HW 1

Dutchak Bohdan

Hide Assignment Information Instructions Working with Individual Classifiers

This assignment is worth 100 points, which is 20% of the overall course grade, This assignment is to be completed individually. Please consult the course syllabus for a description of our academic honesty policy.

Consider the dataset that you performed EDA on, your task is to do the following classification tasks

- Logistic Regression or Naive Bayes or LDA/QDA
- Support Vector Machine
- Decision Tree
- K-Nearest Neighbor

Now determine the performances using the following

- Roc Plot
- AUC
- Confusion Matrix
- Accuracy
- Specificity
- Precision
- Recall (Sensitivity)

Lastly, Calculate the Variance and bias for these algorithms, com pare and discuss the difference.

```
library(MASS)
library(ggplot2)
library(GGally)
library(ggcorrplot)
library(e1071)
library(class)
library(dplyr)
library(foreign)
library(nnet)
library(tidyverse)
library(caret)
library(party)
library(pROC)
library(cvms)
library(tibble)
library(mltest)
library(Metrics)
library(boot)
```

```
options(warn = -1)
data <- read.csv("data/all_seasons.csv")
head(data)
```

	Player	Pos	Ht	Wt	Exp	Age	G	GS	MP
	<chr>	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<dbl>
1	Nick Anderson	SG	198	93	10	32	72	72	29.1
2	Jon Barry	SG	193	88	7	30	62	1	20.7
3	Tyrone Corbin	SF	198	95	14	37	54	5	17.4
4	Tony Delk	PG	185	86	3	26	46	1	14.8
5	Vlade Divac	C	216	110	10	31	82	81	29.0
6	Lawrence Funderburke	PF	206	104	2	29	75	1	13.7

6 rows | 1-10 of 33 columns

```
data$Pos <- replace(data$Pos, data$Pos=='C', 5)
data$Pos <- replace(data$Pos, data$Pos=='PF', 4)
data$Pos <- replace(data$Pos, data$Pos=='SF', 3)
data$Pos <- replace(data$Pos, data$Pos=='SG', 2)
data$Pos <- replace(data$Pos, data$Pos=='PG', 1)
data$Pos <- as.factor(data$Pos)

data <- data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

str(data)
```

```
## 'data.frame':    11071 obs. of  7 variables:
##  $ Pos : Factor w/ 5 levels "1","2","3","4",..: 2 2 3 1 5 4 1 5 2 3 ...
##  $ Ht  : int  198 193 198 185 216 206 180 211 196 208 ...
##  $ Wt  : int  93 88 95 86 110 104 77 120 86 104 ...
##  $ AST : num  1.7 2.4 1.1 1.2 3 0.4 1.7 0.6 0 1.4 ...
##  $ X3PA: num  5.5 2.5 0.8 0.9 0.3 0 1.7 0 2 3.6 ...
##  $ ORB : num  1.2 0.6 0.7 0.8 2.1 1.3 0.1 2.2 0 1 ...
##  $ BLK : num  0.2 0.1 0.1 0.1 1.3 0.3 0 0.8 0 0.1 ...
```

```
set.seed(1)

sample <- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.8,0.2))
train <- data[sample, ]
test <- data[!sample, ]

test.X <- test[2:7]
test.Y <- test[1]

train.X<-train[2:7]
train.Y<-train[1]

message("Shape of test dataframe is ", dim(test)[1], 'x', dim(test)[2])
```

```
## Shape of test dataframe is 2281x7
```

```
message("Shape of train dataframe is ", dim(train)[1], 'x', dim(train)[2])
```

```
## Shape of train dataframe is 8790x7
```

