Capstone project: predicting speed skating training sessions

S van der Zwaard

2023-12-01

Introduction

Context of the problem

The Netherlands is world leading in speed skating and the next generation is getting ready to continue this high standard of international performance. The Netherlands is also a knowledge country, and to keep winning medals, we need to bring this scientific knowledge to practice!

In previous years, I have collaborated with multiple speed skating teams, coaches and embedded scientists. This context provided the unique opportunity to collect speed skating data from talented young skaters on the ice, for which they provided consent.

The coach has shared two seasons of training data, including external training load obtained from the position on the ice rink. This is **tracked using a transponder** that the speed skaters wear around their ankle. The data includes information about the skated distance, number of laps, mean speed and fastest lap. In addition, the coaches also send information for every training session via the phone (e.g. the type of the session). In addition to the transponder data, athletes also wear a heart rate monitor to capture their internal training load. The athletes have uploaded their heart rate (HR) data, and the embedded scientist already performed some pre-processing. Within the current project, I will wrangle the seasonal data that was provided by the speed skaters and help the coach to prescribe the right training sessions.

The coach prescribes interval training sessions and distinguishes these into either intensive or extensive sessions. In general, the coach expects the extensive sessions to be less intense than the intensive sessions, but he is not sure whether this is also the case for his speed skaters. His question therefore is: How does the internal training load differ between intensive and extensive interval sessions?

To help the coach to prescribe his training sessions the most effectively, he would like to know the expected type of a training session. Therefore, he would like to build a prediction model using the type of a training session as a target (i.e. extensive or intensive interval training sessions) and the internal training load parameters and personal characteristics as predictors. We will consider feature engineering, data partitioning, standardization and cross-validation when building the model and think how to evaluate the effectiveness of your model. Also, describe which features you included in the model. How would you use this information to advice the coach to optimize his training prescription?

In summary, the goal of this project is to discover new insights from the the data, use scientific knowledge to interpreted the data and provide relevant feedback on training monitoring and optimization to the coach! This way we can provide the young talented speed skaters with that little, but essential, information to give them the head start, so that the next generation of talented speed skaters is well prepared for the future Winter Olympics.

Scientific background

Providing the best preparation for speed skating athletes to their competitive races requires optimization of training sessions, periodization of training intensities over time. To improve performance, an athlete needs to train sufficient number of hours to allow for physiological adaptations. For the prescription of training, a combination of intensive and extensive training sessions is necessary to keep athletes in shape and at the same time avoid overtraining induced by repeating the same training intensities all the time [1], in which athletes cannot compete for months.

Extensive and intensive interval training are common forms of training for speed skaters. Both training methods use an intermittent training load with a certain amount of rest in between. Extensive interval training sessions consists of many larger intervals combined with short recovery periods, which enables improved endurance of the athlete [2]. Intensive interval training, on the other hand, is more intense and consists of less/shorter intervals at a higher intensity together with prolonged recovery periods in between interval sets, which facilitates the anaerobic capacity of the speed skater, i.e. the capacity to generate muscle power without the use of oxygen [2,3].

Coaches require feedback on their training prescription, such as by monitoring the training load of a specific session and to verify whether the intended training intensity was actually reached by the athletes. It allows for optimizing and adjusting the training program on an individual basis [4]. Typically, training load is evaluated in terms of external training load and internal training load [5]. External training load refers to the physiological work performed by the athlete in terms of the quantity, quality, and organization of exercise (e.g., measured by velocity, acceleration or power), whereas internal training load is defined as the psycho-physiological response to the external load during exercise (e.g., measured by heart rate or lactate production) [5,6]. It is advised to analyze both internal and external training load variables for sufficient insights into training stress [8]. Nonetheless, interactions or coupling between internal and external load can be complex and requires more advanced investigation [9].

Ideally, a speed skating coach wants his training sessions to either be intensive or extensive interval training and exactly know if the training type is actually performed by the speed skaters as planned. To assess this, both internal and external training load measurements may be used as predictors, while the training type being intensive or extensive interval tranining can be considered the target. The purpose of this study is to see how well we can distinguish between intensive and extensive interval training sessions based on measures of training load. Which means that we are talking about a classification problem here.

Aim

The aim of this project was to build a classification model to distinguish extensive and intensive interval training sessions in young speed skaters, based on measurement of the internal and/or external training load and using supervised machine learning techniques.

Overview

This report provides a brief overview of what has been done in the present project, from obtaining the relevant data, preprocessing the data, performing data exploration / exploratory data analysis and building and validating several machine learning algorithms on the data. How well these models predict the type of training session will be evaluated using the performance (F1-score, accuracy, sensitivity) on the 'unseen' data from a final holdout test set.

Dataset

The data is derived from a longitudinal study, and was collected from young talented speed skaters over a period of two consecutive seasons. The group consists of 18 (sub)elite skaters, 8 male and 10 female, trained by the same coach. The skaters were performing their regular training routine, consisting of extensive and intensive training sessions, determined by the coach. Only the extensive and intensive interval sessions were included in this dataset. Variables include date and description of training session, details on the speed skater that performed the session and their corresponding speed recorded from the transponder during each of the segments on the ice rink and heart rate recorded using telemonitoring.

An overview of all variables (including engineered features) in the dataset is provided as a data dictionary in a separate Rmarkdown and html file.

Note that in addition to the 'general dataset' the coach has provided us with raw data from the transponders (one record for each passing of a segment on the ice rink) and from the heart rate monitoring (one record for each second). These raw datasets can be used for additional feature engineering, see below for more details.

Methods / analysis

Data exploration

After loading the general speed skating data from the `data_speedskating.csv` file, we first inspect the dataframe with intensive and extensive interval training sessions.

```
# Load general speed skating data for caption project
  data  <- read.csv('./data/data_speedskating.csv')
```

```
# Preliminary inspection of the data
  head(data,3)
```

```
##         date skater_id skater_type gender session      training_type ice_start ice_duration ice_dista
## 1 2018-10-26         3    Allround      F       1 Extensive interval  18:46:33           55     15.38
## 2 2018-10-26         4    Allround      M       1 Extensive interval  18:46:24           57     19.81
## 3 2018-10-26         6      Sprint      F       1 Extensive interval  18:46:08           55     18.07
##   ice_fastest_lap HR_start HR_duration HR_max_overall HR_z0_min HR_z1_min HR_z2_min HR_z3_min HR_z4_m
## 1          34.135 18:46:33          55            214  1.833333  17.46667  16.21667 15.283333  4.4166
## 2          27.374 18:46:24          57            195  4.733333  15.66667  16.73333  8.716667  8.9000
## 3          32.767 18:46:08          56            197  0.250000  13.76667  19.33333 18.650000  3.8833
```

```
# Further inspection of the data
  summary(data)
```

```
##      date             skater_id    skater_type           gender             session      training_typ
##  Length:141         Min.   : 2.0   Length:141         Length:141         Min.   :1.000   Length:141
##  Class :character   1st Qu.: 6.0   Class :character   Class :character   1st Qu.:1.000   Class :chara
##  Mode  :character   Median :15.0   Mode  :character   Mode  :character   Median :1.000   Mode  :chara
##                     Mean   :12.7                                         Mean   :1.028
##                     3rd Qu.:18.0                                         3rd Qu.:1.000
##                     Max.   :21.0                                         Max.   :2.000
##   ice_start          ice_duration    ice_distance     ice_nr_laps    ice_fastest_lap  HR_start
##  Length:141         Min.   : 4.00   Min.   : 2.325   Min.   : 7.00   Min.   :27.37   Length:141
##  Class :character   1st Qu.:39.00   1st Qu.:12.010   1st Qu.:31.00   1st Qu.:30.18   Class :characte
##  Mode  :character   Median :48.00   Median :15.497   Median :40.00   Median :31.17   Mode  :characte
##                     Mean   :53.87   Mean   :16.027   Mean   :41.82   Mean   :31.38
##                     3rd Qu.:70.00   3rd Qu.:19.759   3rd Qu.:51.00   3rd Qu.:32.78
##                     Max.   :93.00   Max.   :29.832   Max.   :79.00   Max.   :36.92
##  HR_max_overall    HR_z0_min        HR_z1_min        HR_z2_min        HR_z3_min         HR_z4_min
##  Min.   :188.0   Min.   : 0.000   Min.   : 0.000   Min.   : 0.300   Min.   : 0.9667   Min.   : 0.000
##  1st Qu.:195.0   1st Qu.: 0.000   1st Qu.: 2.750   1st Qu.: 9.233   1st Qu.: 7.0333   1st Qu.: 4.817
##  Median :197.0   Median : 0.450   Median : 8.200   Median :12.783   Median :11.0000   Median : 8.533
##  Mean   :201.5   Mean   : 3.913   Mean   : 9.894   Mean   :13.809   Mean   :11.5753   Mean   : 9.130
##  3rd Qu.:199.0   3rd Qu.: 3.567   3rd Qu.:15.667   3rd Qu.:16.850   3rd Qu.:15.2833   3rd Qu.:12.700
##  Max.   :230.0   Max.   :43.133   Max.   :46.300   Max.   :52.400   Max.   :25.9167   Max.   :32.967
```

The dataset contains a clear overview with records of training sessions performed by certain speed skaters. As expected, the data frame contains information about the skaters, training session, data from the transponder (e.g. duration, distance, fastest lap) and heart rate related data (overall maximal heart rate and time in different heart rate zones).

The summary results indicate that there are no missing values in the dataset (as it was already preprocessed by the embedded scientist of the team), and values make mostly sense: maximal duration on the ice and with heart rate is 93 minutes, maximal heart rate is around 200 for these young talented speed skaters. There is one speedskater with a maximal heart rate of 230 which is quite high but not impossible. Also we see that maximal skater_id is 21 while there were only 18 speed skaters expected. Let's look more into this.

```
    # Check number of speed skaters
      unique(data$skater_id) %>% length() %>% print()
```

```
## [1] 18
```

Indeed, there are 18 unique speed skaters. It seems that they received other pseudonymization id's. Now let's check the distribution of males and females:

```
    # Check sex of speed skaters
      data %>%
        group_by(skater_id) %>% filter(row_number()==1) %>%
        group_by(gender) %>% summarise(n=n())
```

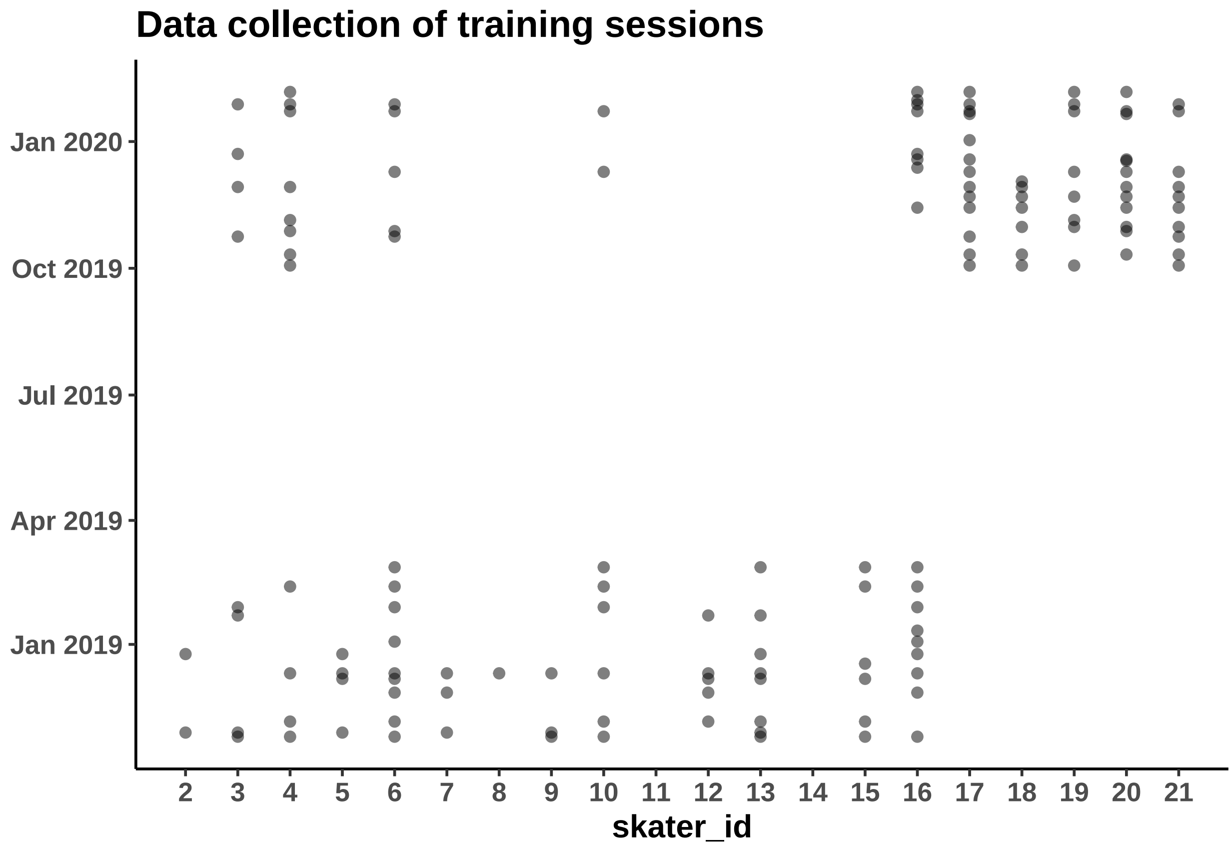
```
## # A tibble: 2 x 2
##   gender     n
##   <chr>  <int>
## 1 F         10
## 2 M          8
```

As expected, there are indeed 10 females and 8 males.

Time for some preprocessing of the general data, such as processing date information and adding a unique identifier to each training session.

But can we look into more detail on when data was collected throughout the two speed skating seasons?

```
# Data visualization: plot collected data throughout the season for each speedskater
ggplot(data, aes(x = skater_id, y = date)) +
       geom_point(alpha =.5) +
       ggtitle("Data collection of training sessions") +
       scale_x_continuous(breaks=seq(1,21,1)) +
       theme_classic() +
       theme(plot.title   = element_text(size=14, face="bold"),
             axis.title.y = element_blank(),
             axis.title.x = element_text(size=12, face="bold"),
             axis.text    = element_text(size=10, face="bold"))
```



We can see that the training sessions are recorded in the period September - March, which corresponds to the competitive season of the speed skaters and corresponds to the time when they skated on the ice. Typically there is no ice in the summer. Also, we see some skaters participating only in the first season, some only in the second season and a few participating in both seasons.

And now looking into the number of intensive and extensive interval sessions:

```
# Check intensive and extensive interval training sessions
data %>% group_by(training_type) %>% summarise(n=n())
```

```
## # A tibble: 2 x 2
##   training_type     n
##   <fct>         <int>
## 1 Ext_Interval    118
## 2 Int_Interval     23
```

Over the course of two consecutive speed skating seasons, we have collected data from 118 extensive interval and 23 intensive interval training sessions, in which both heart rate and speed were recorded. We can see that there is some class imbalance (84-16%), which we'll take a closer look at later in our analysis.

Note that in addition to this general dataset, the coach has provided us with two other files containing the raw data. For the external load - the transponder / speed - this is a recording for each time the skater passed one of the loops on the ice (of which there are 12 in total). For the internal load - the heart rate - this is the heart rate measured using telemonitoring sampled every second. We can use these raw datasets later on to engineer new features that can be added to the training session records in the general dataset.

```r
# Load detailed speed skating data for internal training load (heart rate)
data_i_raw <- read.csv('./data/data_speedskating_internal_raw.csv')

# Load detailed speed skating data for external training load (speed)
data_e_raw <- read.csv('./data/data_speedskating_external_raw.csv')
```

```r
# Inspect raw datasets - internal load
glimpse(data_i_raw)
```

```
## Rows: 444,101
## Columns: 6
## $ date      <chr> "2018-10-29", "2018-10-29", "2018-10-29", "2018-10-29", "2018-10-29", "2018-10-29"
## $ time      <chr> "10:26:24", "10:26:25", "10:26:26", "10:26:27", "10:26:28", "10:26:29", "10:26:30"
## $ session   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
## $ skater_id <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2
## $ HR        <int> 0, 0, 0, 0, 0, 76, 76, 77, 77, 77, 78, 78, 77, 77, 77, 77, 78, 78, 78, 78, 78, 82,
## $ max_HR    <dbl> 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 188, 18
```

```r
# Inspect raw datasets - external load
glimpse(data_e_raw)
```

```
## Rows: 103,947
## Columns: 11
## $ date         <chr> "2018-09-13", "2018-09-13", "2018-09-13", "2018-09-13", "2018-09-13", "2018-09-
## $ time         <chr> "18:46:35", "18:46:36", "18:46:38", "18:46:41", "18:46:44", "18:46:48", "18:46:5
## $ session      <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
## $ skater_id    <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2
## $ lap_id       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3
## $ loop_id_end  <int> 12, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
## $ duration     <dbl> 4.658, 1.093, 3.317, 3.353, 3.296, 5.721, 5.642, 2.742, 2.780, 2.800, 2.315, 3.4
## $ speed        <dbl> 33.68914, 38.04209, 30.12300, 29.79958, 30.31493, 34.73519, 35.25346, 36.43982,
## $ zone         <int> 3, 3, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3
## $ max_speed    <dbl> 54.26999, 54.26999, 54.26999, 54.26999, 54.26999, 54.26999, 54.26999, 54.26999,
## $ acceleration <dbl> 2.009036840, 1.106269450, -0.663173156, -0.026793605, 0.043431830, 0.214621520,
```

Both sets contain relevant information to generate new features that can be added to the general dataset.

Now it would be good to check if there may be outliers or missing data.

```r
# Inspect raw datasets - internal load
summary(data_i_raw)
```

```
##      date               time              session        skater_id           HR            max_HR
##  Length:444101      Length:444101      Min.   :1.000   Min.   : 2.00   Min.   :  0.0   Min.   :188.0
##  Class :character   Class :character   1st Qu.:1.000   1st Qu.: 7.00   1st Qu.:119.0   1st Qu.:195.0
##  Mode  :character   Mode  :character   Median :1.000   Median :16.00   Median :137.0   Median :197.0
##                                        Mean   :1.025   Mean   :13.58   Mean   :138.8   Mean   :202.3
##                                        3rd Qu.:1.000   3rd Qu.:19.00   3rd Qu.:160.0   3rd Qu.:199.0
##                                        Max.   :2.000   Max.   :21.00   Max.   :232.0   Max.   :230.0
```

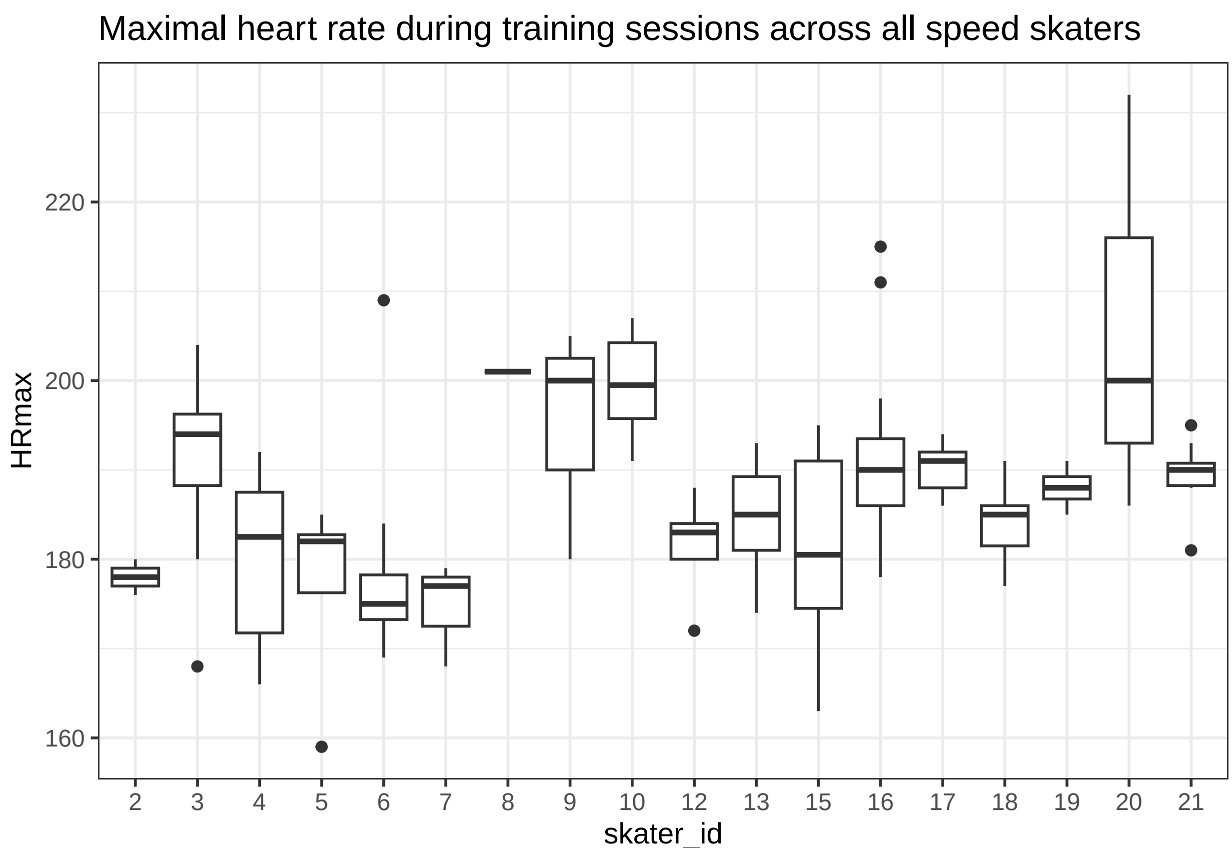
```r
# Inspect raw datasets - external load
summary(data_e_raw)
```

```
##      date               time              session        skater_id         lap_id          loop_id_e
##  Length:103947      Length:103947      Min.   :1.000   Min.   : 2.00   Min.   :        1   Min.   : 1
##  Class :character   Class :character   1st Qu.:1.000   1st Qu.: 6.00   1st Qu.:       24   1st Qu.: 3
##  Mode  :character   Mode  :character   Median :1.000   Median :13.00   Median : 4283844   Median : 7
##                                        Mean   :1.017   Mean   :11.89   Mean   : 5229330   Mean   : 6
##                                        3rd Qu.:1.000   3rd Qu.:17.00   3rd Qu.: 8730387   3rd Qu.: 9
##                                        Max.   :2.000   Max.   :21.00   Max.   :14919529   Max.   :12
##     duration            speed              zone         max_speed       acceleration
##  Min.   :    0.762   Min.   : 0.01952   Min.   :0.000   Min.   :49.47   Min.   : -1.73367
##  1st Qu.:    2.537   1st Qu.:15.88193   1st Qu.:1.000   1st Qu.:50.82   1st Qu.: -0.10982
##  Median :    4.113   Median :31.78053   Median :3.000   Median :53.29   Median : -0.01337
##  Mean   :    6.297   Mean   :28.72268   Mean   :2.384   Mean   :53.12   Mean   :  0.01676
##  3rd Qu.:    6.873   3rd Qu.:40.17612   3rd Qu.:4.000   3rd Qu.:55.07   3rd Qu.:  0.09766
##  Max.   :25541.187   Max.   :55.57047   Max.   :5.000   Max.   :58.45   Max.   :108.68126
```

Both the internal and external raw data do not seem to contain any missings. For the internal data, we see again that there is some very high values around 230 bpm. Let's check if those can be considered to be outliers, by looking at the maximal heart rate of all training sessions across all speed skaters.

```r
# Inspect maximal heart rate from the raw data
data_i_raw %>% group_by(skater_id,date,session) %>% summarise(HRmax = max(HR)) %>%
ggplot(aes(x=factor(skater_id),y=HRmax)) + geom_boxplot() +
xlab('skater_id') + ggtitle('Maximal heart rate during training sessions across all speed skaters
theme_bw()
```

```
## `summarise()` has grouped output by 'skater_id', 'date'. You can override using the `.groups` argumen
```



After reporting boxplots of the maximal heart rate per session across all speed skaters, we see that skater 20 is able to reach very high maximal heart rate values during his/her sessions. The heart rate of 230 is not uncommon as this value is included within the wisker of the boxplot and is not considered to be an outlier.

For the external raw data, we saw that speed values are in the range that you would expected, increasing up to 55-60 km/h, which is quite fast but not uncommon for young elite speed skaters. However, there are also some strange things happening with the maximal acceleration (108 m/s^2) and time duration between two loops on the ice (25541 s). For maximal acceleration you would not expect one to be faster than the fasted human on earth (Usain Bolt), corresponding to approximately 10m/s^2, so these records were replaced by missing values. Same holds for the sample of 25541 s between two passings, which is way too long. The longest time duration you could expect is 1200 s, which is the time it takes to mop the ice rink (and can possibly - but luckily not often - happen during training hours).

Ok, now we can move on to the preprocessing of the raw data. Details on this can be found in the `preprocess_data_raw` script, but contains removal of outliers and calculation of time and relative speed and heart rate (relative to the maximum).

```
# Perform preliminary pre-processing for raw data
data_i_raw <- preprocess_data_raw(data_i_raw, data, 'internal')
data_e_raw <- preprocess_data_raw(data_e_raw, data, 'external')
```

Data partitioning

So similar to the movielens project, it is very important to split our dataset into a train and test data. The train dataset will be used to generate our models, while the test set will serve as unseen data to evaluate our model performance. This will also illustrate the generalizability of the models. Since we are dealing with a slightly imbalanced dataset and rather small number of samples of our minority class, I've used a 65-35%

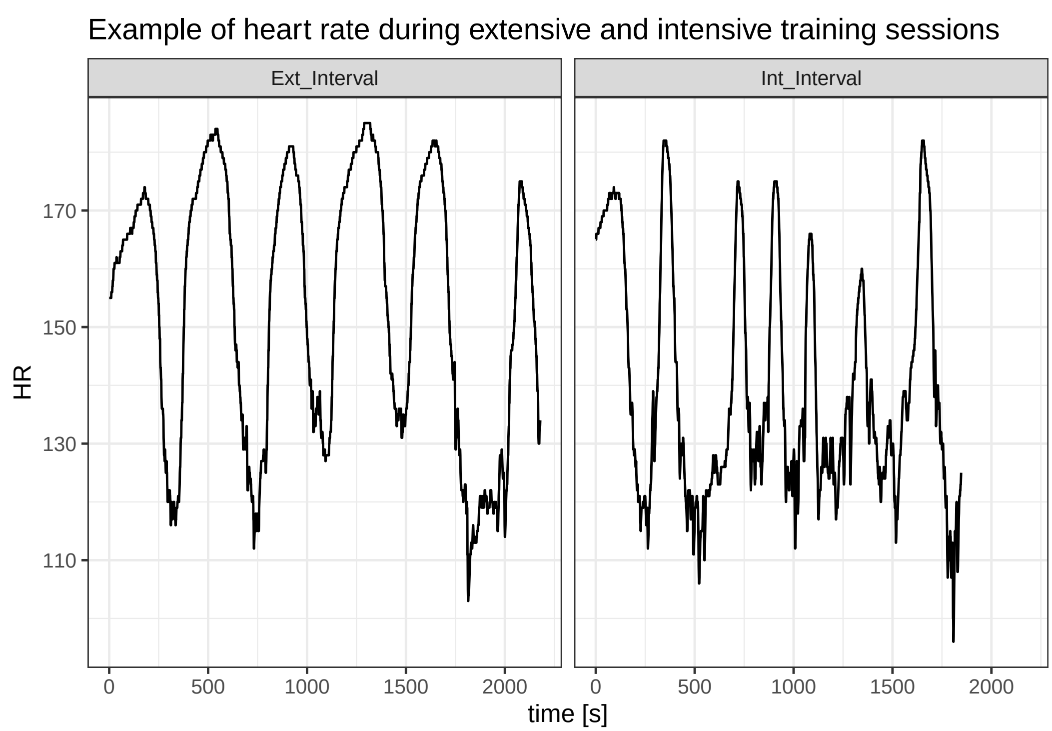
split between train and test, so that there are still sufficient number of intensive interval sessions also in the test set.

```
# Data partitioning: split data into train and holdout test set
set.seed(123)
trainIndex       <- createDataPartition(data$training_type, times=1, p = .65, list=F)
data_train       <- data[trainIndex,]
data_test        <- data[-trainIndex,]
```

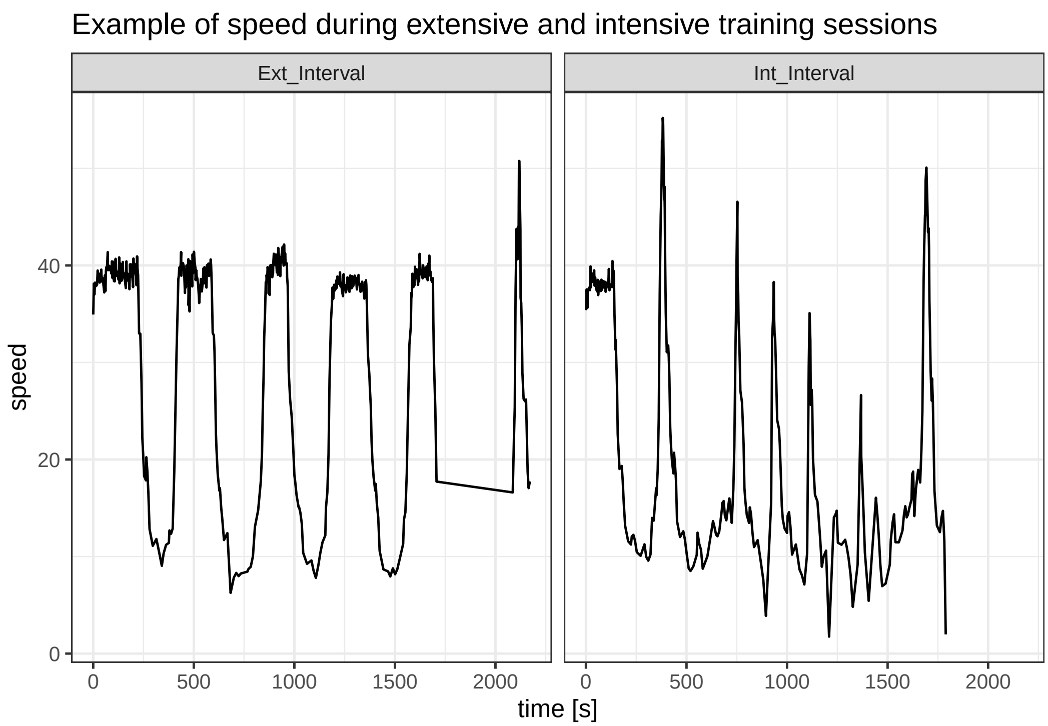
Exploratory data analysis

Now back to the raw data and exploration again. What is the typical HR and speed pattern for an intensive or extensive interval training session?

```
# Inspect heart rate during intensive and extensive interval sessions
data_i_raw %>% filter(skater_id==5, date %in% c('2018-12-07','2018-12-11')) %>%
ggplot(aes(x=time,y=HR)) + geom_line() + facet_wrap(~training_type) +
xlab('time [s]') +
ggtitle('Example of heart rate during extensive and intensive training sessions') +
theme_bw()
```



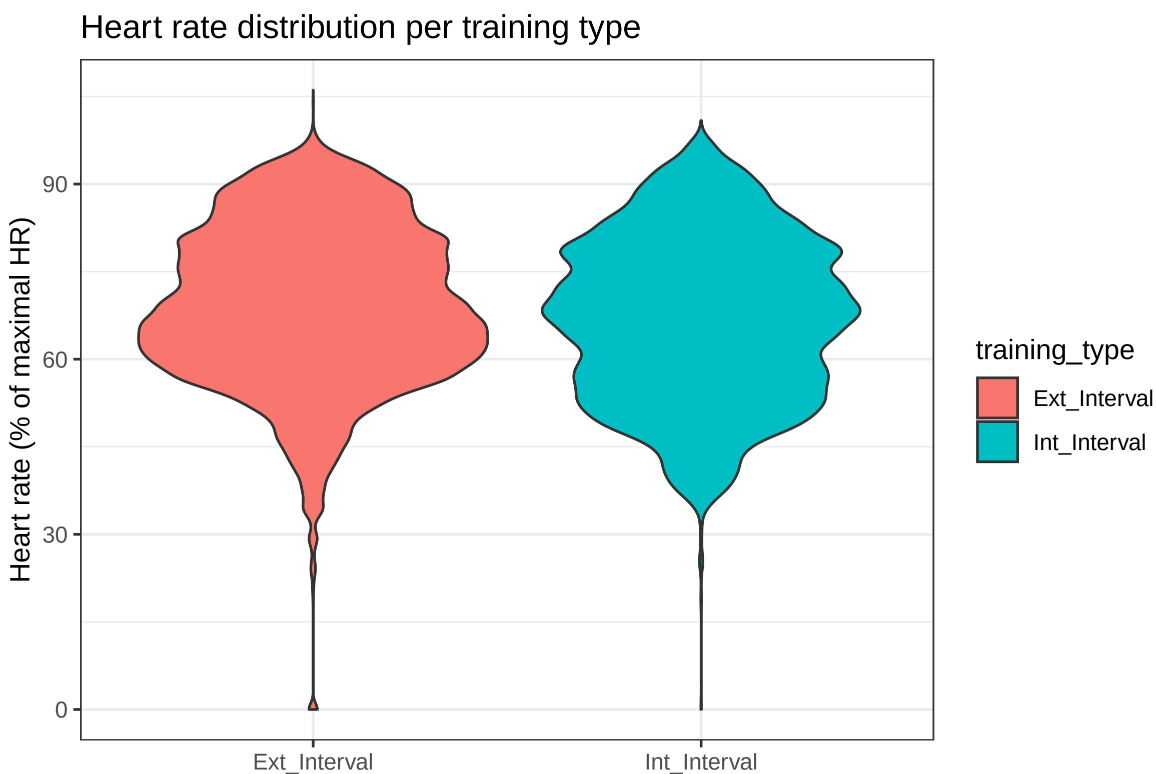

```
# Inspect speed during intensive and extensive interval sessions
data_e_raw %>% filter(skater_id==5, date %in% c('2018-12-07','2018-12-11')) %>%
ggplot(aes(x=time,y=speed)) + geom_line() + facet_wrap(~training_type) +
xlab('time [s]') +
ggtitle('Example of speed during extensive and intensive training sessions') +
theme_bw()
```



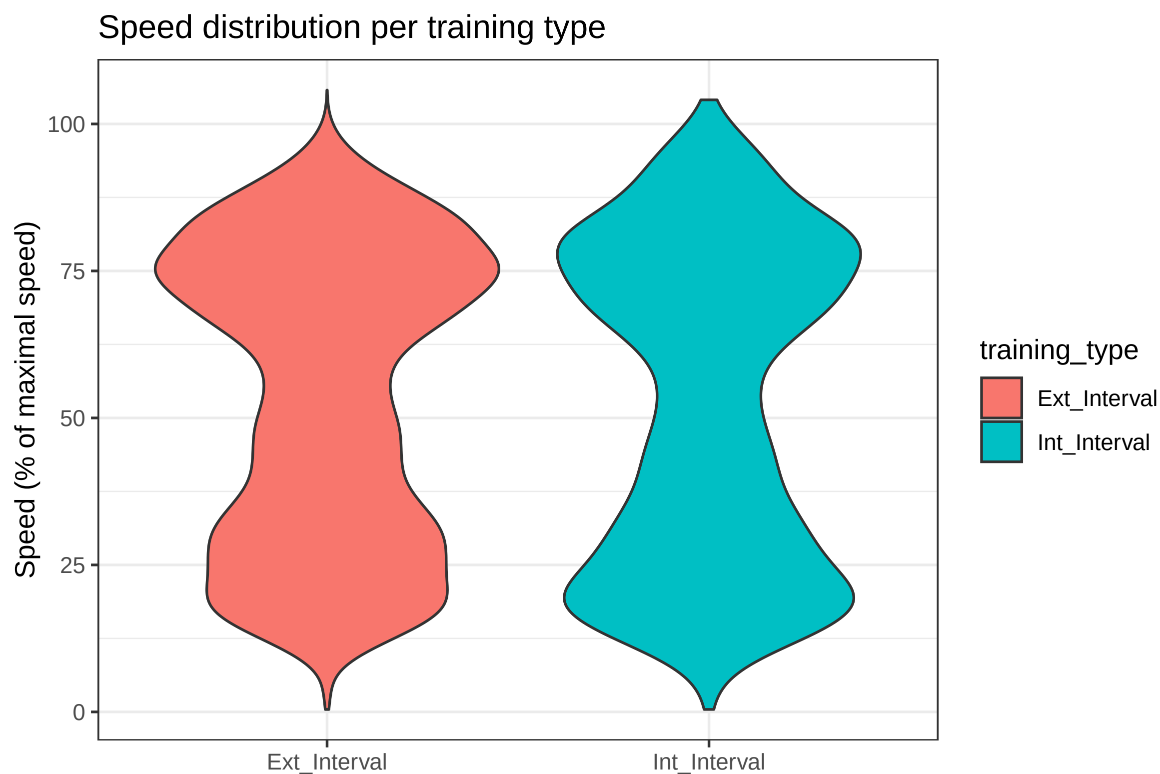
Some clear patterns on the intervals can be seen, with extensive having longer intervals (as expected). Heart rate intensity is not much higher with the intensive interval sessions, but could be explained by the shorter duration of the intervals, as heart rate increases with time at high intensities (there is some cardiac drift).

Are there any differences in relative speed or heart rate between the two training types? Note that we should only look at the training data as this will provide some good information for designing our new features.

```
#Check for distribution of speed on sessions in training data: reveals some differences
 ggplot(data_i_raw %>% filter(!is.na(training_type)) %>% filter(train_id %in% data_train$train_id),
        aes(x=training_type,y=HR_rel, fill=training_type)) +
   geom_violin() +
   xlab('') + ylab('Heart rate (% of maximal HR)') + theme_bw() +
   ggtitle('Heart rate distribution per training type')
```




```
#Check for distribution of speed on sessions in training data: reveals some differences
ggplot(data_e_raw %>% filter(!is.na(training_type)) %>% filter(train_id %in% data_train$train_id),
       aes(x=training_type,y=speed_rel, fill=training_type)) +
    geom_violin() +
    xlab('') + ylab('Speed (% of maximal speed)') + theme_bw() +
    ggtitle('Speed distribution per training type')
```



Overall the distribution patterns show similar shapes, but when looking more closely we can see that both have clear differences at certain intensities. For example for heart rate, the extensive interval sessions peak at ~65% of max HR, while this is ~70% for intensive sessions. For speed, you can see a similar pattern, while the recovery part is typically executed at a lower speed during the intensive sessions. This provides some good input for designing our new features.

Feature engineering

Based on the data exploration, several features are engineered. For all details see the script below. Note that feature engineering is performed separately for the test and train set to avoid contamination between the two datasets.

```
# Perform feature engineering on train set
data_train <- perform_feature_engineering(data_i_raw, data_e_raw, data_train)

# Perform feature engineering separately on final validation set
data_test <- perform_feature_engineering(data_i_raw, data_e_raw, data_test)
```

Engineered features include speed over certain segments on the ice rink, e.g. straights or turns, percentiles of speed and heart rate during the sessions, mean, max, median speed and heart rate, but also variability of speed and heart rate. Also, based on the data exploration, I've added number of passings within a specific speed window (based on %max speed) or number of seconds within a specific heart rate window (based on %max HR). Additionally, information on the season, time of day were included in our final dataset.

Modelling

Alright, now it's time for the exciting part: modelling the training type. Can we predict whether speed skaters performed an intensive or extensive interval training based on the predictors obtained from internal and external training load (i.e. heart rate or speed)? I've selected three types of algorithms to evaluate: a 1) random forest (ensamble), 2) glmnet (elastig net) and 3) support vector machine algorithm.

Modeling is performed using 10-fold cross-validation (with preprocessing per fold) and hyperparameter tuning (mtry for rf, lambda and alpha parameters for glmnet and c parameter for svm). Models are optimised for the ROC metric.

For the final evaluation of model perofrmance, it is best to look at metrics that take into account the model performance on both classes, since dealing with imbalanced dataset. Other said, if you are predicting fraud (that happens in less than 1% of the time) then your model can predict no fraud and is accurate >99% of the time, but still does not tell you anything on when it may be fraud. Similarly, it is equally important to predict intensive interval sessions, even though they occur less frequently. The metrics I will evaluate are:

- F1-score

$$(2xPrecisionxRecall)/(Precision + Recall)$$

- Balanced accuracy

$$(TruePositiveRate + TrueNegativeRate)/2$$

- Sensitivity

$$(TruePositives)/(TruePositives + FalseNegatives)$$

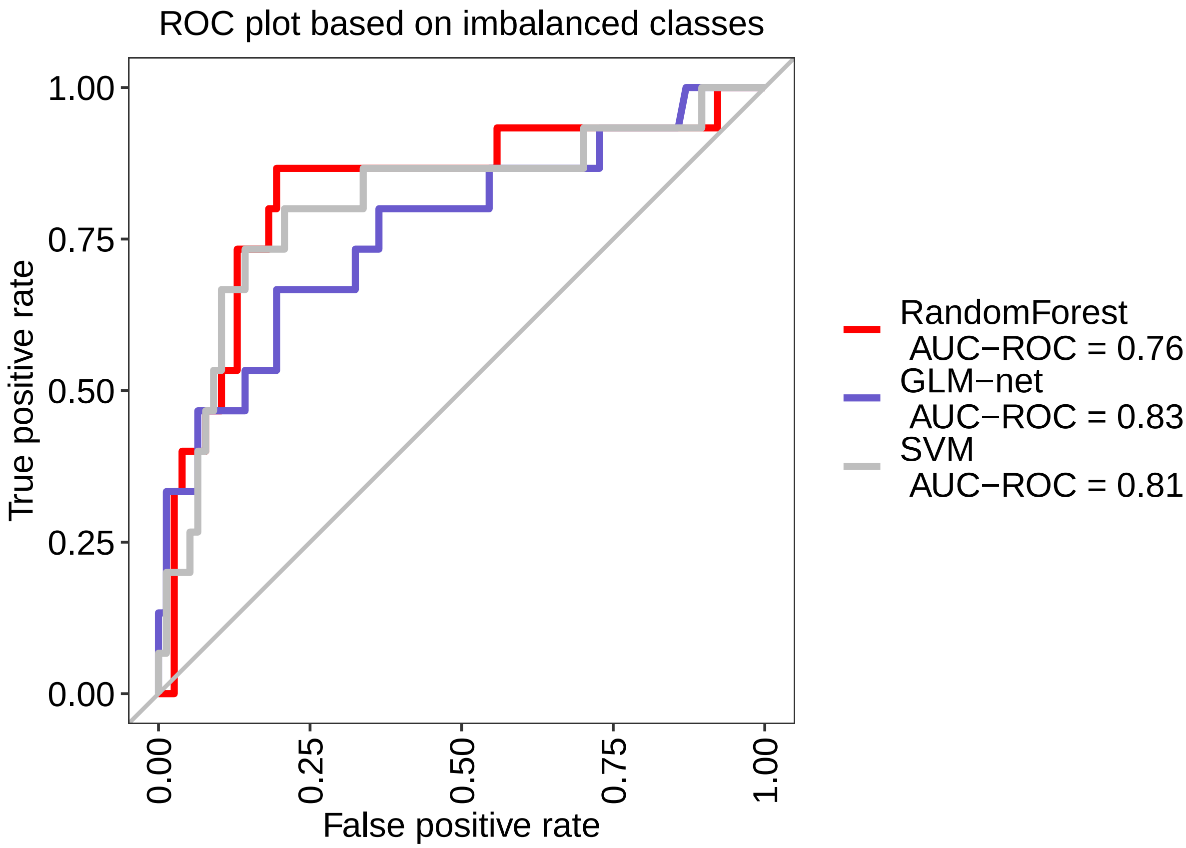
- Specificity

$$(TrueNegatives)(TrueNegatives + FalsePositives)$$

Let's create separate models for each of the three algorithms:

```
# Perform ML modelling machine learning models based on imbalanced classes
       model_def_1 <- perform_ml_modelling(data_train, 'rf')
       model_def_2 <- perform_ml_modelling(data_train, 'glmnet')
       model_def_3 <- perform_ml_modelling(data_train, 'svm')

       # Evaluate machine learning models based on their ROC-curve and AUC
       res <- evalm(list(model_def_1,model_def_2,model_def_3),
                    c('RandomForest','GLM-net','SVM'),
                    title = 'ROC plot based on imbalanced classes',
                    silent=T, plots=F)
       res$roc
```



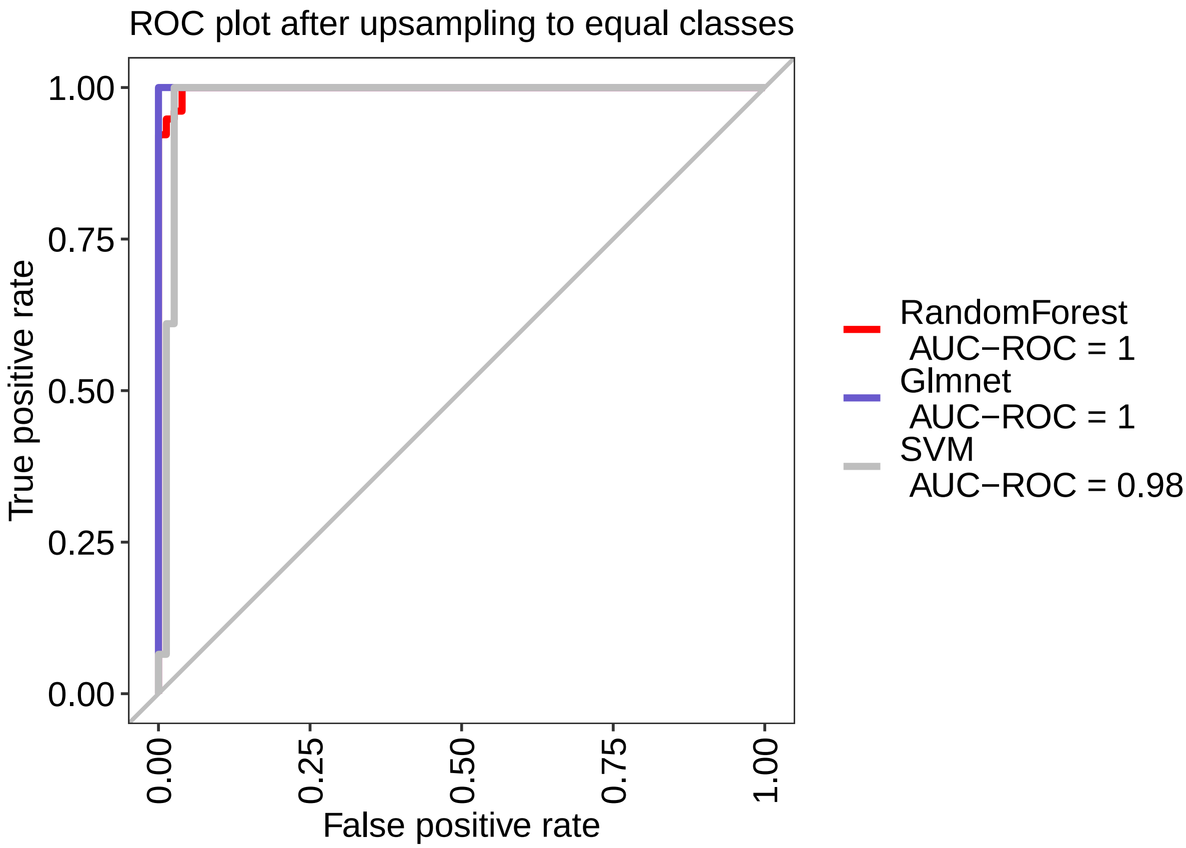
Model performance is already quite ok, with AUC scores of 0.75+. Also we see slightly higher performance for the elastic net. However, this is based on a rather imbalanced dataset. What if we now upsample the minority class (only for the train set) and see if we can improve model performance?

```
# Note that upsampling techniques may be applied on the minority class to improve training
# of the ML models

# Upsample train set to accomodate for class imbalance
data_train_upsample <- upSample(data_train %>% select(-training_type),
                                data_train %>% pull(training_type) %>% as.factor()) %>%
                        rename(training_type = Class)

# Perform ML modelling machine learning models based on equal classes
model_ups_1 <- perform_ml_modelling(data_train_upsample, 'rf')
model_ups_3 <- perform_ml_modelling(data_train_upsample, 'svm')
model_ups_2 <- perform_ml_modelling(data_train_upsample, 'glmnet')

# Evaluate machine learning models based on their ROC-curve and AUC
res <- evalm(list(model_ups_1,model_ups_2,model_ups_3),
             c('RandomForest','Glmnet','SVM'),
             title = 'ROC plot after upsampling to equal classes',
             silent=T, plots=F)
res$roc
```



Indeed model performance on the train set has improved. But how does this translate to the ultimate model performance test: the performance on the test set? Let's take a look at this for our final results.

Results

First check the model performance on the unseen data from the test set. The trained models are now used to predict the classes of either intensive or extensive interval training sessions, based on the different predictors. By comparing predicted values by the actual class of the training sessions in the test set, we get an understanding of model performance.

Below we summarise the important metrics to evaluate model performance for our classification problem: F1-score, balanced accuracy, sensitivity and specificity. With the former two presenting very important information on prediction of the two classes.

```
# Obtain final results on unseen test set for each of the models
# (both before and after upsampling the minority class)
results <- rbind('model_rf_def'  = eval_ml_modelling(model_def_1, data_test, 'metric'),
                 'model_glm_def' = eval_ml_modelling(model_def_2, data_test, 'metric'),
                 'model_svm_def' = eval_ml_modelling(model_def_3, data_test, 'metric'),
                 'model_rf_ups'  = eval_ml_modelling(model_ups_1, data_test, 'metric'),
                 'model_glm_ups' = eval_ml_modelling(model_ups_2, data_test, 'metric'),
                 'model_svm_ups' = eval_ml_modelling(model_ups_3, data_test, 'metric'))

# Process tabular output
results <- as.data.frame(results) %>% rownames_to_column() %>%
           mutate(data = ifelse(str_detect(rowname,'def'),'default','upsampled'),
```

```
                   model = gsub('(.*)_(.*)_(.*)','\\2',rowname)) %>%
             select(model,data,F1,`Balanced Accuracy`,Sensitivity,Specificity)

    knitr::kable(results, digits=5)
```

model	data	F1	Balanced Accuracy	Sensitivity	Specificity
rf	default	0.90244	0.70122	0.90244	0.500
glm	default	0.94118	0.73780	0.97561	0.500
svm	default	0.90698	0.60061	0.95122	0.250
rf	upsampled	0.90909	0.55030	0.97561	0.125
glm	upsampled	0.92500	0.82622	0.90244	0.750
svm	upsampled	0.95000	0.90091	0.92683	0.875

From the table, a couple of things have become clear.

- Balanced accuracy reflects decent performance on the default dataset for rf and glmnet (without upsampling)
- Upsampling improved model performance in terms of F1-score, balanced accuracy and specificity (but not for rf)
- When interested in the minority class (intensive interval training sessions), only the SVM and GLMnet with upsampling are of sufficient performance (high enough specificity).
- The best performing model is the support vector machine after accounting for class imbalance with upsampling.

Now look at this final best model in a bit more detail:

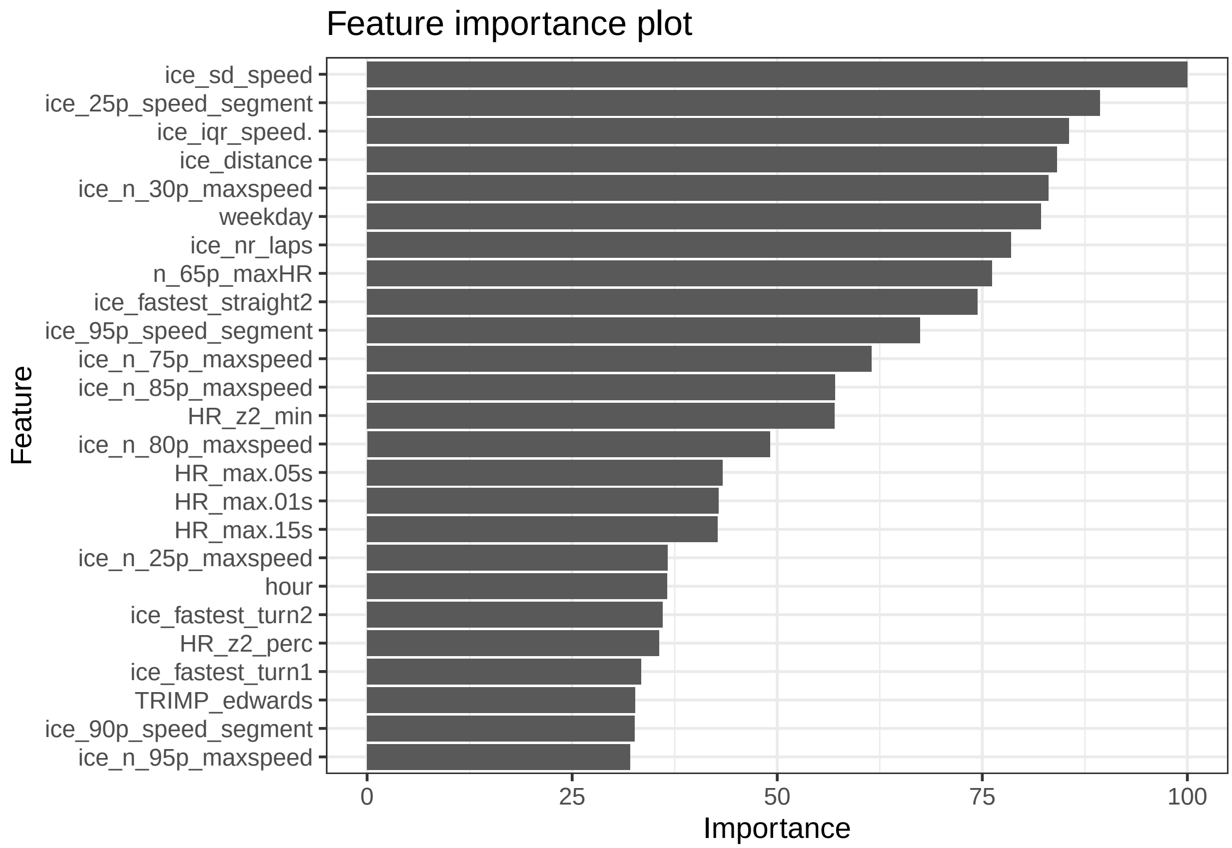
```
        # Obtain confusion matrix from best performing model
        eval_ml_modelling(model_ups_3, data_test, 'cmatrix')
```

```
##                Reference
## Prediction      Ext_Interval Int_Interval
##   Ext_Interval           38            1
##   Int_Interval            3            7
```

The confusion matrix shows that performance on both intensive and extensive training sessions is rather good. With values as high as ~0.90. Of note, the test set contains limit number of intensive sessions, so it would be strongly recommmended to repeat this analysis also with a larger dataset, even though this is quite challeging within the sport science context.

Of course the coach is not only interested in predicting training sessions based on internal and external load, he/she is also interested in knowning what features are relevant for this predction. These are shown below:

```
        # Best performing model based on balanced accuracy is model3 after upsampling.
        # Here I provide the top features based on feature importance scores:
        feature_importance <- varImp(model_ups_3)
        ggplot(feature_importance, top=25) + ggtitle('Feature importance plot') + theme_bw()
```



Important characteristics are recovery speed (at 25th percentile), variability in speed and fastest straight 2, but interestingly enough also the weekday a training is performed. From internal load, maximal heart rate is the most important predictor, but also minutes in zone two are of interest. These features will enable the coach to learn what distinguishes best between intensive and extensive sessions and whether his/her training prescription needs to be altered accordingly.

That concludes the analysis of the current capstone project on speed skating.

Conclusion

I have demonstrated that with an extensive data collection from a sport science perspective, training sessions with intensive and extensive interval training, one can predict training type based on predictors of internal and external load. That is, based on speed during the training sessions as well as the heart rate of the athletes. Optimal model performed very well both on the majority class (extensive; sensivity >0.9) as wel as on the minority class (intensie; specificity ~0.90). Combined metrics show excellent performance as well: F1/balanced accuracy of 0.90-0.95. Importantly, one needs to address the class imbalance, such as using upsampling techniques. However also other techniques could be used (e.g. SMOTE), although improving the number of records is preferred - even though difficult from a sports science perspective.

The coach has received a supervised machine learning model that helps to predict training session type and also has learned what are key features that enable this prediction, including some . Future studies could repeat this analysis in a larger sample, but also translate this to other sports, such as cycling or rowing.

I hope you have learned something new with the current analysis and would be honoured to hear your feedback and suggestions on the report.

References

- [1] Foster C, Rodriguez-Marroyo JA, de Koning JJ. Monitoring Training Loads: The Past, the Present, and the Future. Int J Sports Physiol Perform. 2017 Apr;12(Suppl 2):S22-S28. doi: 10.1123/ijspp.2016-0388. Epub 2017 Mar 2. PMID: 28253038.
- [2] Seiler S. What is best practice for training intensity and duration distribution in endurance athletes? Int J Sports Physiol Perform. 2010 Sep;5(3):276-91. doi: 10.1123/ijspp.5.3.276. PMID: 20861519.
- [3] Ramadhan, Azhari Rezha et al. "Intensive and Extensive Interval Training; Which is Better Against Vo2max Football Athletes?" International Journal of Multidisciplinary Research and Analysis 2022.
- [4] Goudsmit J, Otter RTA, Stoter I, van Holland B, van der Zwaard S, de Jong J, Vos S. Co-Operative Design of a Coach Dashboard for Training Monitoring and Feedback. Sensors (Basel). 2022 Nov 23;22(23):9073. doi: 10.3390/s22239073. PMID: 36501775; PMCID: PMC9737713.
- [5] Haddad M, Stylianides G, Djaoui L, Dellal A, Chamari K. Session-RPE Method for Training Load Monitoring: Validity, Ecological Usefulness, and Influencing Factors. Front Neurosci. 2017 Nov 2;11:612. doi: 10.3389/fnins.2017.00612. PMID: 29163016; PMCID: PMC5673663.
- [6] Impellizzeri FM, Marcora SM, Coutts AJ. Internal and External Training Load: 15 Years On. Int J Sports Physiol Perform. 2019 Feb 1;14(2):270-273. doi: 10.1123/ijspp.2018-0935. Epub 2019 Jan 6. PMID: 30614348.
- [7] Halson SL. Monitoring training load to understand fatigue in athletes. Sports Med. 2014 Nov;44 Suppl 2(Suppl 2):S139-47. doi: 10.1007/s40279-014-0253-z. PMID: 25200666; PMCID: PMC4213373.
- [8] Bourdon PC, Cardinale M, Murray A, Gastin P, Kellmann M, Varley MC, Gabbett TJ, Coutts AJ, Burgess DJ, Gregson W, Cable NT. Monitoring Athlete Training Loads: Consensus Statement. Int J Sports Physiol Perform. 2017 Apr;12(Suppl 2):S2161-S2170. doi: 10.1123/IJSPP.2017-0208. PMID: 28463642.
- [9] van der Zwaard S, Otter RTA, Kempe M, Knobbe A, Stoter IK. Capturing the Complex Relationship Between Internal and External Training Load: A Data-Driven Approach. Int J Sports Physiol Perform. 2023 Apr 20;18(6):634-642. doi: 10.1123/ijspp.2022-0493. PMID: 37080541.

Session information

```r
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Monterey 12.5
##
## Matrix products: default
## BLAS:   /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versi
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK ve
##
## Random number generation:
##  RNG:     Mersenne-Twister
##  Normal:  Inversion
##  Sample:  Rounding
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Europe/Amsterdam
## tzcode source: internal
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] randomForest_4.7-1.1 knitr_1.45           rmarkdown_2.25       MLeval_0.3           kernlab_0.9-
##  [6] ranger_0.16.0        glmnet_4.1-8         Matrix_1.6-1.1       readxl_1.4.3         anytime_0.3
## [11] zoo_1.8-12           caret_6.0-94         lattice_0.21-9       lubridate_1.9.3      forcats_1.0
## [16] stringr_1.5.1        dplyr_1.1.4          purrr_1.0.2          readr_2.1.4          tidyr_1.3.0
## [21] tibble_3.2.1         ggplot2_3.4.4        tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0     timeDate_4022.108    farver_2.1.1         fastmap_1.1.1        pROC_1.18.5
##  [6] digest_0.6.33        rpart_4.1.21         timechange_0.2.0     lifecycle_1.0.4      survival_3.5
## [11] magrittr_2.0.3       compiler_4.3.2       rlang_1.1.2          tools_4.3.2          utf8_1.2.4
## [16] yaml_2.3.7           data.table_1.14.8    labeling_0.4.3       plyr_1.8.9           withr_2.5.2
## [21] nnet_7.3-19          grid_4.3.2           stats4_4.3.2         fansi_1.0.5          e1071_1.7-1
## [26] colorspace_2.1-0     future_1.33.0        globals_0.16.2       scales_1.3.0         iterators_1
## [31] MASS_7.3-60          tinytex_0.49         cli_3.6.1            generics_0.1.3       rstudioapi_
## [36] future.apply_1.11.0  reshape2_1.4.4       tzdb_0.4.0           proxy_0.4-27         splines_4.3
## [41] parallel_4.3.2       cellranger_1.1.0     vctrs_0.6.4          hardhat_1.3.0        hms_1.1.3
## [46] listenv_0.9.0        foreach_1.5.2        gower_1.0.1          recipes_1.0.8        glue_1.6.2
## [51] parallelly_1.36.0    codetools_0.2-19     stringi_1.8.2        gtable_0.3.4         shape_1.4.6
## [56] munsell_0.5.0        pillar_1.9.0         htmltools_0.5.7      ipred_0.9-14         lava_1.7.3
## [61] R6_2.5.1             evaluate_0.23        highr_0.10           renv_1.0.3           class_7.3-2
## [66] Rcpp_1.0.11          nlme_3.1-163         prodlim_2023.08.28   xfun_0.41            ModelMetric
## [71] pkgconfig_2.0.3
```