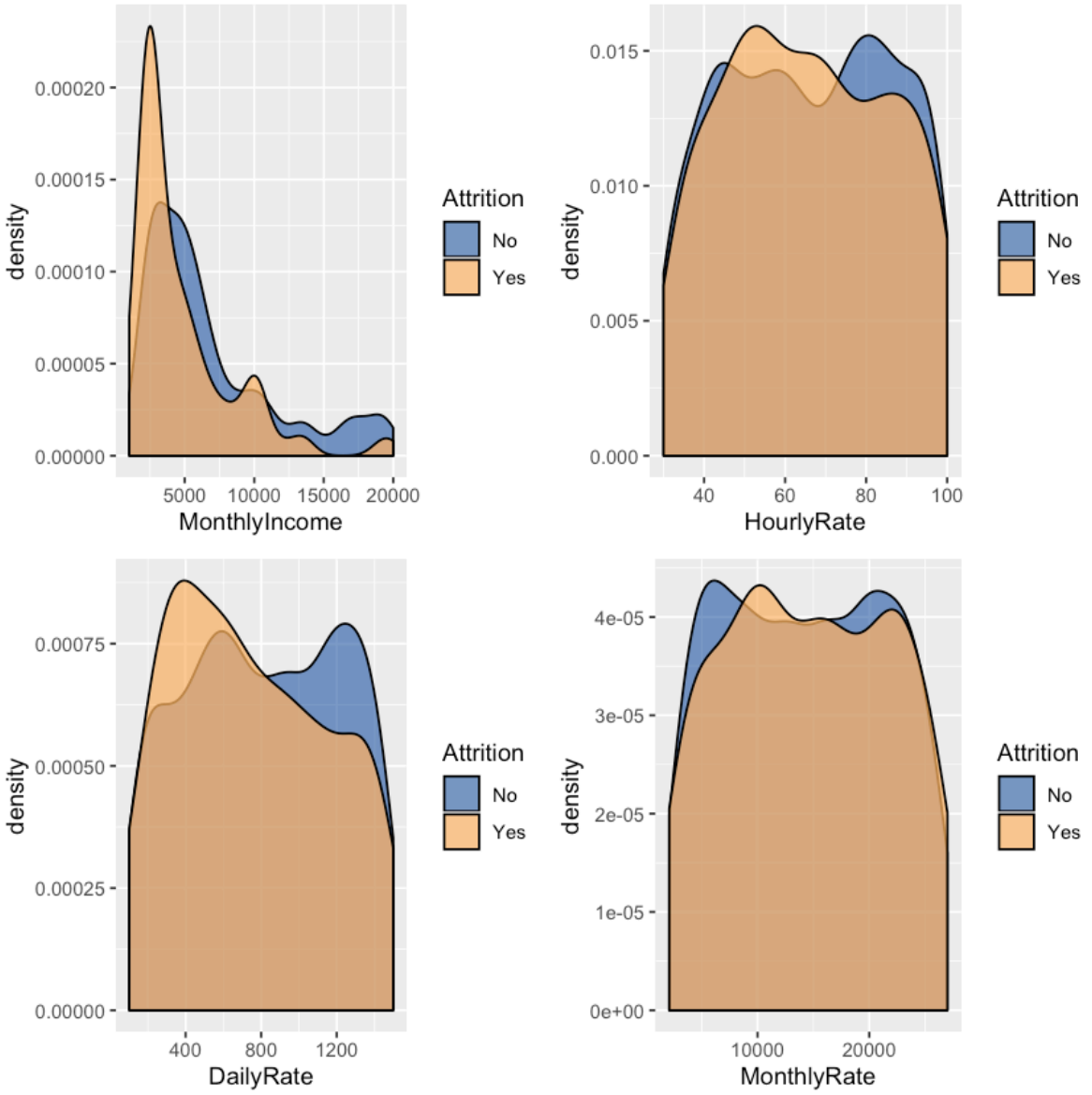
A Quick EDA

In [9]:

```
ibm <- read.csv("~/Desktop/ibm.csv")

suppressMessages(library(ggplot2))
suppressMessages(library(grid))
suppressMessages(library(gridExtra))
suppressMessages(library(dplyr))
suppressMessages(library(rpart))
suppressMessages(library(rpart.plot))
suppressMessages(library(randomForest))
suppressMessages(library(caret))
suppressMessages(library(gbm))
suppressMessages(library(survival))
suppressMessages(library(pROC))
suppressMessages(library(DMwR))
suppressMessages(library(scales))
```

In [10]:

