Autistic Spectrum Disorder (ASD)

In [45]:

```
using CSV, DataFrames, GLM, Random, Statistics, StatsBase, Plots, EvalMetrics, Gadfly
```

In [46]:

```
ENV["COLUMNS"] = 200
```

Out[46]:

```
200
```

In [47]:

```
dt = CSV.read("Autism.csv", DataFrame)
```

Out[47]: 704 rows × 20 columns (omitted printing of 2 columns)

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
1	1	1	1	1	1	0	0	1	1	0	0	26	f	W Europ
2	2	1	1	0	1	0	0	0	1	0	1	24	m	La
3	3	1	1	0	1	1	0	1	1	1	1	27	m	La
4	4	1	1	0	1	0	0	1	1	0	1	35	f	W Europ
5	5	1	0	0	0	0	0	0	1	0	0	40	f	
6	6	1	1	1	1	1	0	1	1	1	1	36	m	Ot
7	7	0	1	0	0	0	0	0	1	0	0	17	f	I
8	8	1	1	1	1	0	0	0	0	1	0	64	m	W Europ
9	9	1	1	0	0	1	0	0	1	1	1	29	m	W Europ

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
10	10	1	1	1	1	0	1	1	1	1	0	17	m	/
11	11	1	1	1	1	1	1	1	1	1	1	33	m	W Europ
12	12	0	1	0	1	1	1	1	0	0	1	18	f	Mi Eas
13	13	0	1	1	1	1	1	0	0	1	0	17	f	
14	14	1	0	0	0	0	0	1	1	0	1	17	m	
15	15	1	0	0	0	0	0	1	1	0	1	17	f	
16	16	1	1	0	1	1	0	0	1	0	1	18	m	Mi Eas
17	17	1	0	0	0	0	0	1	1	1	1	31	m	Mi Eas
18	18	0	0	0	0	0	0	0	1	0	1	30	m	W Europ
19	19	0	0	1	0	1	1	0	0	0	0	35	f	Mi Eas
20	20	0	0	0	0	0	0	1	1	0	1	34	m	
21	21	0	1	1	1	0	0	0	0	0	0	38	m	
22	22	0	0	0	0	0	0	0	0	0	0	27	f	I
23	23	0	0	0	1	0	0	1	1	1	1	27	m	Mi Eas
24	24	0	0	0	0	0	0	0	1	0	1	42	m	Mi Eas

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
25	25	1	1	1	1	0	0	0	1	0	0	43	m	
26	26	0	1	1	0	0	0	0	1	0	0	24	f	
27	27	0	0	0	0	0	0	0	1	0	0	40	m	Pa
28	28	0	0	0	0	0	0	0	1	0	0	40	m	Mi Ea
29	29	0	0	0	0	0	0	0	1	0	0	48	m	I
30	30	0	1	1	0	0	0	0	0	1	1	31	m	Mi Ea
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

In [48]:

```
names(dt)
```

Out[48]:

```
20-element Vector{String}:
 "Column1"
 "A1_Score"
 "A2_Score"
 "A3_Score"
 "A4_Score"
 "A5_Score"
 "A6_Score"
 "A7_Score"
 "A8_Score"
 "A9_Score"
 "A10_Score"
 "age"
 "gender"
 "ethnicity"
 "jundice"
 "contry_of_res"
 "used_app_before"
```

```
"age_desc"
"relation"
"austim"
```

In [49]:
```
sz = size(dt)
```

Out[49]:
```
(704, 20)
```

(a) Variables

In [50]:
```
describe(dt)
```

Out[50]: 20 rows × 7 columns

	variable	mean	min	median	max	nmissing	eltype
	Symbol	**Union...**	**Any**	**Union...**	**Any**	**Int64**	**DataType**
1	Column1	352.5	1	352.5	704	0	Int64
2	A1_Score	0.721591	0	1.0	1	0	Int64
3	A2_Score	0.453125	0	0.0	1	0	Int64
4	A3_Score	0.457386	0	0.0	1	0	Int64
5	A4_Score	0.495739	0	0.0	1	0	Int64
6	A5_Score	0.49858	0	0.0	1	0	Int64
7	A6_Score	0.284091	0	0.0	1	0	Int64
8	A7_Score	0.417614	0	0.0	1	0	Int64
9	A8_Score	0.649148	0	1.0	1	0	Int64
10	A9_Score	0.323864	0	0.0	1	0	Int64
11	A10_Score	0.573864	0	1.0	1	0	Int64
12	age		17		NA	0	String3
13	gender		f		m	0	String1
14	ethnicity		Asian		others	0	String15

	variable	mean	min	median	max	nmissing	eltype
	Symbol	Union...	Any	Union...	Any	Int64	DataType
15	jundice		no		yes	0	String3
16	contry_of_res		Afghanistan		Viet Nam	0	String31
17	used_app_before		no		yes	0	String3
18	age_desc		18 and more		18 and more	0	String15
19	relation		Health care professional		Self	0	String31
20	austim		no		yes	0	String3

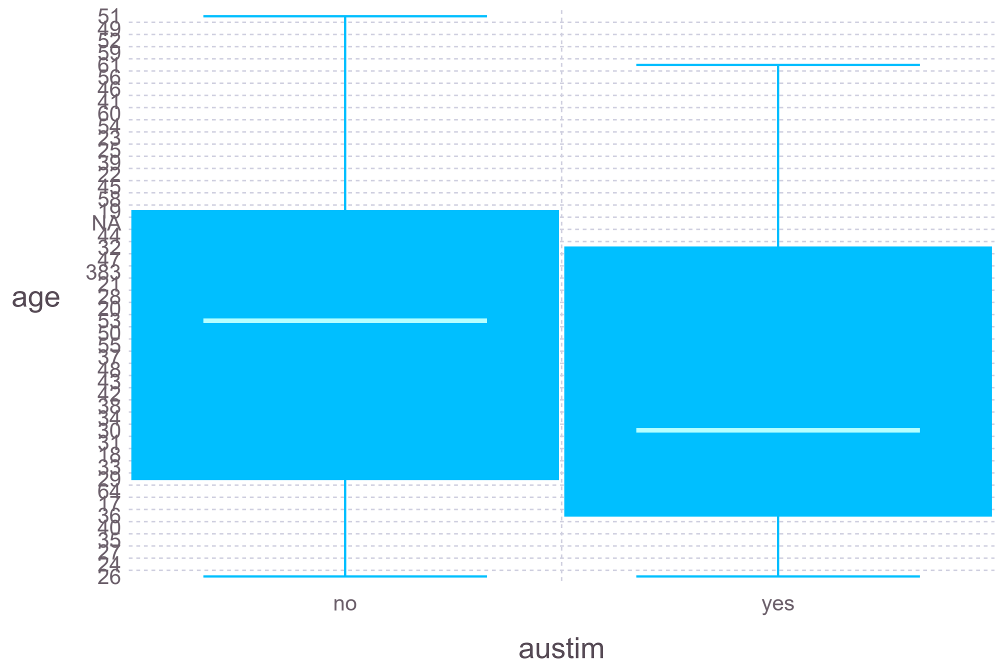
- Variables of interest in an experiment (those that are measured or observed) are called response or dependent variables. Other variables in the experiment that affect the response and can be set or measured by the experimenter are called predictor, explanatory, or independent variables. [Reference: Minitab]
- This variable selection could be a dynamic problem for this situation. In the initail guess, we can consider **austim** as the response variable. But this would not be only the case.
- Potential response/dependent variables: **austim**, **jundice**
- Potential Predictors: **relation**,**used_app_before**, **contry_of_res**, **age**, **gender**, **ethnicity**, ***behavioural features (AQ-10-Adult)*
- However, it may be varied likely according to the context and requirments.
- From the **describe()** function, we observed there is no missing value, although have some "N/A" values. We have to clean those values in order to perform the algorithms into the data