Logit

Since EDA showed, that some features have significant correlation with the target variable, have good distribution and dataset has many observations, I decided to use LOGIT.

```
logit.fit <- multinom(Pos ~ ., data = train)
```

```
logit.pred <- logit.fit %>% predict(test.X)
summary(logit.fit)
```

```
## Call:
## multinom(formula = Pos ~ ., data = train)
##
## Coefficients:
##   (Intercept)        Ht         Wt        AST       X3PA      ORB      BLK
## 2   -51.32427 0.2272984 0.08933343 -0.9690196 0.63144896 1.073986 2.061825
## 3  -133.06304 0.5909153 0.18620138 -1.4008063 0.62531229 1.819329 2.947120
## 4  -205.57534 0.8605128 0.35445186 -2.0284141 0.51189160 2.729614 3.673918
## 5  -286.20298 1.2047126 0.43556042 -2.4483612 0.03091912 2.844946 4.426758
##
## Std. Errors:
##    (Intercept)          Ht          Wt        AST       X3PA       ORB
## 2 0.0387366828 0.004574189 0.009898202 0.04938410 0.03943291 0.2097763
## 3 0.0026560960 0.005881311 0.012518372 0.06635111 0.04914258 0.2401554
## 4 0.0041163848 0.007015817 0.014623873 0.08490815 0.05782517 0.2516934
## 5 0.0009175398 0.007564067 0.015582315 0.10335588 0.07378147 0.2588006
##         BLK
## 2 0.3968997
## 3 0.4612620
## 4 0.4830126
## 5 0.4912126
##
## Residual Deviance: 11283.17
## AIC: 11339.17
```

```
population.logit <- multinom(Pos ~ ., data = data)
```

```
population.pred <- population.logit %>% predict(test.X)
```

Attention, the block belows computes ROC curves and AUC. It contains a lot of code and takes a lot of kernel memory to process, so I did it only for LOGIT. For the same reason I chose One VS Rest approach in order to save time and space. Here I got only 5 plots, but OvO approach would require 20.

ROC Curve and AUC

```
temp_data <- read.csv("data/all_seasons.csv")
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='C', 1)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SG', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PG', 0)
temp_data$Pos <- as.factor(temp_data$Pos)
temp_data <- temp_data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

sample <- sample(c(TRUE, FALSE), nrow(temp_data), replace=TRUE, prob=c(0.8,0.2))
temp_train <- temp_data[sample, ]
temp_test <- temp_data[!sample, ]
temp_test.X <- temp_test[2:7]
temp_test.Y <- temp_test[1]
temp_train.X<-temp_train[2:7]
temp_train.Y<-temp_train[1]

logit5.fit <- glm(Pos~., data=temp_train, family = binomial)
logit5.pred <- predict(logit5.fit, temp_test.X, type="response")
logit5.pred <- ifelse(test=logit5.pred>0.5, yes=1, no=0)

roc5 <- roc(temp_test.Y$Pos, logit5.pred, plot=TRUE)
```

```
message("AUC: ", auc(temp_test.Y$Pos, logit5.pred))
```

```
## AUC: 0.869365747414528
```

```r
temp_data <- read.csv("data/all_seasons.csv")
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='C', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PF', 1)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SG', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PG', 0)
temp_data$Pos <- as.factor(temp_data$Pos)
temp_data <- temp_data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

sample <- sample(c(TRUE, FALSE), nrow(temp_data), replace=TRUE, prob=c(0.8,0.2))
temp_train <- temp_data[sample, ]
temp_test <- temp_data[!sample, ]
temp_test.X <- temp_test[2:7]
temp_test.Y <- temp_test[1]
temp_train.X<-temp_train[2:7]
temp_train.Y<-temp_train[1]

logit4.fit <- glm(Pos~., data=temp_train, family = binomial)
logit4.pred <- predict(logit4.fit, temp_test.X, type="response")
logit4.pred <- ifelse(test=logit4.pred>0.4, yes=1, no=0)

roc4 <- roc(temp_test.Y$Pos, logit4.pred, plot=TRUE)
```

```r
message("AUC: ", auc(temp_test.Y$Pos, logit4.pred))
```

```
## AUC: 0.534073836829405
```

```r
temp_data <- read.csv("data/all_seasons.csv")
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='C', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SF', 1)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SG', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PG', 0)
temp_data$Pos <- as.factor(temp_data$Pos)
temp_data <- temp_data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

sample <- sample(c(TRUE, FALSE), nrow(temp_data), replace=TRUE, prob=c(0.8,0.2))
temp_train <- temp_data[sample, ]
temp_test <- temp_data[!sample, ]
temp_test.X <- temp_test[2:7]
temp_test.Y <- temp_test[1]
temp_train.X<-temp_train[2:7]
temp_train.Y<-temp_train[1]

logit3.fit <- glm(Pos~., data=temp_train, family = binomial)
logit3.pred <- predict(logit3.fit, temp_test.X, type="response")
logit3.pred <- ifelse(test=logit3.pred>0.3, yes=1, no=0)

roc3 <- roc(temp_test.Y$Pos, logit3.pred, plot=TRUE)
```

```
message("AUC: ", auc(temp_test.Y$Pos, logit3.pred))
```

```
## AUC: 0.573637662249855
```

```
temp_data <- read.csv("data/all_seasons.csv")
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='C', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SG', 1)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PG', 0)
temp_data$Pos <- as.factor(temp_data$Pos)
temp_data <- temp_data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

sample <- sample(c(TRUE, FALSE), nrow(temp_data), replace=TRUE, prob=c(0.8,0.2))
temp_train <- temp_data[sample, ]
temp_test <- temp_data[!sample, ]
temp_test.X <- temp_test[2:7]
temp_test.Y <- temp_test[1]
temp_train.X<-temp_train[2:7]
temp_train.Y<-temp_train[1]

logit2.fit <- glm(Pos~., data=temp_train, family = binomial)
logit2.pred <- predict(logit2.fit, temp_test.X, type="response")
logit2.pred <- ifelse(test=logit2.pred>0.2, yes=1, no=0)

roc2 <- roc(temp_test.Y$Pos, logit2.pred, plot=TRUE)
```

```
message("AUC: ", auc(temp_test.Y$Pos, logit2.pred))
```