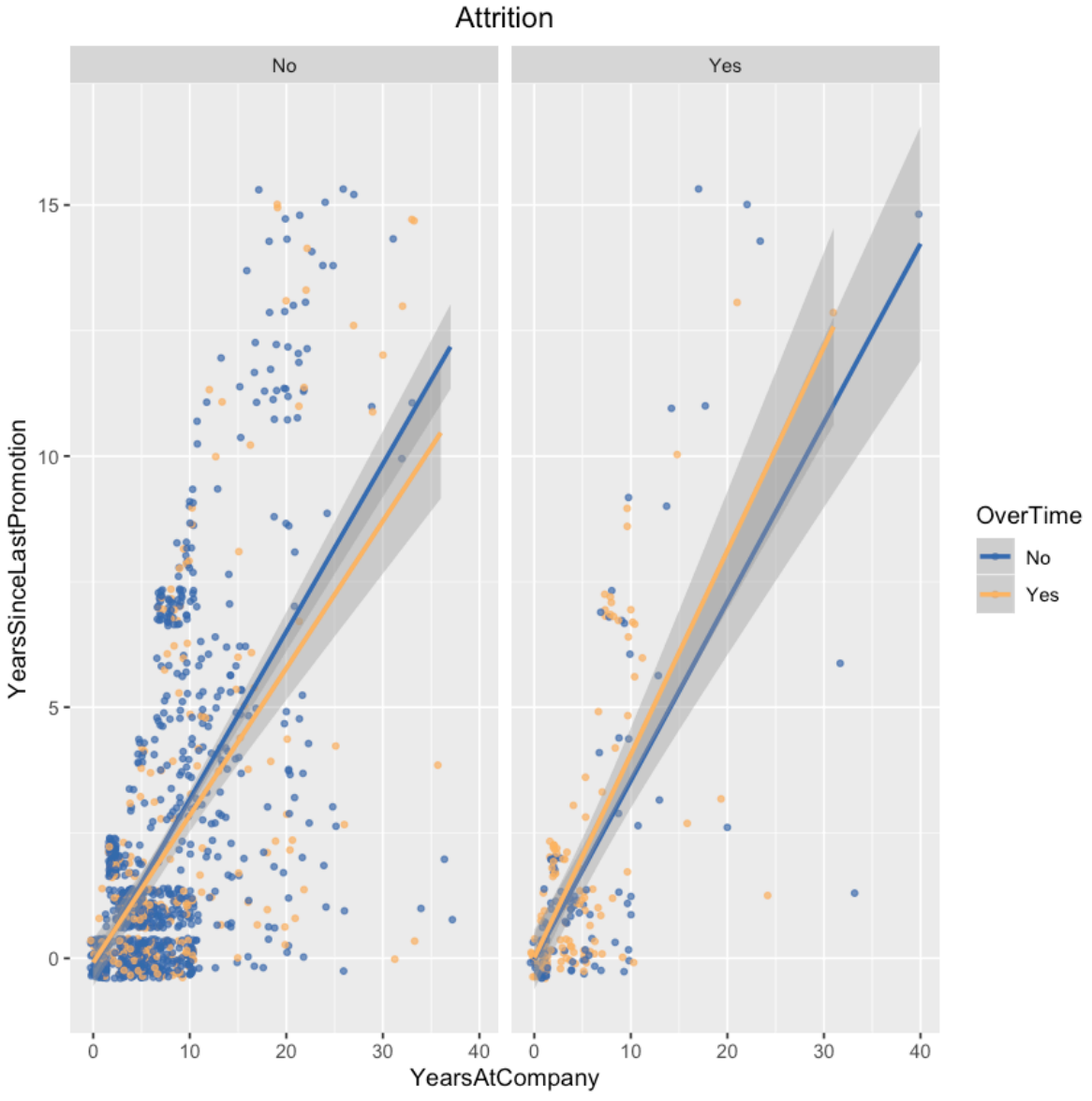
```
g1 <- ggplot(ibm,
             aes(x = MonthlyIncome, fill = Attrition)) +
             geom_density(alpha = 0.7) +
             scale_fill_manual(values = c("#386cb0","#fdb462"))

g2 <- ggplot(ibm,
             aes(x = HourlyRate, fill = Attrition)) +
             geom_density(alpha = 0.7) +
             scale_fill_manual(values = c("#386cb0","#fdb462"))

g3 <- ggplot(ibm,
             aes(x = DailyRate, fill = Attrition)) +
             geom_density(alpha = 0.7) +
             scale_fill_manual(values = c("#386cb0","#fdb462"))

g4 <- ggplot(ibm,
             aes(x = MonthlyRate, fill = Attrition)) +
             geom_density(alpha = 0.7) +
             scale_fill_manual(values = c("#386cb0","#fdb462"))

grid.arrange(g1, g2, g3, g4, ncol = 2, nrow = 2)
```



Monthly income shows something interesting in the data

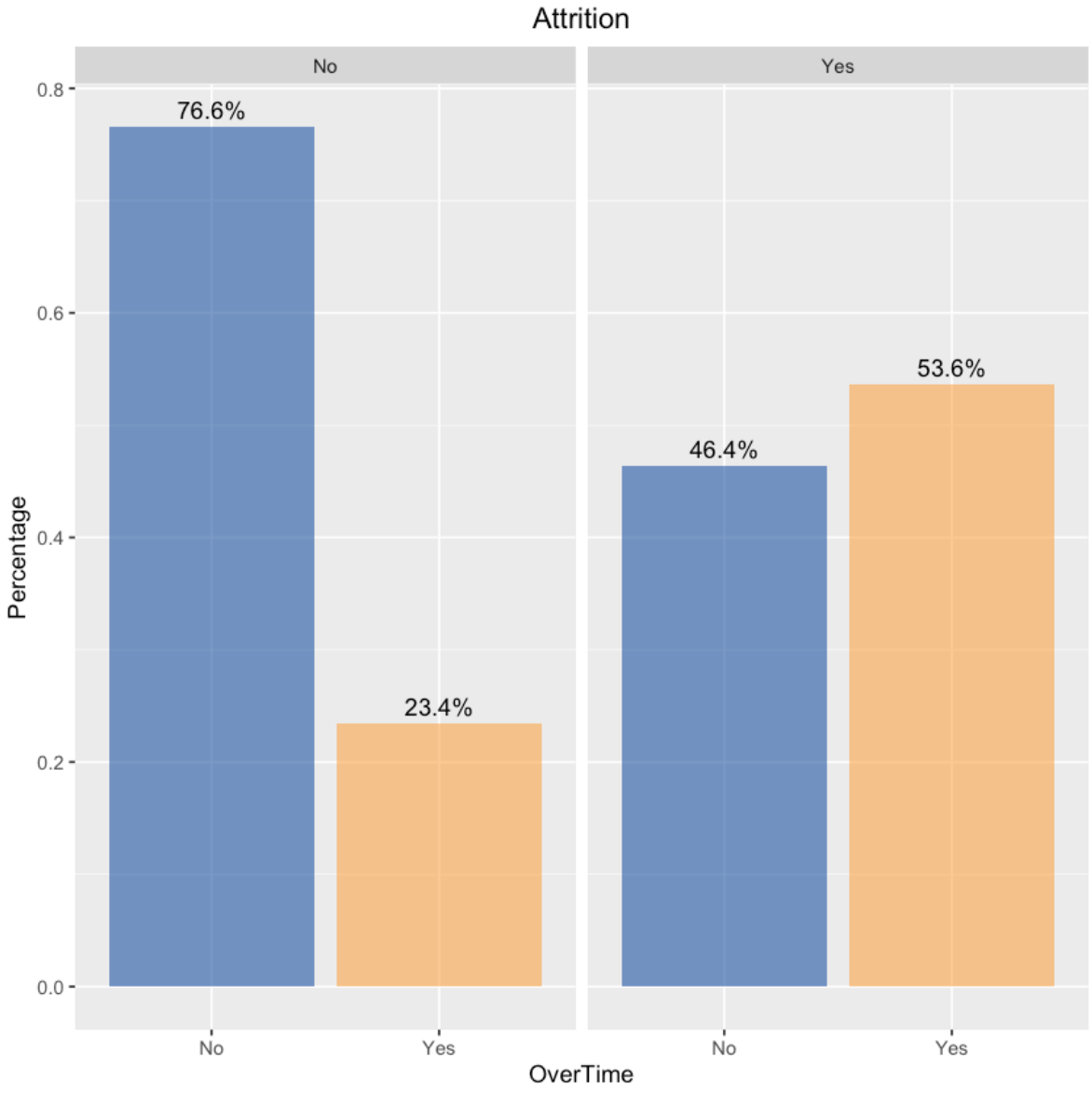
In [11]:

```
ggplot(ibm,
       aes(y = YearsSinceLastPromotion, x = YearsAtCompany, col
our = OverTime)) +
       geom_jitter(size = 1, alpha = 0.7) +
       geom_smooth(method = "gam") +
       facet_wrap(~ Attrition) +
       ggtitle("Attrition") +
       scale_colour_manual(values = c("#386cb0","#fdb462")) +
       theme(plot.title = element_text(hjust = 0.5))
```