(b) Graphical Association

In [51]:

```
Gadfly.plot(dt, x=:austim, y=:age, Geom.boxplot)
```

Out[51]:



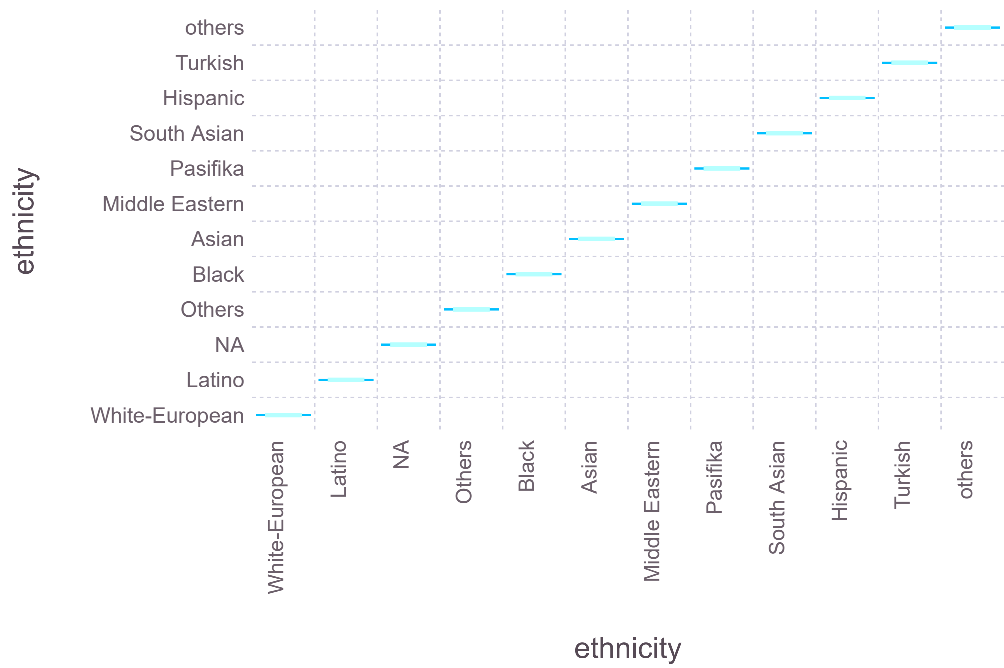
In [52]:

```
Gadfly.plot(dt, x=:austim, y=:relation, Geom.boxplot)
```

Out[52]:

In [55]:
```
Gadfly.plot(dt, x=:ethnicity, y=:ethnicity, Geom.boxplot)
```

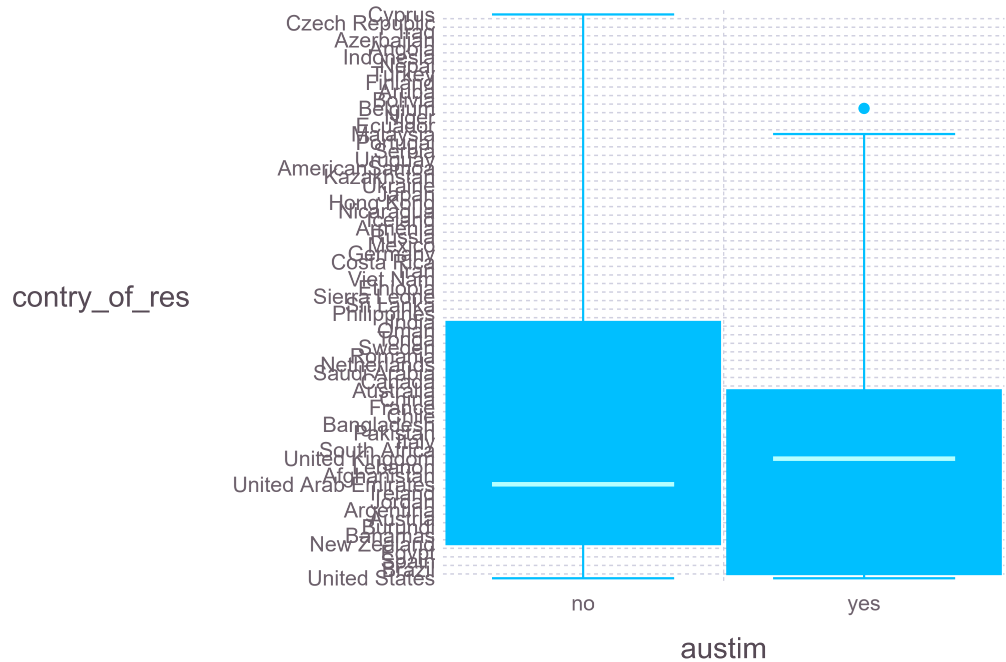
Out[55]:



In [56]:

```
Gadfly.plot(dt, x=:austim, y=:contry_of_res, Geom.boxplot)
```

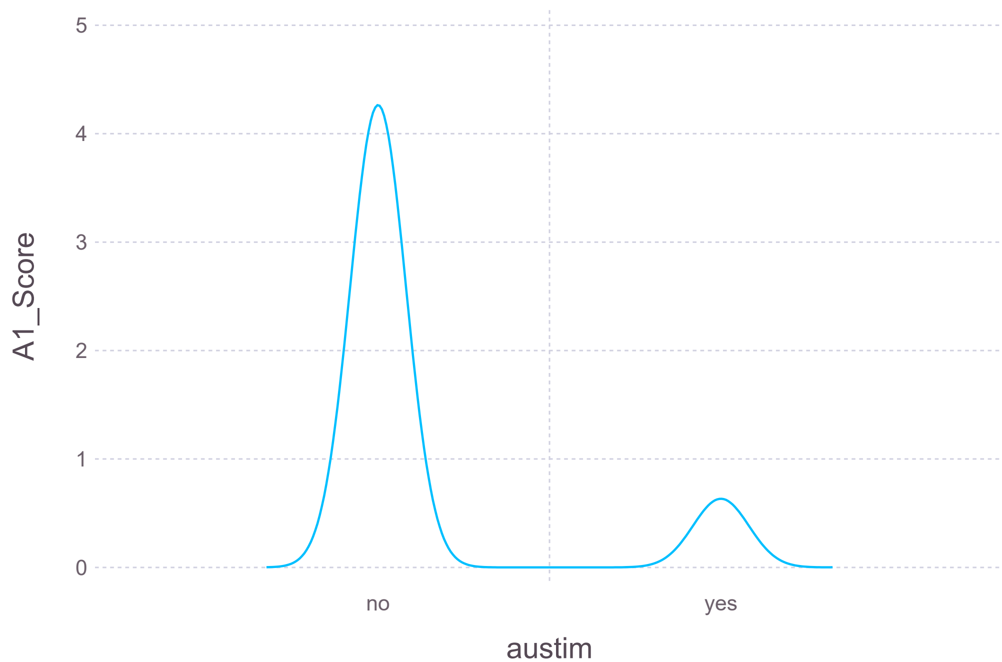
Out[56]:



