	48
8	c4000BC	125	136	93	48
9	c4000BC	131	134	102	51
10	c4000BC	134	134	99	51
11	c4000BC	129	138	95	50
12	c4000BC	134	121	95	53
13	c4000BC	126	129	109	51
14	c4000BC	132	136	100	50
15	c4000BC	141	140	100	51
16	c4000BC	131	134	97	54
17	c4000BC	135	137	103	50
18	c4000BC	132	133	93	53
19	c4000BC	139	136	96	50
20	c4000BC	132	131	101	49
21	c4000BC	126	133	102	51
22	c4000BC	135	135	103	47
23	c4000BC	134	124	93	53
24	c4000BC	128	134	103	50
25	c4000BC	130	130	104	49
26	c4000BC	138	135	100	55
27	c4000BC	128	132	93	53
28	c4000BC	127	129	106	48
29	c4000BC	131	136	114	54
30	c4000BC	124	138	101	46
...	...	...	...	...	...

[10]:
```
using Statistics

# Response matrix Y (dependent variables: mb, bh, bl, nh)
Y = Matrix(skulls[:, [:mb, :bh, :bl, :nh]])

# Encode epoch as categorical
skulls.epoch = categorical(skulls.epoch)
X = hcat(ones(nrow(skulls)), Matrix(skulls[:, [:epoch]]))
```

[10]:
```
150×2 Matrix{Any}:
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
```

```
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")
 1.0  String7("c4000BC")

 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
 1.0  String7("cAD150")
```

[11]:

```
# Mean vector of Y
Y_bar = mean(Y, dims=1)

# Compute the total SSCP matrix
T = (Y .- Y_bar)' * (Y .- Y_bar)

# Compute the within-group SSCP matrix
groups = groupby(skulls, :epoch)
W = zeros(4, 4)  # There are 4 measurements: mb, bh, bl, nh
for g in groups
    Y_g = Matrix(g[:, [:mb, :bh, :bl, :nh]])
    Y_g_bar = mean(Y_g, dims=1)
    W += (Y_g .- Y_g_bar)' * (Y_g .- Y_g_bar)
end

# Between-group SSCP matrix
B = T - W
```

[11]:

```
4×4 Matrix{Float64}:
  502.827  -228.147   -626.627   135.433
 -228.147   229.907    292.28    -66.0667
 -626.627   292.28     803.293  -180.733
  135.433   -66.0667  -180.733    61.2
```

[12]:

```julia
using LinearAlgebra

# Compute the Hotelling-Lawley trace
W_inv = inv(W)
eigvals_ = LinearAlgebra.eigvals

# Compute the eigenvalues of W ¹B
eigvals_result = eigvals_(W_inv * B)

# Hotelling-Lawley trace is the sum of the eigenvalues
hotelling_lawley_trace = sum(eigvals_result)

println("Hotelling-Lawley Trace: ", hotelling_lawley_trace)
```

```
Hotelling-Lawley Trace: 0.48181907888097136
```

[15]:

```julia
using LinearAlgebra

# Define trace function (sum of diagonal elements)
trace(M) = sum(diag(M))

# Compute the matrix (W + B)
W_plus_B = W + B

# Invert the (W + B) matrix
W_plus_B_inv = inv(W_plus_B)

# Compute the product (W + B) ¹ * B
product_matrix = W_plus_B_inv * B

# Compute Pillai's trace (sum of the diagonal elements)
pillai_trace = trace(product_matrix)

println("Pillai's Trace: ", pillai_trace)
```

```
Pillai's Trace: 0.3533055662224513
```

[16]:

```julia
using LinearAlgebra

# Compute the total SSCP matrix (W + B)
W_plus_B = W + B

# Compute the determinant of W and (W + B)
det_W = det(W)
det_W_plus_B = det(W_plus_B)

# Compute Wilks' Lambda
wilks_lambda = det_W / det_W_plus_B
```

```julia
println("Wilks' Lambda: ", wilks_lambda)
```

```
Wilks' Lambda: 0.6635857985781096
```

[17]:

```julia
using SimpleANOVA, DataFrames

# Perform ANOVA for each dependent variable (mb, bh, bl, nh)

# ANOVA for mb
anova_mb = anova(skulls, :mb, [:epoch])

# ANOVA for bh
anova_bh = anova(skulls, :bh, [:epoch])

# ANOVA for bl
anova_bl = anova(skulls, :bl, [:epoch])

# ANOVA for nh
anova_nh = anova(skulls, :nh, [:epoch])

# Display results
println("ANOVA for mb:")
println(anova_mb)
println("\nANOVA for bh:")
println(anova_bh)
println("\nANOVA for bl:")
println(anova_bl)
println("\nANOVA for nh:")
println(anova_nh)
```

```
ANOVA for mb:

Analysis of Variance Results

Effect         SS    DF         MS         F            p          ²
---------------------------------------------------------------
 Total  3563.89    149
 epoch   502.827     4   125.707    5.95461  0.000182632  0.116704
 Error  3061.07    145    21.1108


ANOVA for bh:

Analysis of Variance Results

Effect         SS    DF        MS         F          p          ²
-------------------------------------------------------------
```

```
 Total  3635.17   149
 epoch   229.907    4  57.4767  2.44742  0.0489699  0.0371634
 Error  3405.27   145  23.4846
```

```
ANOVA for bl:

Analysis of Variance Results

Effect        SS   DF        MS        F          p         ²
--------------------------------------------------------------
 Total  4309.26   149
 epoch   803.293    4  200.823   8.30566  4.63639e-6  0.163052
 Error  3505.97   145   24.1791


ANOVA for nh:

Analysis of Variance Results

Effect       SS   DF       MS      F        p          ²
---------------------------------------------------------
 Total  1533.33  149
 epoch    61.2     4  15.3     1.507  0.203179  0.0133396
 Error  1472.13  145  10.1526
```

[ ]:

[ ]: