

Parte A

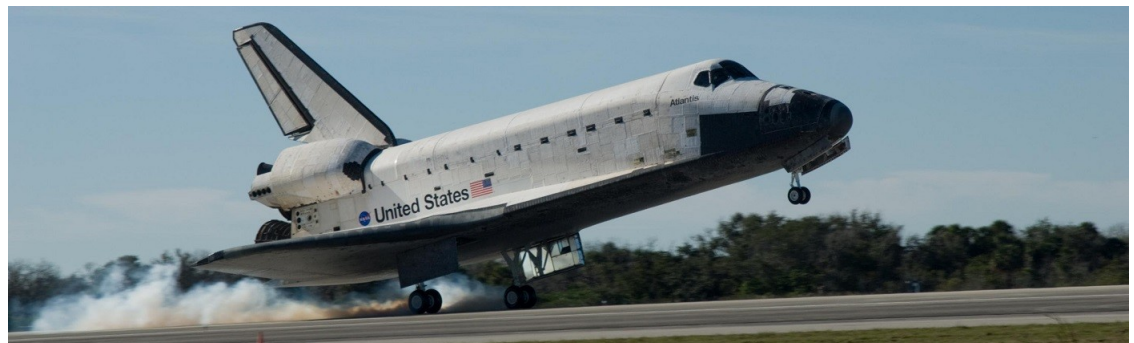
1. Abra la base shuttle de la librería MASS. Se quiere predecir el uso o no del autolander (variable a predecir: “use”). Renombre la base como “base”.

```
> library(MASS)  
> base=shuttle
```

2. Indique de qué trata el problema

Problema del Autolander del Transbordador Espacial

El problema trata acerca de determinar cuando conviene utilizar el sistema de aterrizaje en forma automatica.



3. Muestre un head de la base.

```
> head(base)
  stability error sign wind  magn vis  use
1    xstab   LX   pp head  Light no auto
2    xstab   LX   pp head Medium no auto
3    xstab   LX   pp head Strong no auto
4    xstab   LX   pp tail  Light no auto
5    xstab   LX   pp tail Medium no auto
6    xstab   LX   pp tail Strong no auto
```

4. Comente las variables, cantidad de registros, cantidad de registros de cada clase.

Info del Data Frame	Cantidad		CantVariables		CantRegistros
	Registros	256			
	Variables	7			
	Categoricas		6		
	Decision		1		
	Entrenamiento			253	
	Test			3	

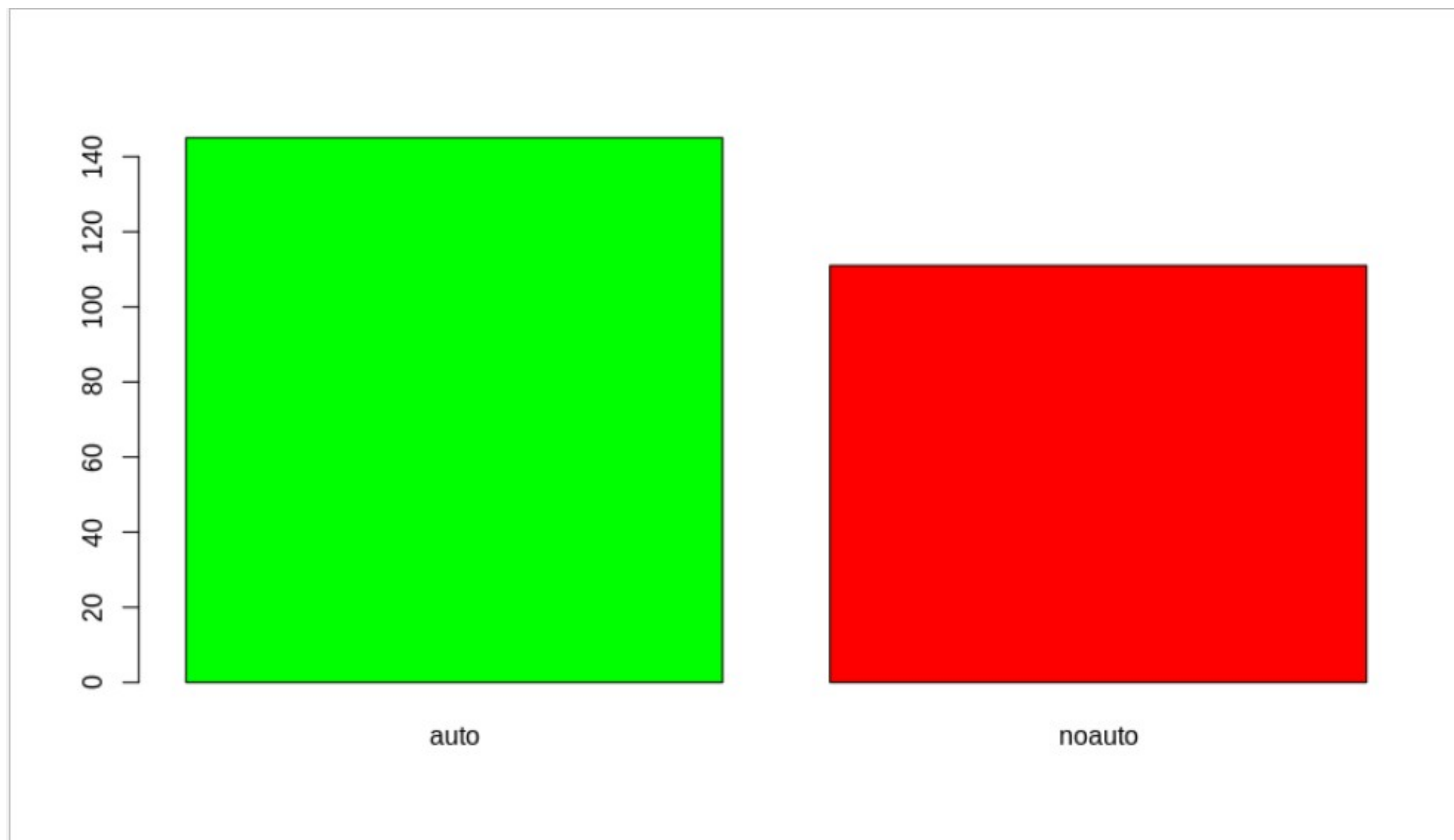
Definicion de las Variables

Variable	Definicion	Valores	Tipo
<i>Categoricas</i>			
stability	Posicion estable o no	stab / xstab	Cualitativa/Nominal
error	tamaño del error	MM / SS / LX / XL	Cualitativa/Nominal
sign	señal de error	pp / nn positive or negative	Cualitativa/Nominal
wind	viento de donde	Head / tail	Cualitativa/Nominal
magn	fuerza del viento	Light / Medium / Strong / Out of Range	Cualitativa/Nominal
vis	visibilidad	Yes / no	Cualitativa/Nominal
<i>Decision</i>			
use	usar el autolander	auto / noauto	Cualitativa/Nominal

```
> summary(base)
stability  error  sign    wind    magn    vis    use
stab :128   LX:64   nn:128  head:128  Light :64  no :128  auto  :145
xstab:128   MM:64   pp:128  tail:128  Medium:64 yes:128  noauto:111
              SS:64
              XL:64
              Out  :64
              Strong:64
```

5. Realice un gráfico de barras de la variable “use”.

```
> barplot(table(shuttle$use), col=c("green", "red"))
```



Parte B

6. Cargue la librería caret, setee la semilla=8 y particione la base en un conjunto de entrenamiento y uno de testeo, utilizando la instrucción createDataPartition de la librería caret. Setee $p=(0.70)$. Indique el código R utilizado.

```
> library(caret)
> set.seed(8)
> particion=createDataPartition(y=shuttle$use, p=0.7, list=FALSE)
> train=shuttle[particion, ]
> test=shuttle[-particion, ]
```

7. Muestre un head, str y un summary del conjunto de entrenamiento y del conjunto de testeo.
¿Cuántos registros quedaron en cada conjunto?
¿Cuántos registros de cada clase quedaron en cada conjunto?

```
> head(train)
  stability error sign wind  magn vis  use
1      xstab  LX   pp head  Light no auto
2      xstab  LX   pp head Medium no auto
5      xstab  LX   pp tail Medium no auto
7      xstab  LX   nn head  Light no auto
11     xstab  LX   nn tail Medium no auto
12     xstab  LX   nn tail Strong no auto
```

```
> str(train)
'data.frame':    180 obs. of  7 variables:
 $ stability: Factor w/ 2 levels "stab","xstab": 2 2 2 2 2 2 2 2 2 2 2 ...
 $ error    : Factor w/ 4 levels "LX","MM","SS",...: 1 1 1 1 1 1 4 4 4 4 ...
 $ sign     : Factor w/ 2 levels "nn","pp": 2 2 2 1 1 1 2 2 2 2 ...
 $ wind     : Factor w/ 2 levels "head","tail": 1 1 2 1 2 2 1 1 1 2 ...
 $ magn     : Factor w/ 4 levels "Light","Medium",...: 1 2 2 1 2 4 1 2 4 2 ...
 $ vis      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ use      : Factor w/ 2 levels "auto","noauto": 1 1 1 1 1 1 1 1 1 1 ...
```

```
> summary(train)
stability  error    sign      wind      magn      vis      use
stab :92    LX:42    nn:88    head:89   Light :45   no :89    auto  :102
xstab:88    MM:51    pp:92    tail:91   Medium:49  yes:91    noauto: 78
              SS:44
              XL:43
              Out  :40
              Strong:46
```

```
> head(test)
  stability error sign wind  magn vis  use
3      xstab  LX   pp head Strong no auto
4      xstab  LX   pp tail  Light no auto
6      xstab  LX   pp tail Strong no auto
8      xstab  LX   nn head Medium no auto
9      xstab  LX   nn head Strong no auto
10     xstab  LX   nn tail  Light no auto
```

```
> str(test)
'data.frame':    76 obs. of  7 variables:
 $ stability: Factor w/ 2 levels "stab","xstab": 2 2 2 2 2 2 2 2 2 2 2 ...
 $ error    : Factor w/ 4 levels "LX","MM","SS",...: 1 1 1 1 1 1 4 4 4 2 ...
 $ sign     : Factor w/ 2 levels "nn","pp": 2 2 2 1 1 1 2 1 1 2 ...
 $ wind     : Factor w/ 2 levels "head","tail": 1 2 2 1 1 2 2 1 2 1 ...
 $ magn     : Factor w/ 4 levels "Light","Medium",...: 4 1 4 2 4 1 1 2 1 1 ...
 $ vis      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ use      : Factor w/ 2 levels "auto","noauto": 1 1 1 1 1 1 1 1 1 1 ...
```

```
> summary(test)
stability  error    sign      wind      magn      vis      use
stab :36    LX:22    nn:40    head:39   Light :19   no :39    auto  :43
xstab:40    MM:13    pp:36    tail:37   Medium:15  yes:37    noauto:33
              SS:20
              XL:21
              Out  :24
              Strong:18
```

Cant.de Registros	
Train	180
Test	76

Cant.de Reg.x Clase		
	auto	noauto
Train	102	78
Test	43	33

Parte C

8. Cree un Árbol de Decisión (con librería rpart) para modelar el problema planteado.

Indique el código R utilizado.

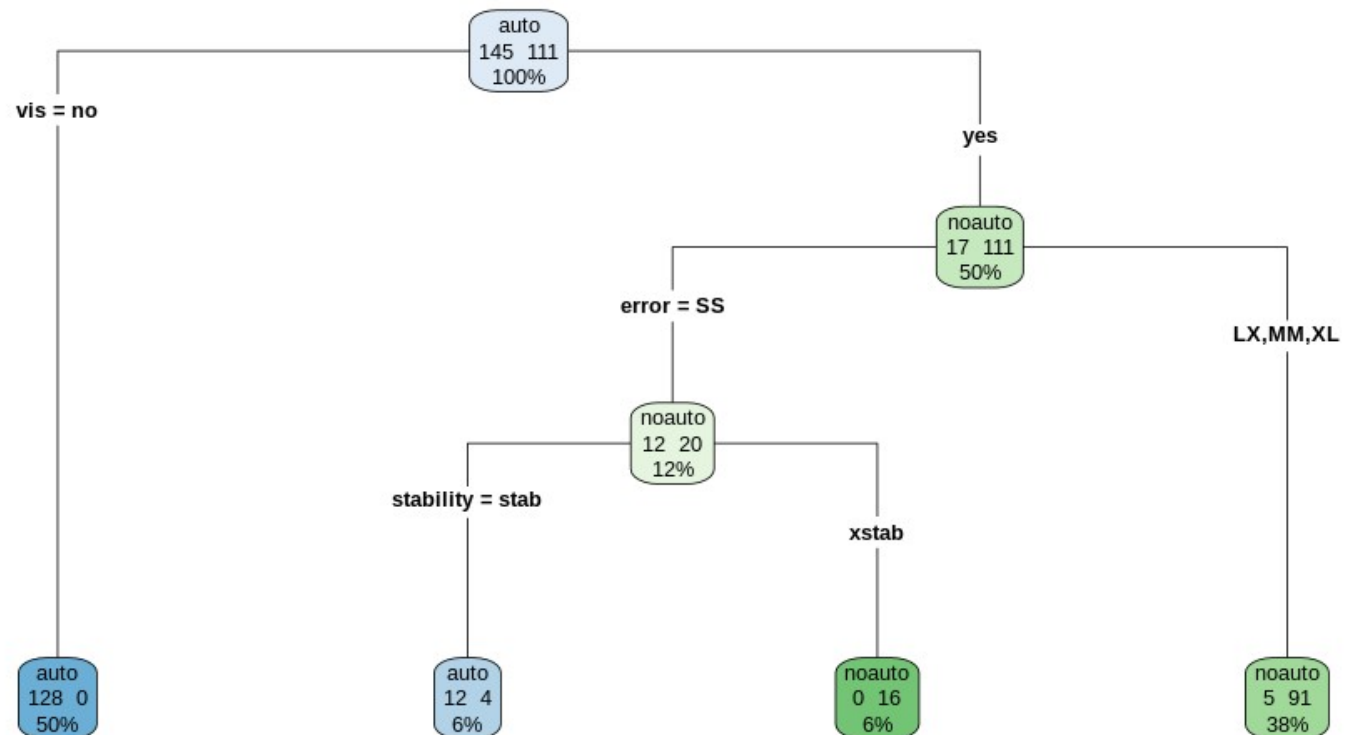
```
> library(rpart)
```

```
> arbol=rpart(use~., shuttle, method="class")
```

9. Grafique el árbol de decisión resultante con la instrucción rpart.plot de la librería rpart.plot.

```
> library(rpart.plot)
```

```
> rpart.plot(arbol, extra=101, type=4, cex=0.8)
```



10. Calcule la matriz de confusión utilizando la instrucción confusionMatrix de la librería caret.
Muestre una imagen de los resultados obtenidos.

```
> pred=predict(arbol, test, type="class")  
> confusionMatrix(pred, test$use)
```

Confusion Matrix and Statistics

	Reference		
Prediction	auto	noauto	
auto	42	2	
noauto	1	31	

Accuracy : 0.9605
95% CI : (0.8889, 0.9918)
No Information Rate : 0.5658
P-Value [Acc > NIR] : 5.333e-15

Kappa : 0.9194

Mcnemar's Test P-Value : 1

Sensitivity : 0.9767
Specificity : 0.9394
Pos Pred Value : 0.9545
Neg Pred Value : 0.9687
Prevalence : 0.5658
Detection Rate : 0.5526
Detection Prevalence : 0.5789
Balanced Accuracy : 0.9581

'Positive' Class : auto

11. ¿Cuántos registros quedaron bien y mal clasificados?

Clasificados			
Bien	42 + 31 =	73	
Mal	1 + 2 =	3	

12. ¿Cómo fue la performance del modelo? (accuracy, sensibilidad y especificidad)