In [113…

```
Gadfly.plot(dt, x=:austim, y=:A1_Score, Geom.density)
```

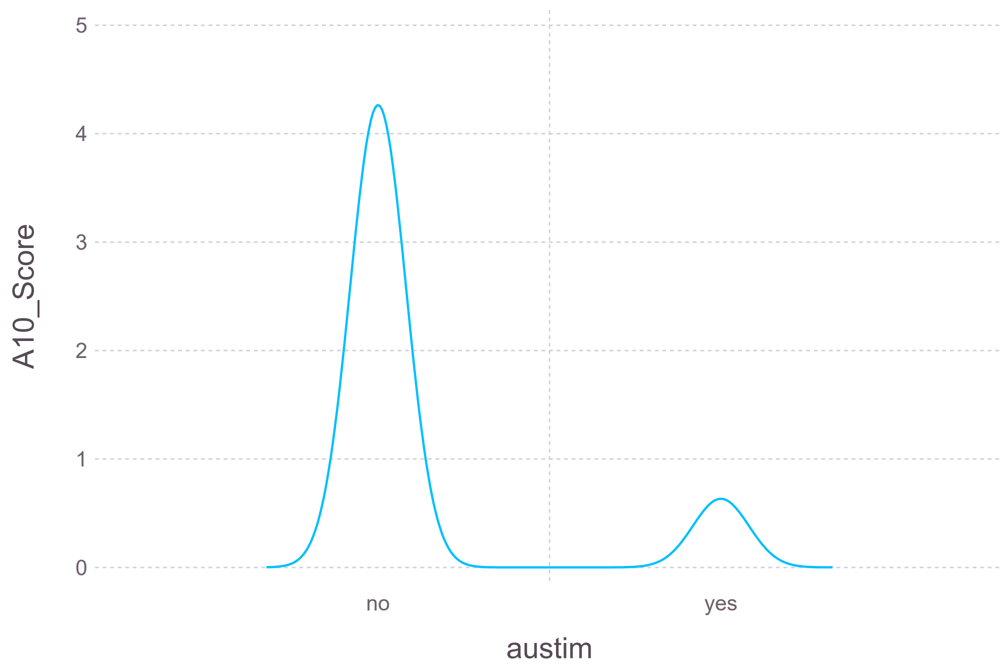
Out[113…



In [114…

```
Gadfly.plot(dt, x=:austim, y=:A10_Score, Geom.density)
```

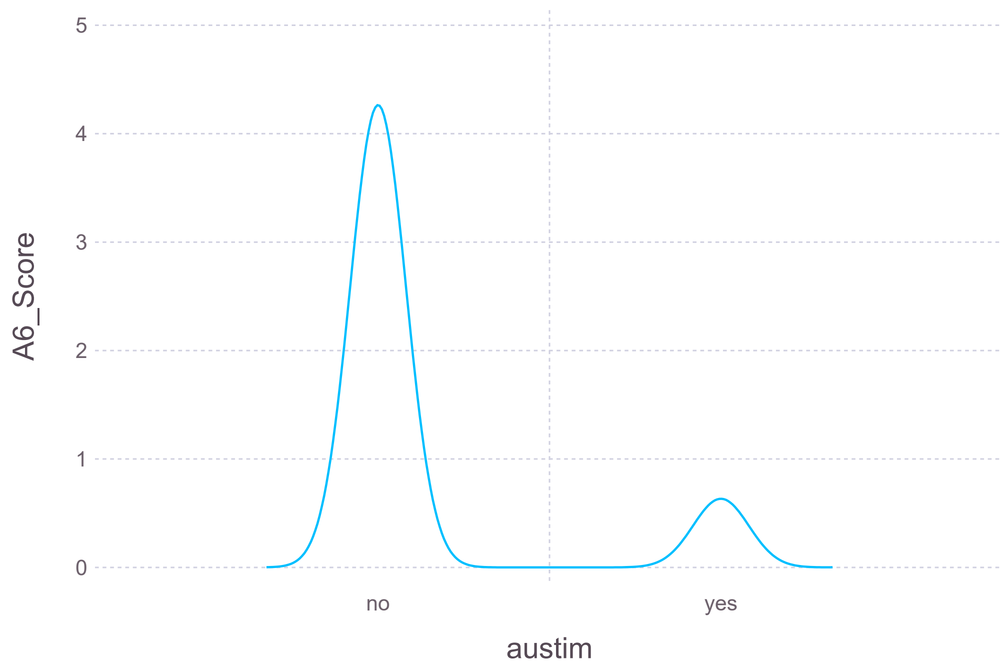
Out[114…



In [115…

```
Gadfly.plot(dt, x=:austim, y=:A6_Score, Geom.density)
```

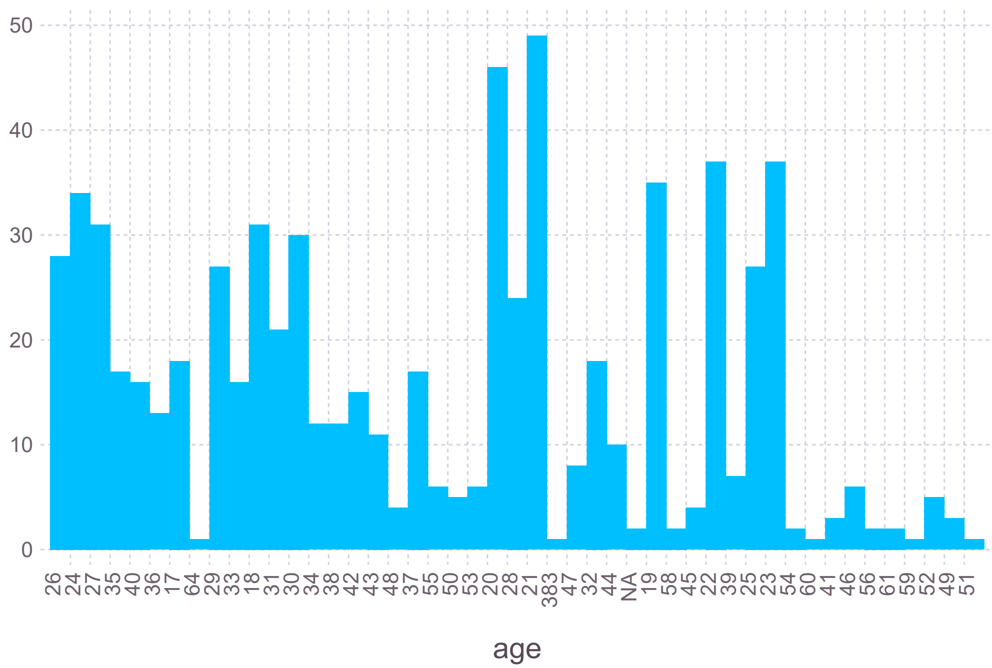
Out[115…



In [117…

```
Gadfly.plot(dt, x= :age, Geom.histogram)
```

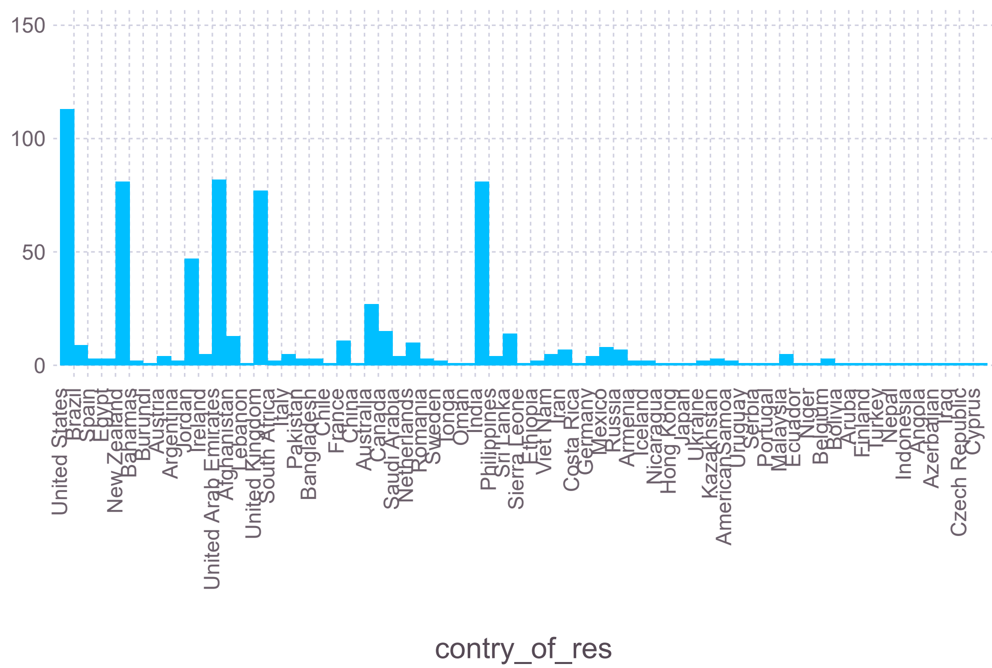
Out[117…



In [118…

```
Gadfly.plot(dt, x= :contry_of_res, Geom.histogram)
```

Out[118…



(c) Fitting a Logistic Regression

preprocessing

In [57]:
```
data = dt[dt[!, :age].!="NA",:]
```

Out[57]: 702 rows × 20 columns (omitted printing of 2 columns)

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
1	1	1	1	1	1	0	0	1	1	0	0	26	f	W Euro
2	2	1	1	0	1	0	0	0	1	0	1	24	m	La

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
3	3	1	1	0	1	1	0	1	1	1	1	27	m	La
4	4	1	1	0	1	0	0	1	1	0	1	35	f	W Euro
5	5	1	0	0	0	0	0	0	1	0	0	40	f	
6	6	1	1	1	1	1	0	1	1	1	1	36	m	Ot
7	7	0	1	0	0	0	0	0	1	0	0	17	f	I
8	8	1	1	1	1	0	0	0	0	1	0	64	m	W Euro
9	9	1	1	0	0	1	0	0	1	1	1	29	m	W Euro
10	10	1	1	1	1	0	1	1	1	1	0	17	m	/
11	11	1	1	1	1	1	1	1	1	1	1	33	m	W Euro
12	12	0	1	0	1	1	1	1	0	0	1	18	f	Mi Ea
13	13	0	1	1	1	1	1	0	0	1	0	17	f	
14	14	1	0	0	0	0	0	1	1	0	1	17	m	
15	15	1	0	0	0	0	0	1	1	0	1	17	f	
16	16	1	1	0	1	1	0	0	1	0	1	18	m	Mi Ea
17	17	1	0	0	0	0	0	1	1	1	1	31	m	Mi Ea

	Column1	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	age	gender	ethn
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	String3	String1	Strir
18	18	0	0	0	0	0	0	0	1	0	1	30	m	W Europ
19	19	0	0	1	0	1	1	0	0	0	0	35	f	Mi Ea:
20	20	0	0	0	0	0	0	1	1	0	1	34	m	
21	21	0	1	1	1	0	0	0	0	0	0	38	m	
22	22	0	0	0	0	0	0	0	0	0	0	27	f	I
23	23	0	0	0	1	0	0	1	1	1	1	27	m	Mi Ea:
24	24	0	0	0	0	0	0	0	1	0	1	42	m	Mi Ea:
25	25	1	1	1	1	0	0	0	1	0	0	43	m	
26	26	0	1	1	0	0	0	0	1	0	0	24	f	
27	27	0	0	0	0	0	0	0	1	0	0	40	m	Pa:
28	28	0	0	0	0	0	0	0	1	0	0	40	m	Mi Ea:
29	29	0	0	0	0	0	0	0	1	0	0	48	m	I
30	30	0	1	1	0	0	0	0	0	1	1	31	m	Mi Ea:
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

