R Project

Martin Le Guennec

2023-05-11

Contents

1 Code description

This code compares several model of prediction on a data set to find the one who is the better predictor.

The data set that will be used in this code is contained in the directory `DAT` and is names `data_R.csv`

Please make sure that there is a file named `apa.csl` and another file named `references.bib`, they are used to cite the articles mentionned in this code.

The file you are currently reading is `main.rmd`, it is the only code you have to run in order to get the full analysis. You can just run all the code or run the blocks of code one by one, some of the parameters can be changed to make the processing time faster. You can also simply `knit` this script to pdf or html.

1.1 Organization of the code

First we load the libraries used in this code and we create a function named `TestModel` that will be used many times in this code.

Then, you will find a part that explains the context and the objective of this code.

Finally, the major part of the code consists of creating models and comparing them

1.2 Library

1.3 Functions

1.4 Load data

2 Context

Despite our growing knowledge of the mechanisms of fatigue during physical effort, traditional strength training does not use any objective means to characterize these. However, individualization and the utilization of scientific advancements have become pivotal in physical preparation. Recognizing this, Teikari & Pietrusz (2021) propose the incorporation of objective measurements to quantify fatigue during training. By leveraging these quantifiable indicators, coaches and athletes can gain valuable insights into the physiological responses to resistance training, enabling more informed and targeted training strategies. This integration of objective signals holds the potential to enhance performance outcomes and optimize training adaptations by tailoring workouts to the individual needs of athletes.

2.1 The experiment

The objective of my experiment was to determine if it was possible to predict fatigue from physiological measurements during different resistance training modalities.

To do this, 7 participants performed two different measurement sessions where they performed squat movements at maximal intented speed. The two sessions differed in the initial mean concentric velocity (MV) that the participants were asked to achieve. This MV was measured using a linear position transducer.

In the first session, they performed squats with the load that allowed them to lift the bar at approximately 0.8 m.s-1; and in the second session, it was with the load that allowed them to lift the bar at approximately 0.5 m.s-1. This speed had to be reached during the first 3 repetitions of a set, otherwise the weight was increased or decreased accordingly.

They continued to perform reps until they lost 20% velocity from the MV of the fastest rep of the set. When they reached this 20% threshold, they had completed a set.

They had to perform as many sets as possible to reach their training load. This training load was calculated for each repetition performed as follow :

$$WL = \sum^{n} \frac{m^2 \times g \times ROM}{MV \times h}$$

where WL is the workload ; n is the number of repetitions performed ; m is the weight lifted (in % repetition maximum) and is normalized with respect to the theoretical maximal weight that the subject can lift for one repetition on the squat ; g is the acceleration of the gravity at the Earth's surface and is worth 9.81 m.s-2 ; ROM is the range of motion (in m) of the barbell during the nth repetition ; MV is the mean concentric velocity (in m.s-1) ; and h height is the height of the subject (in m) and is used to normalized the ROM.

While they perform the squats, their muscular electrical activity is measured with electromyography (EMG), their cortical electrical activity is measured with electroencephalography (EEG), and their muscular metabolic activity is measured with near-infrared spectroscopy (NIRS).

2.1.1 Why measuring velocity ?

It has been shown that the use of velocity allows an objective and precise prescription of training volume and work intensity :

- It allows a quick and precise estimation of the intensity, allowing an individualization of the training and an adaptation to the athlete's daily shape (Weakley et al., 2021 ; Flanagan & Jovanović, 2014)
- Bar speed is also a good indicator of fatigue because of its relationship with other mechanical and metabolic variables (Sánchez-Medina & González-Badillo, 2011)

For these reasons, it is the variable that we will use to estimate the level of fatigue of the subjects during the session.

2.1.2 EMG measurements

The EMG were measured on 9 different muscles :

- Gastrocnemius medialis (GaMe)
- Gastrocnemius lateralis (GaLa)
- Biceps femoris (BiFe)
- Semi-tendinous (SeTe)
- Quadriceps' vastus lateralis (VaLa)
- Quadriceps' rectus femoris (ReFe)
- Quadriceps' vastus medialis (VaMe)
- Gluteus maximus (GlMa)
- Lumber extensors (ExLo)

In this analysis, we will only focus on VaLa, ReFe, VaMe, GlMa and ExLo because the signal for the other were very noisy, therefore, the extracted variables must be erroneus.

For each of these variables, we measured the mean instantaneous mean frequency (IMNF) for each repetition, the mean value of the RMS enveloppe (meanRMS), the area under the curve of the RMS (aucRMS) and the maximum value of the RMS (maxRMS). These values were calculated in Matlab.

2.2 Aim of the code

The aim of this code is to predict the fatigue – estimated from the MV – thanks to the physiological measurements' values. Normally, we should use EEG, EMG and NIRS measurements but I didn't have the time to treat NIRS and EEG data. Therefore, we will lean only on EMG data to predict the fatigue.

Thus, with this code, we will compare different models with different parameters to find the better to predict MV using

$$\hat{Y} = \hat{f}(X)$$

where $\hat{Y}$ is the prediction of Y – in our context, the MV -, $\hat{f}$ is our estimate model of prediction, and X is the set of inputs measured – here the EMG data - used to predict Y (G. James et al., 2021).

For this code, we will compare two estimate model of prediction with different methods

- The first one will predict the MV only with EMG values will be named `estimate.1`

$$\hat{MV} = \sum^{muscle} \Big(\hat{f}(IMNF_{muscle}) + \hat{f}(meanRMS_{muscle}) + \hat{f}(aucRMS_{muscle}) + \hat{f}(maxRMS_{muscle} + \hat{f}(t2maxRMS_{musc}$$

- The second one will predict the MV with the EMG values, and with the subject ID, set number and repetition number as co-variables. It will be named `estimate.2`

$$\hat{MV} = \sum^{muscle} \Big(\hat{f}(IMNF_{muscle}) + \hat{f}(meanRMS_{muscle}) + \hat{f}(aucRMS_{muscle}) + \hat{f}(maxRMS_{muscle}) + \hat{f}(t2maxRMS_{mus}$$
$$+ subjectID + setNumber$$

3 Compare models

As mentioned above, we will compare the two estimates but we will also compare different methods :

- Linear model
- Linear mixed models
- Regression trees
- Random forest

For each of these models, we will try different parameters with

- The two estimates
- Different data sets : the full data set or split into sessions
  - Given that the two session don't correspond to the same modality of resistance training, maybe that fitting a model by session will allow for a better prediction than if fit it for the whole data set

Set the number of iteration

We set a number of 100 iteration that will be common to all models. This number corresponds to the number of times that the data set will be shuffled and then the model trained on this basis.

The higher this number, the higher the prediction accuracy but there is a risk of over fitting the data (*i.e.* the model performs exceptionally well on the training data but fails to generalize well to unseen or new data) and increases the calculation time. Inversely, a low number of iteration per model will certainly reduce the accuracy of its prediction.

The performance of a model is assessed by the root mean square error, calculated as ;

$$RMSE = \sqrt{\frac{1}{n}\sum^{n}(y_i - \hat{y}_i)^2}$$

Smaller RMSE values indicate better predictive accuracy.

3.1 Disclaimer

The interpretation of the results is based of the data that I have obtained when I made this document. However, given that each iteration is made on a random sample of the data set, it is possible that the results differ from what I have seen.

I would like to emphasize that the results I discuss in this script were obtained by making a high number of run (100). These results seem therefore pertinent and I invite you to verify but the processing time can be long.

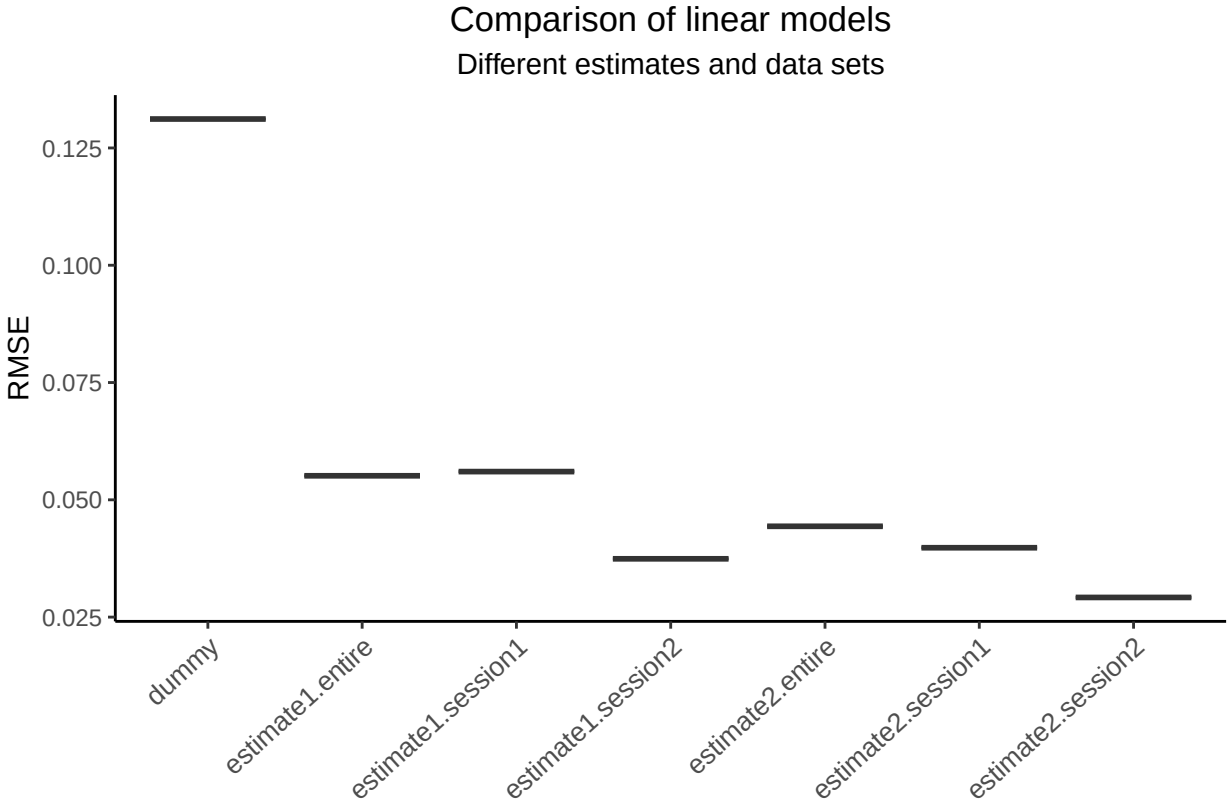
4 Linear model

First, we evaluate the performances of a linear model. The linear model, of multiple linear regression model, assumes that there is approximately a linear relationship between each X_p and Y. The aim of this model, is to determine the coefficients

4.1 Choose parameters

We will compare a model with the estimate model 1 (`estimate1`), and a model with the estimate model 2 (`estimate2`).

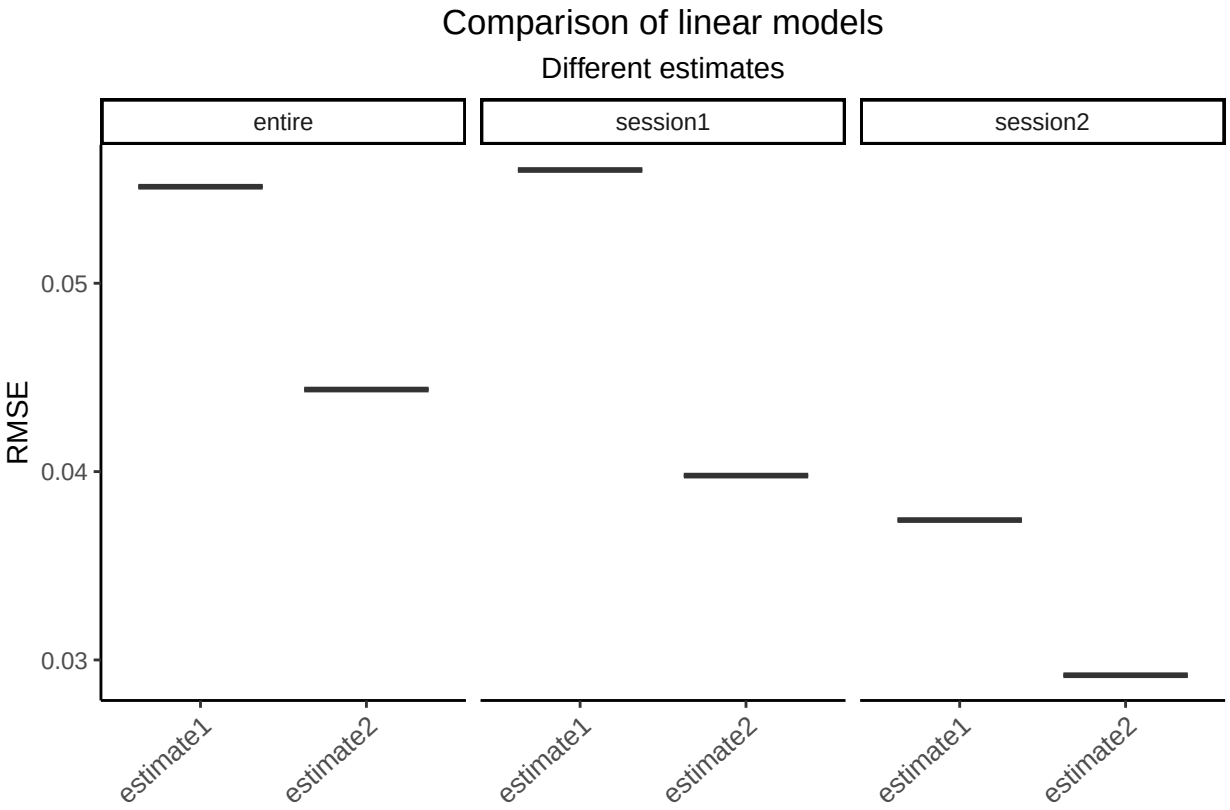
In addition, we are going to compare the performances of each linear model on the entire data set (`entire`,) then on each session individually (`session1` and `session2`).



All models allow a reliable prediction because they are more precise than the dummy prediction.

4.1.1 Estimate comparison

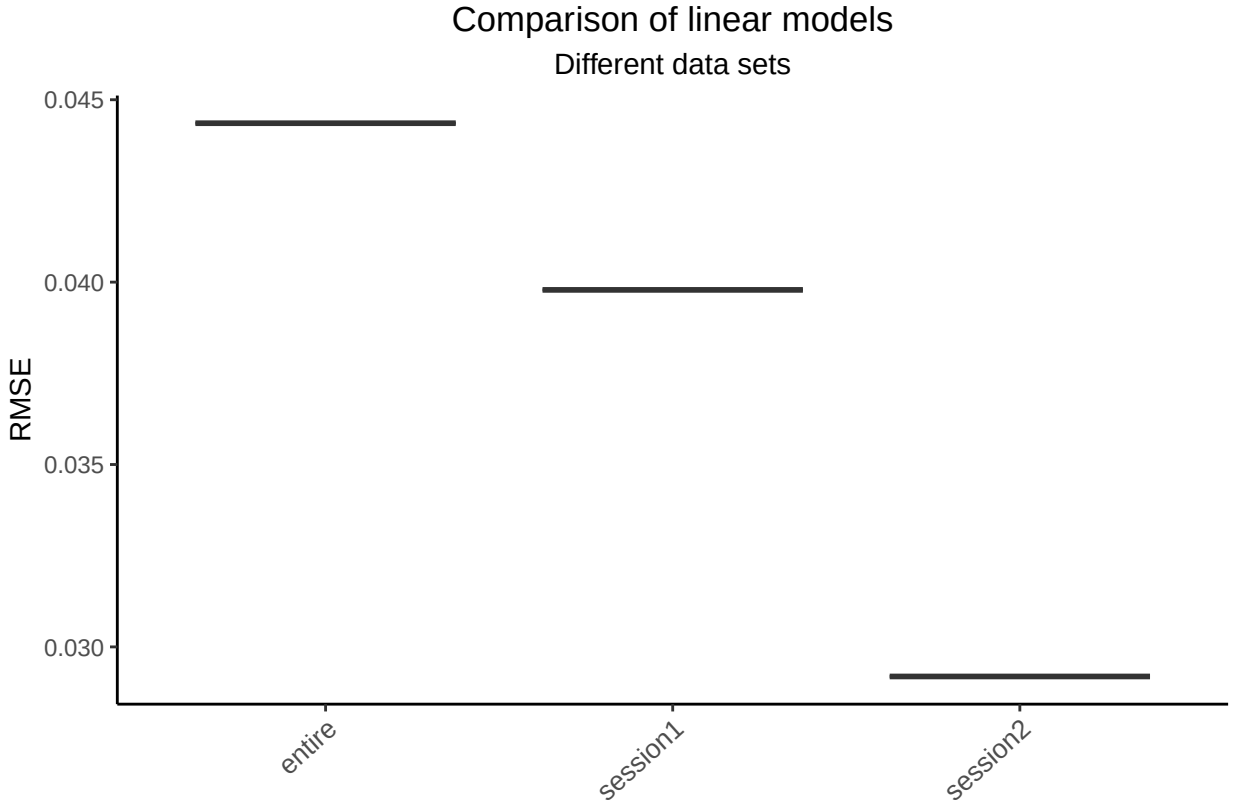
We are going to compare the models by estimate first.



Using the `estimate.2` improve the accuracy of the model.

4.1.2 Data set comparison

We are going to compare the accuracy of the model with the entire data set vs. with separate data set for each session.



Separating the data set by sessions seem to improve the accuracy of the model.

4.2 Interpretation

Now that we have identified the best parameters for our regression model, we can fit it on the whole data set and analyse the results. We will proceed by session.

```
##
## Call:
## lm(formula = estimate.2, data = session.1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.114170 -0.023476 -0.001402  0.024219  0.141746
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.5050737  0.1208209   4.180 3.49e-05 ***
## IMNF_VaMe      0.0013769  0.0011035   1.248 0.212762
## IMNF_ReFe     -0.0001589  0.0004876  -0.326 0.744683
## IMNF_VaLa      0.0034414  0.0009806   3.509 0.000494 ***
## IMNF_GlMa      0.0004782  0.0010447   0.458 0.647366
## IMNF_ExLo      0.0009809  0.0007727   1.269 0.204928
## meanRMS_VaMe   0.5130865  0.1859849   2.759 0.006037 **
## meanRMS_ReFe  -0.3469531  0.1163877  -2.981 0.003027 **
## meanRMS_VaLa  -0.1816238  0.1547766  -1.173 0.241227
## meanRMS_GlMa   0.4768019  0.1124552   4.240 2.71e-05 ***
## meanRMS_ExLo  -0.1902443  0.0621754  -3.060 0.002346 **
## aucRMS_VaMe   -0.3519878  0.1048011  -3.359 0.000849 ***
## aucRMS_ReFe    0.1148219  0.0674214   1.703 0.089244 .
## aucRMS_VaLa    0.0626264  0.0827087   0.757 0.449327
## aucRMS_GlMa   -0.3919628  0.0644220  -6.084 2.49e-09 ***
## aucRMS_ExLo    0.0178928  0.0323724   0.553 0.580727
## maxRMS_VaMe    0.0025032  0.0115942   0.216 0.829160
## maxRMS_ReFe    0.0533918  0.0118295   4.513 8.14e-06 ***
## maxRMS_VaLa    0.0347193  0.0098404   3.528 0.000461 ***
## maxRMS_GlMa    0.0528328  0.0127349   4.149 3.99e-05 ***
## maxRMS_ExLo    0.0912818  0.0252222   3.619 0.000329 ***
## t2maxRMS_VaMe  0.0175799  0.0110778   1.587 0.113221
## t2maxRMS_ReFe  0.0258899  0.0122509   2.113 0.035119 *
## t2maxRMS_VaLa  0.0458428  0.0142104   3.226 0.001346 **
## t2maxRMS_GlMa -0.0210769  0.0220327  -0.957 0.339269
## t2maxRMS_ExLo  0.0282984  0.0103153   2.743 0.006322 **
## ID02HaVi       0.0330467  0.0351726   0.940 0.347944
## ID03MaLu      -0.0880217  0.0306311  -2.874 0.004249 **
## ID04GiMa      -0.0534965  0.0217044  -2.465 0.014079 *
## ID05CaEn      -0.1110579  0.0226346  -4.907 1.29e-06 ***
## ID06AbPa      -0.1192782  0.0223202  -5.344 1.44e-07 ***
## ID07BoLa      -0.1347254  0.0231021  -5.832 1.04e-08 ***
## set           -0.0048224  0.0014301  -3.372 0.000810 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03939 on 454 degrees of freedom
## Multiple R-squared:  0.773,  Adjusted R-squared:  0.757
## F-statistic: 48.32 on 32 and 454 DF,  p-value: < 2.2e-16
```

In our model, at least one of the predictor variables is significantly related to the MV, $F(41, 288) = 76.75$, $p < 0.0001$.

The section that follows was not made for the memory

As we can see however, many coefficients aren't significant. Therefore, we can remove them from the model.

```
##
## Call:
## lm(formula = estimate.updated, data = data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.130782 -0.023709 -0.001471  0.024221  0.150579
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.5652270  0.0535206  10.561  < 2e-16 ***
## IMNF_VaLa      0.0046033  0.0007087   6.495 2.14e-10 ***
## meanRMS_VaMe   0.3080778  0.1068618   2.883 0.004122 **
## meanRMS_ReFe  -0.1728576  0.0329227  -5.250 2.31e-07 ***
## meanRMS_GlMa   0.3796215  0.1042121   3.643 0.000300 ***
## meanRMS_ExLo  -0.2100519  0.0474615  -4.426 1.20e-05 ***
## aucRMS_VaMe   -0.2385510  0.0611443  -3.901 0.000110 ***
## aucRMS_GlMa   -0.3668186  0.0576918  -6.358 4.88e-10 ***
## maxRMS_ReFe    0.0531570  0.0106399   4.996 8.30e-07 ***
## maxRMS_VaLa    0.0269832  0.0055534   4.859 1.62e-06 ***
## maxRMS_GlMa    0.0631148  0.0112184   5.626 3.19e-08 ***
## maxRMS_ExLo    0.1054164  0.0230291   4.578 6.05e-06 ***
## t2maxRMS_ReFe  0.0371698  0.0102469   3.627 0.000318 ***
## t2maxRMS_VaLa  0.0473613  0.0114008   4.154 3.88e-05 ***
## t2maxRMS_ExLo  0.0305217  0.0098915   3.086 0.002152 **
## ID02HaVi       0.0751101  0.0251807   2.983 0.003006 **
## ID03MaLu      -0.0492818  0.0193491  -2.547 0.011187 *
## ID04GiMa      -0.0423566  0.0141722  -2.989 0.002950 **
## ID05CaEn      -0.0926606  0.0122946  -7.537 2.54e-13 ***
## ID06AbPa      -0.1126365  0.0182235  -6.181 1.40e-09 ***
## ID07BoLa      -0.1208195  0.0173711  -6.955 1.20e-11 ***
## set           -0.0048808  0.0011463  -4.258 2.50e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03961 on 465 degrees of freedom
## Multiple R-squared:  0.765,  Adjusted R-squared:  0.7544
## F-statistic: 72.08 on 21 and 465 DF,  p-value: < 2.2e-16
```

This way our model is simpler and we don't lose much prediction power despite having less predictors. We have a adjusted R2 at 0.75 instead of 0.76 with all the values. Therefore, we gain in simplicity and lose only 1% of variance explained.

Now, let's do the same for the session at 0,5 m/s.

```
##
## Call:
## lm(formula = estimate.2, data = session.2)
```

```
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.048464 -0.013775 -0.000248  0.013141  0.067527
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.0048669  0.2085749  -0.023  0.98143
## IMNF_VaMe      0.0054088  0.0029895   1.809  0.07344 .
## IMNF_ReFe      0.0030783  0.0015455   1.992  0.04915 *
## IMNF_VaLa     -0.0007416  0.0022013  -0.337  0.73692
## IMNF_GlMa      0.0026917  0.0016663   1.615  0.10943
## IMNF_ExLo     -0.0005685  0.0004926  -1.154  0.25125
## meanRMS_VaMe   0.4347418  0.2243678   1.938  0.05552 .
## meanRMS_ReFe  -0.1403848  0.1814956  -0.773  0.44108
## meanRMS_VaLa   0.3860620  0.2452583   1.574  0.11865
## meanRMS_GlMa  -0.1580737  0.1498855  -1.055  0.29416
## meanRMS_ExLo   0.0798519  0.1224228   0.652  0.51574
## aucRMS_VaMe   -0.2076387  0.0834057  -2.490  0.01446 *
## aucRMS_ReFe    0.0759090  0.0797054   0.952  0.34323
## aucRMS_VaLa   -0.2207959  0.0851700  -2.592  0.01097 *
## aucRMS_GlMa    0.0629612  0.0602109   1.046  0.29826
## aucRMS_ExLo   -0.0499349  0.0441136  -1.132  0.26038
## maxRMS_VaMe    0.0257725  0.0214830   1.200  0.23313
## maxRMS_ReFe   -0.0041619  0.0025202  -1.651  0.10183
## maxRMS_VaLa    0.0494345  0.0262496   1.883  0.06260 .
## maxRMS_GlMa    0.0126798  0.0191625   0.662  0.50970
## maxRMS_ExLo    0.0251768  0.0213150   1.181  0.24036
## t2maxRMS_VaMe -0.0004386  0.0096199  -0.046  0.96372
## t2maxRMS_ReFe  0.0349433  0.0123082   2.839  0.00549 **
## t2maxRMS_VaLa  0.0176830  0.0095295   1.856  0.06649 .
## t2maxRMS_GlMa -0.0041935  0.0163828  -0.256  0.79851
## t2maxRMS_ExLo  0.0072629  0.0048502   1.497  0.13745
## ID02HaVi      -0.0461015  0.0623552  -0.739  0.46145
## ID03MaLu      -0.0618722  0.0609844  -1.015  0.31279
## ID04GiMa      -0.1741246  0.0404280  -4.307  3.9e-05 ***
## ID05CaEn      -0.1023737  0.0359559  -2.847  0.00536 **
## ID06AbPa      -0.1945638  0.0497681  -3.909  0.00017 ***
## ID07BoLa      -0.1417831  0.0522688  -2.713  0.00787 **
## set           -0.0056544  0.0034400  -1.644  0.10341
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02501 on 99 degrees of freedom
## Multiple R-squared:  0.8637, Adjusted R-squared:  0.8196
## F-statistic:  19.6 on 32 and 99 DF,  p-value: < 2.2e-16


##
## Call:
## lm(formula = estimate.updated, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.11051 -0.02101  0.00017  0.01770  0.11941
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.8261836  0.0712115  11.602  < 2e-16 ***
## IMNF_ReFe    0.0028019  0.0009487   2.953 0.003777 **
## aucRMS_VaMe -0.1292508  0.0218175  -5.924 3.02e-08 ***
## aucRMS_VaLa -0.1288125  0.0241442  -5.335 4.52e-07 ***
## ID02HaVi    -0.1888654  0.0412448  -4.579 1.14e-05 ***
## ID03MaLu    -0.0947900  0.0225143  -4.210 4.93e-05 ***
## ID04GiMa    -0.1352683  0.0199606  -6.777 4.75e-10 ***
## ID05CaEn    -0.0919314  0.0194015  -4.738 5.93e-06 ***
## ID06AbPa    -0.1577876  0.0243752  -6.473 2.15e-09 ***
## ID07BoLa    -0.0838672  0.0225107  -3.726 0.000297 ***
## set         -0.0099623  0.0031922  -3.121 0.002256 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0346 on 121 degrees of freedom
## Multiple R-squared:  0.681,  Adjusted R-squared:  0.6546
## F-statistic: 25.83 on 10 and 121 DF,  p-value: < 2.2e-16
```

This time we lost a lot of variance explanation by suppressing the values because we went from a R2 at 0.82 to a R2 at 0.65. Therefore, we will do the sorting in two steps : before we suppress all the pvalues inferior to 0.10, them we will make a final sorting with the pvalues < 0.05.

```
##
## Call:
## lm(formula = estimate.updated, data = data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.061097 -0.016668 -0.001631  0.012369  0.071315
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.049385   0.127831   0.386  0.69996
## IMNF_VaMe      0.006438   0.002221   2.898  0.00449 **
## IMNF_ReFe      0.003635   0.001264   2.875  0.00481 **
## meanRMS_VaMe   0.577872   0.073806   7.830 2.55e-12 ***
## aucRMS_VaMe   -0.230686   0.025406  -9.080 3.39e-15 ***
## aucRMS_VaLa   -0.093235   0.027622  -3.375  0.00100 **
## maxRMS_VaLa    0.073788   0.022754   3.243  0.00155 **
## t2maxRMS_ReFe  0.022482   0.008305   2.707  0.00782 **
## t2maxRMS_VaLa  0.023151   0.008885   2.606  0.01037 *
## ID02HaVi      -0.046378   0.043083  -1.076  0.28395
## ID03MaLu      -0.096208   0.019819  -4.854 3.80e-06 ***
## ID04GiMa      -0.180581   0.020403  -8.851 1.16e-14 ***
## ID05CaEn      -0.142097   0.016225  -8.758 1.90e-14 ***
## ID06AbPa      -0.196329   0.021213  -9.255 1.32e-15 ***
## ID07BoLa      -0.121092   0.025778  -4.697 7.29e-06 ***
## set           -0.008028   0.002397  -3.349  0.00110 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.02554 on 116 degrees of freedom
## Multiple R-squared:  0.8333, Adjusted R-squared:  0.8118
## F-statistic: 38.67 on 15 and 116 DF,  p-value: < 2.2e-16
```

Thanks to this one more step, we managed to get back to a R2 of 0.81.

4.3 Conclusion

Using the `estimate.2` and splitting the data set by session improves the accuracy of the model's prediction.

However

Using a linear model may not be the best choice due to the presence of multicollinearity, as indicated by the warning message: "Warning: prediction from a rank-deficient fit may be misleading." This warning suggests that there is a high degree of collinearity among the independent variables in your model.

Collinearity creates a problem in estimating the coefficients of the model, making them unstable and susceptible to small changes in the data. Consequently, the predictions from such a model can be unreliable and potentially misleading.

Therefore, we are going to try other models, if they are as precise or more precise we will use them instead of the linear model.

5 Linear mixed models

First, we need to build a linear mixed model by taking into account set and ID as random effects. We make the assumption that intercept and slope can be modified by the subject who performs the squat, and the number of sets performed before.

Now, let's fit it on the session 1

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: estimate.lmer
##    Data: session.1
##
## REML criterion at convergence: -1652.5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.9307 -0.5740  0.0182  0.6050  3.0570
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  set:ID   (Intercept) 0.0008593 0.02931
##  ID       (Intercept) 0.0023161 0.04813
##  Residual             0.0011378 0.03373
## Number of obs: 487, groups:  set:ID, 36; ID, 7
##
## Fixed effects:
##                 Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)    2.700e-01  1.112e-01  3.639e+02   2.428 0.015677 *
## IMNF_VaMe      3.756e-03  1.108e-03  4.372e+02   3.391 0.000760 ***
## IMNF_ReFe     -5.212e-05  5.427e-04  4.042e+02  -0.096 0.923539
## IMNF_VaLa      2.180e-03  9.137e-04  1.673e+02   2.386 0.018136 *
## IMNF_GlMa      2.318e-03  9.769e-04  4.478e+02   2.373 0.018055 *
## IMNF_ExLo      9.678e-04  7.524e-04  4.124e+02   1.286 0.199064
## meanRMS_VaMe   3.743e-01  1.831e-01  4.605e+02   2.044 0.041530 *
## meanRMS_ReFe  -9.856e-02  1.177e-01  4.556e+02  -0.838 0.402692
## meanRMS_VaLa  -2.768e-02  1.509e-01  4.555e+02  -0.183 0.854600
## meanRMS_GlMa   2.160e-01  1.068e-01  4.505e+02   2.022 0.043731 *
## meanRMS_ExLo  -2.279e-01  6.590e-02  4.562e+02  -3.458 0.000596 ***
## aucRMS_VaMe   -2.684e-01  1.031e-01  4.604e+02  -2.604 0.009508 **
## aucRMS_ReFe    1.154e-02  6.465e-02  4.581e+02   0.179 0.858356
## aucRMS_VaLa   -4.315e-02  8.038e-02  4.602e+02  -0.537 0.591648
## aucRMS_GlMa   -1.813e-01  6.152e-02  4.528e+02  -2.948 0.003365 **
## aucRMS_ExLo    4.597e-03  3.562e-02  4.468e+02   0.129 0.897357
## maxRMS_VaMe    1.505e-02  1.111e-02  4.603e+02   1.354 0.176246
## maxRMS_ReFe    2.994e-02  1.206e-02  4.493e+02   2.482 0.013419 *
## maxRMS_VaLa    3.496e-02  9.351e-03  4.585e+02   3.739 0.000208 ***
## maxRMS_GlMa    3.338e-02  1.160e-02  4.469e+02   2.877 0.004208 **
## maxRMS_ExLo    1.131e-01  2.241e-02  4.445e+02   5.046  6.6e-07 ***
## t2maxRMS_VaMe  2.099e-02  9.860e-03  4.388e+02   2.129 0.033796 *
## t2maxRMS_ReFe  1.146e-02  1.086e-02  4.404e+02   1.056 0.291733
## t2maxRMS_VaLa  3.604e-02  1.311e-02  4.549e+02   2.748 0.006235 **
## t2maxRMS_GlMa -5.181e-02  1.987e-02  4.476e+02  -2.607 0.009439 **
## t2maxRMS_ExLo  3.172e-02  9.282e-03  4.468e+02   3.418 0.000689 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 26 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

And for the session 2

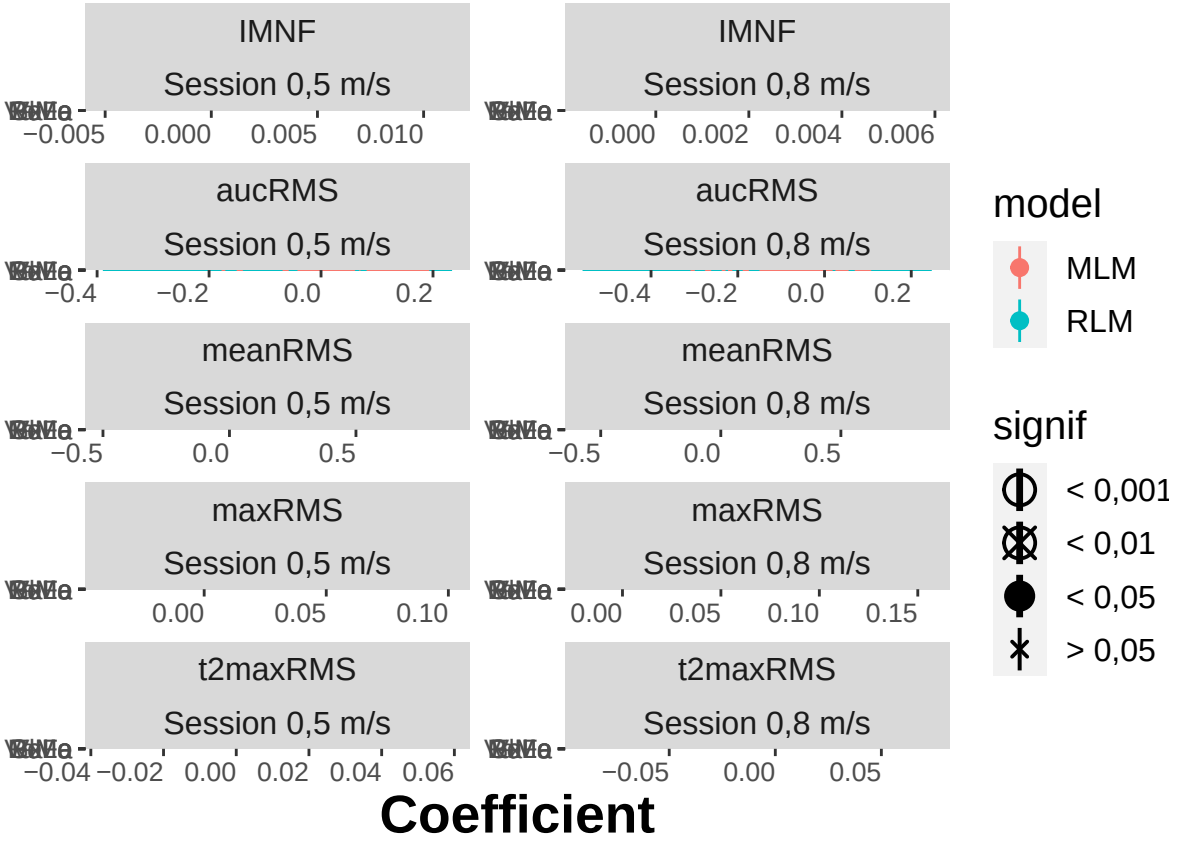
```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: estimate.lmer
##    Data: session.2
##
## REML criterion at convergence: -422.2
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.28493 -0.56593 -0.03351  0.56350  1.80936
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  set:ID   (Intercept) 0.0010683 0.03269
##  ID       (Intercept) 0.0035540 0.05962
##  Residual             0.0003669 0.01915
## Number of obs: 132, groups:  set:ID, 25; ID, 7
##
## Fixed effects:
##                 Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)    0.0559178  0.1610212 84.0023370   0.347  0.72926
## IMNF_VaMe      0.0031102  0.0022360 67.5827338   1.391  0.16880
## IMNF_ReFe      0.0005891  0.0014838 99.1062531   0.397  0.69221
## IMNF_VaLa      0.0004834  0.0017708 70.0729059   0.273  0.78567
## IMNF_GlMa      0.0024178  0.0012877 70.0381167   1.878  0.06461 .
## IMNF_ExLo      0.0017741  0.0006121 88.6511113   2.898  0.00473 **
## meanRMS_VaMe   0.1623643  0.1981257 89.4702092   0.820  0.41468
## meanRMS_ReFe  -0.1170344  0.1580390 86.7998597  -0.741  0.46097
## meanRMS_VaLa   0.3483091  0.1987805 86.9195129   1.752  0.08326 .
## meanRMS_GlMa  -0.0279073  0.1284053 85.0304619  -0.217  0.82847
## meanRMS_ExLo   0.1015632  0.1099498 87.8170910   0.924  0.35816
## aucRMS_VaMe   -0.1450479  0.0731718 87.9673118  -1.982  0.05057 .
## aucRMS_ReFe    0.0550607  0.0697237 89.3091107   0.790  0.43180
## aucRMS_VaLa   -0.2020733  0.0731612 85.3118779  -2.762  0.00703 **
## aucRMS_GlMa    0.0277743  0.0540792 95.0303706   0.514  0.60873
## aucRMS_ExLo   -0.0632422  0.0392547 94.8305928  -1.611  0.11049
## maxRMS_VaMe    0.0217707  0.0173103 80.5133767   1.258  0.21215
## maxRMS_ReFe   -0.0024966  0.0024803 96.4831290  -1.007  0.31665
## maxRMS_VaLa    0.0474903  0.0209527 81.3959050   2.267  0.02607 *
## maxRMS_GlMa   -0.0092793  0.0162452 86.0131731  -0.571  0.56935
## maxRMS_ExLo    0.0455164  0.0190375 86.6690506   2.391  0.01897 *
## t2maxRMS_VaMe -0.0047169  0.0092789 92.2657056  -0.508  0.61242
## t2maxRMS_ReFe  0.0245134  0.0109292 96.2929876   2.243  0.02719 *
```

```
## t2maxRMS_VaLa  0.0190495  0.0078812 81.3213305   2.417  0.01788 *
## t2maxRMS_GlMa  0.0052364  0.0135666 79.0555709   0.386  0.70055
## t2maxRMS_ExLo  0.0055799  0.0042916 83.5831785   1.300  0.19711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 26 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

6 Compare the two linear model

Now that we have fitted the linear models (multiple regression and linear mixed models), we are going to visualize the coefficients. This way, it will be easier to interpret.



This graph is pretty big, I recommand visualizing it with 1000 width pts and 1500 height pts.

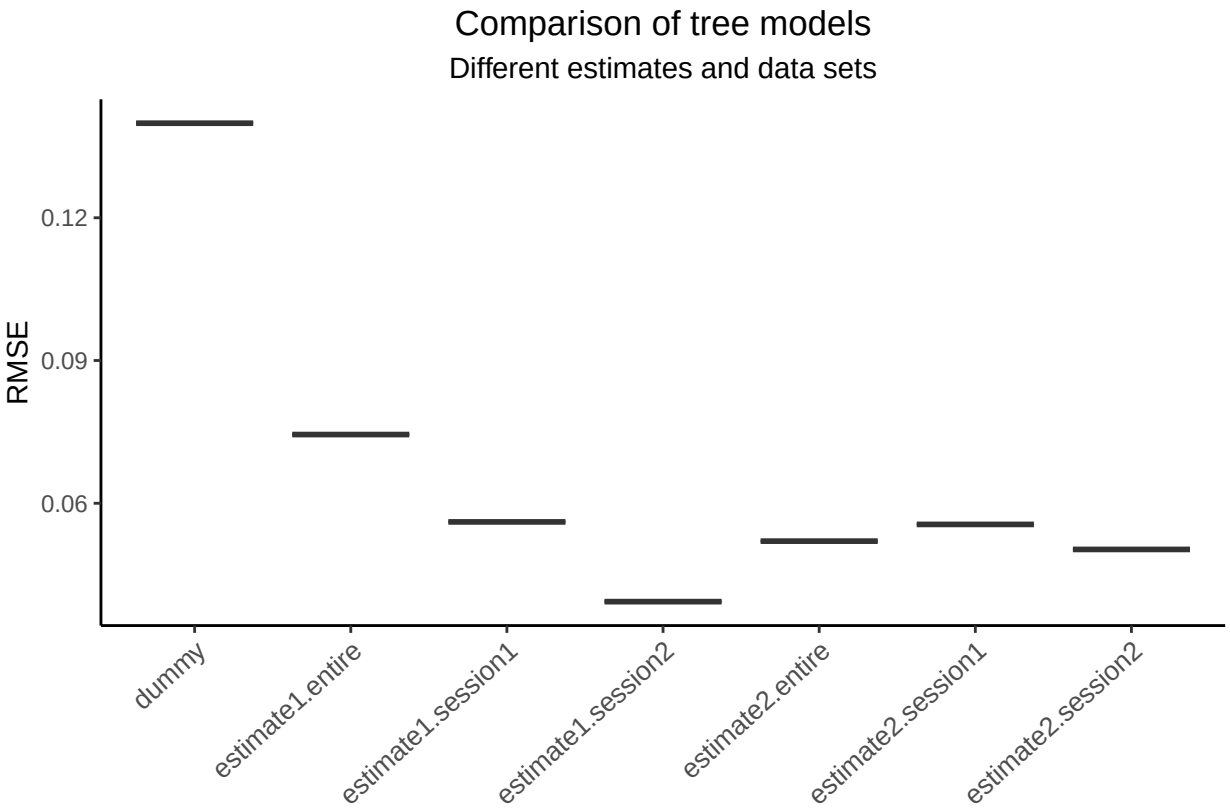
7 Regression trees

Regression tree partitions recursively the input data into subsets based on the values of the input feature. At each step, the algorithm selects a feature X_p and a split point that optimally divides the data, aiming to minimize the variability of the target variable Y (here MV) within each resulting subset.

7.1 Choose the parameters

First, we will compare the accuracy of the tree's prediction with the estimate model 1 (`estimate1`) vs. with the estimate model 2 (`estimate2`). Then, we are going to compare the performances of each linear model on the entire data set (`entire`,) then on each session individually (`session1` and `session2`).

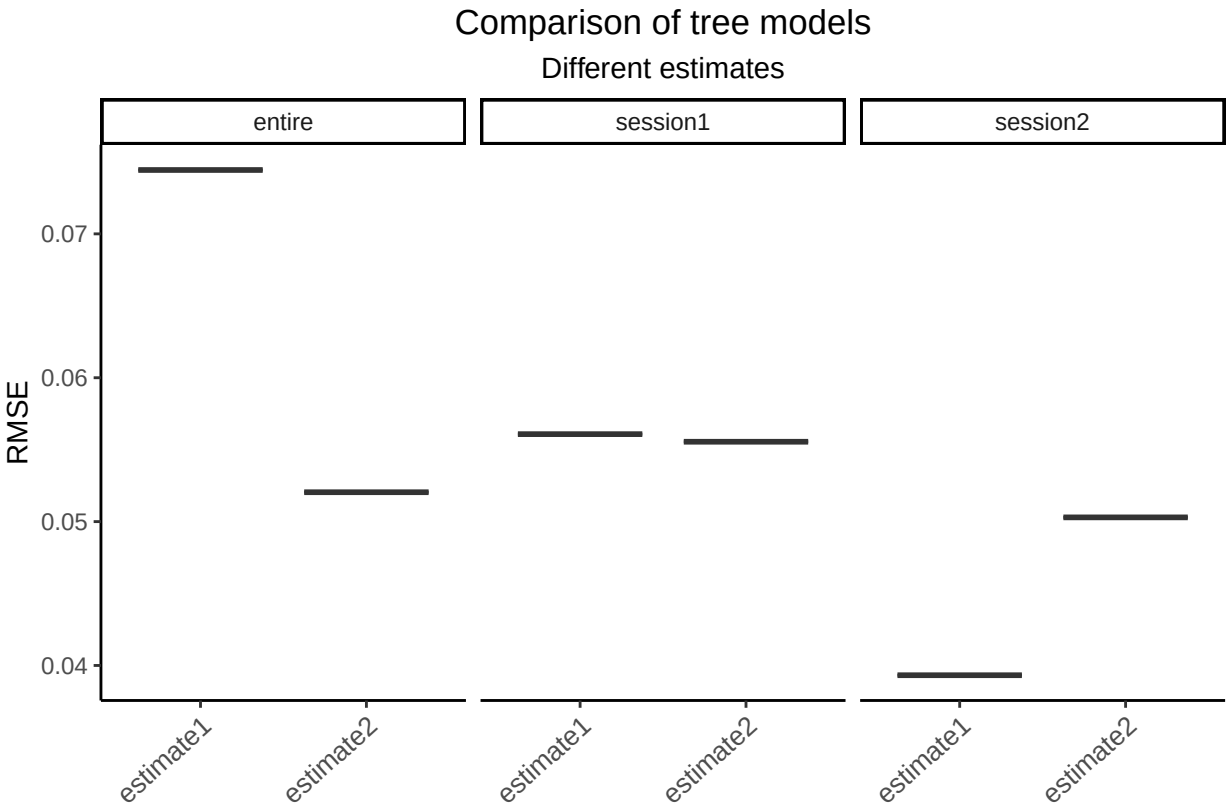
```
## Warning: Expected 2 pieces. Missing pieces filled with 'NA' in 1 rows [1].
```



All models allow a reliable prediction because they are more precise than the dummy prediction.

7.1.1 Estimate comparison

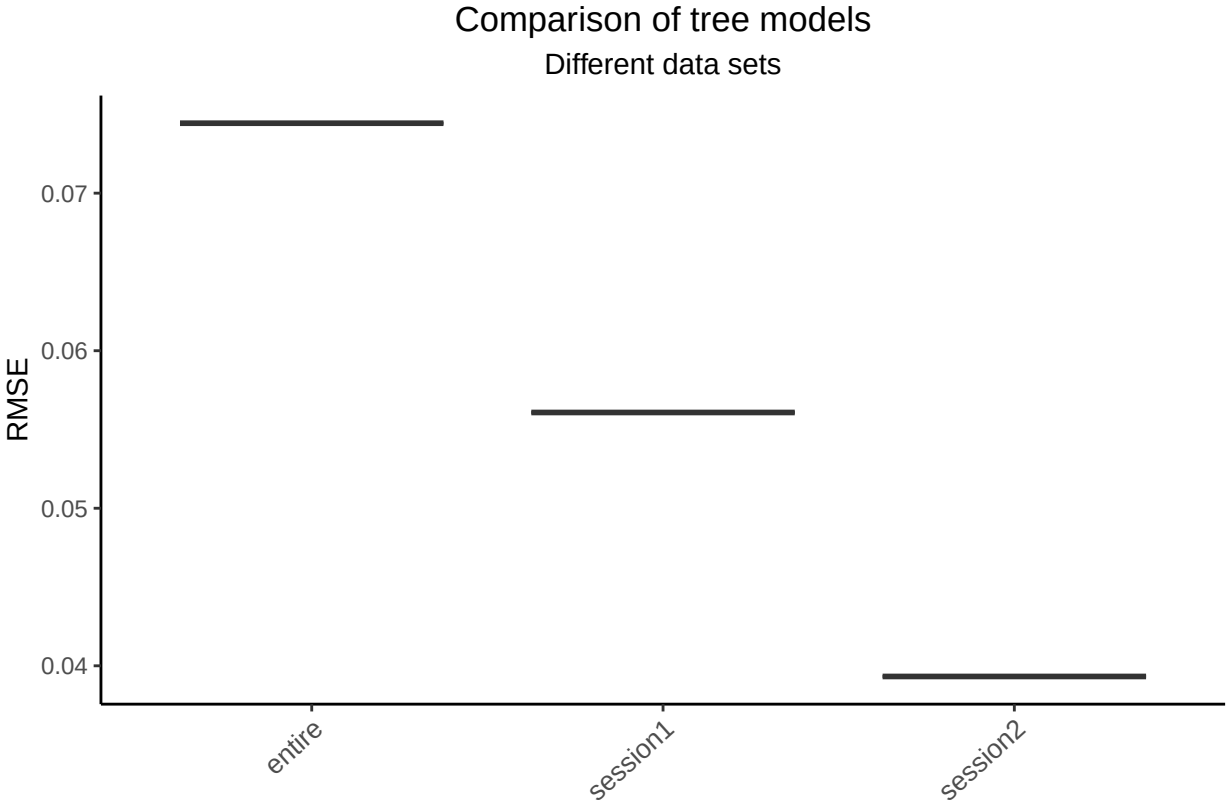
We are going to compare the models by estimate first.



Using the `estimate.2` improve the accuracy of the model.

7.1.2 Data set comparison

We are going to compare the accuracy of the model with the entire data set vs. with separate data set for each session.



Separating the data set by sessions seem to improve the accuracy of the model.

7.1.3 Complexity and number of examples

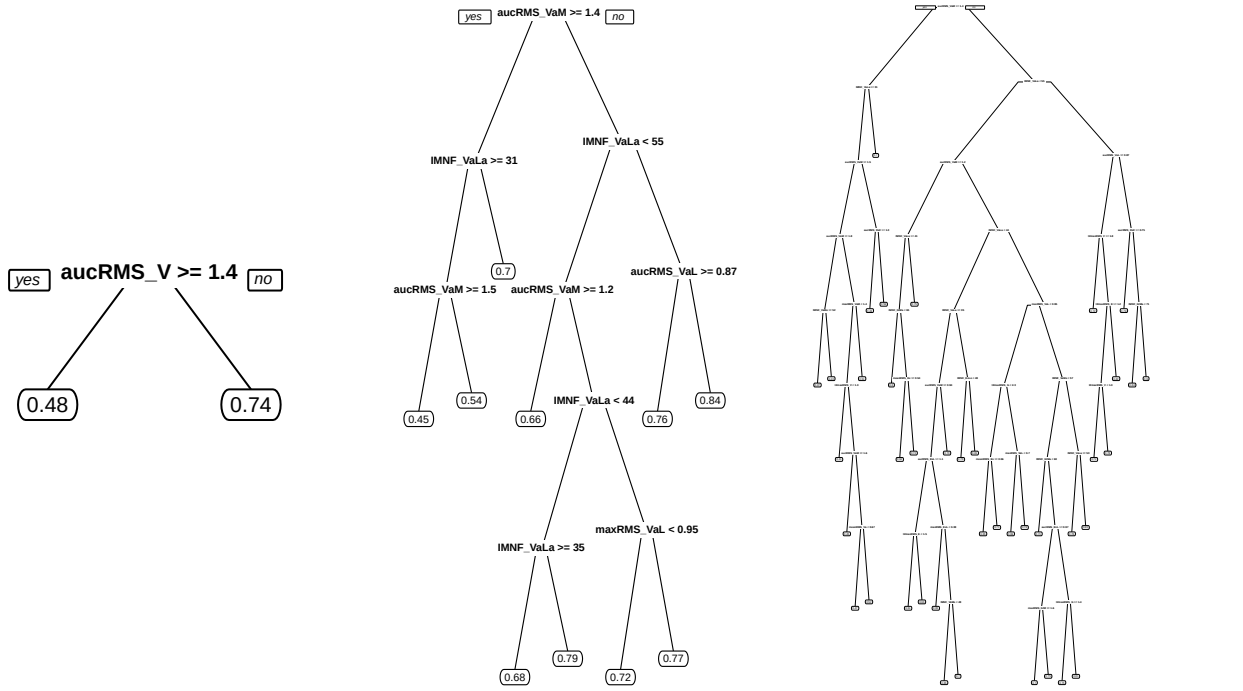
In addition, we will also try different complexity parameters and number of examples for the tree.

The complexity controls the number of splits, or branches, in the tree.

- A low complexity means fewer branch ; it allows an easier interpretation and it is less prone to over fitting, but the predictive accuracy is reduced
- A high complexity means more branch ; it may capture more intricate patterns, but is harder to interpret and more susceptible to over fitting

Let's see an example.

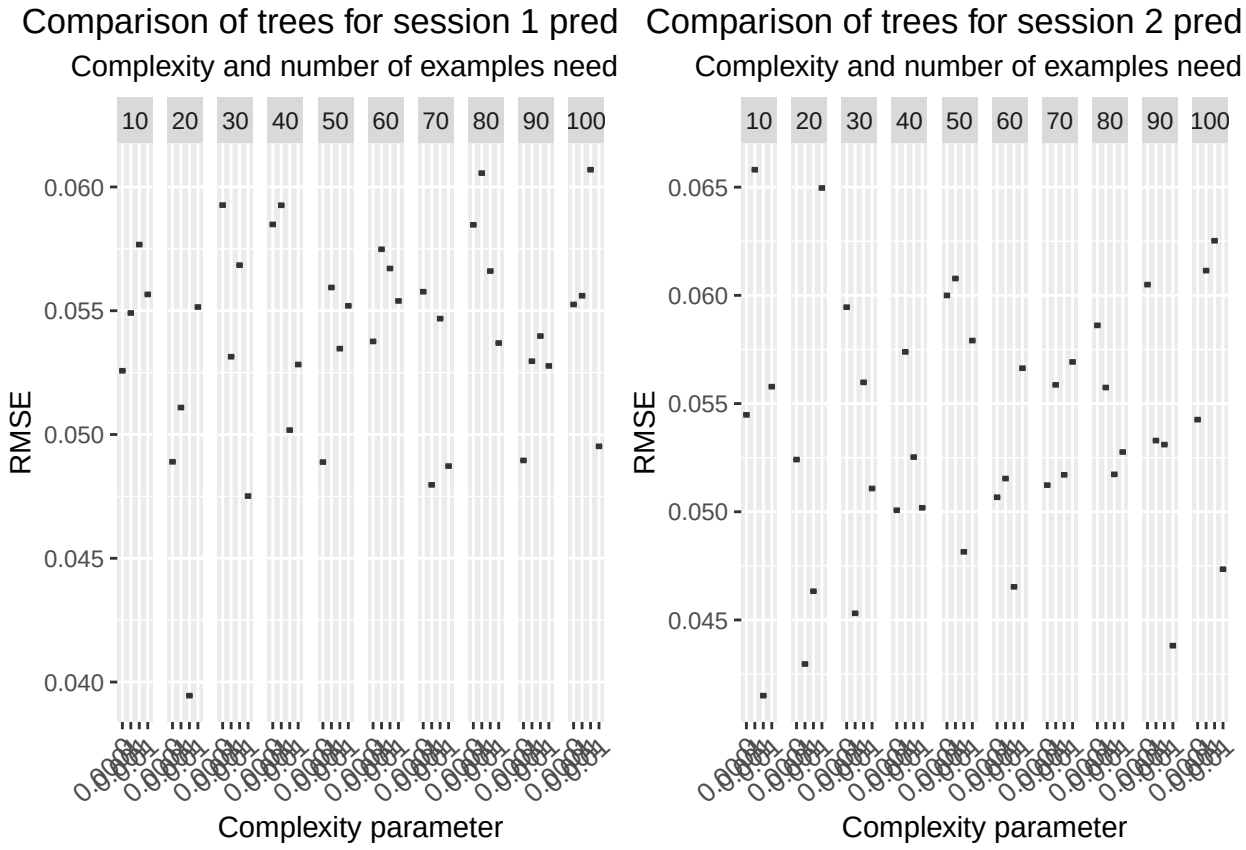
Low complexity **Medium complexity** **High complexity**



The number of examples controls the minimum number of observations that must exist in the data set in order for a split to be attempted.

Let's determine the optimal complexity parameter to predict our data sets. We will compare :

- Complexity parameters : 0, 0.0001, 0.001 and 0.01
- Number of examples : lies in [10, 20, ... 100]



Based on the graphs of the two sessions, we will chose `cp = 0.001` and `nb examples = 20`.

7.1.4 Conclusion

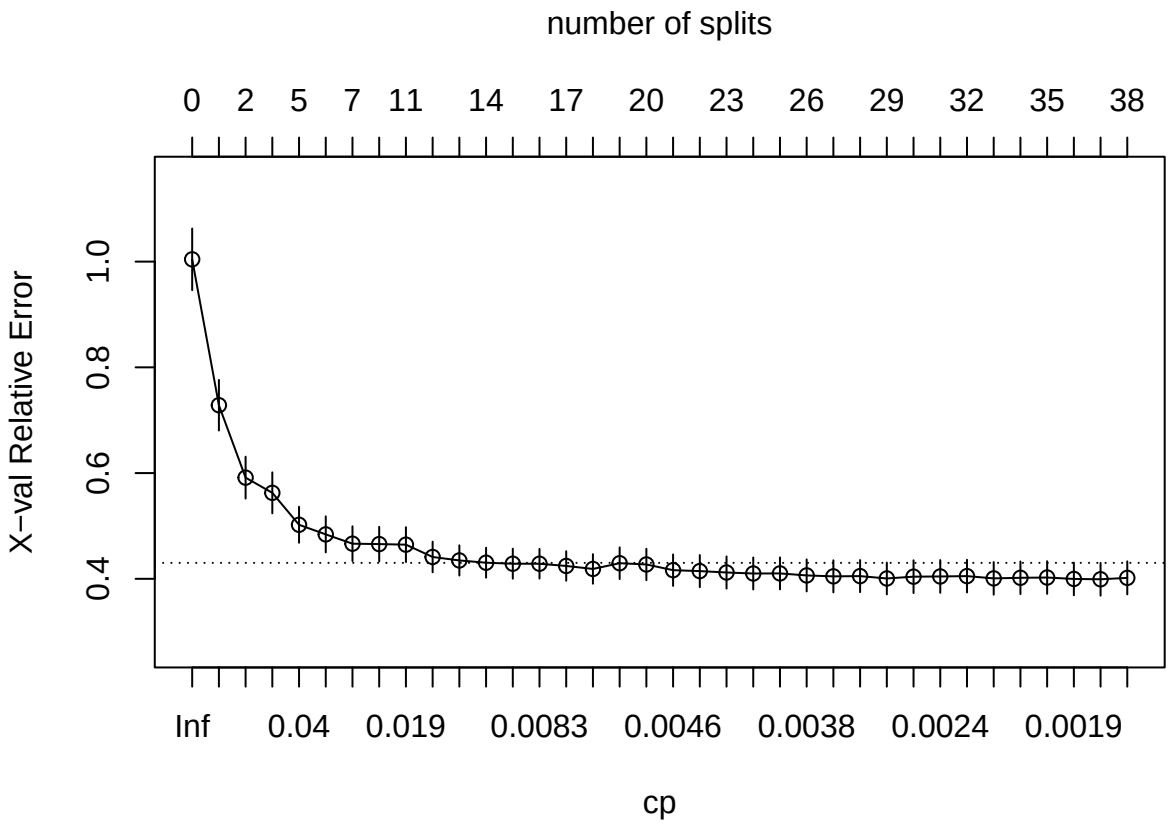
The chosen parameters are the following :

- The `estimate.2`
- Split the data sets by sessions
- The complexity is set at 0.001
- The number of examples needed to split a new branch is 20

7.2 Interpretation

Now, let's see what variables are the most useful to predict the MV with our data sets.

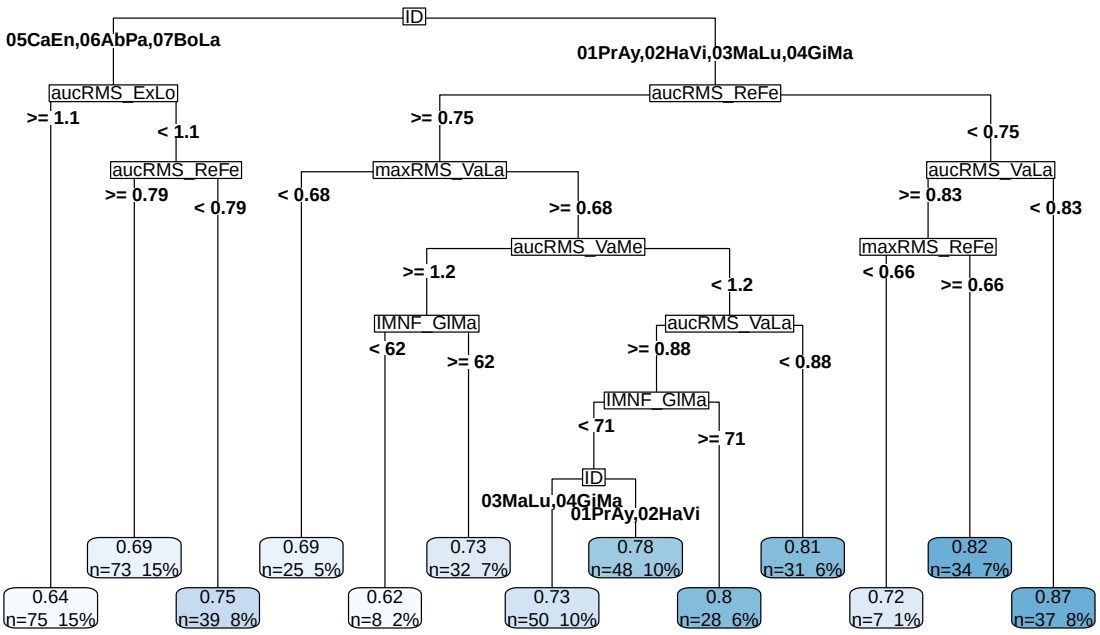
First for session 1



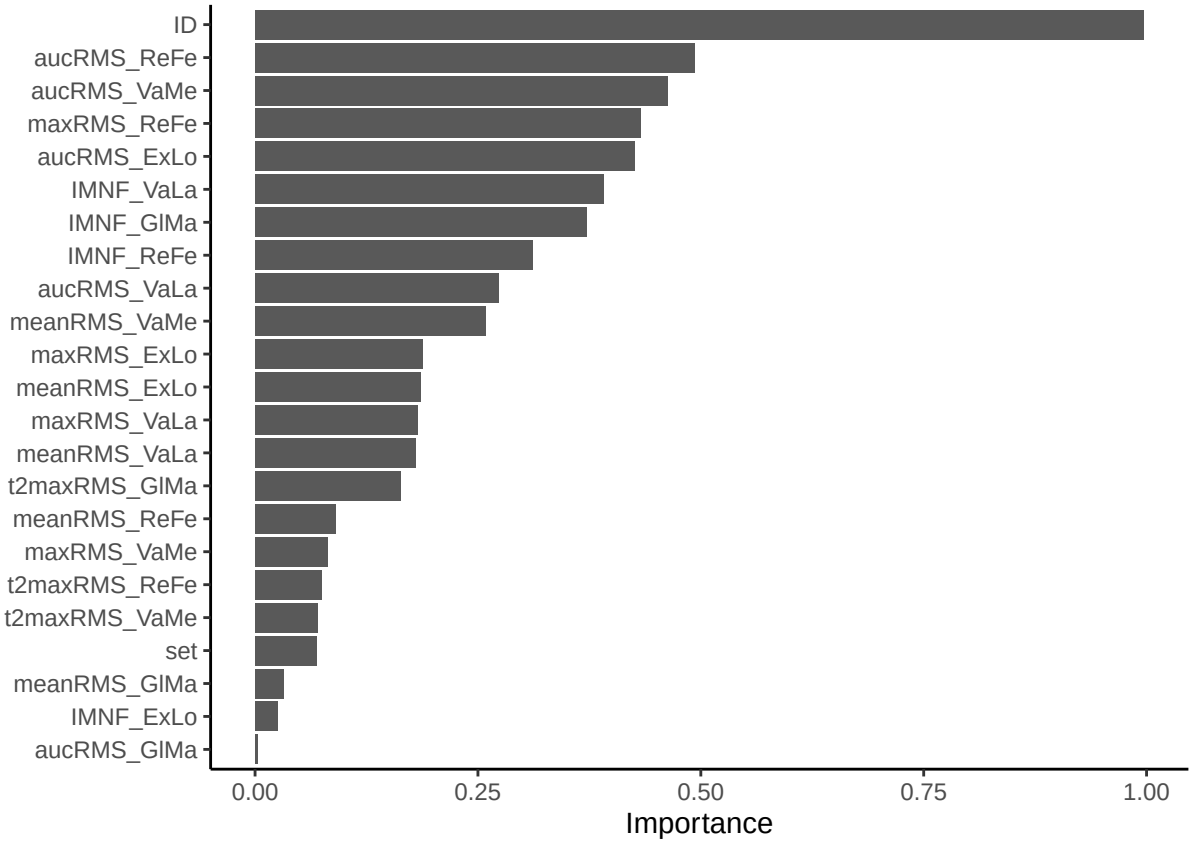
We will now prune the tree with the smallest relative SSE xerror to balance predictive power with simplicity.

The minimum error seem to be with cp = 0.001533046. However, the maximum CP under the dashed lineis at 4 splits. We will use this to prune the tree and then interpret it.

Session 1

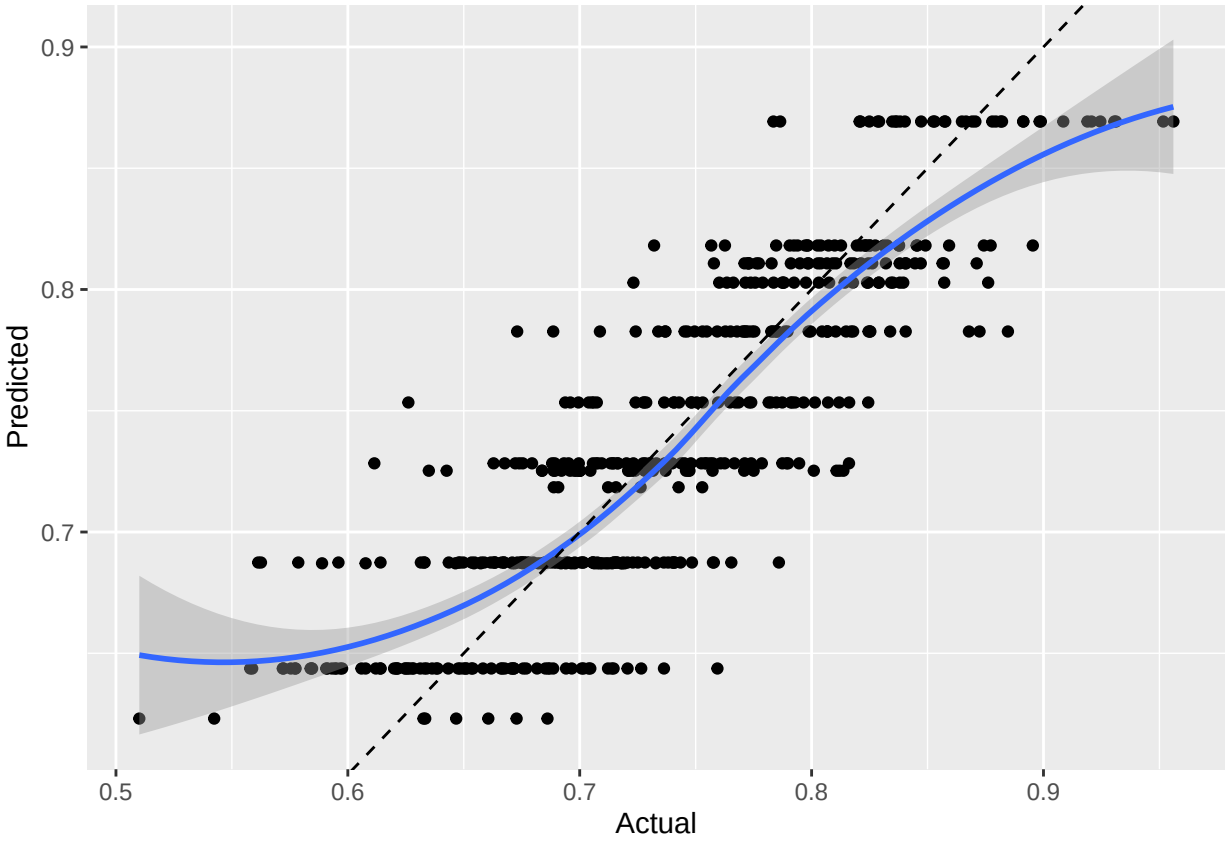


Now we cant see the variable importance for this tree

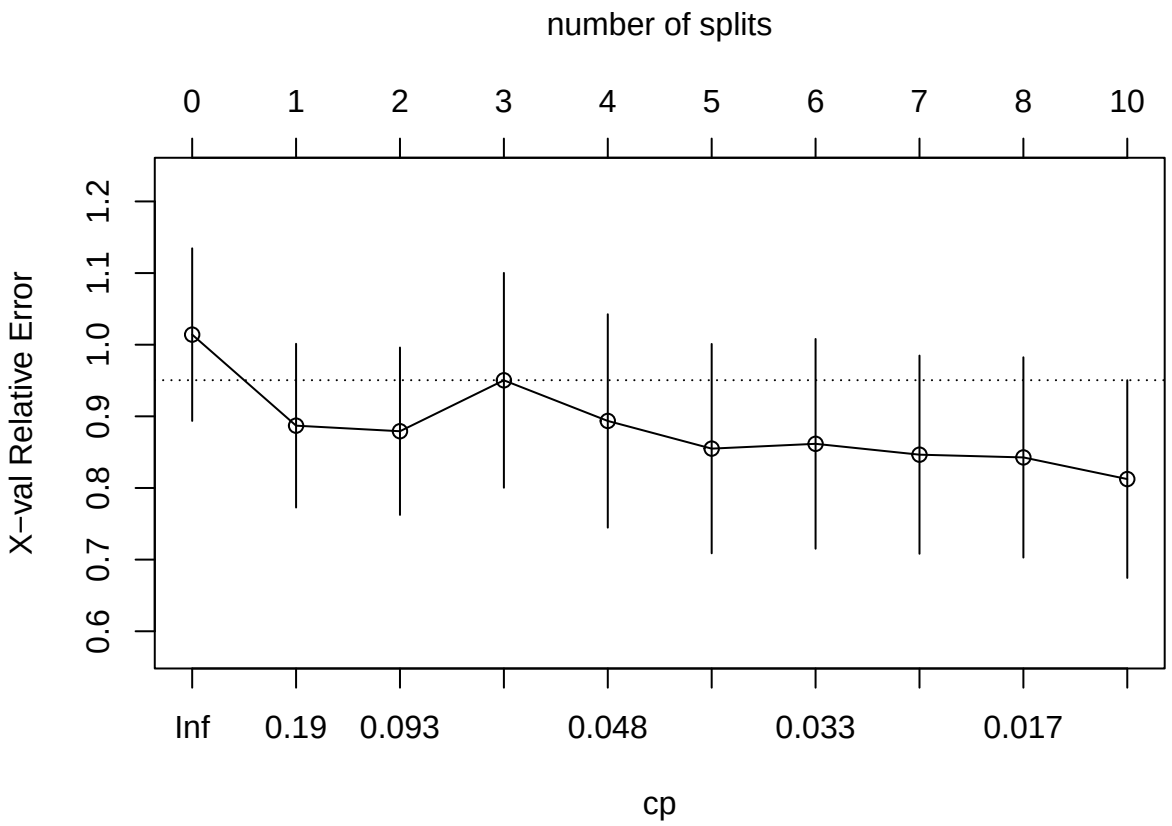


Now let's see the accuracy of the model

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



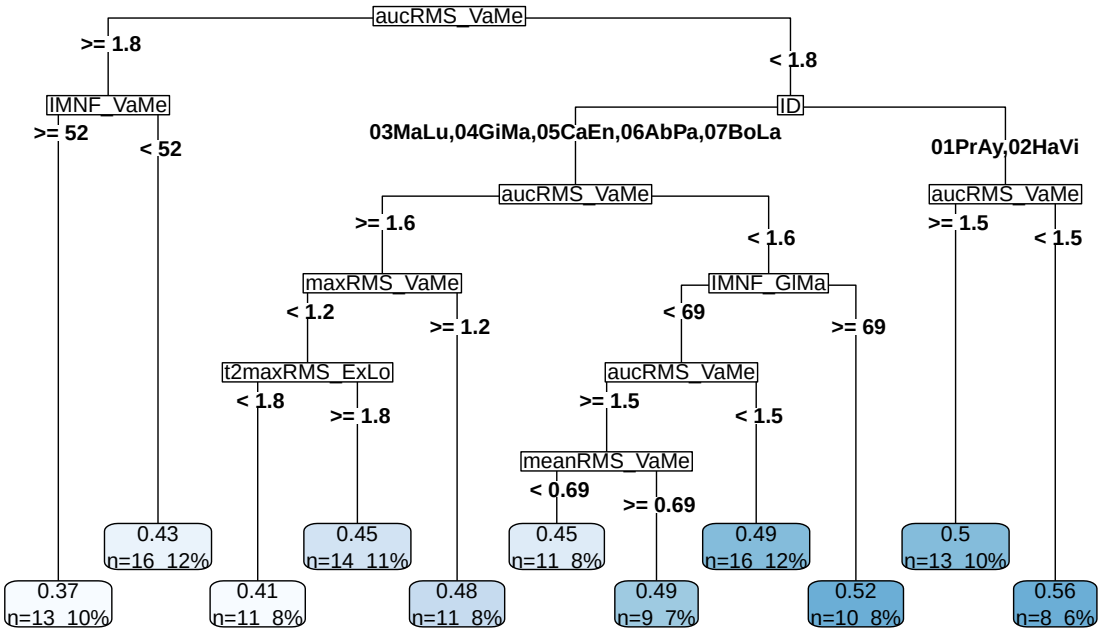
We repeat the same process for the session 2



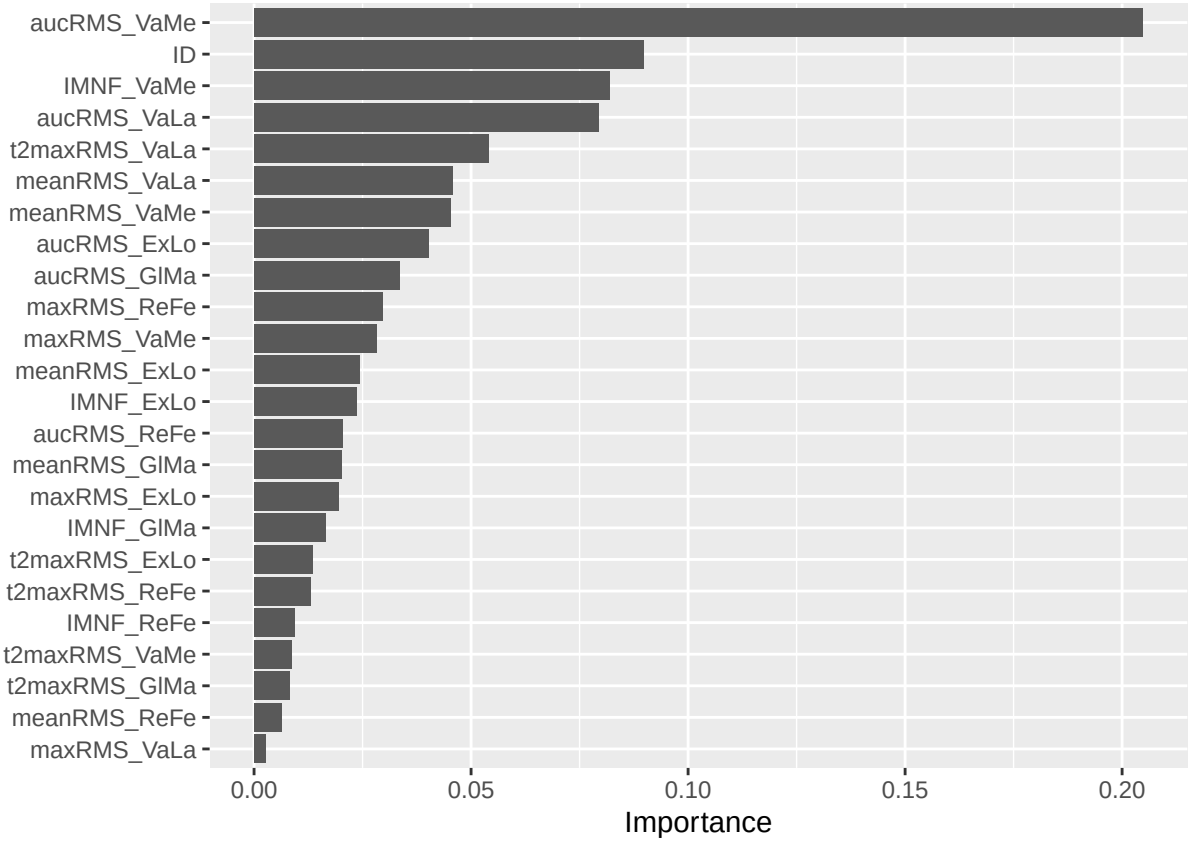
We will now prune the tree with the smallest relative SSE xerror to balance predictive power with simplicity.

The maximum CP under the dashed line is at 2 splits. However, this very low number must be due to the few data for this session. We will use the same number of splits as the session 1.

Session 2



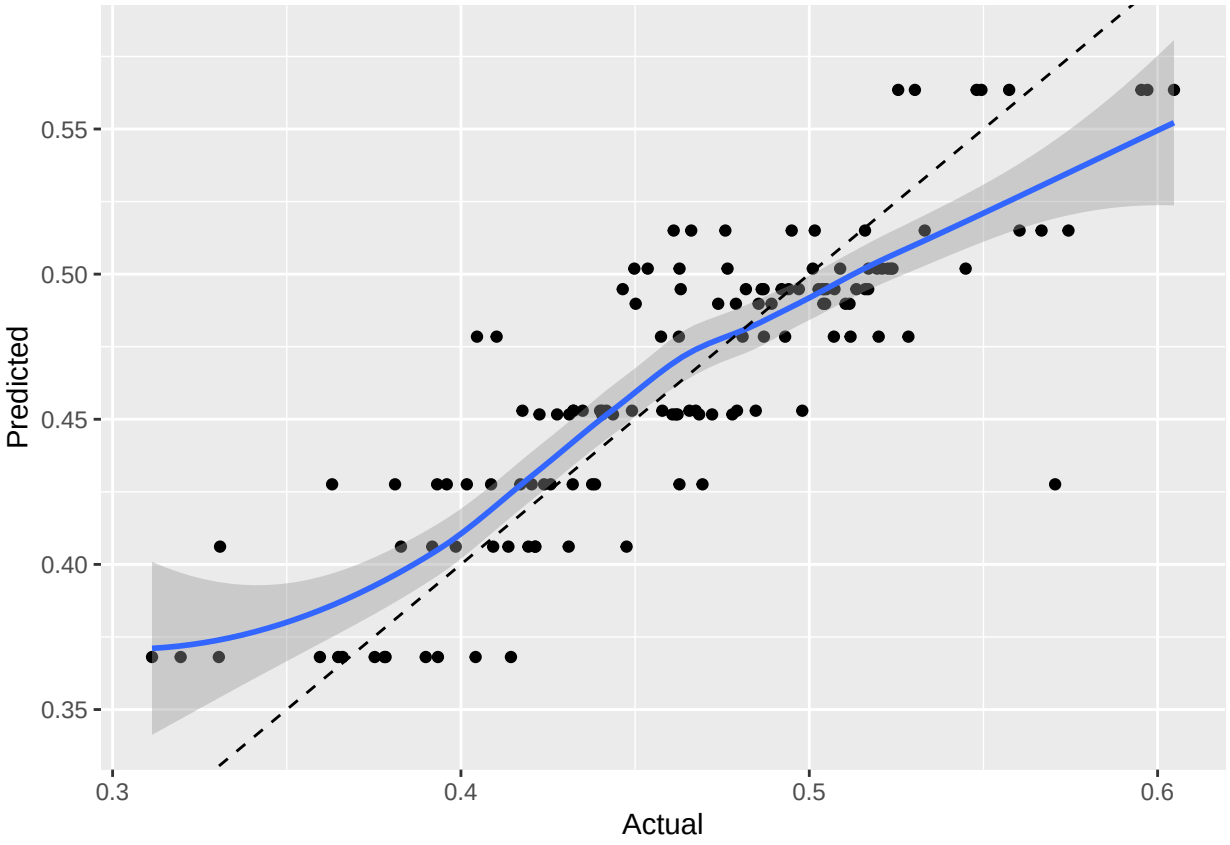
Now we can see the variable importance for this tree




```
##          Variable  Importance
## 1    aucRMS_VaMe 0.204805424
## 2             ID 0.089774432
## 3      IMNF_VaMe 0.081843324
## 4    aucRMS_VaLa 0.079405837
## 5  t2maxRMS_VaLa 0.053975715
## 6   meanRMS_VaLa 0.045849029
## 7   meanRMS_VaMe 0.045319150
## 8    aucRMS_ExLo 0.040300971
## 9    aucRMS_GlMa 0.033539462
## 10   maxRMS_ReFe 0.029562265
## 11   maxRMS_VaMe 0.028259896
## 12  meanRMS_ExLo 0.024324657
## 13     IMNF_ExLo 0.023713778
## 14   aucRMS_ReFe 0.020318145
## 15  meanRMS_GlMa 0.020228135
## 16   maxRMS_ExLo 0.019527888
## 17     IMNF_GlMa 0.016431546
## 18 t2maxRMS_ExLo 0.013510435
## 19 t2maxRMS_ReFe 0.012965559
## 20     IMNF_ReFe 0.009292954
## 21 t2maxRMS_VaMe 0.008597550
## 22 t2maxRMS_GlMa 0.008224820
## 23  meanRMS_ReFe 0.006385257
## 24   maxRMS_VaLa 0.002628589
```

Now let's evaluate the accuracy of the model

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



The model overestimates the low velocities and underestimates the high velocities

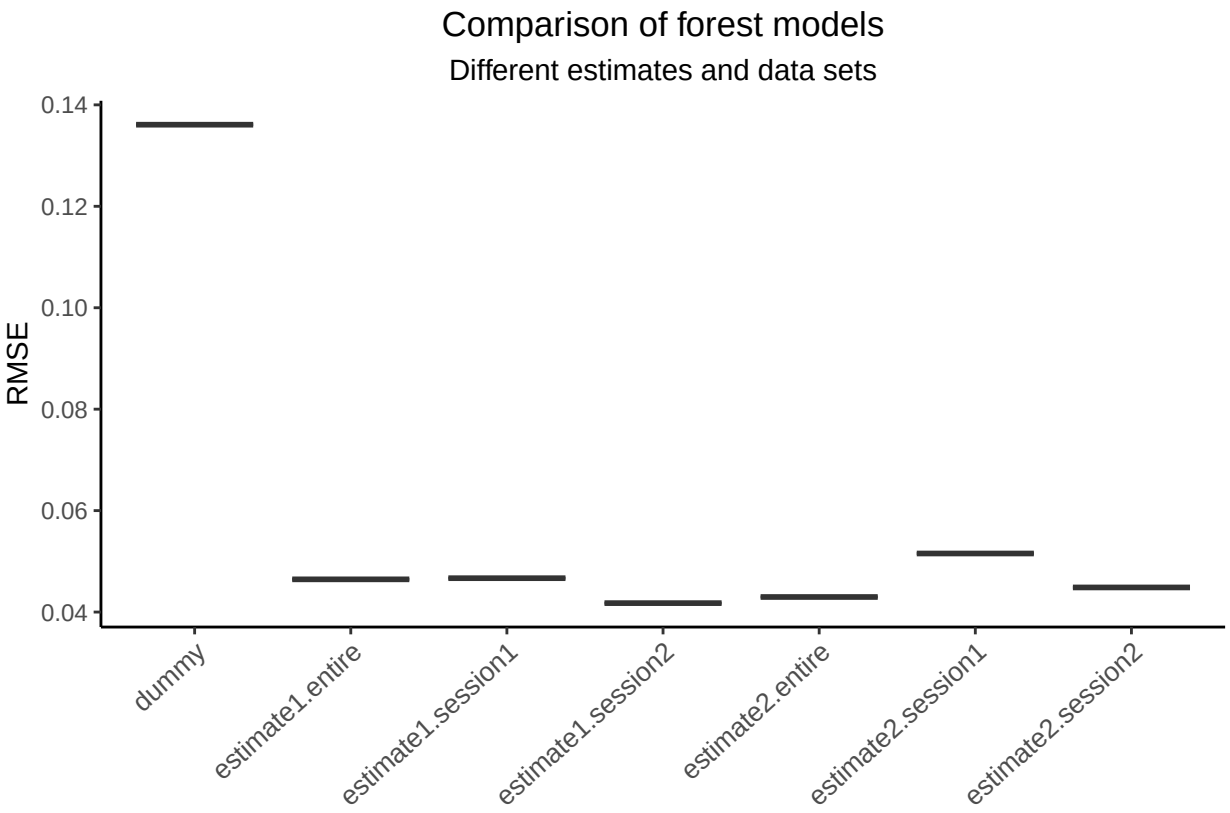
8 Random forest

A random forest combines multiple decision trees to make predictions. Each tree is made on bootstrapped training samples of the input data set ; and each tree is trained on a random sample of m predictors chosen as split of the full set of p predictors.

Therefore, each tree has only access to a part of the full training of the data set and a minority of the available. Then, each tree votes and the forest average those votes.

These characteristics allow a more reliable prediction of Y than with single trees (C. R. James et al., 2010).

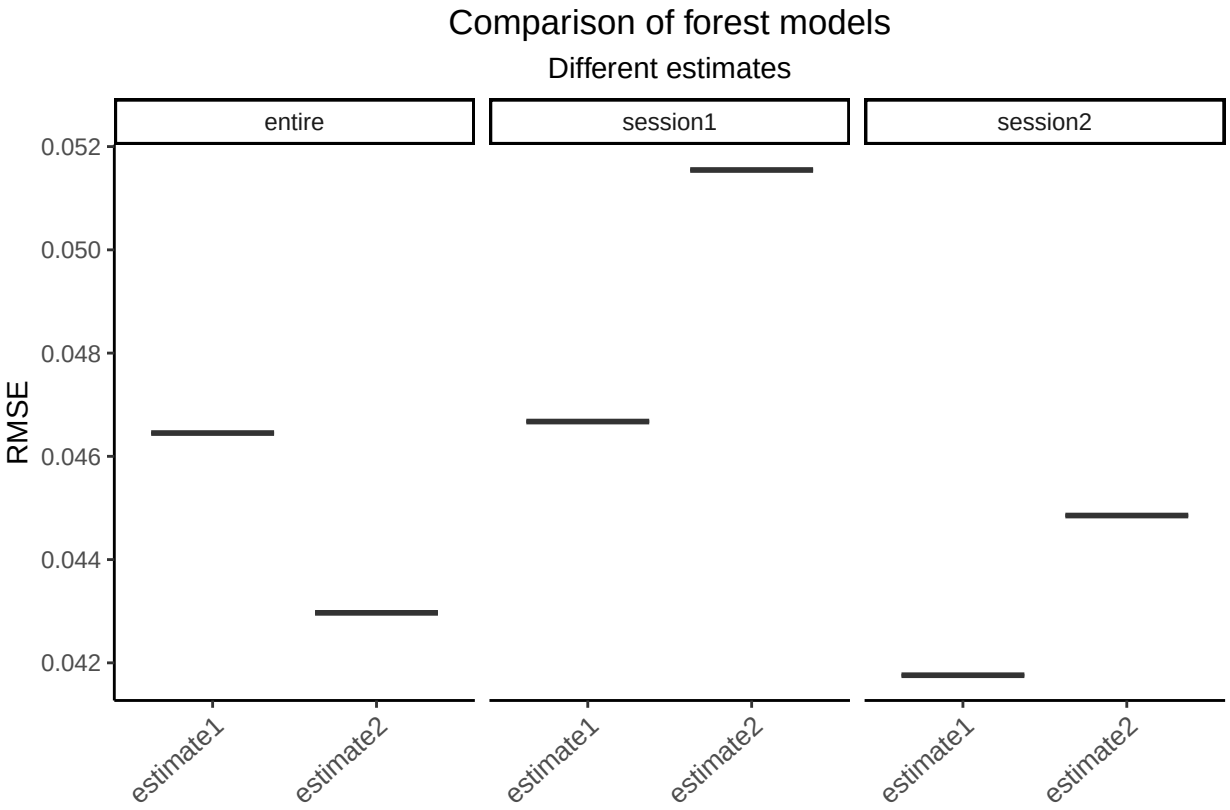
8.1 Choosing the parameters



All models allow a reliable prediction because they are more precise than the dummy prediction.

8.1.1 Estimate comparison

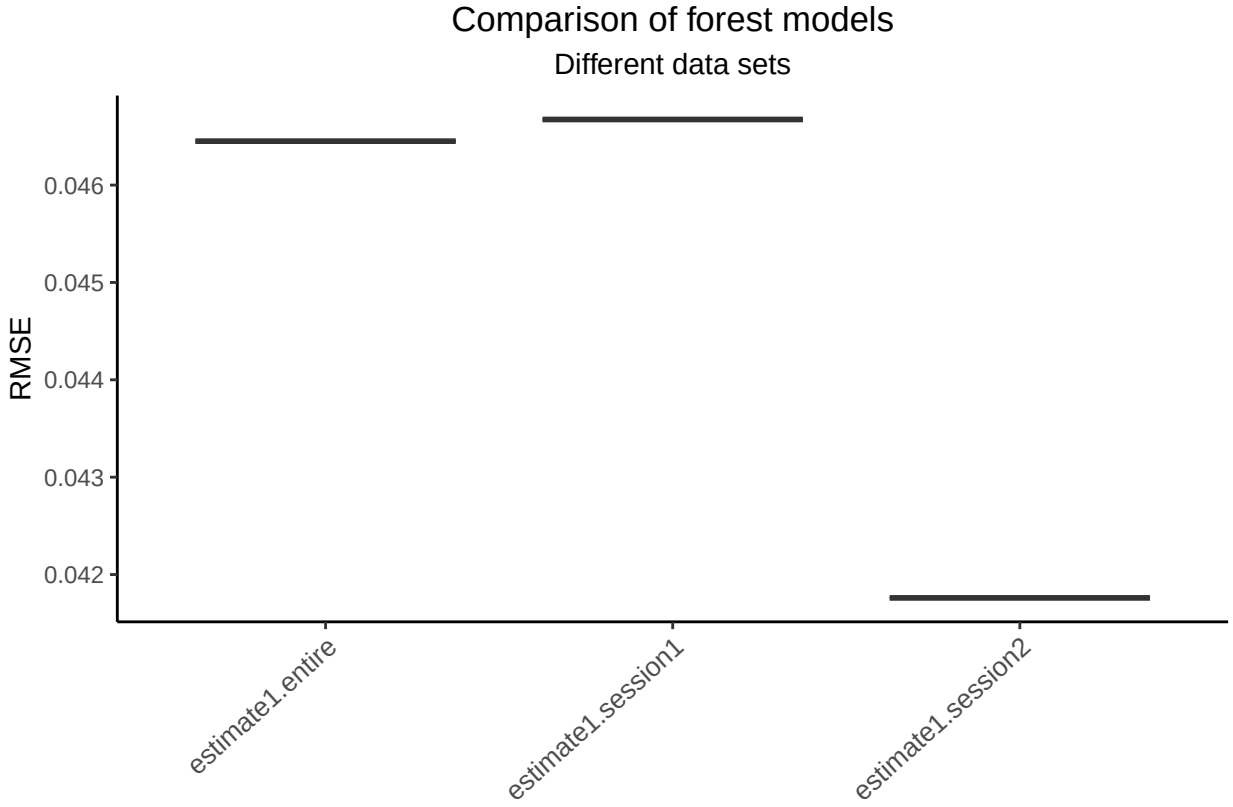
We are going to compare the models by estimate first.



Using the `estimate.2` improve the accuracy of the model.

8.1.2 Data set comparison

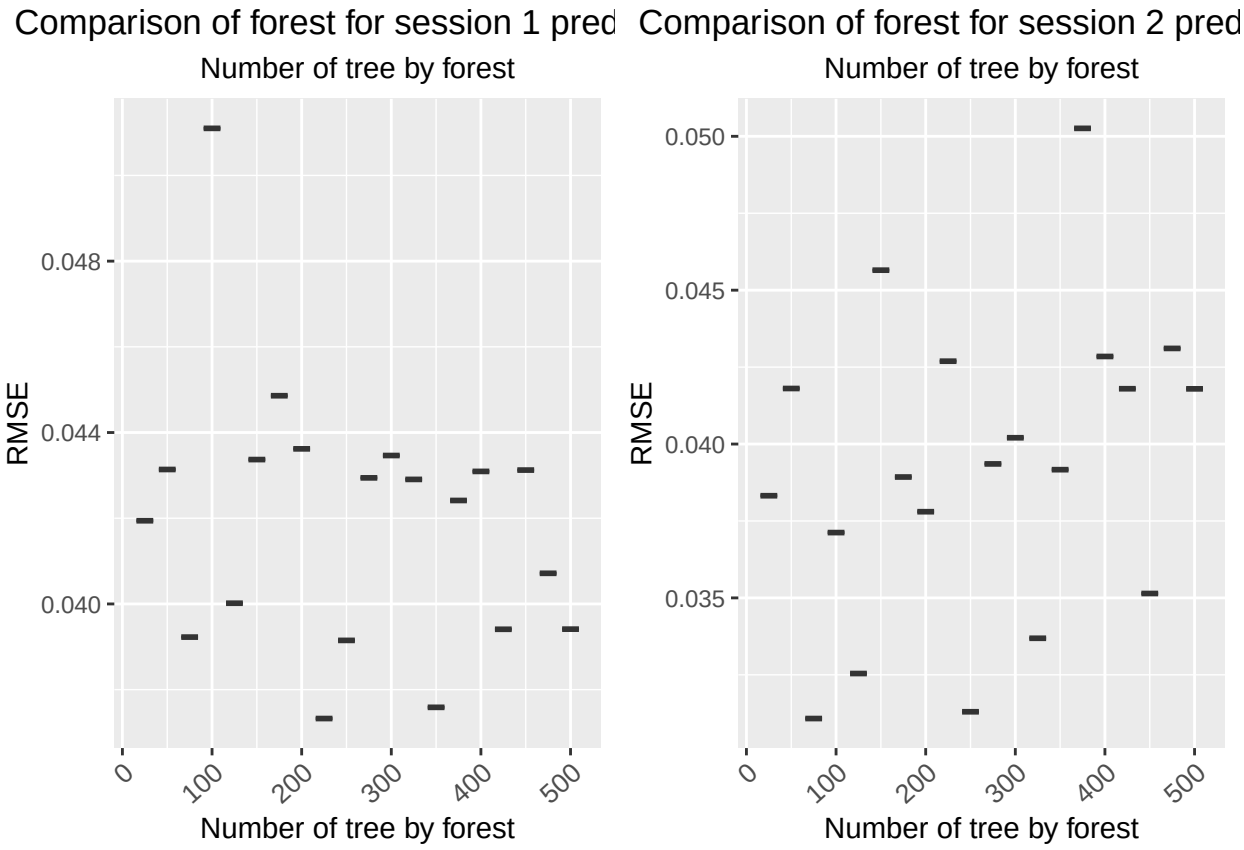
We are going to compare the accuracy of the model with the entire data set vs. with separate data set for each session.



Separating the data set by sessions seem to improve the accuracy of the model.

8.1.3 Number of tree by forests

Now, let's determine the optimal number of tree by forest to predict our data sets.



Choosing 300 trees seem to improve the accuracy of the model.

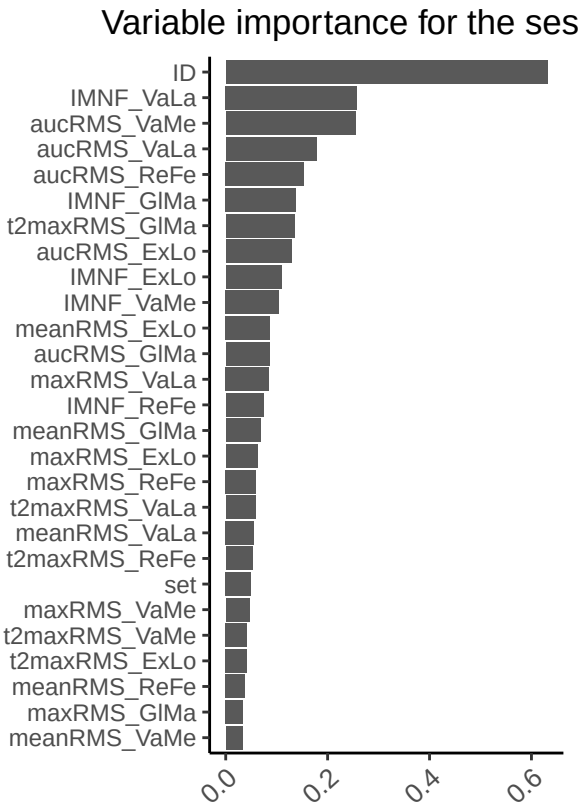
8.1.4 Conclusion

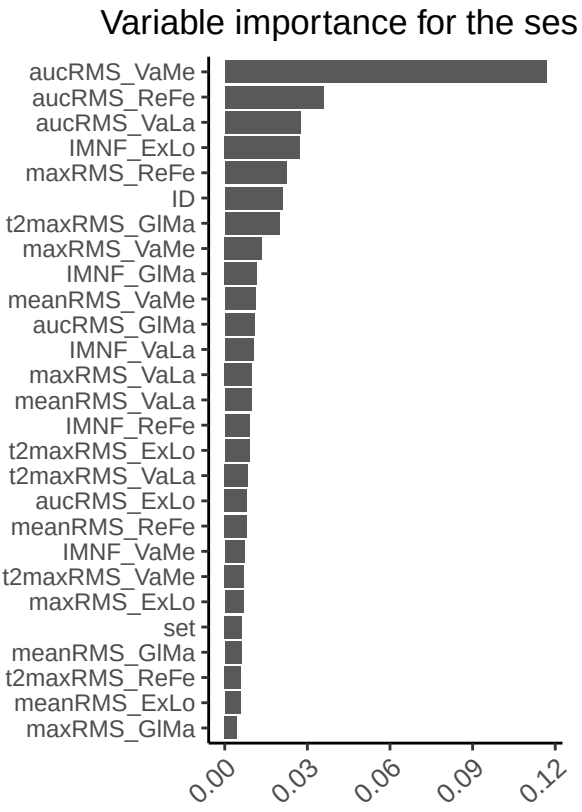
The chosen parameters are the following :

- We split the data set by session
- We include the subject ID, the set and the repetition numbers in the tree prediction
- The number of trees by forest is set at 300

8.2 Interpretations

Now, let's see what variables are the most useful to predict the MV with our data sets.



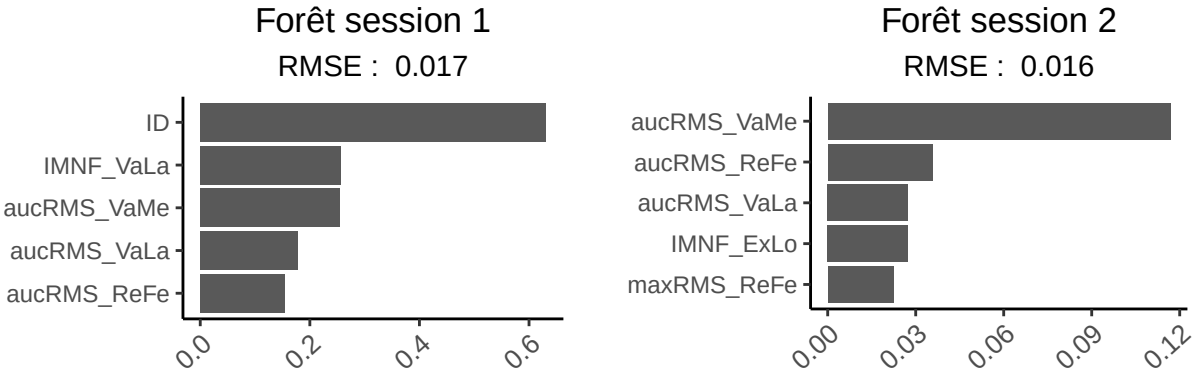



```
##                     variable       value
## aucRMS_VaMe     aucRMS_VaMe 0.116788857
## aucRMS_ReFe     aucRMS_ReFe 0.035793502
## aucRMS_VaLa     aucRMS_VaLa 0.027442323
## IMNF_ExLo         IMNF_ExLo 0.027397883
## maxRMS_ReFe     maxRMS_ReFe 0.022392570
## ID                       ID 0.020874018
## t2maxRMS_GlMa t2maxRMS_GlMa 0.019845721
## maxRMS_VaMe     maxRMS_VaMe 0.013514752
## IMNF_GlMa         IMNF_GlMa 0.011685433
## meanRMS_VaMe   meanRMS_VaMe 0.011242725
## aucRMS_GlMa     aucRMS_GlMa 0.010743023
## IMNF_VaLa         IMNF_VaLa 0.010465160
## maxRMS_VaLa     maxRMS_VaLa 0.009968798
## meanRMS_VaLa   meanRMS_VaLa 0.009796516
## IMNF_ReFe         IMNF_ReFe 0.009040298
## t2maxRMS_ExLo t2maxRMS_ExLo 0.009028684
## t2maxRMS_VaLa t2maxRMS_VaLa 0.008460426
## aucRMS_ExLo     aucRMS_ExLo 0.008154971
## meanRMS_ReFe   meanRMS_ReFe 0.008045102
## IMNF_VaMe         IMNF_VaMe 0.007252257
## t2maxRMS_VaMe t2maxRMS_VaMe 0.006999181
## maxRMS_ExLo     maxRMS_ExLo 0.006787443
## set                     set 0.006253227
```

```
## meanRMS_GlMa   meanRMS_GlMa 0.006003057
## t2maxRMS_ReFe t2maxRMS_ReFe 0.005878297
## meanRMS_ExLo   meanRMS_ExLo 0.005799193
## maxRMS_GlMa     maxRMS_GlMa 0.004420092
```

Once again, we observe different variable importance depending on the session we analyze.

As mentioned for the tree, we only keep the 5 more important variables to visualize.



The variable importance order is not the same compared to tree models.

9 Compare all the models

Now that we've established the best parameters for each model, let's compare the performance of each of them.

9.0.1 Statistical comparisons

We want to compare our models to see if their predicition is statistically different from one. We do that thanks to the Kruskall-Wallis test

For both session, there are differences between the models' accuracy. We are going to run pairwise comparisons to identify which model is more accurate.

Now we are going to viauslize this

9.0.2 Stats about the models

10 Compare the models to the regressions

References

Flanagan, E., & Jovanović, M. (2014). Researched Applications of Velocity Based Strength Training. *J. Australian Strength Cond.*, *22*, 58–69.

James, C. R., Scheuermann, B. W., & Smith, M. P. (2010). Effects of two neuromuscular fatigue protocols on landing performance. *Journal of Electromyography and Kinesiology*, *20*(4), 667–675. https://doi.org/10.1016/j.jelekin.2009.10.007

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning with Applications in R* (Second Edition).

Sánchez-Medina, L., & González-Badillo, J. J. (2011). Velocity Loss as an Indicator of Neuromuscular Fatigue during Resistance Training. *Medicine & Science in Sports & Exercise*, *43*(9), 1725–1734. https://doi.org/10.1249/MSS.0b013e318213f880

Teikari, P., & Pietrusz, A. (2021). *Precision strength training: Data-driven artificial intelligence approach to strength and conditioning*. SportRxiv. https://doi.org/10.31236/osf.io/w734a

Weakley, J., Mann, B., Banyard, H., McLaren, S., Scott, T., & Garcia-Ramos, A. (2021). Velocity-Based Training: From Theory to Application. *Strength & Conditioning Journal*, *43*(2), 31–49. https://doi.org/10.1519/SSC.0000000000000560