$$\text{Accuracy} = (42+31) / (42+31+1+2) = 0.9605$$

$$\text{Sensibilidad} = 42 / (42 + 1) = 0.9767$$

$$\text{Especificidad} = 31 / (31 + 2) = 0.9394$$

13. Según el árbol de decisión creado, ¿cuándo debería usarse el autolander?

Cuando

No hay visibilidad o

Hay visibilidad, tamaño del error es SS y se encuentra en posición Estable

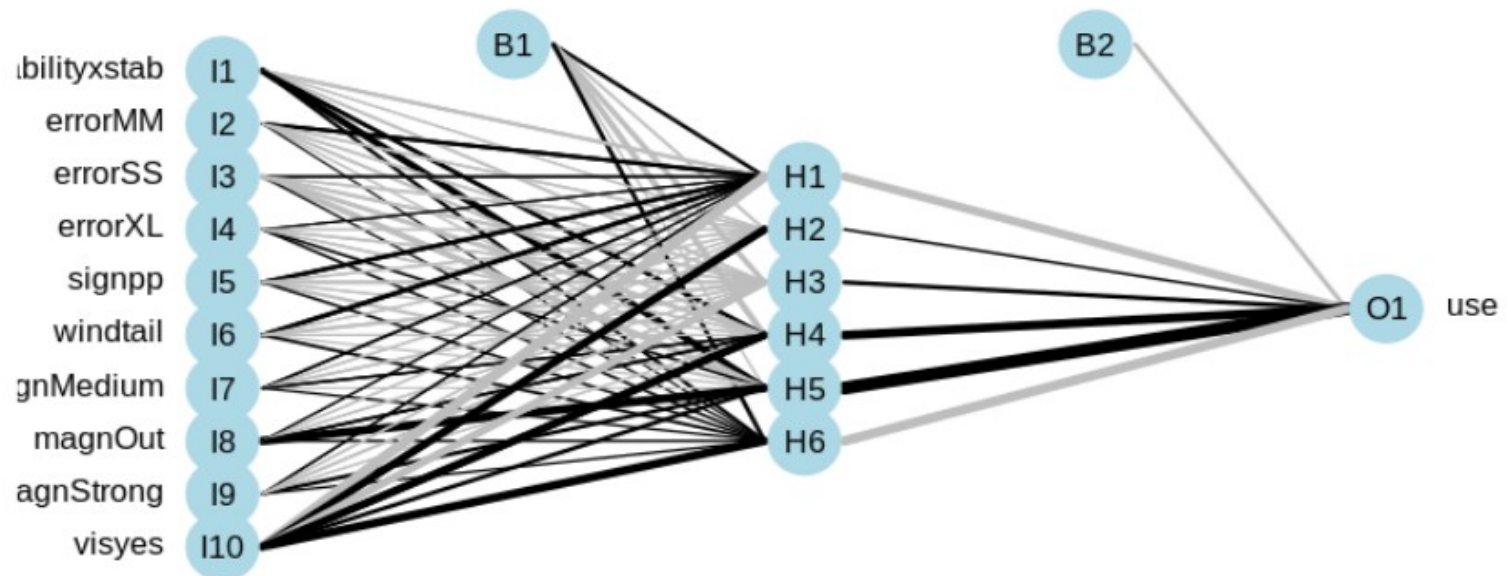
Parte D

14. Setee la semilla=8 y cree una Red Neuronal (con librería nnet) para modelar el problema planteado, con size=6 y maxit=5000. Indique el código R utilizado.

```
> library(nnet)
> set.seed(8)
> red=nnet(use~., train, size=6, maxit=5000)
# weights: 73
initial value 123.906943
iter 10 value 3.359699
iter 20 value 0.052594
iter 30 value 0.001692
iter 40 value 0.000349
final value 0.000080
converged
```

15. Grafique la Red Neuronal resultante.

```
> library(NeuralNetTools)
> plotnet(red)
```



16. Calcule la matriz de confusión utilizando la instrucción confusionMatrix de la librería caret.
Muestre una imagen de los resultados obtenidos.

```
> pred=predict(red, test, type="class")
> library(caret)
> confusionMatrix(factor(pred), test$use)
```

Confusion Matrix and Statistics

	Reference	
Prediction	auto	noauto
auto	43	2
noauto	0	31

Accuracy : 0.9737
95% CI : (0.9082, 0.9968)
No Information Rate : 0.5658
P-Value [Acc > NIR] : 2.766e-16

Kappa : 0.9461

Mcnemar's Test P-Value : 0.4795

Sensitivity : 1.0000
Specificity : 0.9394
Pos Pred Value : 0.9556
Neg Pred Value : 1.0000
Prevalence : 0.5658
Detection Rate : 0.5658
Detection Prevalence : 0.5921
Balanced Accuracy : 0.9697

'Positive' Class : auto

17. ¿Cuántos registros quedaron bien y mal clasificados? ¿Cómo fue la performance del modelo? (accuracy, sensibilidad y especificidad)

Clasificados			
Bien	43 + 31 =	74	
Mal	0 + 2 =	2	

$$\text{Accuracy} = (43+31) / (43+31+0+2) = 0.9737$$

$$\text{Sensibilidad} = 43 / (43 + 0) = 1$$

$$\text{Especificidad} = 31 / (31 + 2) = 0.9394$$

Conclusion

Comparando las matrices de confusion del Arbol y la Red Neuronal, veo que la Red Neuronal tiene mayor Exactitud, Sensibilidad y Especificidad.

Con lo cual, en este caso, utilizaria la Red Neuronal en vez del Arbol de Decision.

```
#-----  
library(MASS)  
base=shuttle  
str(base)  
fix(base)  
head(base)  
summary(base)  
barplot(table(shuttle$use), col=c("green", "red"))  
#-----  
library(caret)  
set.seed(8)  
particion=createDataPartition(y=shuttle$use, p=0.7, list=FALSE)  
train=shuttle[particion, ]  
test=shuttle[-particion, ]  
head(train)  
str(train)  
summary(train)  
head(test)  
str(test)  
summary(test)  
#-----  
library(rpart)  
arbol=rpart(use~., shuttle, method="class")  
summary(arbol)  
library(rpart.plot)  
rpart.plot(arbol, extra=101, type=4, cex=0.8)  
pred=predict(arbol, test, type="class")  
table(pred, test$use)  
confusionMatrix(pred, test$use)  
#-----  
library(nnet)  
set.seed(8)  
red=nnet(use~., train, size=6, maxit=5000)  
library(NeuralNetTools)  
plotnet(red)  
pred=predict(red, test, type="class")  
library(caret)  
confusionMatrix(factor(pred), test$use)  
#-----
```