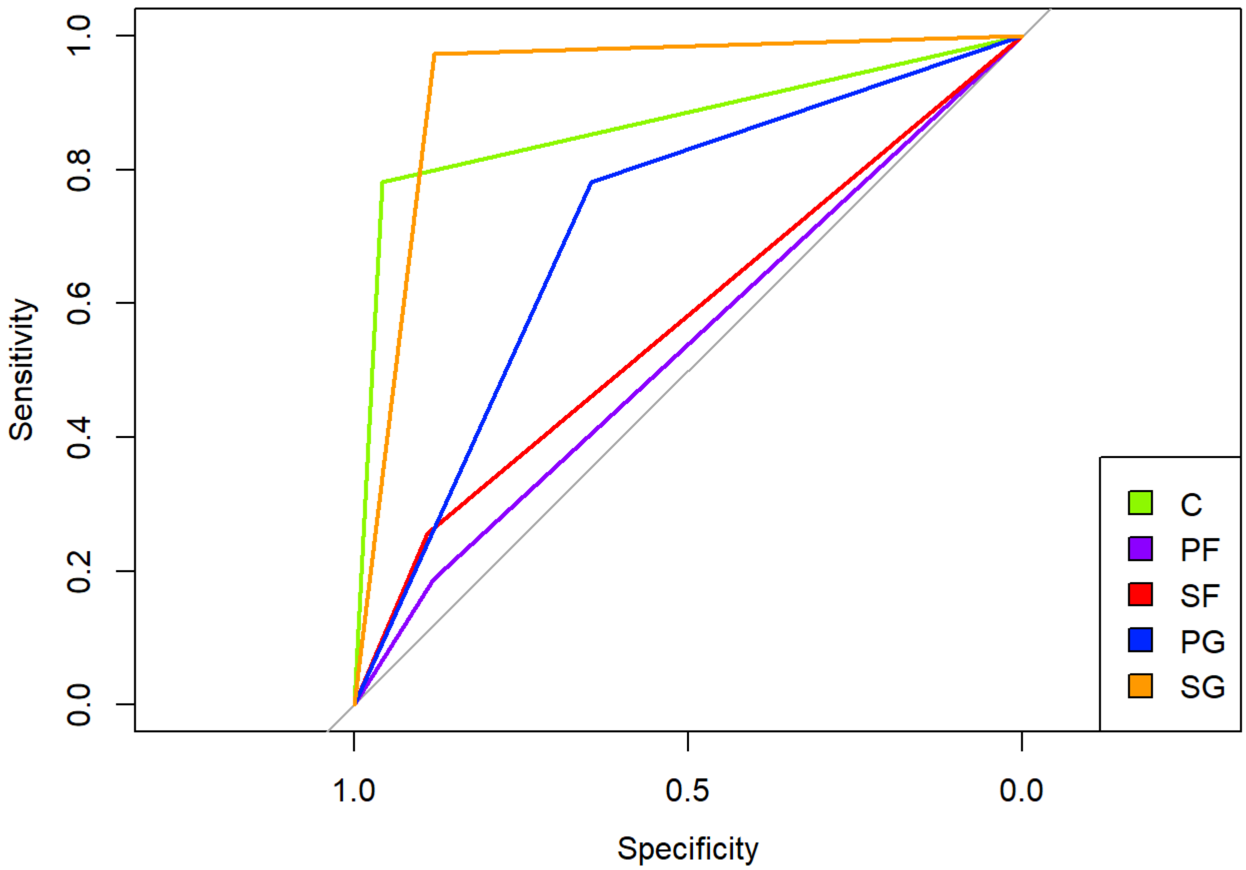
```
## AUC: 0.712834758833427
```

```
temp_data <- read.csv("data/all_seasons.csv")
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='C', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SF', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='SG', 0)
temp_data$Pos <- replace(temp_data$Pos, temp_data$Pos=='PG', 1)
temp_data$Pos <- as.factor(temp_data$Pos)
temp_data <- temp_data[,c('Pos','Ht','Wt','AST','X3PA','ORB','BLK')]

sample <- sample(c(TRUE, FALSE), nrow(temp_data), replace=TRUE, prob=c(0.8,0.2))
temp_train <- temp_data[sample, ]
temp_test <- temp_data[!sample, ]
temp_test.X <- temp_test[2:7]
temp_test.Y <- temp_test[1]
temp_train.X<-temp_train[2:7]
temp_train.Y<-temp_train[1]

logit1.fit <- glm(Pos~., data=temp_train, family = binomial)
logit1.pred <- predict(logit1.fit, temp_test.X, type="response")
logit1.pred <- ifelse(test=logit1.pred>0.1, yes=1, no=0)

roc1 <- roc(temp_test.Y$Pos, logit1.pred, plot=TRUE)
```

```
message("AUC: ", auc(temp_test.Y$Pos, logit1.pred))
```

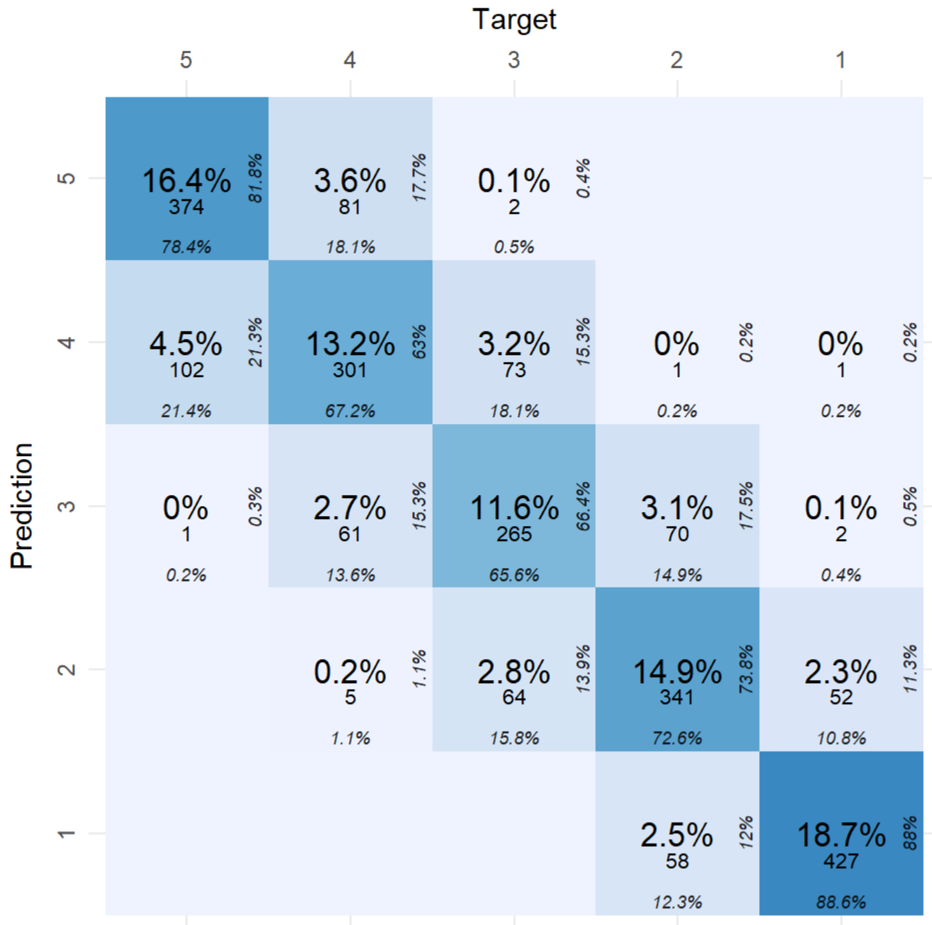
```
## AUC: 0.925752913312346
```

```
plot(roc5, col='#8df801')
lines(roc4, col='#8c00ff')
lines(roc3, col='#ff0000')
lines(roc2, col='#0026ff')
lines(roc1, col='#ff9900')
legend(x='bottomright',legend=c('C','PF','SF','PG','SG'),fill=c('#8df801','#8c00ff','#ff0000','#
0026ff','#ff9900'))
```



As a result we can notice, that “marginal” positions (Center, Point guard) have the best results. It is because they are on the edges of the distributions. Center players tend to be the highest, the haviest, shoots the least 3’s, don’t have assists and have the most blocks and rebounds, while Point guards Vice Versa. Other Positions are between them, so their performance differ depending on the strategy, personal stats of the player etc.

```
plot_confusion_matrix(confusion_matrix(targets=test.Y$Pos, prediction=logit.pred))
```




```
stats <- ml_test(logit.pred, test.Y$Pos, output.as.table = FALSE)
logit.results <- data.frame(
    Accuracy = round(rep(stats$accuracy, 5),4),
    Specificity = round(stats$specificity,4),
    Precision = round(stats$precision,4),
    Sensitivity = round(stats$recall,4),
    Bias = rep(mean(abs(as.integer(population.pred) - as.integer(logit.pred))), 5),
    Variance = rep(var(as.integer(population.pred), as.integer(logit.pred)), 5)
)
message("Accuracy: ", round(stats$accuracy,4)*100, "%\n")
```

```
## Accuracy: 74.88%
```

```
logit.results
```

	Accuracy <dbl>	Specificity <dbl>	Precision <dbl>	Sensitivity <dbl>	Bias <dbl>	Variance <dbl>
1	0.7488	0.9567	0.8804	0.8859	0	2.064605
2	0.7488	0.9187	0.7381	0.7255	0	2.064605
3	0.7488	0.9150	0.6642	0.6559	0	2.064605

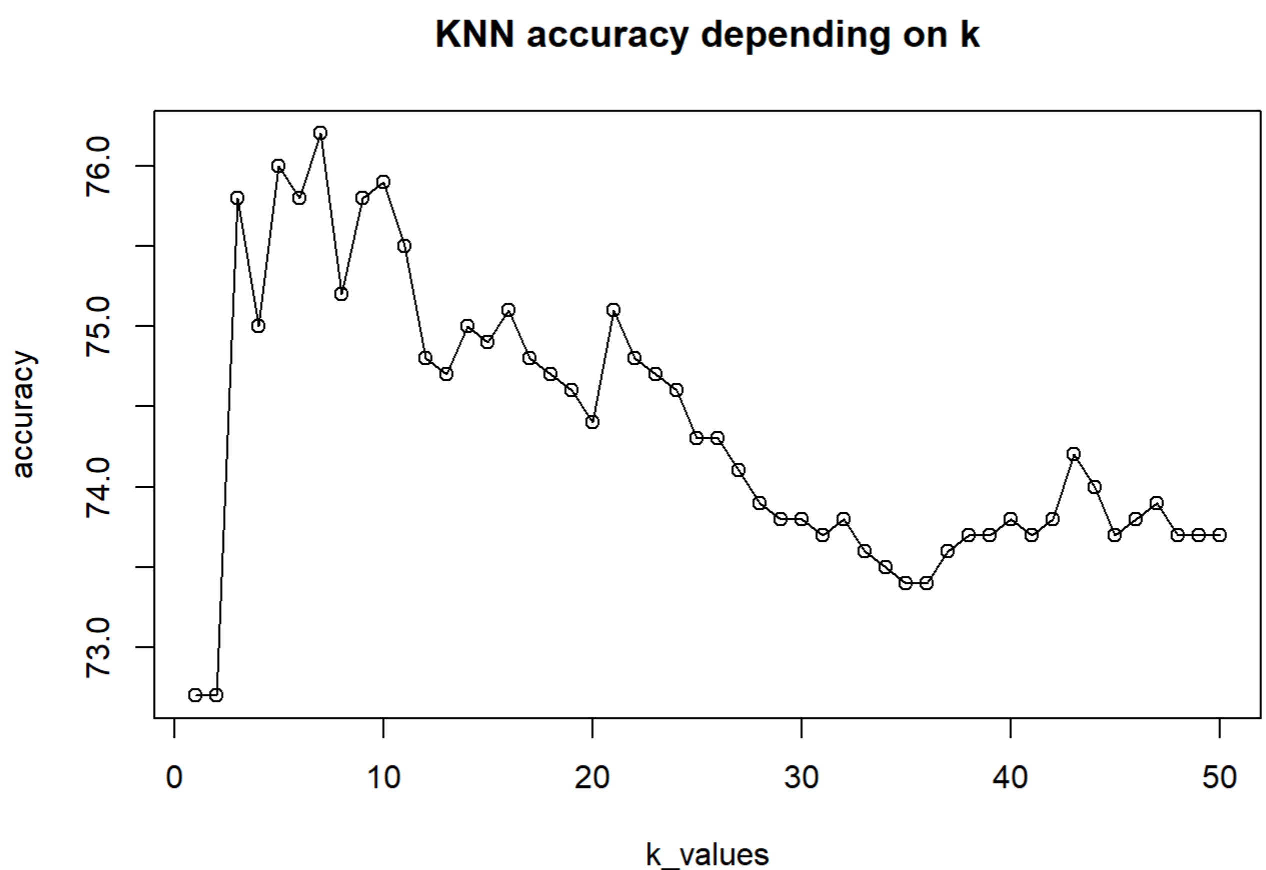
	Accuracy <dbl>	Specificity <dbl>	Precision <dbl>	Sensitivity <dbl>	Bias <dbl>	Variance <dbl>
4	0.7488	0.8883	0.6297	0.6719	0	2.064605
5	0.7488	0.9414	0.8184	0.7841	0	2.064605

