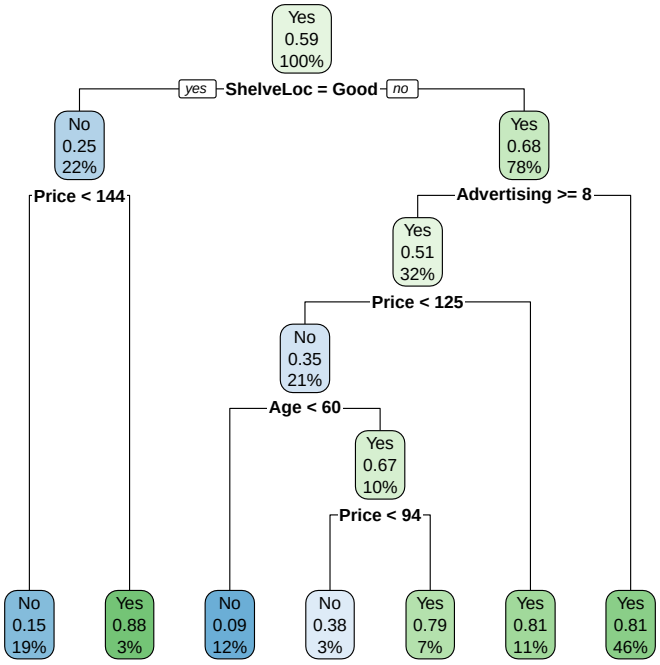
November 1, 2023

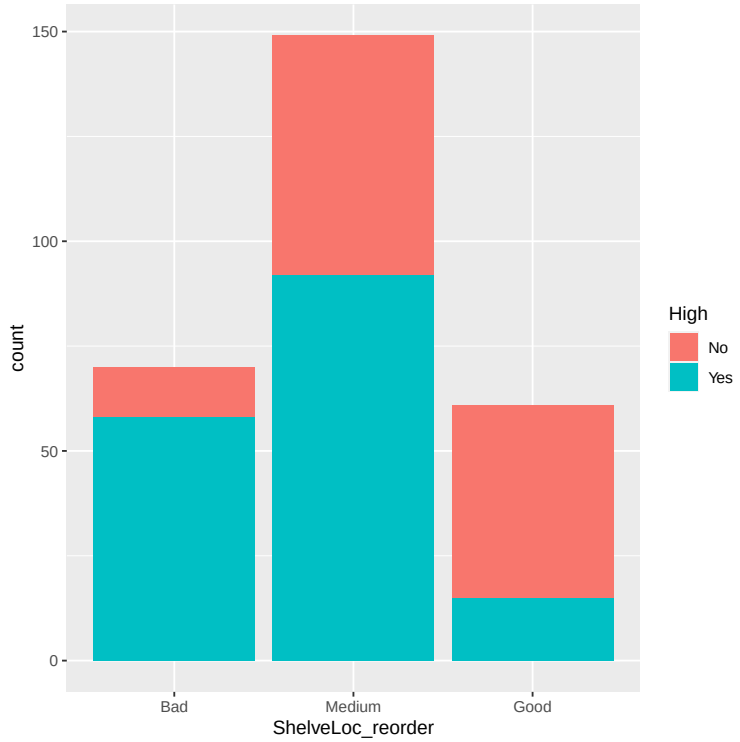
The results below are generated from an R script.

```r
# Liberías necesarias para resolver el ejercicio
library(ISLR2)
library(dplyr)
library(caTools) # Particiones de los datos
library(rpart) # Para árboles de decisión
library(rpart.plot)
library(ggplot2)
library(caret) # Para la matriz de confusión


# Datos
attach(Carseats)

## The following objects are masked from Carseats (pos = 3):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US
## The following objects are masked from Carseats (pos = 4):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US
## The following objects are masked from Carseats (pos = 8):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US

# Creamos una nueva variable respuesta binaria


# Creamos el data frame

df = Carseats %>%
      mutate(High = factor(ifelse(Sales>=8,"No","Yes"))) %>%
      select(-Sales)

# Partición de los datos

# Mediante una semilla conseguimos que el ejercicio sea reproducible
set.seed(121)

# Usamos el 70% de la base de datos como conjunto de entrenamiento y el resto como conjunto de test
sample = sample.split(df$High, SplitRatio=0.7)
train  = subset(df, sample==TRUE)
test   = subset(df, sample==FALSE)
```

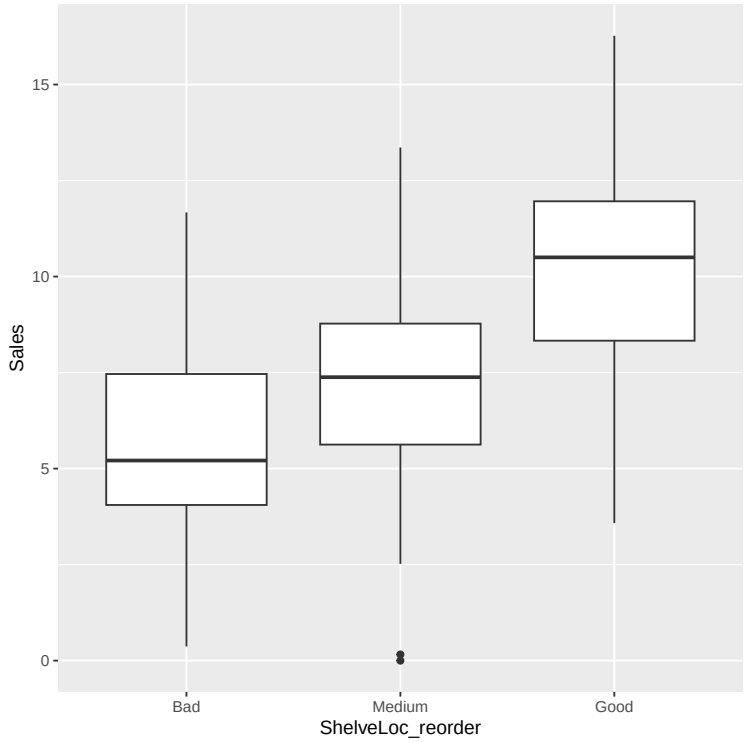
```
# Entrenamos un modelo sobre la muestra de entrenamiento empleando todas las variables

fit.dt = rpart(High~., data = train, method = 'class')
rpart.plot(fit.dt, extra = 106)
```




```
# La variable más importante es:
fit.dt$variable.importance

##   ShelveLoc       Price Advertising         Age   CompPrice          US   Education
##   19.486155   18.342365   10.575532    9.955631    6.354683    4.737580    2.558957
##  Population      Income
##    2.547655    1.973958

# Relación entre la variable respuesta y la variable más importante
# reordenamos la variable
train %>%
  mutate(ShelveLoc_reorder=factor(ShelveLoc,levels=c("Bad","Medium","Good")))%>%
  ggplot(aes(x = ShelveLoc_reorder, fill = High)) +
  geom_bar()
```




```
# Podemos visualizar su relación con la variable respuesta original como sigue
# reordenamos la variable
df %>%
  mutate(ShelveLoc_reorder=factor(ShelveLoc,levels=c("Bad","Medium","Good")))%>%
  ggplot(aes(ShelveLoc_reorder, Sales)) +
  geom_boxplot()
```




```
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(fit.dt, train, type = 'class')
cf = confusionMatrix(prediction, as.factor(train$High),positive="Yes")
print(cf)
```

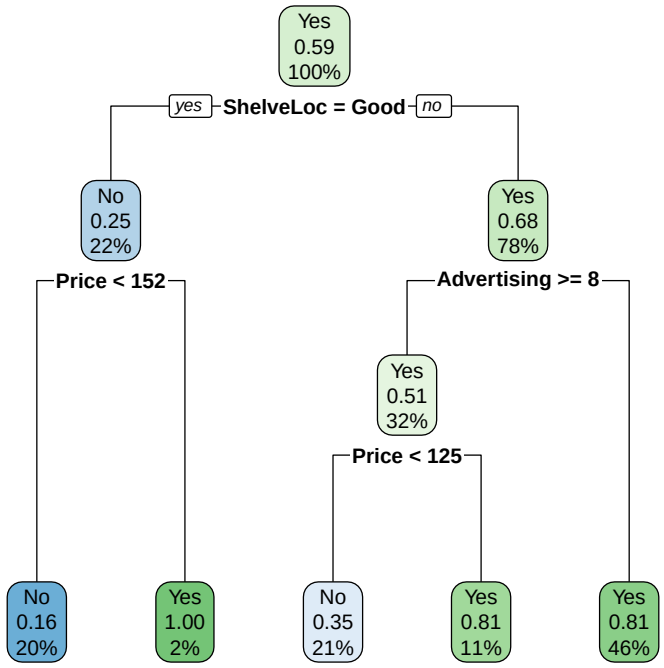
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   80  14
##        Yes  35 151
##
##                Accuracy : 0.825
##                  95% CI : (0.7753, 0.8676)
##     No Information Rate : 0.5893
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6282
##
##  Mcnemar's Test P-Value : 0.004275
##
##             Sensitivity : 0.9152
##             Specificity : 0.6957
##          Pos Pred Value : 0.8118
##          Neg Pred Value : 0.8511
##              Prevalence : 0.5893
##          Detection Rate : 0.5393
##    Detection Prevalence : 0.6643
##       Balanced Accuracy : 0.8054
```

```
##
##        'Positive' Class : Yes
##
```

```
# sobre la partición de validación
prediction = predict(fit.dt, test, type = 'class')
cf = confusionMatrix(prediction, as.factor(test$High),positive="Yes")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  27  10
##        Yes 22  61
##
##                Accuracy : 0.7333
##                  95% CI : (0.6449, 0.8099)
##     No Information Rate : 0.5917
##     P-Value [Acc > NIR] : 0.0008589
##
##                   Kappa : 0.4264
##
##  Mcnemar's Test P-Value : 0.0518299
##
##             Sensitivity : 0.8592
##             Specificity : 0.5510
##          Pos Pred Value : 0.7349
##          Neg Pred Value : 0.7297
##              Prevalence : 0.5917
##          Detection Rate : 0.5083
##    Detection Prevalence : 0.6917
##       Balanced Accuracy : 0.7051
##
##        'Positive' Class : Yes
##
```

```
# Ajustamos un modelo con menos profundidad para evitar el sobreajuste.
control = rpart.control(minsplit = 4,
                        minbucket = round(5 / 3),
                        maxdepth = 3,
                        cp = 0)
tune.fit = rpart(High~., data = train, method = 'class', control = control)
rpart.plot(tune.fit, extra = 106)
```




```
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(tune.fit, train, type = 'class')
cf = confusionMatrix(prediction, as.factor(train$High),positive="Yes")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   85  30
##        Yes  30 135
##
##                Accuracy : 0.7857
##                  95% CI : (0.733, 0.8323)
##     No Information Rate : 0.5893
##     P-Value [Acc > NIR] : 2.758e-12
##
##                   Kappa : 0.5573
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8182
##             Specificity : 0.7391
##          Pos Pred Value : 0.8182
##          Neg Pred Value : 0.7391
##              Prevalence : 0.5893
##          Detection Rate : 0.4821
##    Detection Prevalence : 0.5893
##       Balanced Accuracy : 0.7787
```

```
##
##        'Positive' Class : Yes
##
```

```
# sobre la partición de validación
prediction = predict(tune.fit, test, type = 'class')
cf = confusionMatrix(prediction, as.factor(test$High),positive="Yes")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  32  15
##        Yes 17  56
##
##                Accuracy : 0.7333
##                  95% CI : (0.6449, 0.8099)
##     No Information Rate : 0.5917
##     P-Value [Acc > NIR] : 0.0008589
##
##                   Kappa : 0.4446
##
##  Mcnemar's Test P-Value : 0.8596838
##
##             Sensitivity : 0.7887
##             Specificity : 0.6531
##          Pos Pred Value : 0.7671
##          Neg Pred Value : 0.6809
##              Prevalence : 0.5917
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.6083
##       Balanced Accuracy : 0.7209
##
##        'Positive' Class : Yes
##
```

The R session information (including the OS info, R version and all packages used):

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
```

```
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] caret_6.0-94     lattice_0.21-9   ggplot2_3.4.3    rpart.plot_3.1.1 rpart_4.1.19
## [6] caTools_1.18.2   dplyr_1.1.3      ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4         xfun_0.40            recipes_1.0.8        tzdb_0.4.0
##  [5] vctrs_0.6.3          tools_4.3.1          bitops_1.0-7         generics_0.1.3
##  [9] stats4_4.3.1         parallel_4.3.1       proxy_0.4-27         tibble_3.2.1
## [13] fansi_1.0.5          highr_0.10           ModelMetrics_1.2.2.2 pkgconfig_2.0.3
## [17] Matrix_1.6-1.1       data.table_1.14.8    lifecycle_1.0.3      stringr_1.5.0
## [21] compiler_4.3.1       farver_2.1.1         tinytex_0.47         munsell_0.5.0
## [25] codetools_0.2-19     htmltools_0.5.6.1    class_7.3-22         yaml_2.3.7
## [29] prodlim_2023.08.28   pillar_1.9.0         MASS_7.3-60          gower_1.0.1
## [33] iterators_1.0.14     foreach_1.5.2        nlme_3.1-163         parallelly_1.36.0
## [37] lava_1.7.2.1         tidyselect_1.2.0     digest_0.6.33        stringi_1.7.12
## [41] future_1.33.0        reshape2_1.4.4       purrr_1.0.2          listenv_0.9.0
## [45] labeling_0.4.3       splines_4.3.1        fastmap_1.1.1        grid_4.3.1
## [49] colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3       survival_3.5-7
## [53] utf8_1.2.3           e1071_1.7-13         future.apply_1.11.0  readr_2.1.4
## [57] withr_2.5.1          scales_1.2.1         lubridate_1.9.3      timechange_0.2.0
## [61] rmarkdown_2.25       globals_0.16.2       nnet_7.3-19          timeDate_4022.108
## [65] hms_1.1.3            evaluate_0.22        knitr_1.44           hardhat_1.3.0
## [69] rlang_1.1.1          Rcpp_1.0.11          glue_1.6.2           pROC_1.18.4
## [73] ipred_0.9-14         rstudioapi_0.15.0    R6_2.5.1             plyr_1.8.9
```

```
Sys.time()
```

```
## [1] "2023-11-01 19:20:14 CET"
```