In [58]:
```
replace!(data.ethnicity, "NA" => "0" )
```

```
replace!(data.ethnicity, "White-European" => "1" )
replace!(data.ethnicity, "Latino" => "2" )
replace!(data.ethnicity, "Black" => "3" )
replace!(data.ethnicity, "Asian" => "4" )
replace!(data.ethnicity, "Middle Eastern " => "5" )
replace!(data.ethnicity, "Pasifika" => "6" )
replace!(data.ethnicity, "South Asian" => "7" )
replace!(data.ethnicity, "Hispanic" => "8" )
replace!(data.ethnicity, "Turkish" => "9" )
replace!(data.ethnicity, "Others" => "10" )
replace!(data.ethnicity, "others" => "11" )
```

Out[58]:

```
702-element PooledArrays.PooledVector{String15, UInt32, Vector{UInt32}}:
 "1"
 "2"
 "2"
 "1"
 "0"
 "10"
 "3"
 "1"
 "1"
 "4"
 "1"
 "5"
 "0"
 ⋮
 "1"
 "1"
 "1"
 "2"
 "9"
 "4"
 "6"
 "1"
 "8"
 "0"
 "7"
 "1"
```

In [59]:

```
data[!, :age] = parse.(Float64, data.age);
data[!, :ethnicity] = parse.(Float64, data.ethnicity);
```

In [60]:

```
data[!, :b_austim] = ifelse.(data.austim .== "yes", 1, 0);
data[!, :b_jundice] = ifelse.(data.jundice .== "yes", 1, 0);
data[!, :b_used_app_before] = ifelse.(data.used_app_before .== "yes", 1, 0);
data[!, :b_gender] = ifelse.(data.gender.== "f", 1, 0);
data[!, :b_age] = ifelse.(data.age .>= mean(data.age), 1, 0);
```

In [61]:

```
describe(data)
```

Out[61]: 25 rows × 7 columns

	variable	mean	min	median	max	nmissing	eltype
	Symbol	Union...	Any	Union...	Any	Int64	DataType
1	Column1	353.283	1	353.5	704	0	Int64
2	A1_Score	0.723647	0	1.0	1	0	Int64
3	A2_Score	0.452991	0	0.0	1	0	Int64
4	A3_Score	0.458689	0	0.0	1	0	Int64
5	A4_Score	0.497151	0	0.0	1	0	Int64
6	A5_Score	0.498575	0	0.0	1	0	Int64
7	A6_Score	0.2849	0	0.0	1	0	Int64
8	A7_Score	0.417379	0	0.0	1	0	Int64
9	A8_Score	0.650997	0	1.0	1	0	Int64
10	A9_Score	0.324786	0	0.0	1	0	Int64
11	A10_Score	0.574074	0	1.0	1	0	Int64
12	age	29.698	17.0	27.0	383.0	0	Float64
13	gender		f		m	0	String1
14	ethnicity	3.0584	0.0	3.0	11.0	0	Float64
15	jundice		no		yes	0	String3
16	contry_of_res		Afghanistan		Viet Nam	0	String31
17	used_app_before		no		yes	0	String3

	variable	mean	min	median	max	nmissing	eltype
	Symbol	Union...	Any	Union...	Any	Int64	DataType
18	age_desc		18 and more		18 and more	0	String15
19	relation		Health care professional		Self	0	String31
20	austim		no		yes	0	String3
21	b_austim	0.12963	0	0.0	1	0	Int64
22	b_jundice	0.0982906	0	0.0	1	0	Int64
23	b_used_app_before	0.017094	0	0.0	1	0	Int64
24	b_gender	0.478632	0	0.0	1	0	Int64
25	b_age	0.396011	0	0.0	1	0	Int64

In [62]:

```
select!(data, Not([:austim, :jundice, :age_desc, :Column1, :age, :gender, :used_app_before, :contry_of_res]))
```

Out[62]: 702 rows × 17 columns

	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	ethnicity	relation	b_austim
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	String31	Int64
1	1	1	1	1	0	0	1	1	0	0	1.0	Self	0
2	1	1	0	1	0	0	0	1	0	1	2.0	Self	1
3	1	1	0	1	1	0	1	1	1	1	2.0	Parent	1
4	1	1	0	1	0	0	1	1	0	1	1.0	Self	1
5	1	0	0	0	0	0	0	1	0	0	0.0	NA	0
6	1	1	1	1	1	0	1	1	1	1	10.0	Self	0
7	0	1	0	0	0	0	0	1	0	0	3.0	Self	0
8	1	1	1	1	0	0	0	0	1	0	1.0	Parent	0
9	1	1	0	0	1	0	0	1	1	1	1.0	Self	0

	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	ethnicity	relation	b_austim
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	String31	Int64
10	1	1	1	1	0	1	1	1	1	0	4.0	Health care professional	1
11	1	1	1	1	1	1	1	1	1	1	1.0	Relative	0
12	0	1	0	1	1	1	1	0	0	1	5.0	Parent	0
13	0	1	1	1	1	1	0	0	1	0	0.0	NA	0
14	1	0	0	0	0	0	1	1	0	1	0.0	NA	0
15	1	0	0	0	0	0	1	1	0	1	0.0	NA	0
16	1	1	0	1	1	0	0	1	0	1	5.0	Parent	1
17	1	0	0	0	0	0	1	1	1	1	5.0	Self	0
18	0	0	0	0	0	0	0	1	0	1	1.0	Self	0
19	0	0	1	0	1	1	0	0	0	0	5.0	Self	1
20	0	0	0	0	0	0	1	1	0	1	0.0	NA	0
21	0	1	1	1	0	0	0	0	0	0	0.0	NA	0
22	0	0	0	0	0	0	0	0	0	0	3.0	Self	0
23	0	0	0	1	0	0	1	1	1	1	5.0	Self	0
24	0	0	0	0	0	0	0	1	0	1	5.0	Relative	0
25	1	1	1	1	0	0	0	1	0	0	0.0	NA	0
26	0	1	1	0	0	0	0	1	0	0	0.0	NA	0
27	0	0	0	0	0	0	0	1	0	0	6.0	Self	1
28	0	0	0	0	0	0	0	1	0	0	5.0	Parent	1
29	0	0	0	0	0	0	0	1	0	0	3.0	Self	0
30	0	1	1	0	0	0	0	0	1	1	5.0	Self	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

In [63]:
```
sum(data.b_austim), length(data.b_austim) - sum(data.b_austim)
```
Out[63]:
```
(91, 611)
```
In [64]:
```
n, p = size(data)
```
Out[64]:
```
(702, 17)
```
In [66]:
```
Random.seed!(19247923)
ind = randperm(n);
train_df = data[1:500,:];
test_df = data[501:702,:];
```
Out[66]: 202 rows × 17 columns