In [12]:

```
ggplot(ibm,
       aes(x = OverTime, group = Attrition)) +
       geom_bar(aes(y = ..prop.., fill = factor(..x..)),
                    stat="count",
                    alpha = 0.7) +
       geom_text(aes(label = scales::percent(..prop..), y = ..p
rop.. ),
                    stat= "count",
                    vjust = -.5) +
       labs(y = "Percentage", fill= "OverTime") +
       facet_grid(~Attrition) +
       scale_fill_manual(values = c("#386cb0","#fdb462")) +
       theme(legend.position = "none", plot.title = element_tex
t(hjust = 0.5)) +
       ggtitle("Attrition")
```

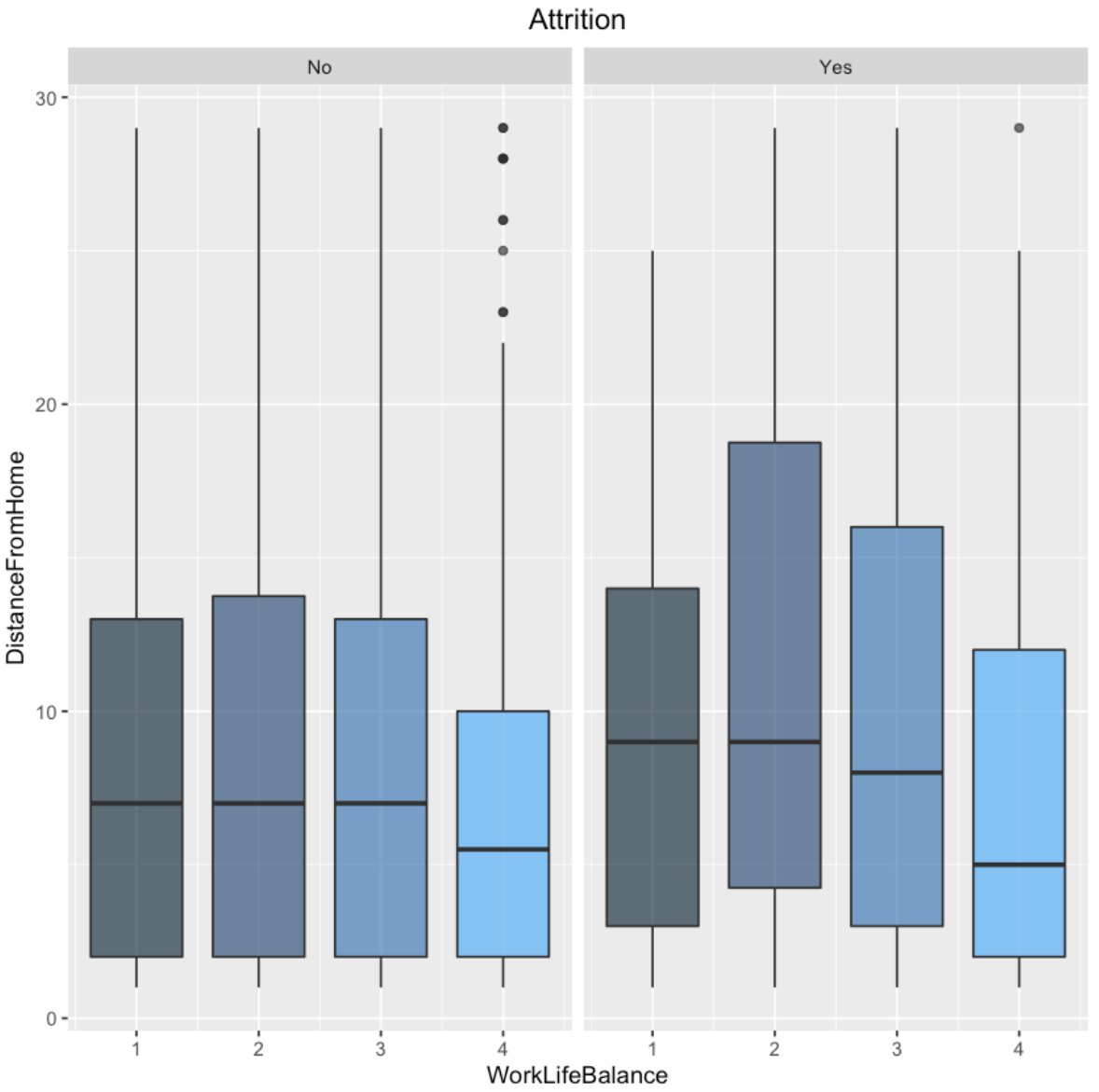


First graph shows Years at company in relation to Years since last promotion, grouped by both attrition and overtime. This is an interesting issue, since a high correlation between these two variables (longer you are in the company, less chance you have to be promoted, so to speak) may mean that people are not really growing within the company. However, since this is a simulated dataset we cannot compare it with some norms outside it, so we can compare certain groups within our set, e.g. those who are working overtime and those who are not.

Here we can note two things. Firstly, there is a relatively higher percentage of people working overtime in the group of those who left, and this observation was confirmed by our barchart. Secondly, while things seem to be going in the right direction for the group of people who still work for IBM (higher correlation between years since last promotion and years at company for those who don't work overtime), you can see that the opposite is happening in the other group. It seems that there may be a pattern of people leaving because they are not promoted although they work hard (as promotion is a viable variable in our data). This is only an assumption at this point, since the confidence intervals (gray area around the lines) are getting wider, meaning there is not that much certainty about this, especially at higher values of X and Y (probably due to the lack of data).

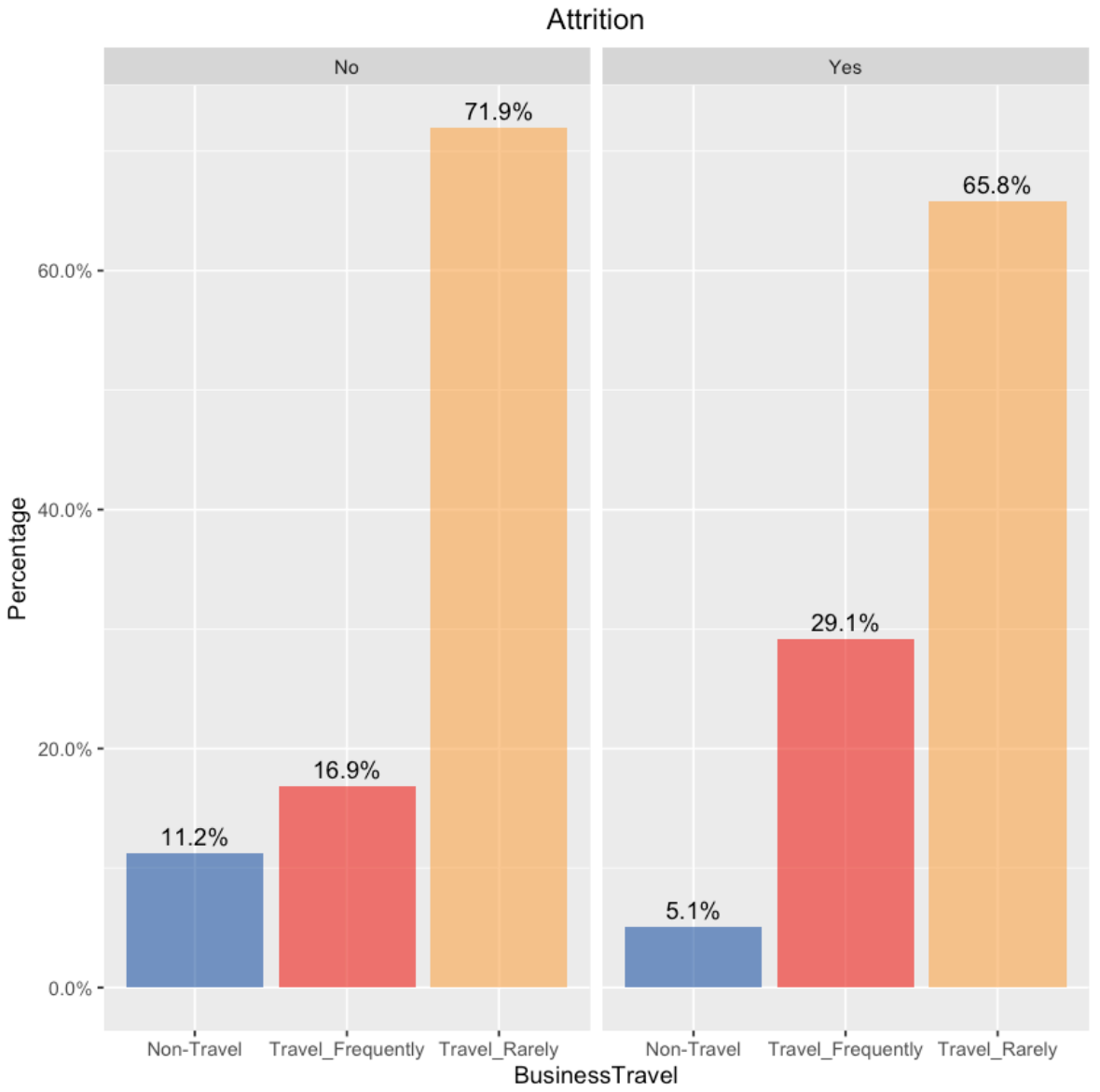
In [13]:

```
ggplot(ibm,
        aes(x= WorkLifeBalance, y=DistanceFromHome, group = Work
LifeBalance, fill = WorkLifeBalance)) +
        geom_boxplot(alpha=0.7) +
        theme(legend.position="none") +
        facet_wrap(~ Attrition) +
        ggtitle("Attrition") +
        theme(plot.title = element_text(hjust = 0.5))
```


Attrition
No
Yes
DistanceFromHome
30
20
10
0
1
2
3
4
WorkLifeBalance

In [14]:

```
ggplot(ibm,
       aes(x= BusinessTravel,  group=Attrition)) +
       geom_bar(aes(y = ..prop.., fill = factor(..x..)),
                  stat="count",
                  alpha = 0.7) +
       geom_text(aes(label = scales::percent(..prop..), y = ..p
rop.. ),
                  stat= "count",
                  vjust = -.5) +
       labs(y = "Percentage", fill="Business Travel") +
       facet_grid(~Attrition) +
       scale_y_continuous(labels=percent) +
       scale_fill_manual(values = c("#386cb0","#ef3b2c", "#fdb4
62")) +
       theme(legend.position = "none", plot.title = element_tex
t(hjust = 0.5)) +
       ggtitle("Attrition")
```


Attrition
No
Yes
71.9%
65.8%
60.0%
40.0%
20.0%
0.0%
Percentage
29.1%
16.9%
11.2%
5.1%
Non-Travel
Travel_Frequently
Travel_Rarely
Non-Travel
Travel_Frequently
Travel_Rarely
BusinessTravel

We can see that there are some patterns in the work-life balance if we compare the two groups within the Attrition variable.

Those who rated their work-life balance relatively low were commuting from a bit farther away in comparison with those who rated their work-life balance as very good. This difference is more pronounced in the group of those who are not in the company anymore, suggesting a possibly influential attrition factor. If the distance is measured in kilometers or miles, this of course doesn't represent a long distance, but since we're dealing with a simulated dataset, it might play a role in predicting our attrition class.

Furthermore, there seems to be a clear indication that those who left travelled more frequently compared to others. This might have also been an important reason behind their resignation. Here I presume that travelling means staying somewhere else overnight, or for a longer period of time, which is why we may connect this to work-life balance issues.

Let's proceed to the modeling part!

Modeling (Decision Tree)

In [16]:

```
set.seed(3221)

# Getting rid of long variable names & certain unuseful variable
s

levels(ibm$JobRole) <- c("HC", "HR", "Lab", "Man", "MDir", "RsD"
, "RsSci", "SlEx", "SlRep")
levels(ibm$EducationField) <- c("HR", "LS", "MRK", "MED", "NA",
"TD")
ibm <- ibm[c(-9,-10,-22,-27)]

# Creating train & test sets

n <- nrow(ibm)
rnd <- sample(n, n * .70)
train <- ibm[rnd,]
test <- ibm[-rnd,]

# Modeling

dtree <- rpart(Attrition ~., data = train)
preds <- predict(dtree, test, type = "class")

rocv <- roc(as.numeric(test$Attrition), as.numeric(preds))
rocv$auc

prop.table(table(test$Attrition, preds, dnn = c("Actual", "Predi
cted")),1)
```

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.643817204301075

```
      Predicted
Actual         No        Yes
   No  0.95430108 0.04569892
   Yes 0.66666667 0.33333333
```

A rather poor AUC with a rather poor sensitivity. It seems that building a single tree will not bring us anywhere.

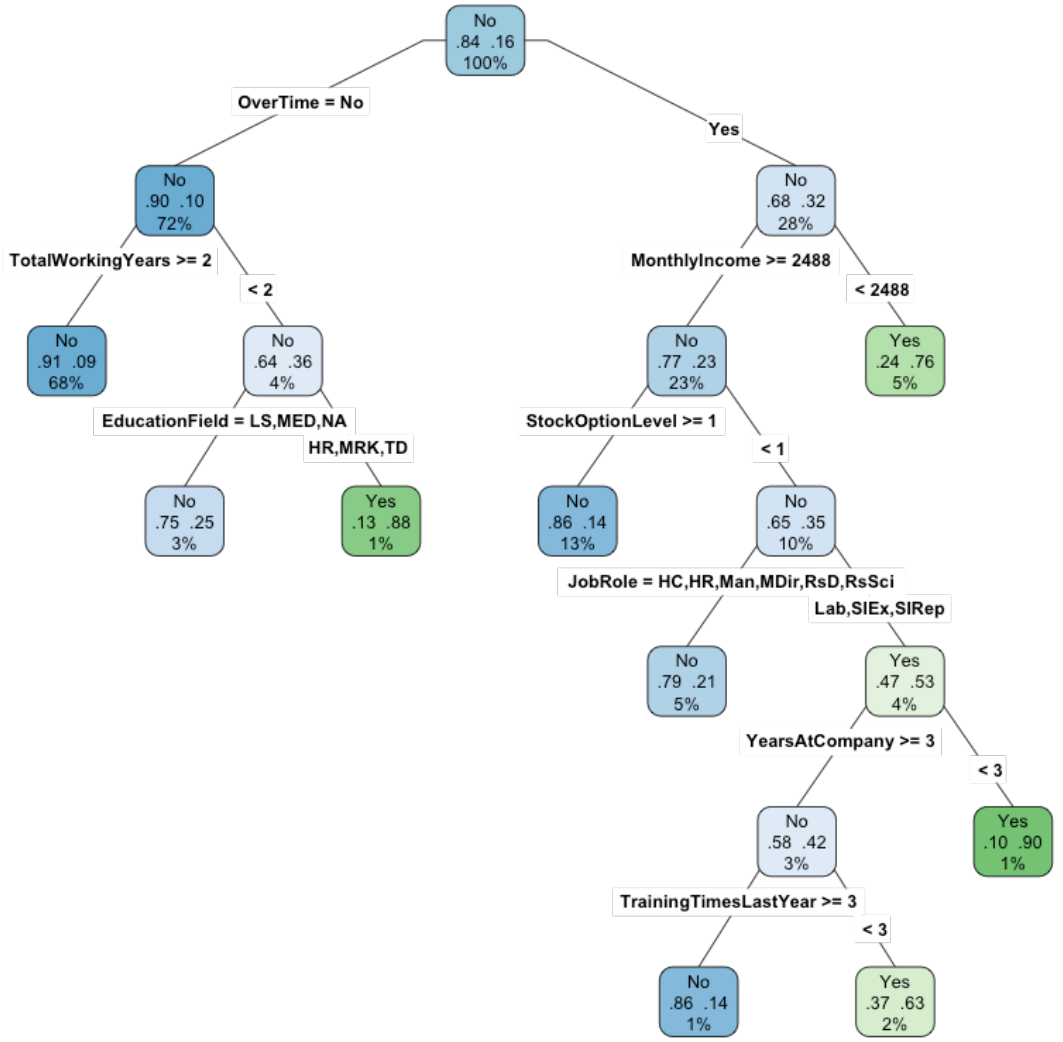
However, while not being useful in general, such a model can nevertheless help us see some patterns. Let us plot the tree and see if we can find any.

In [17]:

```
# Pruning & plotting the tree

dtreepr <- prune(dtree, cp = 0.01666667)
predspr <- predict(dtreepr, test, type = "class")

rocvpr <- roc(as.numeric(test$Attrition), as.numeric(predspr))
rocvpr$auc

rpart.plot(dtreepr,
           type = 4,
           extra = 104,
           tweak = 0.9,
           fallen.leaves = F)
```

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.633356708742403



We have pruned the tree a bit, just so that it is not too crowded and too incomprehensible (our AUC did not suffer much as you can see).

You can see that the most important variables seem to be overtime and monthly income - something we have already discerned through our graphic EDA. Remember, the sensitivity of this model is quite low, which is why we would in principle advise against any general interventions on this basis.

However, we can see that a major percentage of those who left can be relatively reliably identified using the criteria of combined overtime and monthly income. If we consider them jointly, this points to another factor: effort-reward imbalance. This is why it is not entirely unuseful after all to plot a decision tree - it makes you see some patterns that you might have forgotten during your EDA.

Modeling (Random Forest & simple GBM)

In [18]:

```
set.seed(2343)

# Random forest

fit.forest <- randomForest(Attrition ~., data = train)
rfpreds <- predict(fit.forest, test, type = "class")

rocrf <- roc(as.numeric(test$Attrition), as.numeric(rfpreds))
rocrf$auc
```

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.568431510051426

In [19]:

```
set.seed(3433)

# Setting the basic train control used in all GBM models

ctrl <- trainControl(method = "cv",
                     number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

# Simple GBM

gbmfit <- train(Attrition ~.,
                data = train,
                method = "gbm",
                verbose = FALSE,
                metric = "ROC",
                trControl = ctrl)

gbmpreds <- predict(gbmfit, test)

rocgbm <- roc(as.numeric(test$Attrition), as.numeric(gbmpreds))
rocgbm$auc
```

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.591514726507714

That comes as a shock - I did not expect Random Forest and GBM to perform so poorly, and this is the first time I see a single decision tree outperforming them. I've tried and tuning the mtry and ntree parameters in RF gives you around 0.020 of a lift, but you are still below the decision tree (I guess even tuning the GBM wouldn't help much more; feel free to try). Apparently this may happen when a single tree is already stable enough.

Modeling (GBM with weighting, SMOTE and up & down-sampling)

Maybe we should tackle the class imbalance? Note that usually this would be considered if the ratio between classes is 1:10 or higher; in our case it's 1:5, but still it may be justified since we have seen with the decision tree that our main problem is predicting those who actually leave (sensitivity).

I will try different techniques: weighting (punishing the errors in the minority class), down-sampling (randomly removing cases from the majority class), up-sampling (randomly replicating instances in the minority class) and SMOTE (downsampling and synthesizing new minority cases).

In [20]:

```
ctrl$seeds <- gbmfit$control$seeds

# Weighting

model_weights <- ifelse(train$Attrition == "No",
                        (1/table(train$Attrition)[1]) * 0.5,
                        (1/table(train$Attrition)[2]) * 0.5)

weightedfit <- train(Attrition ~ .,
                     data = train,
                     method = "gbm",
                     verbose = FALSE,
                     weights = model_weights,
                     metric = "ROC",
                     trControl = ctrl)
```

```
weightedpreds <- predict(weightedfit, test)
rocweight <- roc(as.numeric(test$Attrition), as.numeric(weighted
preds))
rocweight$auc

# SMOTE

ctrl$sampling <- "smote"

smotefit <- train(Attrition ~.,
                  data = train,
                  method = "gbm",
                  verbose = FALSE,
                  metric = "ROC",
                  trControl = ctrl)

smotepreds <- predict(smotefit, test)
rocsmote <- roc(as.numeric(test$Attrition), as.numeric(smotepred
s))
rocsmote$auc

# UP-sampling

ctrl$sampling <- "up"

upfit <- train(Attrition ~.,
               data = train,
               method = "gbm",
               verbose = FALSE,
               metric = "ROC",
               trControl = ctrl)

uppreds <- predict(upfit, test)
rocup <- roc(as.numeric(test$Attrition), as.numeric(uppreds))
rocup$auc

# DOWN-sampling

ctrl$sampling <- "down"

downfit <- train(Attrition ~.,
                 data = train,
                 method = "gbm",
                 verbose = FALSE,
                 metric = "ROC",
```

```
                  trControl = ctrl)

downpreds <- predict(downfit, test)
rocdown <- roc(as.numeric(test$Attrition), as.numeric(downpreds)
)
rocdown$auc
```

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.780329593267882

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.681626928471248

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.73872136512389

```
Setting levels: control = 1, case = 2
Setting direction: controls < cases
```

0.750467508181393

Comparing the models

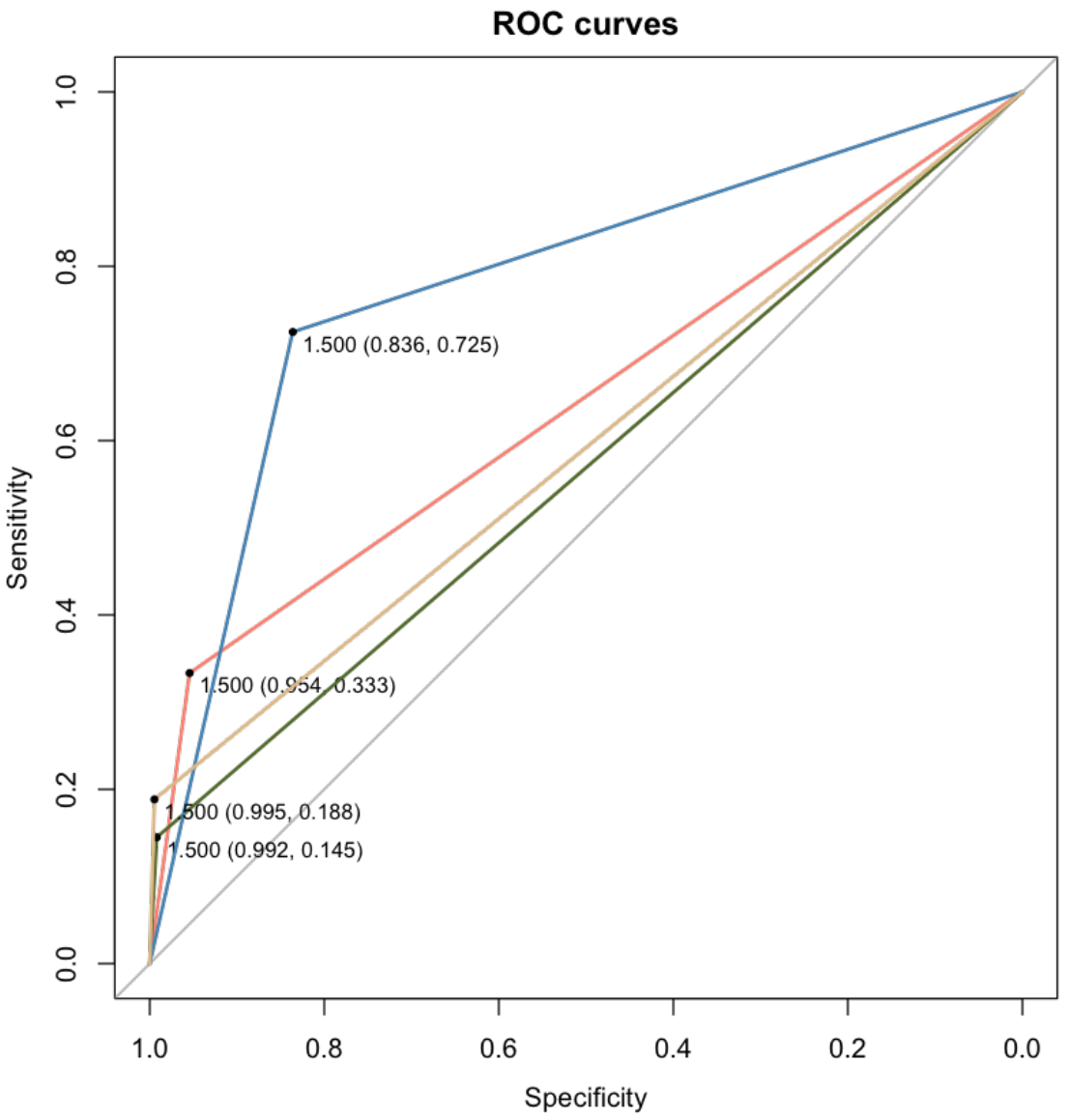
Indeed, all the different techniques perform better (AUC from 0.68 to 0.78) than both simple GBM and Random Forest.

I will plot and compare only the best algorithm, which is the GBM with weighting (0.78). We can see that its specificity is a bit lower (12% to 16%), but this seems reasonable since we have a 40%-55% gain in sensitivity (72.5%)! It is really doing far better if we compare it to the sensitivity of RF (15.9%), GBM (18.8%) or decision tree (33%).

This is the reason why I prefer AUC to e.g. accuracy. The latter can be - especially in the case of class imbalance - inflated by the fact that our model has a high specificity (correctly identifying the negatives), but we cannot really generalise and extrapolate anything from such a model (e.g. it may be quite misleading to say that "low income" is the reason behind people resigning, if we only see that people who stay have higher income and there is no pattern in those who leave). Moreover, we may not be able to find out which - if any - department or job role is at risk, since our information (predictions) about employees at risk is really scarce - meaning we would also be having difficulties addressing them individually in case we would want to do so.

In [21]:

```
plot(rocv, ylim = c(0,1), print.thres = T, print.thres.cex = 0.8
, main = "ROC curves", col = "salmon")
plot(rocrf, ylim = c(0,1), print.thres = T, print.thres.cex = 0.
8, col = "darkolivegreen", add = T)
plot(rocweight, ylim = c(0,1), print.thres = T, print.thres.cex
= 0.8, col = "steelblue", add = T)
plot(rocgbm, ylim = c(0,1), print.thres = T, print.thres.cex = 0
.8, col = "burlywood", add = T)
```


ROC curves
Sensitivity
Specificity
1.0
0.8
0.6
0.4
0.2
0.0
1.500 (0.836, 0.725)
1.500 (0.954, 0.333)
1.500 (0.995, 0.188)
1.500 (0.992, 0.145)

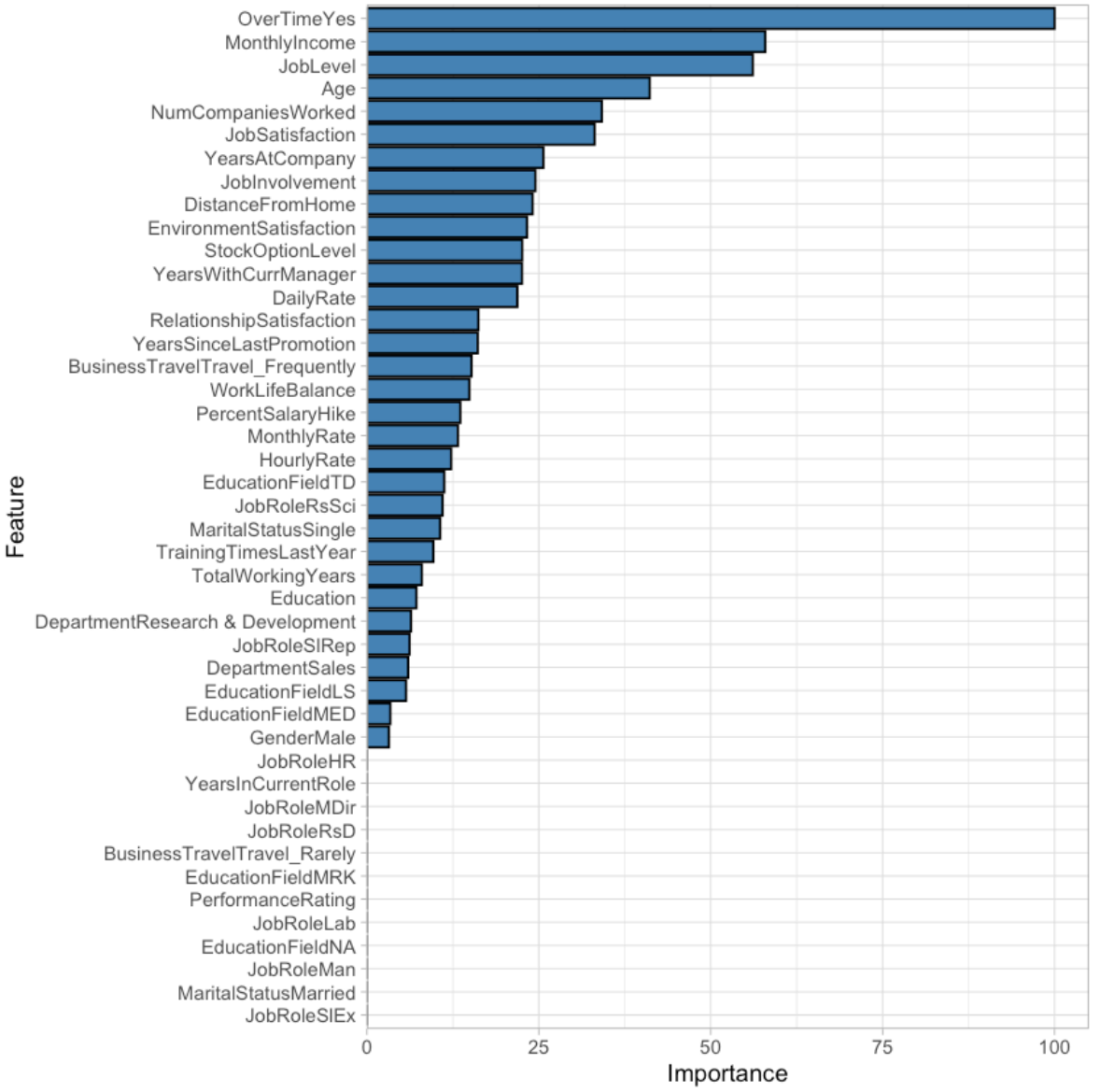
Making sense of our models & analyses

How can we help ourselves with these findings? Complex algorithms aren't easy to interpret, but there are several ways in which they can be useful:

We can examine the variable importance list, and see which factors in general are helpful in determining the outcome (e.g. attrition); this can be also useful in determining where should we carry out our (HR) audit first, We can use our model to calculate the probabilities of leaving for each and everyone of e.g. our new employees; we can also make new variables from these probabilities, e.g. determining who one has the highest possibility to leave and has at the same time high performance rating, works several hours, and contributes in a meaningful way to our company .. We can then convey this information to our management who can perhaps assess the situation, and speak to the person in a tactful way, We can evaluate our organisational tree with regards to these probabilities; e.g. we can assess which department or role has the highest possibility of leaving, and then channel our efforts there, or do additional analyses on that department/role (either quantitative or e.g. focus groups). Let us plot the variable importance list from our best model.

In [22]:

```
ggplot(varImp(weightedfit)) +
geom_bar(stat = 'identity', fill = 'steelblue', color = 'black')
+
scale_y_continuous(limits = c(0, 105), expand = c(0, 0)) +
theme_light()
```


OverTimeYes
MonthlyIncome
JobLevel
Age
NumCompaniesWorked
JobSatisfaction
YearsAtCompany
JobInvolvement
DistanceFromHome
EnvironmentSatisfaction
StockOptionLevel
YearsWithCurrManager
DailyRate
RelationshipSatisfaction
YearsSinceLastPromotion
BusinessTravelTravel_Frequently
WorkLifeBalance
PercentSalaryHike
MonthlyRate
HourlyRate
EducationFieldTD
JobRoleRsSci
MaritalStatusSingle
TrainingTimesLastYear
TotalWorkingYears
Education
DepartmentResearch & Development
JobRoleSlRep
DepartmentSales
EducationFieldLS
EducationFieldMED
GenderMale
JobRoleHR
YearsInCurrentRole
JobRoleMDir
JobRoleRsD
BusinessTravelTravel_Rarely
EducationFieldMRK
PerformanceRating
JobRoleLab
EducationFieldNA
JobRoleMan
MaritalStatusMarried
JobRoleSlEx
Feature
0
25
50
75
100
Importance

The top 5 factors that influence the attrition seem to be:

Overtime, Monthly income, Job level, Age, Number of companies worked for. Two of these are already familiar to us from our EDA and decision tree plot - it seems that we should indeed do something about those who work overtime and then leave and those who have a low monthly income (which is probably also linked to the job level).

We should also delve a bit more into the matter of age & number of companies the person worked for. Isn't that simply linked to people who retire, and who e.g. probably worked for many companies throughout their life? Or to the fact that we frequently hire freelancers for some temporary positions? If not, what could be wrong there? What policies and/or services are we lacking?

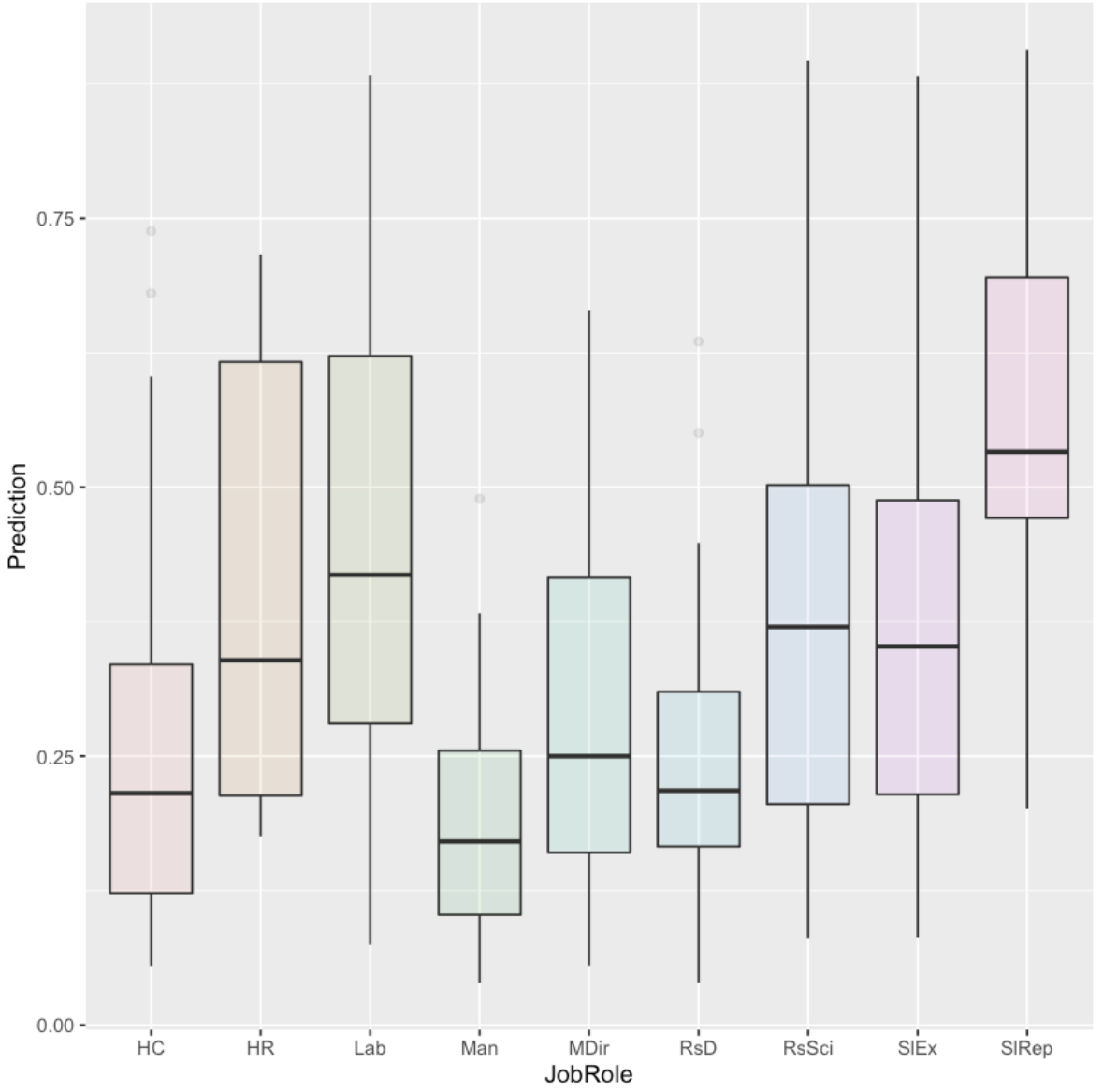
Last but not least, the fact that all three variables linked (directly or indirectly) to work-life balance (distance from home, business travel, and work-life balance as such) have their place among the top 20 variables could also be a sign that something should be done in this area. Remember, we've already observed this pattern during the visualization phase.

We could prioritise and organise an HR audit for each of these areas, just to see what can be improved or what is lacking.

Let us also check whether a certain job role has a higher probability of leaving (according to our model's classification applied to the test set).

In [23]:

```
weightedprobs <- predict(weightedfit, test, type = "prob")
test$Prediction <- weightedprobs$Yes

ggplot(test,
       aes(x=JobRole, y=Prediction, fill=JobRole)) +
       geom_boxplot(alpha=0.1) +
       theme(legend.position="none")
```



It's good that we checked!

Something should be done about those sales representatives. Why do they have more than 50% probability to leave on average?

In [ ]: