Titanic Data Set

Step 1 : Load file in R and view it

setwd("C:/Users/Trupti/Desktop")

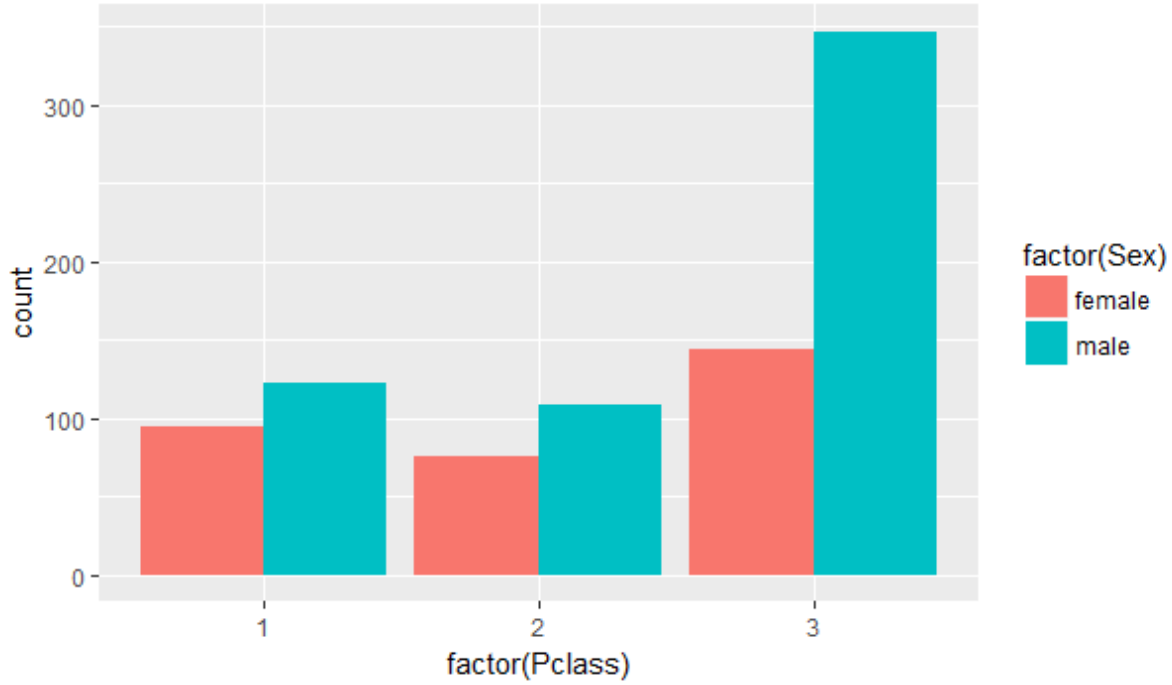
titanictrain<-read.csv("titanictrain.csv", header=TRUE);

titanictrain

Understanding the Data :

We plot a graph to study the relation between the independent variables

ggplot(titanictrain, aes(x=factor(Pclass),fill=factor(Sex)))+geom_bar(position = "dodge")



Step 2 :

We build and plot the decision tree in R

titanictrain_decisiontree<-rpart(Survived~Pclass+Sex+Age+SibSp+Parch+Fare+Embarked, data= titanictrain,method="class") #build decision tree

library(rattle)

library(RColorBrewer)

library(rpart.plot)

#Plot visually appealing decision tree

fancyRpartPlot(titanictrain_decisiontree)

summary(titanictrain_decisiontree)

```
Call:
rpart(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch +
    Fare + Embarked, data = titanictrain, method = "class")
  n= 891

          CP nsplit rel error    xerror       xstd
1 0.44444444      0 1.0000000 1.0000000 0.04244576
2 0.03070175      1 0.5555556 0.5555556 0.03574957
3 0.02339181      3 0.4941520 0.5175439 0.03482337
4 0.02046784      4 0.4707602 0.5000000 0.03437157
5 0.01023392      5 0.4502924 0.5116959 0.03467453
6 0.01000000      8 0.4181287 0.5058480 0.03452394

Variable importance
     Sex     Fare   Pclass    SibSp    Parch      Age Embarked
      47       18       13        7        6        5        4

Node number 1: 891 observations,    complexity param=0.4444444
  predicted class=0  expected loss=0.3838384  P(node) =1
    class counts:   549   342
   probabilities: 0.616 0.384
  left son=2 (577 obs) right son=3 (314 obs)
  Primary splits:
      Sex      splits as  RL,           improve=124.426300, (0 missing)
      Pclass   < 2.5      to the right, improve= 43.781830, (0 missing)
      Fare     < 10.48125 to the left,  improve= 37.941940, (0 missing)
      Embarked splits as  RRLL,         improve= 12.865410, (0 missing)
      Parch    < 0.5      to the left,  improve=  9.157774, (0 missing)
  Surrogate splits:
      Fare     < 77.6229  to the left,  agree=0.679, adj=0.089, (0 split)
      Parch    < 0.5      to the left,  agree=0.678, adj=0.086, (0 split)
      Embarked splits as  RLLL,         agree=0.650, adj=0.006, (0 split)

Node number 2: 577 observations,    complexity param=0.02339181
  predicted class=0  expected loss=0.1889081  P(node) =0.647587
    class counts:   468   109
   probabilities: 0.811 0.189
  left son=4 (553 obs) right son=5 (24 obs)
  Primary splits:
      Age      < 6.5      to the right, improve=10.788930, (124 missing)
      Fare     < 26.26875 to the left,  improve=10.216720, (0 missing)
      Pclass   < 1.5      to the right, improve=10.019140, (0 missing)
      Parch    < 0.5      to the left,  improve= 3.350327, (0 missing)
      Embarked splits as  -RLL,         improve= 3.079304, (0 missing)

Node number 3: 314 observations,    complexity param=0.03070175
  predicted class=1  expected loss=0.2579618  P(node) =0.352413
    class counts:    81   233
   probabilities: 0.258 0.742
  left son=6 (144 obs) right son=7 (170 obs)
  Primary splits:
      Pclass   < 2.5      to the right, improve=31.163130, (0 missing)
```

```
      Fare     < 48.2     to the left,  improve=10.114210, (0 missing)
      SibSp    < 2.5      to the right, improve= 9.372551, (0 missing)
      Parch    < 3.5      to the right, improve= 5.140857, (0 missing)
      Embarked splits as  RRLL,         improve= 3.750944, (0 missing)
  Surrogate splits:
      Fare     < 25.69795 to the left,  agree=0.799, adj=0.563, (0 split)
      Embarked splits as  RRLR,         agree=0.637, adj=0.208, (0 split)
      SibSp    < 1.5      to the right, agree=0.592, adj=0.111, (0 split)
      Parch    < 1.5      to the right, agree=0.567, adj=0.056, (0 split)
      Age      < 18.5     to the left,  agree=0.564, adj=0.049, (0 split)

Node number 4: 553 observations
  predicted class=0  expected loss=0.1681736  P(node) =0.620651
    class counts:   460    93
   probabilities: 0.832 0.168

Node number 5: 24 observations,    complexity param=0.02046784
  predicted class=1  expected loss=0.3333333  P(node) =0.02693603
    class counts:     8    16
   probabilities: 0.333 0.667
  left son=10 (9 obs) right son=11 (15 obs)
  Primary splits:
      SibSp  < 2.5      to the right, improve=8.8888890, (0 missing)
      Pclass < 2.5      to the right, improve=3.8095240, (0 missing)
      Fare   < 20.825   to the right, improve=2.6666670, (0 missing)
      Age    < 1.5      to the right, improve=0.6095238, (0 missing)
  Surrogate splits:
      Pclass   < 2.5      to the right, agree=0.792, adj=0.444, (0 split)
      Fare     < 26.95    to the right, agree=0.750, adj=0.333, (0 split)
      Embarked splits as  -RLR,         agree=0.708, adj=0.222, (0 split)

Node number 6: 144 observations,    complexity param=0.03070175
  predicted class=0  expected loss=0.5  P(node) =0.1616162
    class counts:    72    72
   probabilities: 0.500 0.500
  left son=12 (27 obs) right son=13 (117 obs)
  Primary splits:
      Fare     < 23.35    to the right, improve=10.051280, (0 missing)
      Embarked splits as  -RRL,         improve= 7.071429, (0 missing)
      SibSp    < 2.5      to the right, improve= 4.571429, (0 missing)
      Age      < 38.5     to the right, improve= 3.875163, (42 missing)
      Parch    < 1.5      to the right, improve= 3.773262, (0 missing)
  Surrogate splits:
      SibSp < 2.5      to the right, agree=0.882, adj=0.37, (0 split)
      Parch < 1.5      to the right, agree=0.882, adj=0.37, (0 split)

Node number 7: 170 observations
  predicted class=1  expected loss=0.05294118  P(node) =0.1907969
    class counts:     9   161
   probabilities: 0.053 0.947

Node number 10: 9 observations
  predicted class=0  expected loss=0.1111111  P(node) =0.01010101
    class counts:     8     1
   probabilities: 0.889 0.111

Node number 11: 15 observations
  predicted class=1  expected loss=0  P(node) =0.01683502
    class counts:     0    15
   probabilities: 0.000 1.000

Node number 12: 27 observations
```

```
  predicted class=0  expected loss=0.1111111  P(node) =0.03030303
    class counts:    24     3
   probabilities: 0.889 0.111

Node number 13: 117 observations,    complexity param=0.01023392
  predicted class=1  expected loss=0.4102564  P(node) =0.1313131
    class counts:    48    69
   probabilities: 0.410 0.590
  left son=26 (63 obs) right son=27 (54 obs)
  Primary splits:
      Embarked splits as  -RRL,          improve=2.6048030, (0 missing)
      Age      < 16.5     to the right, improve=2.4685870, (34 missing)
      Fare     < 7.8875   to the right, improve=2.0325270, (0 missing)
      SibSp    < 0.5      to the right, improve=0.3076923, (0 missing)
      Parch    < 1.5      to the left,  improve=0.1582418, (0 missing)
  Surrogate splits:
      Fare < 7.7625   to the right, agree=0.667, adj=0.278, (0 split)

Node number 26: 63 observations,    complexity param=0.01023392
  predicted class=0  expected loss=0.4920635  P(node) =0.07070707
    class counts:    32    31
   probabilities: 0.508 0.492
  left son=52 (37 obs) right son=53 (26 obs)
  Primary splits:
      Fare  < 10.825   to the left,  improve=1.34653300, (0 missing)
      Age   < 27.5     to the right, improve=0.97840760, (5 missing)
      Parch < 0.5      to the left,  improve=0.71428570, (0 missing)
      SibSp < 0.5      to the right, improve=0.08821734, (0 missing)
  Surrogate splits:
      SibSp < 0.5      to the left,  agree=0.746, adj=0.385, (0 split)
      Parch < 0.5      to the left,  agree=0.746, adj=0.385, (0 split)
      Age   < 11       to the right, agree=0.635, adj=0.115, (0 split)

Node number 27: 54 observations
  predicted class=1  expected loss=0.2962963  P(node) =0.06060606
    class counts:    16    38
   probabilities: 0.296 0.704

Node number 52: 37 observations
  predicted class=0  expected loss=0.4054054  P(node) =0.04152637
    class counts:    22    15
   probabilities: 0.595 0.405

Node number 53: 26 observations,    complexity param=0.01023392
  predicted class=1  expected loss=0.3846154  P(node) =0.0291807
    class counts:    10    16
   probabilities: 0.385 0.615
  left son=106 (10 obs) right son=107 (16 obs)
  Primary splits:
      Fare  < 17.6     to the right, improve=3.23269200, (0 missing)
      SibSp < 0.5      to the right, improve=0.72599300, (0 missing)
      Age   < 13       to the right, improve=0.33893560, (2 missing)
      Parch < 0.5      to the left,  improve=0.04578755, (0 missing)
  Surrogate splits:
      SibSp < 1.5      to the right, agree=0.692, adj=0.2, (0 split)
      Parch < 1.5      to the right, agree=0.654, adj=0.1, (0 split)

Node number 106: 10 observations
  predicted class=0  expected loss=0.3  P(node) =0.01122334
    class counts:     7     3
   probabilities: 0.700 0.300
```

```
Node number 107: 16 observations
  predicted class=1  expected loss=0.1875  P(node) =0.01795735
    class counts:     3    13
   probabilities: 0.188 0.812

function (x = stop("no 'x' arg"), type = 0, extra = 0, under = FALSE,
    clip.right.labs = TRUE, nn = FALSE, ni = FALSE, yesno = TRUE,
    fallen.leaves = FALSE, branch = if (fallen.leaves) 1 else 0.2,
    uniform = TRUE, left = TRUE, xflip = FALSE, yflip = FALSE,
    Margin = 0, space = 1, gap = NULL, digits = 2, varlen = -8,
    faclen = 3, cex = NULL, tweak = 1, compress = TRUE, ycompress = unifor
m,
    trace = FALSE, snip = FALSE, snip.fun = NULL, box.col = 0,
    box.palette = 0, pal.thresh = NULL, pal.node.fun = FALSE,
    border.col = col, round = NULL, leaf.round = NULL, shadow.col = 0,
    prefix = "", suffix = "", xsep = NULL, under.font = font,
    under.col = 1, under.cex = 0.8, split.cex = 1, split.font = 2,
    split.family = family, split.col = 1, split.box.col = 0,
    split.border.col = 0, split.lty = 1, split.lwd = NULL, split.round = 0,

    split.shadow.col = 0, split.prefix = "", right.split.prefix = NULL,
    split.suffix = "", right.split.suffix = NULL, facsep = ",",
    eq = " = ", lt = " < ", ge = " >= ", branch.col = if (identical(branch.
type,
        0)) 1 else "gray", branch.lty = 1, branch.lwd = NULL,
    branch.type = 0, branch.tweak = 1, min.branch.width = 0.002,
    branch.fill = branch.col, nn.cex = NULL, nn.font = 3, nn.family = "",
    nn.col = 1, nn.box.col = 0, nn.border.col = nn.col, nn.lty = 1,
    nn.lwd = NULL, nn.round = 0.3, yes.text = "yes", no.text = "no",
    node.fun = NULL, split.fun = NULL, FUN = "text", nspace = branch,
    minbranch = 0.3, do.par = TRUE, add.labs = TRUE, clip.left.labs = FALS
E,
    fam.main = "", yshift = 0, yspace = space, shadow.offset = 0.4,
    split.adj = NULL, split.yshift = 0, split.space = space,
    split.yspace = yspace, split.shadow.offset = shadow.offset,
    nn.adj = 0.5, nn.yshift = 0, nn.space = 0.8, nn.yspace = 0.5,
    ygap = gap/2, under.ygap = 0.5, yesno.yshift = 0, xcompact = TRUE,
    ycompact = uniform, xcompact.ratio = 0.8, min.inter.height = 4,
    max.auto.cex = 1, min.auto.cex = 0.15, ycompress.cex = 0.7,
    accept.cex = 1.1, shift.amounts = c(1.5, 2), Fallen.yspace = 0.1,
    boxes.include.gap = FALSE, legend.x = NULL, legend.y = NULL,
    legend.cex = 1, ...)
{
    check.dots <- function(dots) {
        legal.dots.args <- c("adj", "cex.main", "cex.sub", "col",
            "col.main", "col.sub", "family", "font", "lty", "lwd",
            "main", "mar", "sub", "xlim", "xpd", "ylim")
        if (length(dots) > 0) {
            names <- names(dots)
            pmatch <- pmatch(names, legal.dots.args, duplicates.ok = TRUE)
            if (any(is.na(pmatch))) {
                ibad <- (1:length(dots))[is.na(pmatch)]
                stop0("prp: illegal argument \"", names[ibad][1],
                  "\"")
            }
            duplicated <- duplicated(pmatch)
            if (any(duplicated))
                stop0("prp: duplicated argument \"", names[duplicated][1],

                  "\"")
        }
    }
```

```
    merge1 <- function(vec, split.vec) {
        split.vec <- recycle(split.vec, nodes)
        split.vec[is.leaf] <- recycle(vec, nodes)[is.leaf]
        split.vec
    }
    draw.labs <- function(draw.shadows1, draw.split.shadows1) {
        draw.labs1 <- function(labs, boxes, yspace, cex, font,
            family, col, draw.shadows1, make.space.for.shadows,
            shadow.col, round) {
            draw.under.text <- function() {
                height1 <- my.strheight("M", cex, font, family)
                cex <- under.cex * cex
                under.height <- my.strheight(sep.labs$under.box,
                  cex, under.font, family)
                x <- xy$x
                y <- boxes$y1 - under.ygap * height1 - 0.5 *
                  under.height
                width <- 0.5 * my.strwidth(sep.labs$under.box,
                  cex, under.font, family)
                height <- 0.5 * my.strheight(sep.labs$under.box,
                  cex, under.font, family)
                if (make.space.for.shadows)
                  height <- 1.4 * height
                if (draw.shadows1)
                  draw.shadow(x - 1.2 * width, y - height, x +
                    1.2 * width, y + height, xlim, ylim, 0, shadow.col,
                    shadow.offset)
                else {
                  rect(x - 1.2 * width, y - height, x + 1.2 *
                    width, y + height, col = bg, border = 0)
                  FUN(x, y, sep.labs$under.box, cex = cex, font = under.fo
nt,
                    family = family, col = under.col)
                }
            }
            FUN <- check.func.args(FUN, "FUN argument to the prp",
                graphics::text)
            sep.labs <- separate.labs(labs)
            xy <- get.box.centers(boxes)
            if (!all(nchar(sep.labs$under.box) == 0))
                draw.under.text()
            if (!draw.shadows1)
                FUN(xy$x, xy$y, sep.labs$in.box, cex = cex, font = font,
                  family = family, col = col)
        }
        if (boxes.include.gap) {
            printf("boxes.include.gap is TRUE\n")
            split.space <- split.space + gap/2
            split.yspace <- split.yspace + ygap/2
            space <- space + gap/2
            yspace <- yspace + ygap/2
            gap <- ygap <- 0
        }
        small.underspace <- type == TYPE.all.under && is.box.invisible(spl
it.box.col,
            split.border.col, bg)
        split.boxes <- draw.boxes(if (is.fancy(type))
            "left"
        else "default", draw.split.shadows1, split.labs, node.xy,
            xlim, ylim, nodes, branch, Margin, xflip, yflip,
            main, sub, col.main, cex.main, col.sub, cex.sub,
            split.cex * cex, split.font, split.family, split.adj,
```

```
            split.yshift, split.box.col, split.border.col, split.lty,
            split.lwd, split.space, split.yspace, split.round *
                split.strheight, under.cex, under.font, under.ygap,
            ygap, split.shadow.col, split.shadow.offset, bg,
            small.underspace, split.strwidth, split.strheight)
        if (!draw.split.shadows1)
            draw.labs1(split.labs, split.boxes, split.yspace,
                split.cex * cex, split.font, split.family, split.col,
                draw.split.shadows1, draw.split.shadows, split.shadow.col,

                split.round)
        if (is.fancy(type)) {
            right.split.boxes <- draw.boxes("right", draw.split.shadows1,
                right.split.labs, node.xy, xlim, ylim, nodes,
                branch, Margin, xflip, yflip, main, sub, col.main,
                cex.main, col.sub, cex.sub, split.cex * cex,
                split.font, split.family, split.adj, split.yshift,
                split.box.col, split.border.col, split.lty, split.lwd,
                split.space, split.yspace, split.round * split.strheight,
                under.cex, under.font, under.ygap, ygap, split.shadow.col,

                split.shadow.offset, bg)
            if (!draw.split.shadows1)
                draw.labs1(right.split.labs, right.split.boxes,
                  split.yspace, split.cex * cex, split.font,
                  split.family, split.col, draw.split.shadows1,
                  draw.split.shadows, split.shadow.col, split.round)
        }
        node.boxes <- draw.boxes("default", draw.shadows1, node.labs,
            node.xy, xlim, ylim, nodes, branch, Margin, xflip,
            yflip, main, sub, col.main, cex.main, col.sub, cex.sub,
            cex, font, family, adj, yshift, box.col, border.col,
            lty, lwd, space, yspace, round * strheight, under.cex,
            under.font, under.ygap, ygap, shadow.col, shadow.offset,
            bg)
        draw.labs1(node.labs, node.boxes, yspace, cex, font,
            family, col, draw.shadows1, draw.shadows, shadow.col,
            round)
        if (yesno && !is.fancy(type) && !snip)
            draw.yes.no(yesno, yes.text, no.text, type, draw.shadows1,
                xflip, left, branch, xlim, ylim, node.xy, lwd,
                yesno.yshift, split.boxes, split.cex * cex, split.box.col,

                split.border.col, split.shadow.col, split.shadow.offset,
                nn.cex, nn.font, nn.family, nn.col, nn.box.col,
                nn.border.col, nn.lty, nn.round, bg)
        if (nn || ni)
            draw.node.numbers(nn, ni, draw.shadows1, type, branch,
                Margin, xflip, yflip, cex, main, sub, col.main,
                cex.main, col.sub, cex.sub, xlim, ylim, node.xy,
                is.leaf, nodes, node.labs, font, family, box.col,
                border.col, shadow.col, under.cex, under.font,
                under.ygap, ygap, split.labs, split.cex * cex,
                split.font, split.family, split.box.col, split.border.col,

                split.shadow.col, nn.cex, nn.font, nn.family,
                nn.col, nn.box.col, nn.border.col, nn.lty, nn.lwd,
                nn.round, split.adj, split.space, split.yspace,
                split.yshift, yshift, adj, space, yspace, shadow.offset,
                nn.adj, nn.yshift, nn.space, nn.yspace, bg)
        list(node.boxes = node.boxes, split.boxes = split.boxes)
    }
```

```
if (!inherits(x, "rpart"))
    stop0("the object passed to prp is not an rpart object")
obj <- x
dots <- match.call(expand.dots = FALSE)$...
check.dots(dots)
adj <- eval.parent(dots$adj)
if (is.null(adj))
    adj <- par("adj")
cex.main <- eval.parent(dots$cex.m)
cex.sub <- eval.parent(dots$cex.s)
col <- eval.parent(dots$col)
if (is.null(col))
    col <- par("col")
col.main <- eval.parent(dots$col.m)
if (is.null(col.main))
    col.main <- par("col.main")
col.sub <- eval.parent(dots$col.s)
if (is.null(col.sub))
    col.sub <- par("col.sub")
family <- eval.parent(dots$fam)
if (is.null(family))
    family <- par("family")
font <- eval.parent(dots$fo)
if (is.null(font))
    font <- par("font")
lty <- eval.parent(dots$lt)
if (is.null(lty))
    lty <- par("lty")
lwd <- eval.parent(dots$lw)
if (is.null(lwd))
    lwd <- par("lwd")
main <- eval.parent(dots$mai)
mar <- eval.parent(dots$mar)
sub <- eval.parent(dots$sub)
xlim <- eval.parent(dots$xl)
xpd <- eval.parent(dots$xp)
ylim <- eval.parent(dots$yl)
if (is.null(under.col))
    under.col <- col
if (is.null(border.col))
    border.col <- col
if (is.null(branch.lwd))
    branch.lwd <- lwd
if (is.null(split.lwd))
    split.lwd <- lwd
if (is.null(nn.lwd))
    nn.lwd <- lwd
if (is.null(split.adj))
    split.adj <- adj
class.stats <- NULL
if (obj$method == "class" || is.class.response(obj))
    class.stats <- get.class.stats(obj)
bg <- get.bg()
border.col <- set.zero.to.bg(border.col, bg)
shadow.col <- set.zero.to.bg(shadow.col, bg)
under.col <- set.zero.to.bg(under.col, bg)
split.col <- set.zero.to.bg(split.col, bg)
split.box.col <- set.zero.to.bg(split.box.col, bg)
split.shadow.col <- set.zero.to.bg(split.shadow.col, bg)
nn.col <- set.zero.to.bg(nn.col, bg)
nn.box.col <- set.zero.to.bg(nn.box.col, bg)
nn.border.col <- set.zero.to.bg(nn.border.col, bg)
```

```
    stopifnot(is.numeric(type) && length(type) == 1 && floor(type) ==
        type)
    if (type < TYPE.default || type > TYPE.fancy.all)
        stop0("type must be ", TYPE.default, "...", TYPE.fancy.all,
            ", you have type=", type)
    under <- check.boolean(under)
    clip.left.labs[1] <- check.boolean(clip.left.labs[1])
    clip.right.labs[1] <- check.boolean(clip.right.labs[1])
    nn <- check.boolean(nn)
    ni <- check.boolean(ni)
    stopifnot((is.numeric(yesno) || is.logical(yesno)) && length(yesno) ==

        1 && floor(yesno) == yesno)
    if (yesno < 0 || yesno > 2)
        stop0("yesno must be 0, 1, or 2.  You have yesno=", yesno)
    stopifnot(is.character(yes.text) && length(yes.text) == 1)
    stopifnot(is.character(no.text) && length(no.text) == 1)
    fallen.leaves <- check.boolean(fallen.leaves)
    uniform <- check.boolean(uniform)
    left <- check.boolean(left)
    xflip <- check.boolean(xflip)
    yflip <- check.boolean(yflip)
    do.par <- check.boolean(do.par)
    snip <- check.boolean(snip)
    if (snip) {
        branch.col = "black"
        branch.lty = 1
    }
    compress <- check.boolean(compress)
    ycompress <- check.boolean(ycompress)
    xcompact <- check.boolean(xcompact)
    ycompact <- check.boolean(ycompact)
    add.labs <- check.boolean(add.labs)
    boxes.include.gap <- check.boolean(boxes.include.gap)
    stopifnot(all(split.round >= 0))
    stopifnot(all(nn.round >= 0))
    stopifnot(tweak > 0 && tweak <= 10)
    stopifnot(max.auto.cex >= 1)
    stopifnot(min(shift.amounts) >= 0 && max(shift.amounts) <=
        10)
    stopifnot(xcompact.ratio > 0 && xcompact.ratio <= 2)
    stopifnot(min.auto.cex >= 0 && min.auto.cex <= 1)
    stopifnot(branch >= 0 && branch <= 1)
    if (!is.null(snip.fun))
        check.func.args(snip.fun, "snip.fun", function(tree) NULL)
    if (length(family) != 1 || length(split.family) != 1 || length(nn.fami
ly) !=
        1)
        stop0("prp: family argument must be length 1 (family cannot be vec
torized)")
    stopifnot(is.numeric(digits) && length(digits) == 1 && floor(digits) =
=
        digits && digits >= 0)
    if (digits == 0)
        digits <- getOption("digits")
    if (!is.na.or.zero(branch.type)) {
        branch <- if (branch > 0.5)
            1
        else 0
        ycompact <- FALSE
    }
    auto.cex <- FALSE
```

```
    if (is.null(cex)) {
        auto.cex <- TRUE
        cex <- 1
    }
    if (is.null(split.cex))
        split.cex <- 1
    if (fallen.leaves)
        compress <- FALSE
    if (!is.null(obj$frame$splits))
        stop0("Old-style rpart object?  (frame$splits is NULL)")
    frame <- obj$frame
    is.leaf <- is.leaf(frame)
    nodes <- as.numeric(row.names(frame))
    if (is.auto(extra, n = 1))
        extra <- get.default.extra(obj, class.stats)
    node.fun.name <- deparse(substitute(node.fun))
    node.labs <- internal.node.labs(obj, node.fun, node.fun.name,
        type, extra, under, xsep, digits, varlen, prefix, suffix,
        class.stats)
    temp <- handle.box.palette.args(obj, trace, box.col, box.palette,
        pal.thresh, pal.node.fun, node.fun.name, class.stats,
        node.labs)
    box.col <- temp$box.col
    box.palette <- temp$box.palette
    split.labs <- split.labs.wrapper(obj, split.fun, deparse(substitute(sp
lit.fun)),
        split.prefix, split.suffix, right.split.prefix, right.split.suffix,

        type, clip.left.labs, clip.right.labs, xflip, digits,
        varlen, faclen, facsep, eq, lt, ge)
    if (do.par) {
        init.plot(1, 1, Margin, xflip, yflip, main, sub, col.main,
            cex.main, col.sub, cex.sub)
        par <- par("mar", "xpd", "family")
        on.exit(par(par))
        if (is.null(mar)) {
            mar <- par$mar
            if (is.null(sub))
                mar[1] <- 1
            if (is.null(main))
                mar[3] <- 1
            mar[2] <- mar[4] <- 1
        }
        if (is.null(xpd))
            xpd <- NA
        par(mar = mar, xpd = xpd)
        par(new = TRUE)
    }
    if (is.fancy(type)) {
        right.split.labs <- split.labs[match(2 * nodes + 1, nodes)]
        split.labs <- split.labs[match(2 * nodes, nodes)]
        if (!left)
            stop0("left=FALSE is not yet supported with type=3 or 4")
    }
    else {
        if (left != xflip)
            split.labs <- split.labs[match(2 * nodes, nodes)]
        else split.labs <- split.labs[match(2 * nodes + 1, nodes)]
    }
    draw.shadows <- !is.invisible(shadow.col, bg)
    draw.split.shadows <- !is.invisible(split.shadow.col, bg)
    adj <- recycle(adj, nodes)
```

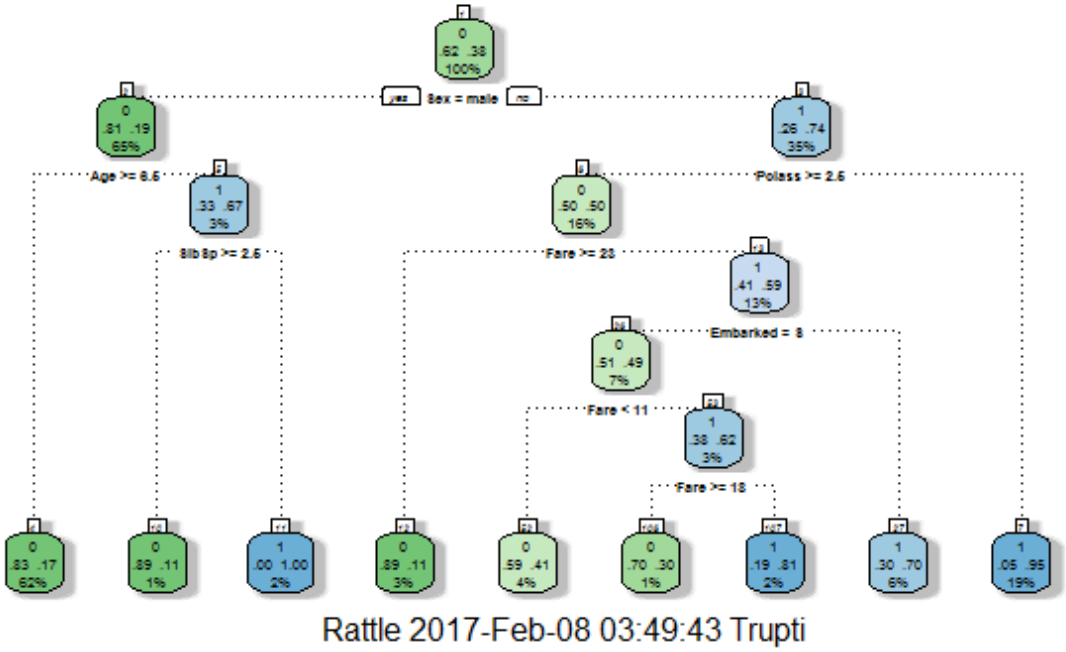
```
    space <- recycle(space, nodes)
    yspace <- recycle(yspace, nodes)
    shadow.offset <- recycle(shadow.offset, nodes)
    under.cex <- recycle(under.cex, nodes)
    under.ygap <- recycle(under.ygap, nodes)
    split.cex <- recycle(split.cex, nodes)
    split.adj <- recycle(adj, nodes)
    split.space <- recycle(split.space, nodes)
    split.yspace <- recycle(split.yspace, nodes)
    split.shadow.offset <- recycle(split.shadow.offset, nodes)
    nn.adj <- recycle(nn.adj, nodes)
    nn.space <- recycle(nn.space, nodes)
    nn.yspace <- recycle(nn.yspace, nodes)
    temp <- get.yshift(type, nodes, is.leaf, cex, node.labs,
        yshift, yspace, under.cex, split.labs, split.cex, split.yshift,
        split.yspace, ygap)
    yshift <- temp$yshift
    split.yshift <- temp$split.yshift
    if (yesno == 2 && !is.fancy(type))
        split.labs <- ifelse(split.labs == "NA", "NA", paste(yes.text,
            split.labs, no.text))
    layout <- get.layout(obj, type, nn, yesno, fallen.leaves,
        branch, uniform, Margin, cex, auto.cex, compress, ycompress,
        trace, main, sub, node.labs, font, family, box.col, border.col,
        under.font, under.cex, split.labs, right.split.labs,
        split.cex, split.font, split.family, split.box.col, split.border.c
ol,
        nspace, minbranch, adj, yshift, space, yspace, split.adj,
        split.yshift, split.space, split.yspace, gap, ygap, under.ygap,
        xcompact, ycompact, xcompact.ratio, min.inter.height,
        max.auto.cex, min.auto.cex, ycompress.cex, accept.cex,
        shift.amounts, Fallen.yspace, bg)
    if (yesno == 2 && !is.fancy(type))
        split.labs <- ifelse(split.labs == "NA", "NA", substr(split.labs,
            nchar(yes.text) + 1, nchar(split.labs) - nchar(no.text) -
                1))
    cex <- layout$cex
    gap <- layout$gap
    ygap <- layout$ygap
    space <- pmax(0.25, layout$node.space)
    yspace <- pmax(0.25, layout$node.yspace)
    if (is.null(xlim))
        xlim <- layout$xlim
    stopifnot(is.numeric(xlim) && length(xlim) == 2)
    if (is.null(ylim))
        ylim <- layout$ylim
    stopifnot(is.numeric(ylim) && length(ylim) == 2)
    split.yshift <- layout$split.yshift
    if (trace > 0) {
        tweak.msg <- if (tweak == 1)
            ""
        else sprintf(" (before applying tweak %g)", tweak)
        printf("cex %.3g%s   xlim c(%.3g, %.3g)   ylim c(%.3g, %.3g)\n",
            cex[1], tweak.msg, xlim[1], xlim[2], ylim[1], ylim[2])
    }
    if (!auto.cex && tweak != 1)
        warning0("cex and tweak both specified, applying both")
    cex <- tweak * cex
    all.cex <- merge1(cex, split.cex * cex)
    split.lwd <- recycle(cex * split.lwd, nodes)
    branch.lwd <- recycle(cex * branch.lwd, nodes)
    nn.lwd <- recycle(cex * nn.lwd, nodes)
```

```
    lwd <- recycle(cex * lwd, nodes)
    node.xy <- layout$node.xy
    init.plot(xlim, ylim, Margin, xflip, yflip, main, sub, col.main,