5 rows

KNN

```
n = 50
k_values = seq(1,n,by=1)
accuracy = rep(0,n)

for (i in 1:n){
    knn.pred <- knn(train.X, test.X, cl=train.Y$Pos, k=i)
    accuracy[i] = round(mean(knn.pred == test.Y$Pos),3)*100
}
```

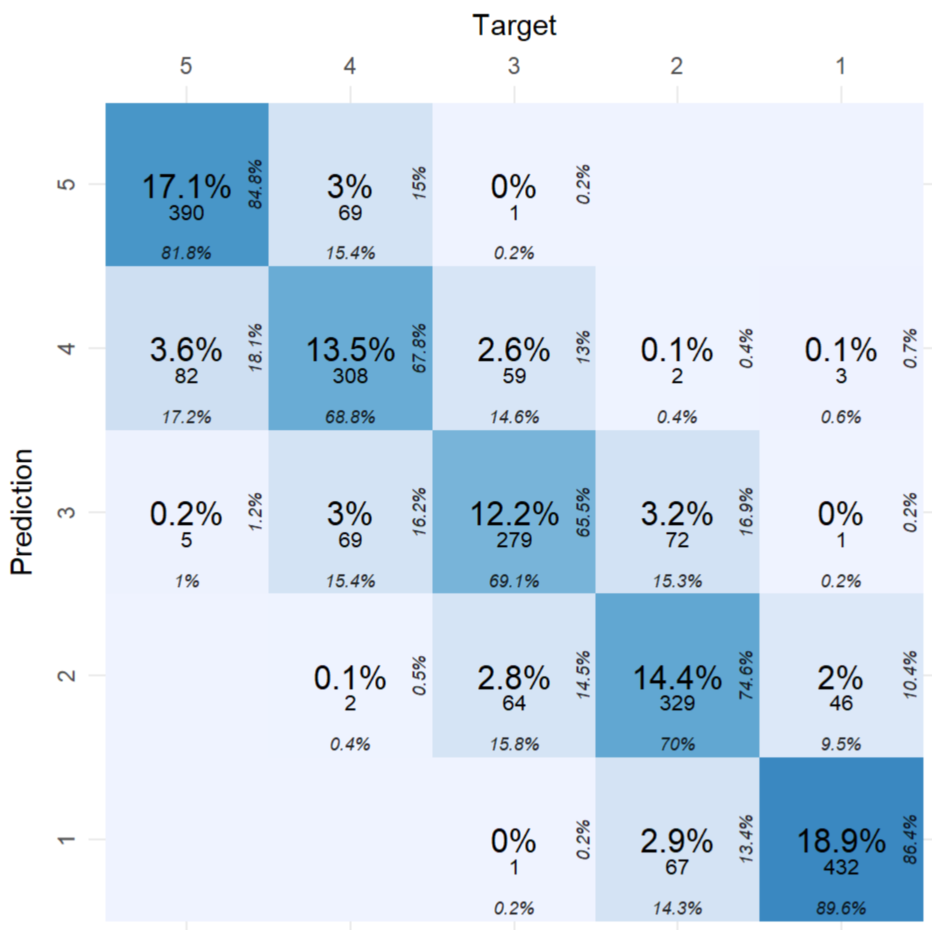
```
plot(k_values, accuracy, type="o", main="KNN accuracy depending on k", ylim=c(min(accuracy),max
(accuracy)))
```



Very flexible model with $k = 7$ shows the best results. I will use it in further computations.

```
knn.pred <- knn(train.X, test.X, cl=train.Y$Pos, k=7)
population.knn <- knn(data[2:7], test.X, cl=data$Pos, k=7)
```

```
plot_confusion_matrix(confusion_matrix(targets=test.Y$Pos, prediction=knn.pred))
```




```
stats <- ml_test(knn.pred, test.Y$Pos, output.as.table = FALSE)
knn.results <- data.frame(
    Accuracy = round(rep(stats$accuracy, 5),4),
    Specificity = round(stats$specificity,4),
    Precision = round(stats$precision,4),
    Sensitivity = round(stats$recall,4),
    Bias = rep(mean(abs(as.integer(population.knn) - as.integer(knn.pred))), 5),
    Variance = rep(var(as.integer(population.knn), as.integer(knn.pred)), 5)
)
message("Accuracy: ", round(stats$accuracy,4)*100, "%\n")
```

```
## Accuracy: 76.19%
```

```
knn.results
```

Accuracy <dbl>	Specificity <dbl>	Precision <dbl>	Sensitivity <dbl>	Bias <dbl>	Variance <dbl>

	Accuracy	Specificity	Precision	Sensitivity	Bias	Variance
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.7619	0.9505	0.8640	0.8963	0.08154318	2.028007
2	0.7619	0.9264	0.7460	0.7000	0.08154318	2.028007
3	0.7619	0.9085	0.6549	0.6906	0.08154318	2.028007
4	0.7619	0.9074	0.6784	0.6875	0.08154318	2.028007
5	0.7619	0.9506	0.8478	0.8176	0.08154318	2.028007

