Absenteeism

Bilal Khaiar

12/17/2019

Introduction

In this project, an attempt to build machine learning algorithms that can predict hours of absenteeism of workers will be made. Multiple algorithms will be tried and an ensemble will be built from the best models. This dataset was created with records of absenteeism at work from July 2007 to July 2010 at a courier company in Brazil. The dataset was acquired from UCI Machine Learning Repository. For more information click (here).

Methods

Packages

The following packages are needed to run this project's code.

```
pack <- c("knitr", "tidyverse", "textreadr", "lubridate","KRLS", "rmarkdown",
          "stringr", "scales", "caret", "MASS", "foreach","fastDummies",
          "import", "foba", "Matrix", "penalized","bst")

# Check if all packages are already installed, if not, install them.
new.packages <- pack[!(pack %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)

# loop to load all packages.
lapply(pack, FUN = function(X) {
    do.call("require", list(X))
})
```

Preparing the Data

The data set and the attached info file will both be downloaded.

```
# load dataset from UCI repo
temp <- tempfile()
download.file(
  "https://archive.ics.uci.edu/ml/machine-learning-databases/00445/Absenteeism_at_work_AAA.zip",
  temp)
data <- read.csv2(unz(temp, "Absenteeism_at_work.csv"))
unlink(temp)

# load info file
temp <- tempfile()
download.file(
  "https://archive.ics.uci.edu/ml/machine-learning-databases/00445/Absenteeism_at_work_AAA.zip",
```

```
              temp)
info <- read_docx(unzip(temp, "Attribute Information.docx"), skip = 1)[1:44]
unlink(temp)
```

Exploratory Analysis

In order to get insights into the dataset, some exploratory analysis will be made.

```
str(data)
```

```
## 'data.frame':    740 obs. of  21 variables:
##  $ ID                              : int  11 36 3 7 11 3 10 20 14 1 ...
##  $ Reason.for.absence              : int  26 0 23 7 23 23 22 23 19 22 ...
##  $ Month.of.absence                : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ Day.of.the.week                 : int  3 3 4 5 5 6 6 6 2 2 ...
##  $ Seasons                         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Transportation.expense          : int  289 118 179 279 289 179 361 260 155 235 ...
##  $ Distance.from.Residence.to.Work: int  36 13 51 5 36 51 52 50 12 11 ...
##  $ Service.time                    : int  13 18 18 14 13 18 3 11 14 14 ...
##  $ Age                             : int  33 50 38 39 33 38 28 36 34 37 ...
##  $ Work.load.Average.day           : Factor w/ 38 levels "205.917","222.196",..: 7 7 7 7 7 7 7 7 7 7
##  $ Hit.target                      : int  97 97 97 97 97 97 97 97 97 97 ...
##  $ Disciplinary.failure            : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Education                       : int  1 1 1 1 1 1 1 1 1 3 ...
##  $ Son                             : int  2 1 0 2 2 0 1 4 2 1 ...
##  $ Social.drinker                  : int  1 1 1 1 1 1 1 1 1 0 ...
##  $ Social.smoker                   : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Pet                             : int  1 0 0 0 1 0 4 0 0 1 ...
##  $ Weight                          : int  90 98 89 68 90 89 80 65 95 88 ...
##  $ Height                          : int  172 178 170 168 172 170 172 168 196 172 ...
##  $ Body.mass.index                 : int  30 31 31 24 30 31 27 23 25 29 ...
##  $ Absenteeism.time.in.hours       : int  4 0 2 4 2 2 8 4 40 8 ...
```

The dataset has 21 variables and 751 observations. Although some are categorical in nature, all variables are stored as integers except the work load averages variable.

Check missing values.

```
# loop to check if any variable has NAs
sum(apply(data, 2, anyNA))
```

```
## [1] 0
```

No NAs where found.

Explore target variable.

```
# count unique values in tahe target variable
length(unique(data$Absenteeism.time.in.hours))
```

```
## [1] 19
```

```
# check how many times each value appear.
summary(as.factor(data$Absenteeism.time.in.hours))
```

```
##   0   1   2   3   4   5   7   8  16  24  32  40  48  56  64  80 104 112 120
##  44  88 157 112  60   7   1 208  19  16   6   7   1   2   3   3   1   2   3
```

The target variable has 19 unique values. Some of those values appear only once or twice. This is seen clearly on values on the higher end. This will probably make predictions of those values hard due to the lack of data on them.

The most frequent target observation is 8 at 208 observations. That makes sense as those could be workers that were absent for 1 day only.

Explore the reasons variable

```
sort(unique(data$Reason.for.absence))
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 21 22 23 24 25
## [26] 26 27 28
```

```
# check what does zero value stand for in the reasons variable by exploring
# - target values for those observations.
data$Absenteeism.time.in.hours[data$Reason.for.absence == 0]
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [39] 0 0 0 0 0
```

Reasons values range from 0 to 28 with no absenteeism due to reason 20. The zero reason is most likely an attribute that was assigned to workers who were never absent.

A list with names of reasons will be created from the info file.

```
# get reasons names form info file. Skip the 23rd line as reason 20 is absent from the data
reason_n <- c(0,info[4:22],info[24])

# list non ICD reasons
c <- c("patient follow-up", "medical consultation",
       "blood donation", "laboratory examination",
       "unjustified absence", "physiotherapy",
       "dental consultation") # list non ICD reasons
c # add non ICD reasons
```

```
## [1] "patient follow-up"      "medical consultation"   "blood donation"
## [4] "laboratory examination" "unjustified absence"    "physiotherapy"
## [7] "dental consultation"
```

```
reason_n[22:28] <- c
reason_n
```

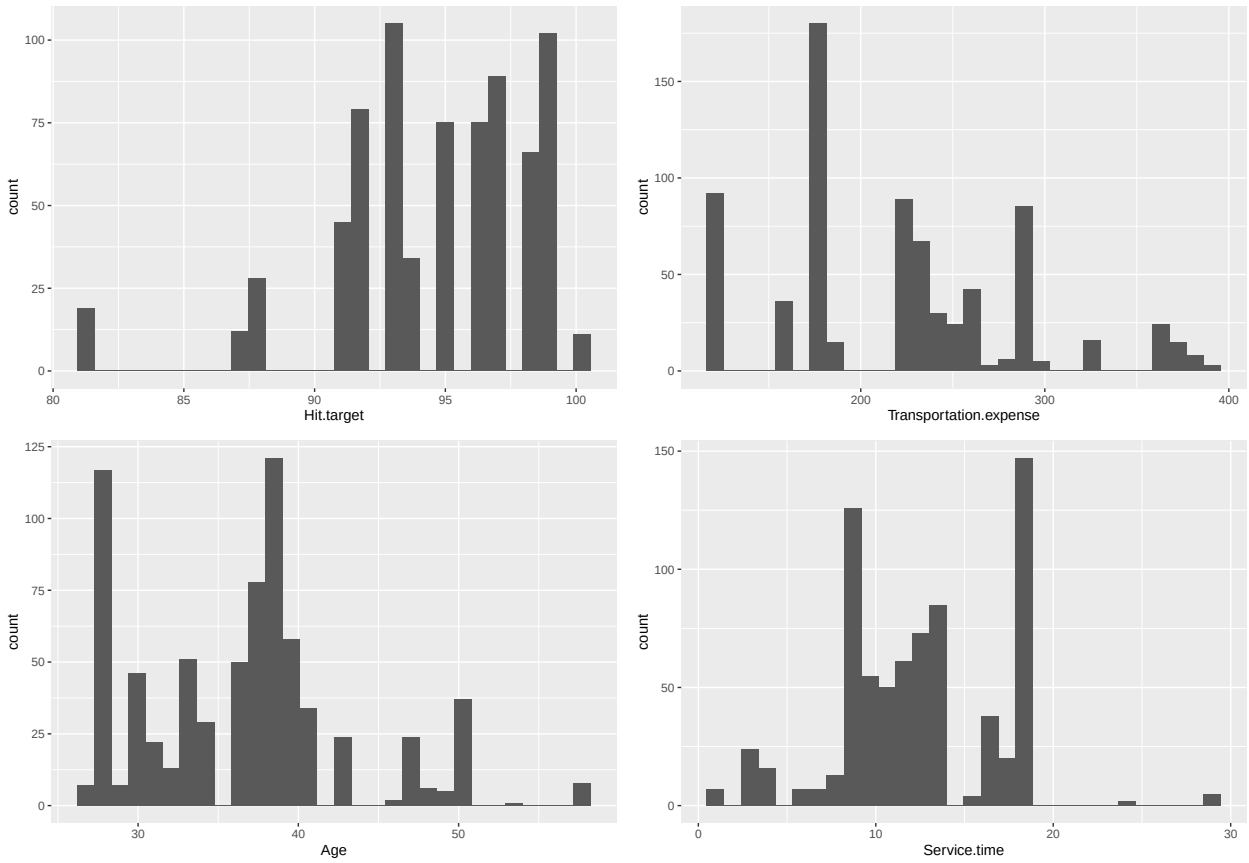
```
##  [1] "0"
##  [2] "I Certain infectious and parasitic diseases"
##  [3] "II Neoplasms"
##  [4] "III Diseases of the blood and blood-forming organs and certain disorders involving the immune m
##  [5] "IV Endocrine, nutritional and metabolic diseases"
##  [6] "V Mental and behavioural disorders"
##  [7] "VI Diseases of the nervous system"
##  [8] "VII Diseases of the eye and adnexa"
##  [9] "VIII Diseases of the ear and mastoid process"
## [10] "IX Diseases of the circulatory system"
## [11] "X Diseases of the respiratory system"
## [12] "XI Diseases of the digestive system"
## [13] "XII Diseases of the skin and subcutaneous tissue"
## [14] "XIII Diseases of the musculoskeletal system and connective tissue"
## [15] "XIV Diseases of the genitourinary system"
## [16] "XV Pregnancy, childbirth and the puerperium"
```

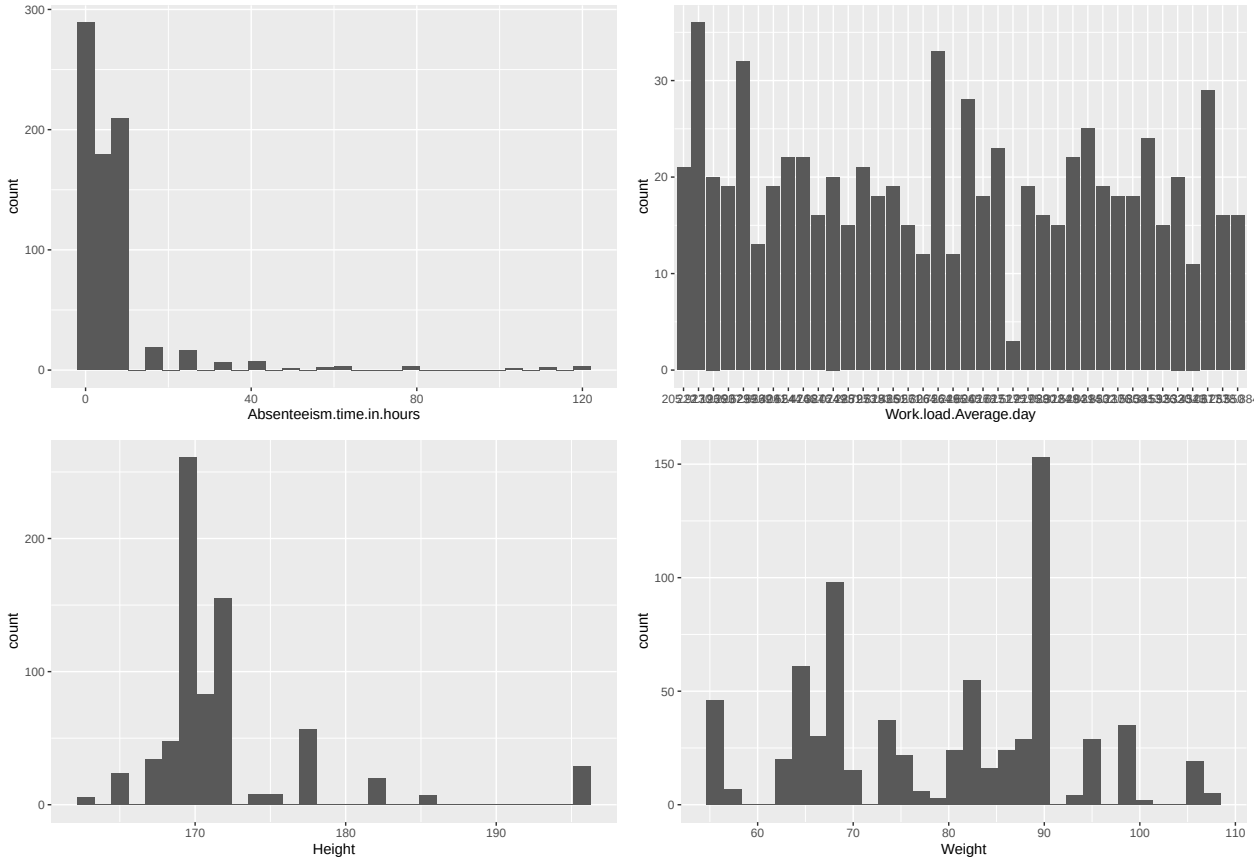
```
## [17] "XVI Certain conditions originating in the perinatal period"
## [18] "XVII Congenital malformations, deformations and chromosomal abnormalities"
## [19] "XVIII Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified"
## [20] "XIX Injury, poisoning and certain other consequences of external causes"
## [21] "XXI Factors influencing health status and contact with health services."
## [22] "patient follow-up"
## [23] "medical consultation"
## [24] "blood donation"
## [25] "laboratory examination"
## [26] "unjustified absence"
## [27] "physiotherapy"
## [28] "dental consultation"
```

Visualization

An attempt to visualize some aspect of the data will be made in order the better understand the distributions and the relationship between some variables.

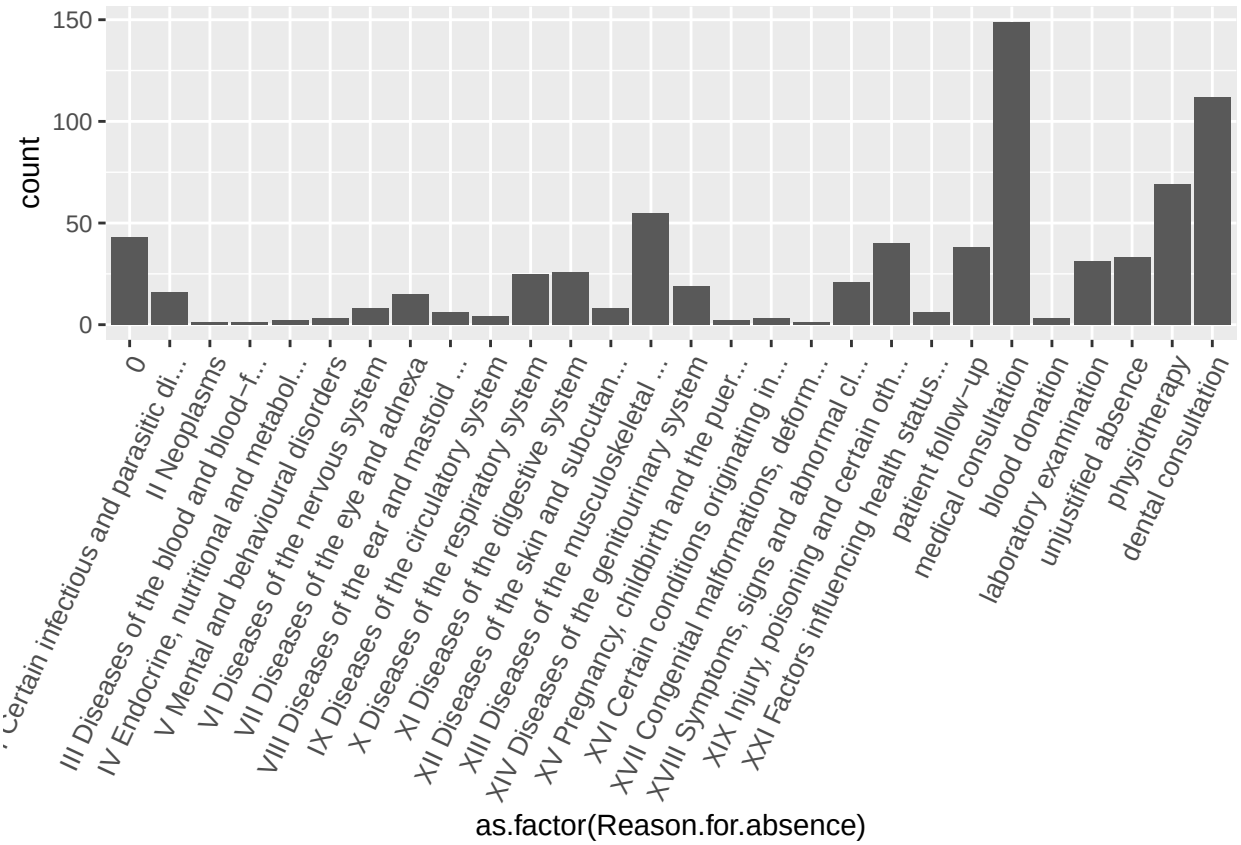
```
# histograms of numerical variables
data %>% ggplot(aes(Hit.target)) + geom_histogram()
data %>% ggplot(aes(Transportation.expense)) + geom_histogram()
data %>% ggplot(aes(Age)) + geom_histogram()
data %>% ggplot(aes(Service.time)) + geom_histogram()
data %>% ggplot(aes(Absenteeism.time.in.hours)) + geom_histogram()
data %>% ggplot(aes(Work.load.Average.day)) + geom_histogram(stat="count")
data %>% ggplot(aes(Height)) + geom_histogram()
data %>% ggplot(aes(Weight)) + geom_histogram()
```





Almost all variables are not normally distributed. The height variable can probably be smoothed to normal but normalization will avoided for all variables

```
# histogram of reasons
data %>% ggplot(aes(as.factor(Reason.for.absence))) +
  geom_histogram(stat = 'count') +
  scale_x_discrete(label = str_trunc(reason_n, 40)) +
  theme(axis.text.x = element_text(angle = 65, hjust = 1, size = 10))
```



Most common reason for absence is medical consultations.

In order to get a quick glimps into how each variable causes variations on the target, a loop that will calculate the standard deviation for the target variable when each variable is grouped into bins will be made.

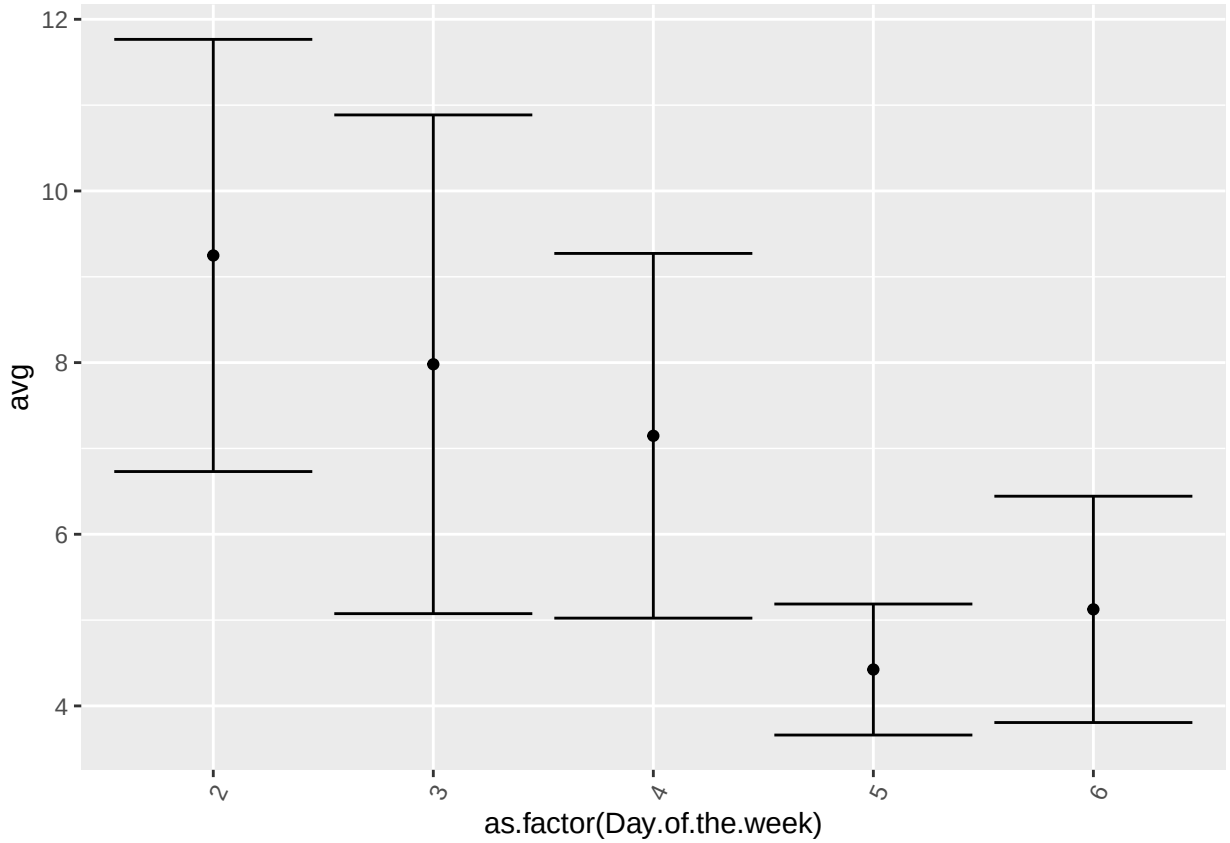
```
# create a vectore wiht varaible names
variable <- colnames(data)[1:20]
# initialize as a list
data1 <- list()
# create a loop that calculate target sd for each variable grouped
for(i in variable){
  f <- data %>%
    group_by_at(i) %>%
    summarise(median = median(Absenteeism.time.in.hours))
  data1[[i]] <- sd(f$median)
}

kable(arrange(tibble(var = names(data1),
                     absent.sd = unlist(data1)),
              desc(absent.sd)))
```

var	absent.sd
Reason.for.absence	5.3660948
ID	2.8522826
Distance.from.Residence.to.Work	2.8296054
Transportation.expense	2.7851021
Age	2.6544294

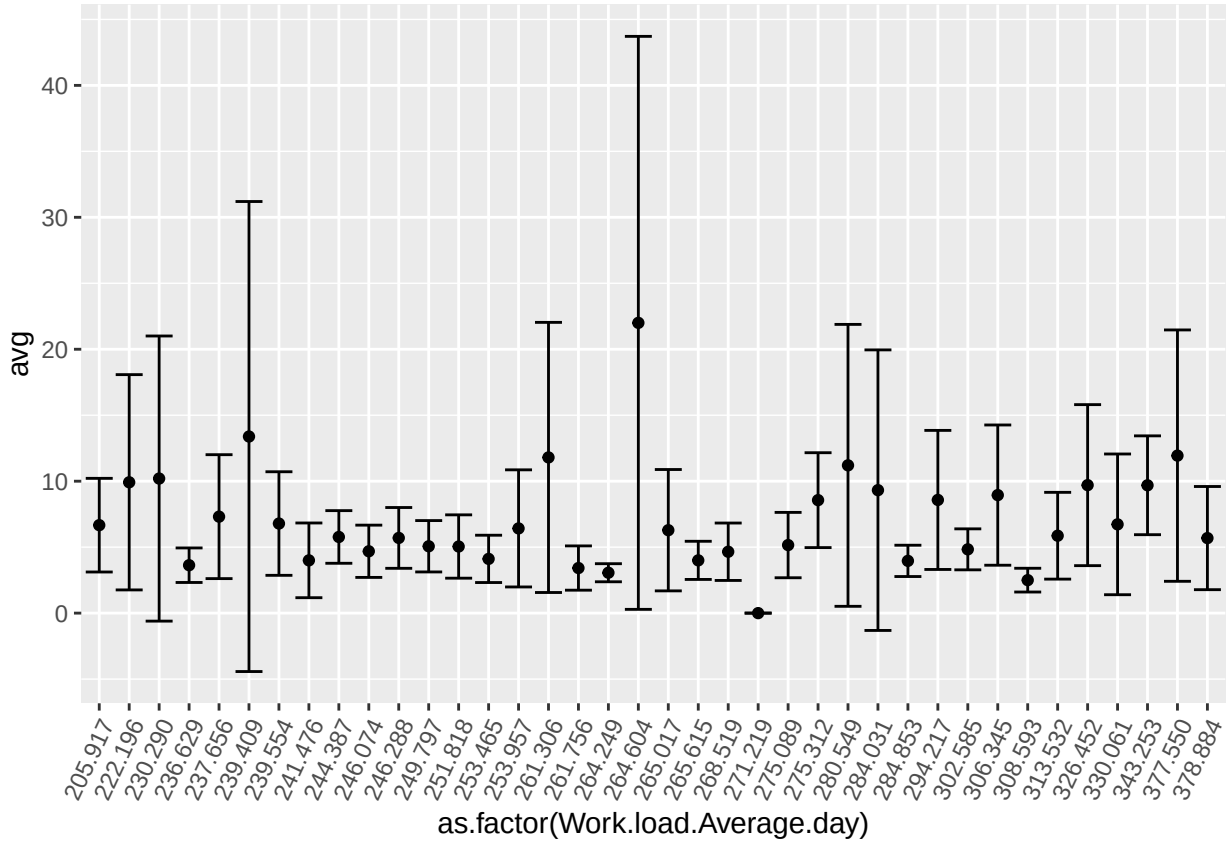
var	absent.sd
Son	2.5884358
Weight	2.5858342
Height	2.4842913
Body.mass.index	2.4796970
Education	2.3935678
Service.time	2.3916043
Disciplinary.failure	2.1213203
Pet	1.9429359
Work.load.Average.day	1.9105159
Month.of.absence	1.7394370
Hit.target	0.9674179
Social.drinker	0.7071068
Seasons	0.5773503
Day.of.the.week	0.4472136
Social.smoker	0.3535534

A couple of error bars will be made to check how some variables can affect the target.

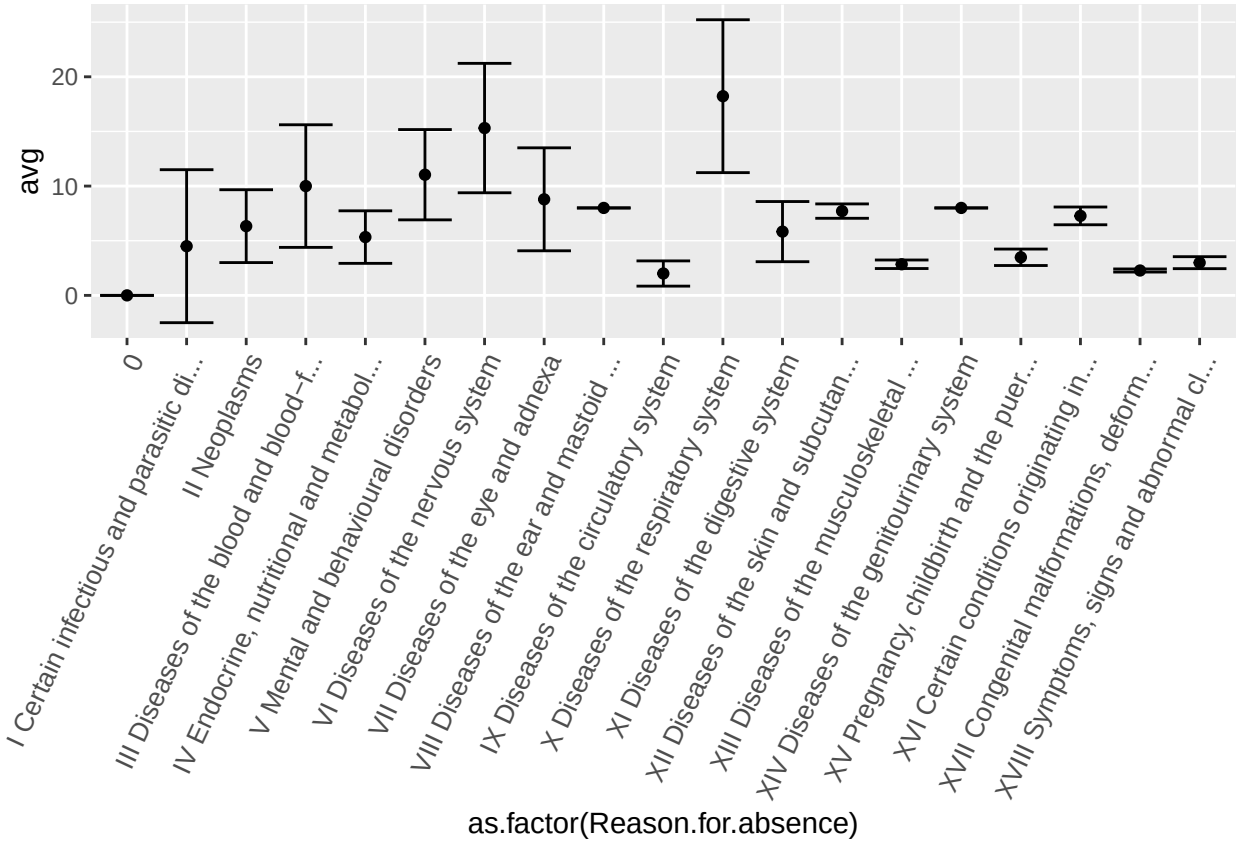
```
# error bar showing effect of week day on absenteeism
data %>% group_by(Day.of.the.week) %>%
  summarise(n = n(),
            avg = mean(Absenteeism.time.in.hours),
            sd = sd(Absenteeism.time.in.hours),
            se = sd/sqrt(n)) %>%
  ggplot(aes(as.factor(Day.of.the.week), avg)) + geom_point() +
  geom_errorbar(aes(ymin=avg - 2*se, ymax=avg+2*se)) +
  theme(axis.text.x = element_text(angle = 65, hjust = 1))
```




```
# error bar showing effect of Work load on absenteeism
data %>% group_by(Work.load.Average.day) %>%
  summarise(n = n(),
            avg = mean(Absenteeism.time.in.hours),
            sd = sd(Absenteeism.time.in.hours),
            se = sd/sqrt(n)) %>%
  ggplot(aes(as.factor(Work.load.Average.day), avg)) + geom_point() +
  geom_errorbar(aes(ymin=avg - 2*se, ymax=avg+2*se)) +
  theme(axis.text.x = element_text(angle = 65, hjust = 1))
```




```
# error bar showing effect of Reason for absence on absenteeism

data %>% group_by(Reason.for.absence) %>%
  summarise(n = n(),
            avg = mean(Absenteeism.time.in.hours),
            sd = sd(Absenteeism.time.in.hours),
            se = sd/sqrt(n)) %>%
  filter(se <= 3.5) %>%
  ggplot(aes(as.factor(Reason.for.absence), avg)) + geom_point() +
  scale_x_discrete(label = str_trunc(reason_n, 40)) +
  theme(axis.text.x = element_text(angle = 65, hjust = 1, size = 10)) +
  geom_errorbar(aes(ymin=avg - 2*se, ymax=avg+2*se))
```



Most of the averages' standard errors overlap. This shows that if each variable is used on its own, it will not be a good predictor in a liner model. When compared, the reason variable seems to give more information than two other variables.

Wrangling

Categorical data will be changed into class factor and the work load averages will be converted into a numerical vector.

```
# changing variable class according to type of data.
data_w <- data %>%
  mutate_at(c(1,2,3,4,5), as.factor) %>%
  mutate(Work.load.Average.day =
           as.numeric(levels(Work.load.Average.day))[Work.load.Average.day])
```

An index for numeric variables will be created in order ease analysis of numeric features.

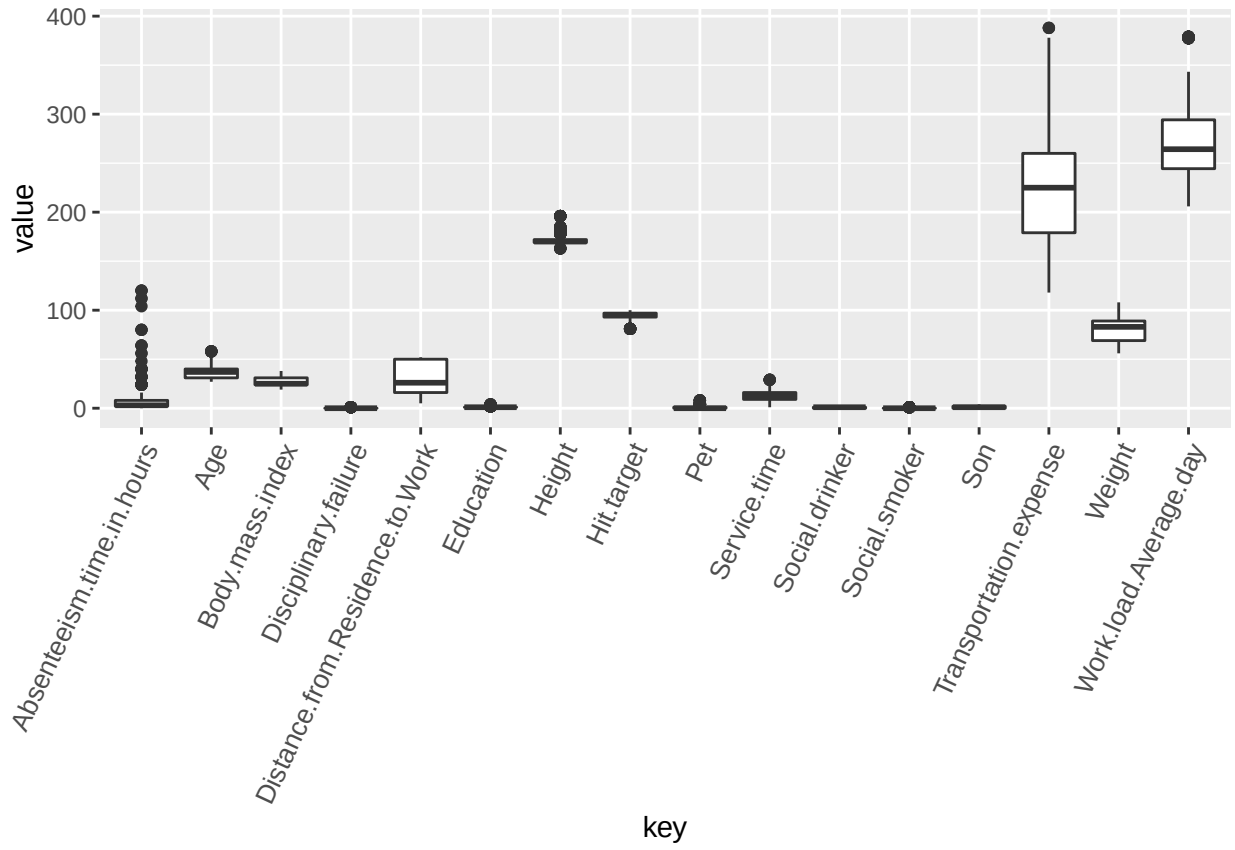
```
# create a list of numeric variables' indices
nums <- unlist(lapply(data_w, is.numeric))
```

Preprocessing

Outliers

A box plot will be created for all numerical variables to visualize outliers.

```
data_w %>% gather(key = "key", value ="value", -c(names(data_w[,!nums]))) %>%
  ggplot(aes(key,value)) + geom_boxplot() +
  theme(axis.text.x = element_text(angle = 65, hjust = 1, size = 10))
```



Variables that had visible outliers in the previous plot will be examined

```
summary(data_w$Height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   163.0   169.0   170.0   172.1   172.0   196.0
```

```
summary(data_w$Hit.target)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   81.00   93.00   95.00   94.59   97.00  100.00
```

```
summary(data_w$Service.time)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    9.00   13.00   12.55   16.00   29.00
```

```
summary(data_w$Absenteeism.time.in.hours)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.000   3.000   6.924   8.000 120.000
```

Outliers can decrease model accuracy. A function that can process the outliers will be created. It will be called caps_outliers().

For the purpose of this project, outliers will be defined according to Tukey Fences.

Outliers will be defined as values falling below Q1-1.5(Q3-Q1) or above Q3+1.5(Q3-Q1) where Q1 and Q3 represent the 1st and the 3rd quartiles respectively. For more information, click (here).

Outliers will be exchanged with values from the 5th and the 95th percentiles. This process can be called capping.

Any changes to the target variable should be done with care. Depending on the purpose of the project, this can artificially increase accuracy because the target will not represent real world data any more. We will basically be predicting a new version of the data that is trimmed and relatively easy to predict.

In this project, outliers from the target variable will not be capped.

```
# create a function that caps outliers.
caps_outliers <- function(x){
  qnt <- quantile(x, c(0.25, 0.75))
  cap <- quantile(x, c(0.05, 0.95))
  fence <- 1.5 * IQR(x)
  x[x < (qnt[1] - fence)] <- cap[1]
  x[x > (qnt[2] + fence)] <- cap[2]
  x
}

# create a new numerical variables index that doesn't include the target
nums_t <- nums
nums_t[21] <- FALSE

# use the new function to cap outliers from all numerical variables except target
data_w[,nums_t] <- apply(data_w[,nums_t],2,caps_outliers)
```

Scaling

Numerical variables will be scaled to avoid possible model bias towards variables with longer ranges. All variable will be represented with values from 0 to 1.

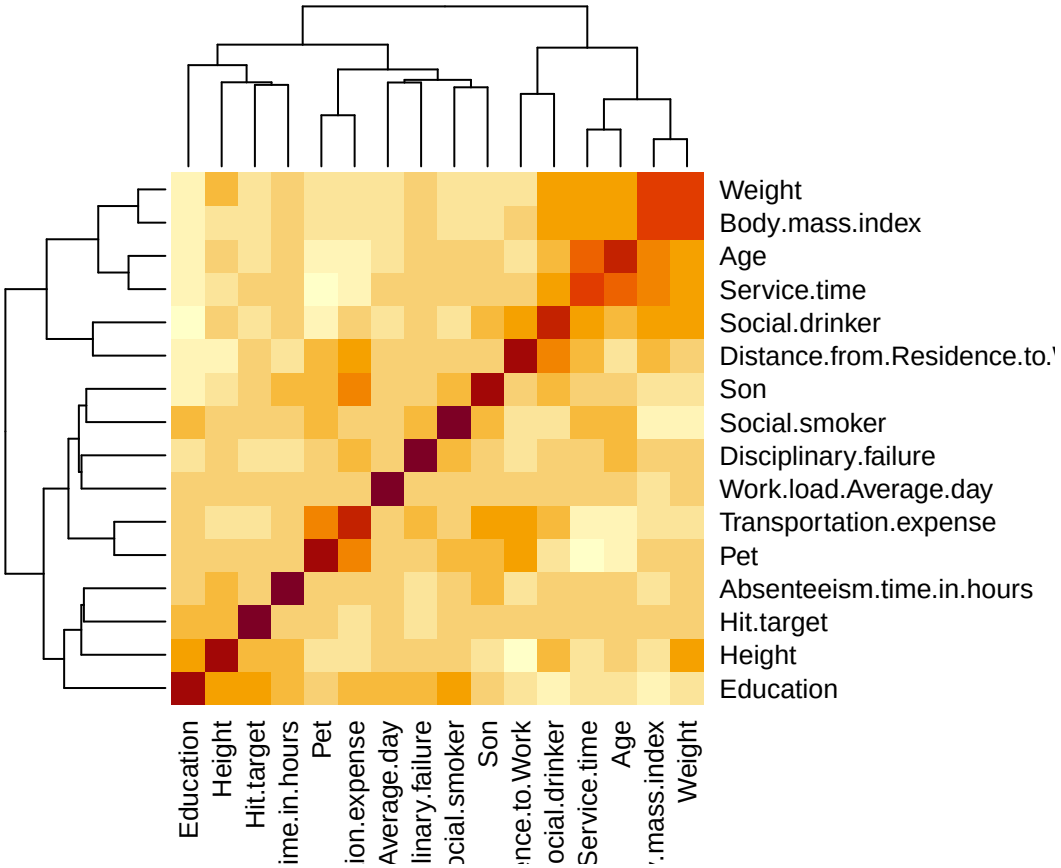
```
# rescale numerical variable to have a value from 0 to 1.
data_w[,nums] <- apply(data_w[,nums], 2, scales::rescale)
```

Multicollinearity

Multicollinearity can be thought of as high correlation between two or more features. It can reduce prediction accuracy as it makes isolating the effect of each variable on the target less accurate.

A Correlation matrix will be created and visualized as a heat map.

```
# create correlation matrix
cor.dat <- cor(data_w[,nums])

# create heat map
heatmap(cor.dat)
```




```
# check correlation between BMI and weight
with(data_w,c(cor(Weight, Body.mass.index)))
```

```
## [1] 0.9041169
```

Since weight and body mass index are highly correlated, the body mass index variable will be removed to avoid collinearity. Intuitively, it is calculated from weight and height and as both are avaible in the data, it was removed.

```
# remove BMI due to high correlation with weight
data_w$Body.mass.index <- NULL
```

Encoding catogorical variable

Many machine learning algorithms do not support categorical values. All categorical variables will binarized using one hot encoding technique. Each categorical variable will be represented with a number of new variables equal to the number of levels of the original variable. Each new variable will hold the value of either one and zero to represent the presence or absence of each specific level from each observation.

```
data_w <- dummy_cols(data_w, remove_selected_columns = TRUE)

dim(data_w)
```

```
## [1] 740 101
```

Due to binarization, the data dimensions increased 5 folds from 20 to 101 predictors. High dimensional data can result in low preforming machine learning algorithms.

Principal Component Analysis (PCA)

PCA is a dimension reduction technique that transforms the data into a set of values of linearly uncorrelated variables called principal components. The first principal component has the highest variance, and each following component has the highest possible variance while being orthogonal to the previous components. It can be used to reduce dimensions without losing data variability.

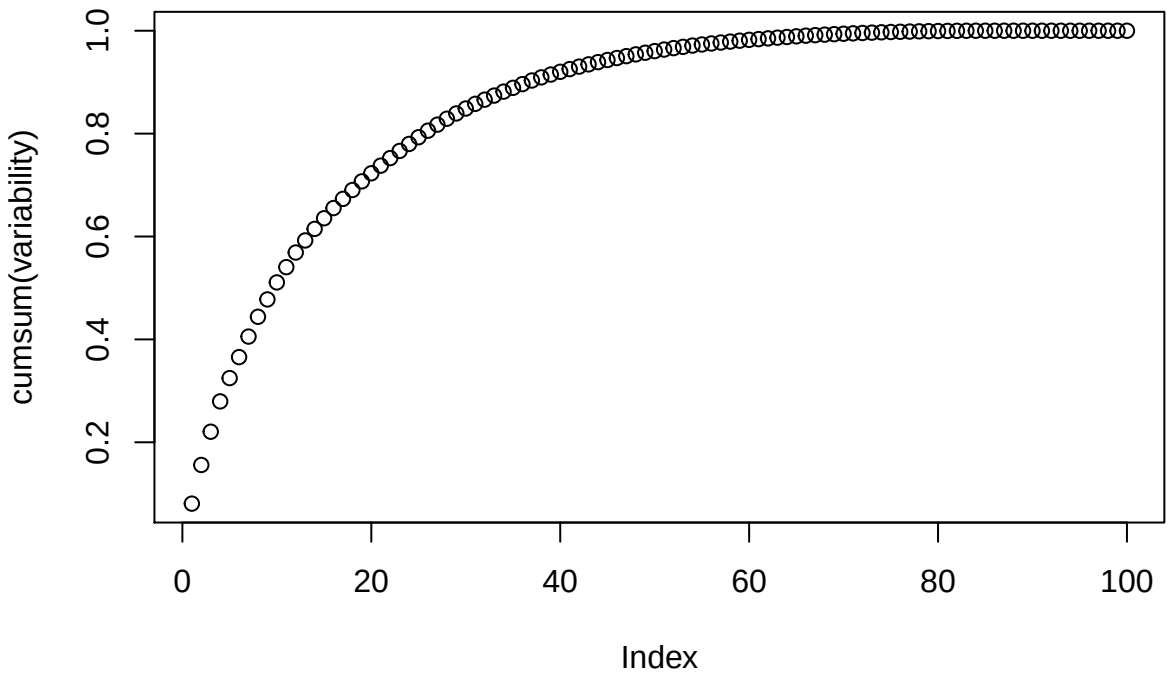
By reducing dimensions, it might be possible to get a more accurate model. The downside is that some of the interpretability of the model is lost due to the transformed nature of the features. If PCA technique is used , it not possible to calculate variable importance in predicting the target.

The target variable will be separated from the data before transformation.

```
# remove target from data
data_w$Absenteeism.time.in.hours -> y
data_w$Absenteeism.time.in.hours <- NULL

# calculate the principal components of the rest of the data
pca <-  prcomp(data_w)
dim(pca$x)
```

```
## [1] 740 100
```

```
# Proportion of variance calculated
variability <- pca$sdev^2/sum(pca$sdev^2)
# plot variability maintained vs number of components
plot(cumsum(variability))
```



```r
# check how much variability is kept if the first 45 components were used.
sum(variability[1:45])
```

```
## [1] 0.9431094
```

By using the first 45 components we keep about 94% of the variability.

```r
data_pc <- pca$x[,1:45]
```

Building Models

Partitioning the data

The data and the target will be split into a test and a train set the test set will be 20% of the data.

```r
# split data and target.
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y,times = 1, p = 0.2, list = FALSE)
train <- data_pc[-test_index,]
test <- data_pc[test_index,]
train_y <- y[-test_index]
test_y <- y[test_index]
```

Models

Training

A Loop will try to fit different models using the caret train function. The following is a brief explanation of each model: 1. **lm** stands for Linear Regression Model. 2. **mlp** stands for Multi-Layer Perceptron. It is part of The Stuttgart Neural Network Simulator (SNNS) library containing many standard implementations of neural networks. For more information click (here). 3. **knn** stands for the K-nearest Neighbors Algorithm. 4. **penalized"* stands for Penalized Linear Regression. Panelized regression models in general are models that are penalized for having too many variables, the regression coefficients are shrunk toward zero. For more information about this specific model click (here) 5.** krlsPoly** stands for Polynomial Kernel Regularized Least Squares. It fits multidimensional data for regression and classification problems without relying on linearity or additivity assumptions. For more information click ([here(https://cran.r-project.org/web/packages/KRLS/KRLS.pdf)). 6. **foba** stands for Greedy variable selection for ridge regression. As the name describes, it uses a ridge regression penalty model to select variables. For more information click (here). 7. **BstLm** stands for Boosted Linear Model. Boosted models are generally ensembles of several models. For more information click (here).

```r
# list models
models <- c("lm","mlp","knn", "penalized","krlsPoly","foba","BstLm")

# train models on the train set
fits <- lapply(models, function(model){
  set.seed(2, sample.kind = "Rounding")
  print(model)
  caret::train(train,train_y, method = model)
})
names(fits) <- models
```

Prediction

Predict test set's target using each model.

```
# predict using models
y_hat <- sapply(fits, function(x){
  predict(x, test)
})
```

Accuracy

The accuracy of each model will be calculated using Root Mean Square Error (RMSE).

```
# calculate RMSEs
rmses <- apply(y_hat, 2, function(x){
  RMSE(x,test_y)
})
rmses
```

```
##         lm        mlp        knn  penalized   krlsPoly       foba      BstLm
## 0.09277300 0.09650409 0.09389580 0.08909802 0.09449498 0.08910233 0.08887852
```

RMSEs are difficult to interpret due the scaled nature of the target.

Ensemble

Predictions from the best three models will be averaged into an ensemble.

```
# Ensembles by mean
ensemble <- apply(y_hat[,c(4,6,7)], 1, mean)

RMSE(ensemble, test_y)
```

```
## [1] 0.08842436
```

The ensemble has a lower rmse than the best model.

Results

RMSE will be rescaled to the original range by reversing the min max scaling formula. This is done to improve interpretability and to get insights from those predictions.

```
# unscaled target
real_y <- data$Absenteeism.time.in.hours

# rescale models' RMSEs to real scale
rmses_res <- rmses*(max(real_y)-min(real_y))-min(real_y)
rmses_res
```

```
##        lm       mlp       knn penalized  krlsPoly      foba     BstLm
##  11.13276  11.58049  11.26750  10.69176  11.33940  10.69228  10.66542
```

```
# rescale ensemble RMSE
paste("ensemble",
      round(RMSE(ensemble, test_y)*(max(real_y)-min(real_y))-min(real_y),
            digits = 4))
```

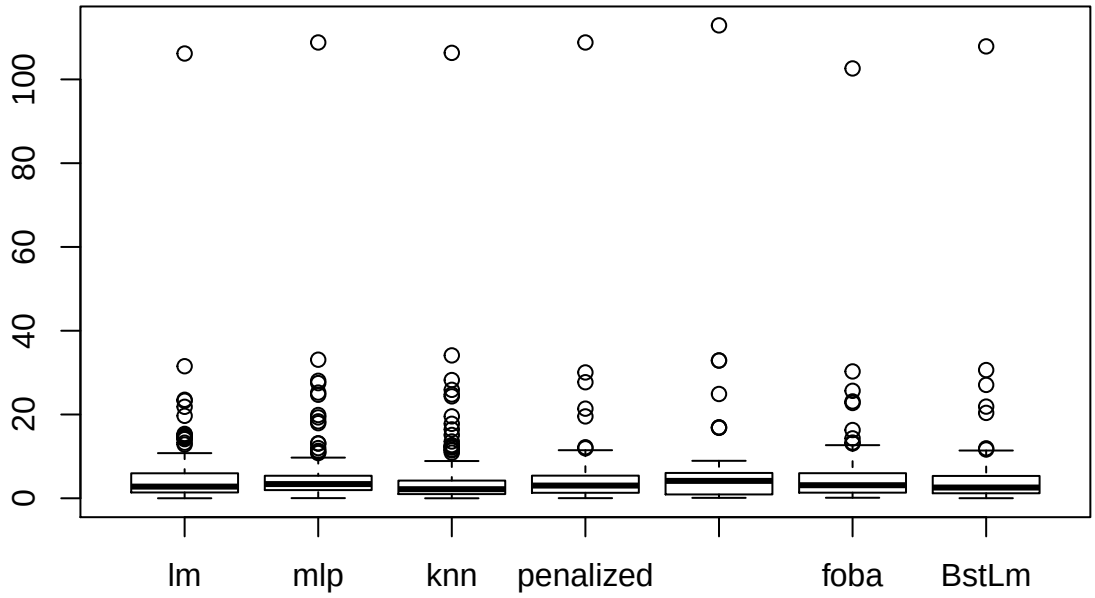
```
## [1] "ensemble 10.6109"
```

Models' accuracies range from 10.6654219 to 11.5804909 That would be the average error that we might get in our prediction measured in hours of absenteeism per month. The ensemble gave the least RMSE.

Explore model weaknesses

An attempt to further study the models will be made. The relationship between the target variable and the amount of error per model will be explored.

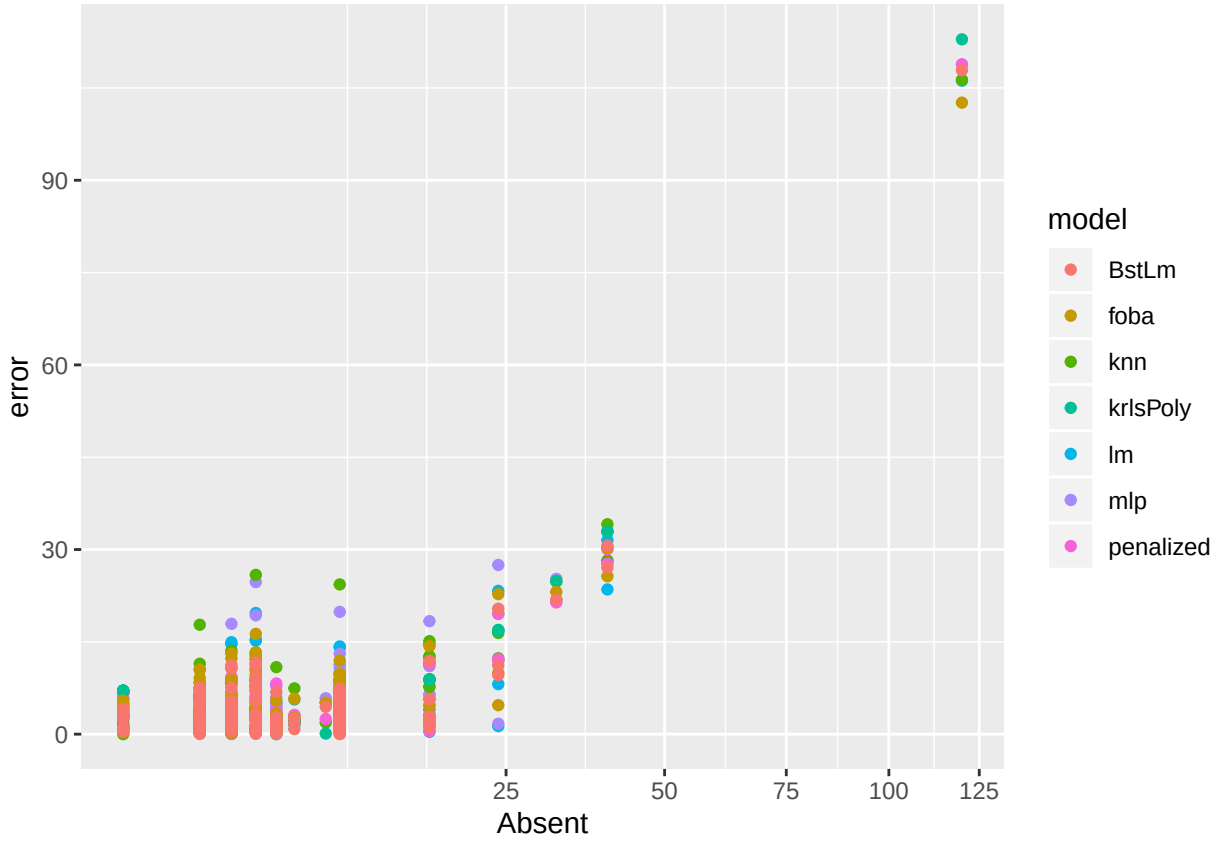
```
# predictions will be rescaled to original scale
y_hats_rescaled <- y_hat*(max(real_y)-min(real_y))-min(real_y)

# error vectors will be created to measure each models predction error
error <- as.data.frame(abs(y_hats_rescaled - test_y*(max(real_y)-min(real_y))-min(real_y)))
boxplot(error)
```



The boxplot shows that for each model, there is one value with an extremely high error in prediction.

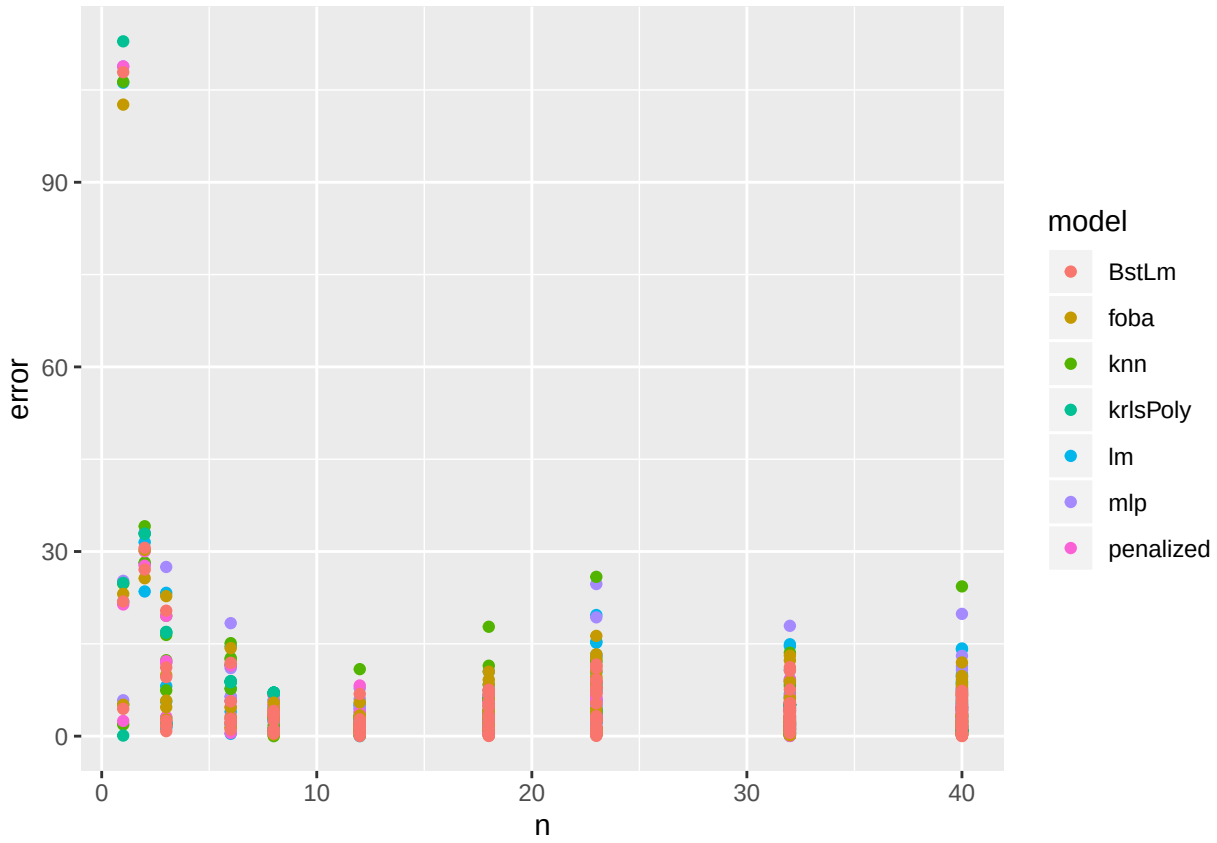
The amount of error made will be plotted against absenteeism in hours

```
# unscaled test set target
Absent <- data$Absenteeism.time.in.hours[test_index]

# bind errors and target
test_ns <- cbind(Absent,error)

# plot errors against test target
test_ns %>% gather(key = "model", value = "error", -Absent) %>%
  ggplot(aes(x = Absent, y = error, color = model)) +
  geom_point() + scale_x_sqrt()
```



We can see that for workers with 120 hours of absenteeism, all of our models' performances were highly inaccurate.

```
# plot errors against amount of data on Absenteeism
test_ns  %>%  group_by(Absent)  %>%  mutate(n = n()) %>% ungroup() %>%
  gather(key = "model", value = "error", -c(n,Absent)) %>%
  ggplot(aes(x = n, y = error, color = model)) +
  geom_point()
```



This plot shows that the lack of accuracy for this subgroup is actually related to the lack of data about high absenteeism.

Conclusion

Low amount of data on high absenteeism limited all models' performances. In order to be able to use some models that only take numerical variables, we had to encode categorical variables and in turn reduce dimensions using PCA. This process took away our ability to check feature importance and further understand the data.

Limitations and future work

Using other models that can handle categorical features can probably improve our understanding of what increases absenteeism for employees. Further tuning of parameters for the used models can probably improve reduces error. It couldn't be done in this project due to time constraints.

Ethical considerations

This project was done for educational and training purposes. It will never be used for any practical decision making. Using data on workers weights and number of sons to predict their absenteeism sounds appalling and can lead to real discrimination in the work place and in the hiring process. Reasonable ethical discretion should always be used when collecting data and building machine learning algorithms.

References and Citations

1. Introduction to Data Science by Rafael A. Irizarry (here).
2. L1 penalized estimation in the Cox proportional hazards model by Goeman, J. J., Biometrical Journal 52(1), 70–84. (here)
3. Exploratory Data Analysis by Tukey, John W. (here)
4. Feature Normalization and Likelihood-based Similarity Measures for Image Retrieval by Selim Aksoy and Robert M. Haralick. (here)
5. Martiniano, A., Ferreira, R. P., Sassi, R. J., & Affonso, C. (2012). Application of a neuro fuzzy network in prediction of absenteeism at work. In Information Systems and Technologies (CISTI), 7th Iberian Conference on (pp. 1-4). IEEE.