        cex.main, col.sub, cex.sub, fam.main = fam.main, cex = cex[1],
        trace = trace, hide.title = FALSE)
    split.strwidth <- my.strwidth("M", split.cex * cex, split.font,
        split.family)
    strheight <- my.strheight("M", cex, font, family)
    split.strheight <- my.strheight("M", split.cex * cex, split.font,
        split.family)
    node.boxes <- split.boxes <- NA
    if (add.labs) {
        if (is.null(round))
            round <- max(1, 2 * min(space, yspace))
        stopifnot(all(round >= 0))
        round <- recycle(round, nodes)
        if (is.null(leaf.round))
            leaf.round <- round
        stopifnot(all(leaf.round >= 0))
        leaf.round <- recycle(leaf.round, nodes)
        round[is.leaf] <- leaf.round[is.leaf]
        if (draw.shadows || draw.split.shadows)
            draw.labs(draw.shadows, draw.split.shadows)
    }
    branch.xy <- draw.branches(obj, branch.type, branch.col,
        branch.lty, branch.lwd, branch.fill, branch.tweak, node.labs,
        split.labs, node.xy, strheight, type, branch, xflip,
        yflip, Margin, space, yspace, cex, font, family, adj,
        box.col, border.col, under.cex, under.font, under.ygap,
        split.cex, split.font, split.family, split.adj, split.yshift,
        split.box.col, split.border.col, split.space, split.yspace,
        main, sub, col.main, cex.main, col.sub, cex.sub, xlim,
        ylim, yshift, ygap, bg, min.branch.width)
    if (add.labs) {
        temp <- draw.labs(FALSE, FALSE)
        node.boxes <- temp$node.boxes
        split.boxes <- temp$split.boxes
    }
    snipped.nodes <- NULL
    if (snip) {
        temp <- do.snip(obj, nodes, split.labs, node.xy, branch.xy,
            branch.lwd, xlim, ylim, digits, snip.fun, cex)
        obj <- temp$obj
        snipped.nodes <- temp$snipped.nodes
    }
    rv <- list(obj = obj, snipped.nodes = snipped.nodes, xlim = xlim,
        ylim = ylim, x = node.xy$x, y = node.xy$y, branch.x = branch.xy$x,

        branch.y = branch.xy$y, labs = node.labs, cex = cex,
        boxes = node.boxes, split.labs = "", split.cex = split.cex,
        split.box = split.boxes)
    possible.palette.legend(rv, class.stats, box.col, box.palette,
        legend.x, legend.y, legend.cex)
    invisible(rv)
}
```

prp(titanictrain_decisiontree)

prp



Step 3: Predict for Dependant variable : Survived

#prediction

prediction <- predict(titanictrain_decisiontree, titanictrain, type = "class")

plot(prediction)

Step 4: Save the response on a csv

#prediction in csv format

solution <- data.frame(PassengerId = titanictrain$PassengerId, Survived = prediction)

write.csv(solution, file = "solution.csv", row.names = FALSE)

Conclusion

It can be concluded that Sex(gender) is the root node and then depending on the value of this root node the predictive analysis shapes as shown in the above graph. The split analysis for female is higher(more) than that of a male signifying that the number of women surviving is higher in number overall.