	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	ethnicity	relation	b_austim	b_
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	String31	Int64	
1	0	1	0	0	1	0	1	1	0	0	1.0	Self	0	
2	1	0	1	1	1	0	0	1	0	1	1.0	Self	1	
3	1	0	1	1	1	0	1	1	1	1	1.0	Relative	1	
4	1	0	1	0	1	0	1	1	0	1	0.0	NA	1	
5	1	0	1	0	0	0	1	0	0	1	4.0	Self	0	
6	0	1	1	0	1	0	0	1	0	0	1.0	Self	0	
7	1	1	1	1	1	1	1	1	1	1	1.0	Self	1	
8	0	1	0	0	0	0	0	0	0	1	1.0	Self	0	
9	0	1	1	1	1	1	1	1	0	1	1.0	Self	0	
10	1	1	1	1	1	1	1	0	1	1	1.0	Self	0	
11	1	1	1	0	0	0	0	0	0	1	4.0	Self	0	
12	1	0	1	1	0	0	0	0	0	0	1.0	Self	0	
13	1	1	1	1	1	0	1	1	1	1	1.0	Self	0	

	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	A8_Score	A9_Score	A10_Score	ethnicity	relation	b_austim	b_
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	String31	Int64	
14	1	1	1	1	1	1	0	1	1	1	1.0	Self	0	
15	1	1	0	1	0	0	1	0	0	0	1.0	Parent	1	
16	1	0	1	1	1	0	1	0	1	1	3.0	Self	0	
17	0	1	0	0	0	0	0	0	1	1	0.0	NA	0	
18	0	0	1	1	0	0	0	0	0	1	5.0	Self	0	
19	1	1	1	1	1	0	0	1	1	1	1.0	Self	1	
20	1	1	1	1	1	0	0	1	0	1	1.0	Self	0	
21	1	1	1	1	1	0	1	0	1	0	1.0	Self	1	
22	1	1	0	1	1	1	0	0	0	1	3.0	Parent	1	
23	1	1	1	1	0	1	0	1	1	1	1.0	Self	0	
24	1	0	0	0	0	0	1	1	0	1	10.0	Self	0	
25	0	0	0	1	0	0	1	1	0	1	4.0	Self	0	
26	0	0	0	0	1	0	0	1	0	1	0.0	NA	0	
27	1	1	1	1	1	1	1	1	1	1	1.0	Self	1	
28	0	1	0	0	0	0	0	1	0	0	3.0	Self	0	
29	1	0	0	0	1	0	0	0	0	0	3.0	Self	0	
30	1	1	0	0	0	0	1	1	0	0	2.0	Relative	0	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

In [67]:

```
describe(train_df)
```

Out[67]: 17 rows × 7 columns

variable	mean	min	median	max	nmissing	eltype

	variable	mean	min	median	max	nmissing	eltype
	Symbol	Union...	Any	Union...	Any	Int64	DataType
1	A1_Score	0.706	0	1.0	1	0	Int64
2	A2_Score	0.426	0	0.0	1	0	Int64
3	A3_Score	0.438	0	0.0	1	0	Int64
4	A4_Score	0.488	0	0.0	1	0	Int64
5	A5_Score	0.47	0	0.0	1	0	Int64
6	A6_Score	0.286	0	0.0	1	0	Int64
7	A7_Score	0.396	0	0.0	1	0	Int64
8	A8_Score	0.654	0	1.0	1	0	Int64
9	A9_Score	0.302	0	0.0	1	0	Int64
10	A10_Score	0.548	0	1.0	1	0	Int64
11	ethnicity	3.114	0.0	3.0	10.0	0	Float64
12	relation		Health care professional		Self	0	String31
13	b_austim	0.122	0	0.0	1	0	Int64
14	b_jundice	0.076	0	0.0	1	0	Int64
15	b_used_app_before	0.016	0	0.0	1	0	Int64
16	b_gender	0.484	0	0.0	1	0	Int64
17	b_age	0.354	0	0.0	1	0	Int64

In [68]:

```
fm = @formula(b_austim ~ A1_Score+ A2_Score+ A3_Score + A4_Score + A5_Score + A6_Score + A7_Score + A8_Score + A9_Score +
```

Out[68]:

```
FormulaTerm
Response:
  b_austim(unknown)
Predictors:
  A1_Score(unknown)
  A2_Score(unknown)
```

```
A3_Score(unknown)
A4_Score(unknown)
A5_Score(unknown)
A6_Score(unknown)
A7_Score(unknown)
A8_Score(unknown)
A9_Score(unknown)
A10_Score(unknown)
b_jundice(unknown)
b_used_app_before(unknown)
relation(unknown)
b_gender(unknown)
ethnicity(unknown)
b_age(unknown)
```

In [69]:

```
logit = glm(fm, train_df, Binomial(), LogitLink())
```

Out[69]:

```
StatsModels.TableRegressionModel{GeneralizedLinearModel{GLM.GlmResp{Vector{Float64}, Binomial{Float64}, LogitLink}, GLM.DensePredChol{Float64, LinearAlgebra.Cholesky{Float64, Matrix{Float64}}}}, Matrix{Float64}}

b_austim ~ 1 + A1_Score + A2_Score + A3_Score + A4_Score + A5_Score + A6_Score + A7_Score + A8_Score + A9_Score + A10_Score + b_jundice + b_used_app_before + relation + b_gender + ethnicity + b_age

Coefficients:
```

	Coef.	Std. Error	z	Pr(>|z|)	Lower 95%	Upper 95%
(Intercept)	-1.3333	1.42516	-0.94	0.3495	-4.12657	1.45996
A1_Score	0.339191	0.372202	0.91	0.3621	-0.390311	1.06869
A2_Score	-0.303536	0.322251	-0.94	0.3462	-0.935135	0.328064
A3_Score	0.240503	0.357539	0.67	0.5012	-0.460261	0.941267
A4_Score	0.822067	0.379531	2.17	0.0303	0.0781989	1.56593
A5_Score	-0.340174	0.365566	-0.93	0.3521	-1.05667	0.376321
A6_Score	-0.107447	0.398465	-0.27	0.7874	-0.888424	0.673531
A7_Score	-0.582551	0.34414	-1.69	0.0905	-1.25705	0.0919512
A8_Score	0.153813	0.327713	0.47	0.6388	-0.488494	0.796119
A9_Score	0.178936	0.391388	0.46	0.6475	-0.588171	0.946043
A10_Score	0.736277	0.354373	2.08	0.0377	0.0417193	1.43084
b_jundice	0.862837	0.450941	1.91	0.0557	-0.0209919	1.74666
b_used_app_before	0.0166539	1.17238	0.01	0.9887	-2.28117	2.31448
relation: NA	-2.87421	1.46599	-1.96	0.0499	-5.74749	-0.000925049
relation: Others	-1.70903	1.87374	-0.91	0.3617	-5.38149	1.96343
relation: Parent	-1.21979	1.39604	-0.87	0.3823	-3.95599	1.5164
relation: Relative	-1.43978	1.46106	-0.99	0.3244	-4.3034	1.42385

```
relation: Self      -2.31119     1.35395    -1.71    0.0878  -4.96489     0.342514
b_gender             0.278783    0.316922    0.88    0.3790  -0.342372    0.899939
ethnicity           -0.0716548   0.0707726  -1.01    0.3113  -0.210367    0.067057
b_age                1.22625     0.317861    3.86    0.0001   0.603259    1.84925
─────────────────────────────────────────────────────────────────────────────────────
```

In [70]:

```
deviance(logit)
```

Out[70]:

```
310.02862189167666
```

(e) Accuracy & ROC

In [75]:

```
tpred = predict(logit, test_df);
target = test_df.b_austim;
```

In [76]:

```
using RCall
```

In [77]:

```
R"library(ROCR)"
```

Out[77]:

```
RObject{StrSxp}
[1] "ROCR"      "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"
```

In [78]:

```
@rput tpred
@rput target
```

Out[78]:

```
202-element Vector{Int64}:
 0
 1
 1
 1
 0
 0
 1
 0
 0
 0
 0
 0
```

```
0
⋮
0
0
0
0
1
0
0
0
0
0
0
0
```

threshold 0.5

In [125…

```
using EvalMetrics

th = 0.5
tpred_demo =ifelse.(tpred .>= th , 1 , 0);
accuracy(target, tpred_demo)
```

Out[125…

```
0.8415841584158416
```

In [126…

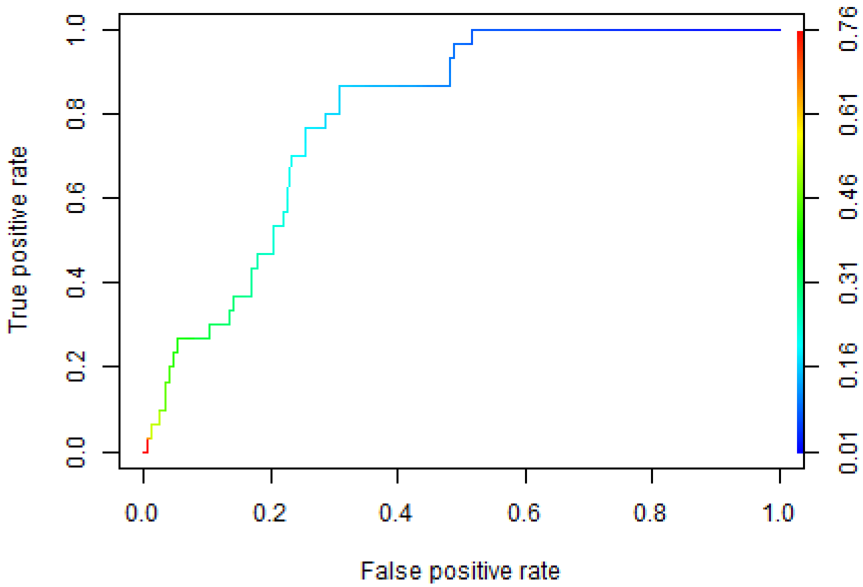
```
R"pred= prediction(tpred,target)"
R""" perf= performance(pred, "tpr", "fpr") """
```

Out[126…

```
RObject{S4Sxp}
A performance instance
  'False positive rate' vs. 'True positive rate' (alpha: 'Cutoff')
  with 194 data points
```

In [127…

```
R"plot(perf,colorize=TRUE)"
```



Out[127…
```
RObject{NilSxp}
NULL
```

In [136…
```
thres = 0.6
tpredbin=ifelse.(tpred .>= thres , 1 , 0);
```

In [137…
```
using EvalMetrics

accuracy(target, tpredbin)
```

Out[137…
```
0.8514851485148515
```

(f) Confusion Matrix

In [84]:
```
using MLBase

confusion_matrix = MLBase.roc(target, tpredbin)
```

Out[84]:
```
ROCNums{Int64}
  p = 30
  n = 172
  tp = 1
  tn = 171
  fp = 1
  fn = 29
```

- p --> number of actual positive
- n --> number of actual negative

In [87]:
```
TP = confusion_matrix.tp
TN = confusion_matrix.tn
FP = confusion_matrix.fp
FN = confusion_matrix.fn;
```

In [88]:
```
acrcy = (TP+TN) / (TP+TN+FP+FN)
```

Out[88]:
```
0.8514851485148515
```

In [89]:
```
sensitivity = (TP) / (TP + FN)
```

Out[89]:
```
0.03333333333333333
```

In [90]:
```
specificity = (TN) / (FP + TN)
```

Out[90]:
```
0.9941860465116279
```

In [91]:
```
ppv = (TP) / (TP + FP)
```

Out[91]:
```
0.5
```

In [539…
```
npv = (TN) / (TN + FN)
```

Out[539…
```
0.855
```

The probability of the model yeils a correct resutl is around 85% The probability that a positve test will truly have a disease is 50% & the negative test truly be disease free is 85% . This is really important (npv) to interpret the model is truly worked or not in this context. The probability that a true negative will test negative is also very high about 99%, that is also a good indication of the model.

(d) Dropping Insignificant Variables

In [92]:
```
select!(data, Not([:b_gender, :relation, :A3_Score, :A6_Score, :A9_Score, :b_used_app_before]))
```

Out[92]: 702 rows × 11 columns

	A1_Score	A2_Score	A4_Score	A5_Score	A7_Score	A8_Score	A10_Score	ethnicity	b_austim	b_jundice	b_age
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	Int64	Int64	Int64
1	1	1	1	0	1	1	0	1.0	0	0	0
2	1	1	1	0	0	1	1	2.0	1	0	0
3	1	1	1	1	1	1	1	2.0	1	1	0
4	1	1	1	0	1	1	1	1.0	1	0	1
5	1	0	0	0	0	1	0	0.0	0	0	1
6	1	1	1	1	1	1	1	10.0	0	1	1
7	0	1	0	0	0	1	0	3.0	0	0	0
8	1	1	1	0	0	0	0	1.0	0	0	1
9	1	1	0	1	0	1	1	1.0	0	0	0
10	1	1	1	0	1	1	0	4.0	1	1	0
11	1	1	1	1	1	1	1	1.0	0	0	1
12	0	1	1	1	1	0	1	5.0	0	0	0
13	0	1	1	1	0	0	0	0.0	0	0	0
14	1	0	0	0	1	1	1	0.0	0	0	0
15	1	0	0	0	1	1	1	0.0	0	0	0

	A1_Score	A2_Score	A4_Score	A5_Score	A7_Score	A8_Score	A10_Score	ethnicity	b_austim	b_jundice	b_age
	Int64	Int64	Int64	Int64	Int64	Int64	Int64	Float64	Int64	Int64	Int64
16	1	1	1	1	0	1	1	5.0	1	0	0
17	1	0	0	0	1	1	1	5.0	0	0	1
18	0	0	0	0	0	1	1	1.0	0	0	1
19	0	0	0	1	0	0	0	5.0	1	0	1
20	0	0	0	0	1	1	1	0.0	0	1	1
21	0	1	1	0	0	0	0	0.0	0	0	1
22	0	0	0	0	0	0	0	3.0	0	0	0
23	0	0	1	0	1	1	1	5.0	0	0	0
24	0	0	0	0	0	1	1	5.0	0	1	1
25	1	1	1	0	0	1	0	0.0	0	0	1
26	0	1	0	0	0	1	0	0.0	0	1	0
27	0	0	0	0	0	1	0	6.0	1	1	1
28	0	0	0	0	0	1	0	5.0	1	1	1
29	0	0	0	0	0	1	0	3.0	0	0	1
30	0	1	0	0	0	0	1	5.0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

In [94]:

```
fm1 = @formula(b_austim ~ A1_Score+ A2_Score+ A4_Score + A5_Score + A7_Score + A8_Score + A10_Score + b_jundice + ethnici
```

Out[94]:

```
FormulaTerm
Response:
  b_austim(unknown)
Predictors:
  A1_Score(unknown)
  A2_Score(unknown)
  A4_Score(unknown)
  A5_Score(unknown)
  A7_Score(unknown)
  A8_Score(unknown)
```

```
A10_Score(unknown)
b_jundice(unknown)
ethnicity(unknown)
b_age(unknown)
```

In [96]:

```
logit1 = glm(fm1, train_df, Binomial(), LogitLink())
```

Out[96]:

```
StatsModels.TableRegressionModel{GeneralizedLinearModel{GLM.GlmResp{Vector{Float64}, Binomial{Float64}, LogitLink}, GLM.DensePredChol{Float64, LinearAlgebra.Cholesky{Float64, Matrix{Float64}}}}, Matrix{Float64}}

b_austim ~ 1 + A1_Score + A2_Score + A4_Score + A5_Score + A7_Score + A8_Score + A10_Score + b_jundice + ethnicity + b_age

Coefficients:
```

	Coef.	Std. Error	z	Pr(>|z|)	Lower 95%	Upper 95%
(Intercept)	-3.53714	0.513309	-6.89	<1e-11	-4.54321	-2.53108
A1_Score	0.406249	0.360302	1.13	0.2595	-0.29993	1.11243
A2_Score	-0.141633	0.303972	-0.47	0.6413	-0.737406	0.45414
A4_Score	0.976446	0.338083	2.89	0.0039	0.313814	1.63908
A5_Score	-0.224946	0.324375	-0.69	0.4880	-0.860709	0.410816
A7_Score	-0.641624	0.327622	-1.96	0.0502	-1.28375	0.000502766
A8_Score	0.0535938	0.313191	0.17	0.8641	-0.56025	0.667437
A10_Score	0.705219	0.330836	2.13	0.0330	0.0567915	1.35365
b_jundice	0.880337	0.437691	2.01	0.0443	0.0224792	1.7382
ethnicity	-0.0345328	0.060589	-0.57	0.5687	-0.153285	0.0842196
b_age	1.32572	0.299988	4.42	<1e-05	0.737759	1.91369

In [97]:

```
deviance(logit1)
```

Out[97]:

```
321.4325161608459
```

In [98]:

```
tpred1 = predict(logit, test_df);
```

In [101…

```
@rput tpred1
R"pred1= prediction(tpred1,target)"
R""" perf1= performance(pred1, "tpr", "fpr") """
```

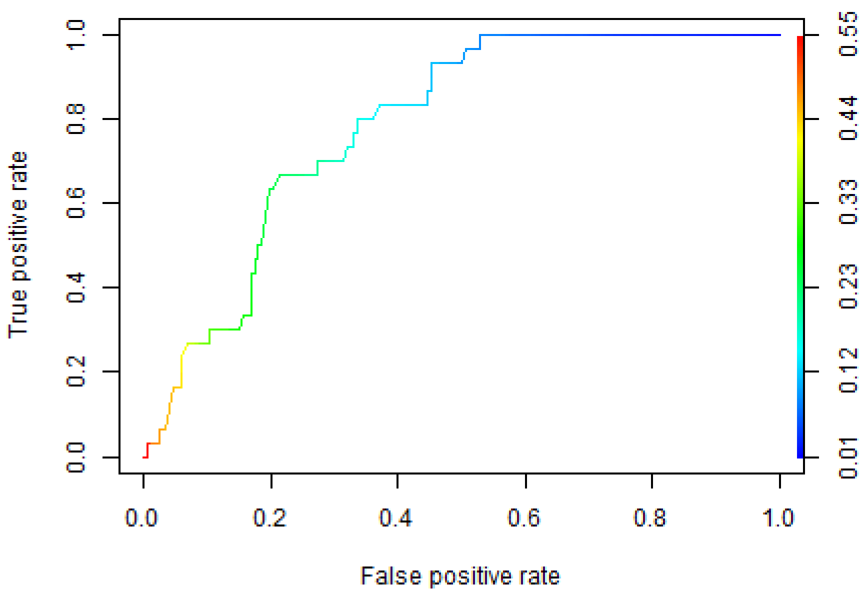
Out[101…

```
RObject{S4Sxp}
```

```
A performance instance
  'False positive rate' vs. 'True positive rate' (alpha: 'Cutoff')
  with 168 data points
```

In [102…

```
R"plot(perf1,colorize=TRUE)"
```



Out[102…

```
RObject{NilSxp}
NULL
```

In [150…

```
thres = 0.6
tpredbin1=ifelse.(tpred1 .>= thres , 1 , 0);
```

In [151…

```
accuracy(target, tpredbin1)
```

Out[151…

```
0.8514851485148515
```

In [152…

```
using MLBase

confusion_matrix1 = MLBase.roc(target, tpredbin1)
```

Out[152…
```
ROCNums{Int64}
  p = 30
  n = 172
  tp = 0
  tn = 172
  fp = 0
  fn = 30
```

In [153…
```
TP1 = confusion_matrix1.tp
TN1 = confusion_matrix1.tn
FP1 = confusion_matrix1.fp
FN1 = confusion_matrix1.fn;
```

In [154…
```
npv = (TN1) / (TN1 + FN1)
```

Out[154…
```
0.8514851485148515
```

In [155…
```
specificity = (TN1) / (FP1 + TN1)
```

Out[155…
```
1.0
```

Overall, the logistic model also perfomed well when dropping some insignificant variables

In [ ]:
```

```