5 rows

Decision Tree

```
tree <- ctree(Pos~., data=train)
tree
```

```
##
##   Conditional inference tree with 109 terminal nodes
##
## Response:  Pos
## Inputs:  Ht, Wt, AST, X3PA, ORB, BLK
## Number of observations:  8790
##
## 1) Ht <= 190; criterion = 1, statistic = 6935.294
##   2) AST <= 4.2; criterion = 1, statistic = 92.893
##     3) X3PA <= 6.3; criterion = 1, statistic = 117.591
##       4) X3PA <= 3.1; criterion = 1, statistic = 93.152
##         5) Ht <= 188; criterion = 1, statistic = 51.854
##           6) AST <= 1.5; criterion = 0.999, statistic = 13.946
##             7) Wt <= 85; criterion = 0.984, statistic = 9.002
##               8)*  weights = 164
##             7) Wt > 85
##               9)*  weights = 94
##           6) AST > 1.5
##             10) ORB <= 0.4; criterion = 0.996, statistic = 11.548
##               11)*  weights = 293
##             10) ORB > 0.4
##               12)*  weights = 75
##         5) Ht > 188
##           13) AST <= 1.2; criterion = 1, statistic = 25.357
##             14) Wt <= 87; criterion = 0.994, statistic = 13.971
##               15)*  weights = 60
##             14) Wt > 87
##               16)*  weights = 56
##           13) AST > 1.2
##             17) ORB <= 0.3; criterion = 1, statistic = 27.6
##               18)*  weights = 146
##             17) ORB > 0.3
##               19) BLK <= 0; criterion = 0.999, statistic = 17.556
##                 20)*  weights = 22
##               19) BLK > 0
##                 21)*  weights = 72
##       4) X3PA > 3.1
##         22) Ht <= 188; criterion = 1, statistic = 29.401
##           23) AST <= 2.2; criterion = 0.996, statistic = 11.483
##             24)*  weights = 35
##           23) AST > 2.2
##             25) X3PA <= 4.2; criterion = 0.989, statistic = 9.752
##               26)*  weights = 69
##             25) X3PA > 4.2
##               27)*  weights = 40
##         22) Ht > 188
##           28)*  weights = 79
##     3) X3PA > 6.3
##       29)*  weights = 14
##   2) AST > 4.2
##     30) BLK <= 0.7; criterion = 0.998, statistic = 13.023
##       31) AST <= 6.8; criterion = 0.988, statistic = 9.492
```

```
##         32) BLK <= 0.2; criterion = 0.987, statistic = 9.434
##           33)*  weights = 277
##         32) BLK > 0.2
##           34)*  weights = 99
##       31) AST > 6.8
##         35)*  weights = 147
##     30) BLK > 0.7
##       36)*  weights = 7
## 1) Ht > 190
##   37) Ht <= 206; criterion = 1, statistic = 4886.68
##     38) Ht <= 198; criterion = 1, statistic = 2757.636
##       39) Wt <= 104; criterion = 1, statistic = 553.447
##         40) Ht <= 196; criterion = 1, statistic = 390.633
##           41) Wt <= 89; criterion = 1, statistic = 175.933
##             42) AST <= 5; criterion = 1, statistic = 45.545
##               43) X3PA <= 1.2; criterion = 1, statistic = 25.696
##                 44) AST <= 0.6; criterion = 1, statistic = 21.426
##                   45)*  weights = 53
##                 44) AST > 0.6
##                   46) BLK <= 0.2; criterion = 0.98, statistic = 11.431
##                     47)*  weights = 81
##                   46) BLK > 0.2
##                     48)*  weights = 12
##               43) X3PA > 1.2
##                 49) AST <= 2.6; criterion = 0.999, statistic = 17.267
##                   50) BLK <= 0; criterion = 0.954, statistic = 9.703
##                     51)*  weights = 24
##                   50) BLK > 0
##                     52)*  weights = 113
##                 49) AST > 2.6
##                   53) Ht <= 193; criterion = 0.983, statistic = 11.742
##                     54)*  weights = 53
##                   53) Ht > 193
##                     55)*  weights = 37
##             42) AST > 5
##               56)*  weights = 33
##           41) Wt > 89
##             57) AST <= 7.5; criterion = 1, statistic = 91.499
##               58) AST <= 2.2; criterion = 1, statistic = 38.006
##                 59) ORB <= 0.8; criterion = 1, statistic = 45.427
##                   60) X3PA <= 0.9; criterion = 1, statistic = 25.763
##                     61) Ht <= 193; criterion = 0.999, statistic = 20.45
##                       62)*  weights = 57
##                     61) Ht > 193
##                       63)*  weights = 86
##                   60) X3PA > 0.9
##                     64) Wt <= 98; criterion = 0.982, statistic = 13.9
##                       65)*  weights = 260
##                     64) Wt > 98
##                       66) BLK <= 0.1; criterion = 0.985, statistic = 11.97
##                         67)*  weights = 36
##                       66) BLK > 0.1
```

```
##                            68)*  weights = 26
##                    59) ORB > 0.8
##                      69) Ht <= 193; criterion = 0.978, statistic = 11.187
##                        70)*  weights = 19
##                      69) Ht > 193
##                        71)*  weights = 56
##                  58) AST > 2.2
##                    72) X3PA <= 2.7; criterion = 0.999, statistic = 17.284
##                      73) BLK <= 0.1; criterion = 0.986, statistic = 12.067
##                        74)*  weights = 22
##                      73) BLK > 0.1
##                        75)*  weights = 41
##                    72) X3PA > 2.7
##                      76) AST <= 4.5; criterion = 0.968, statistic = 10.461
##                        77)*  weights = 85
##                      76) AST > 4.5
##                        78)*  weights = 19
##                57) AST > 7.5
##                  79)*  weights = 11
##            40) Ht > 196
##              80) Wt <= 97; criterion = 1, statistic = 46.362
##                81) AST <= 1.3; criterion = 0.963, statistic = 12.33
##                  82) ORB <= 0.2; criterion = 0.963, statistic = 10.12
##                    83)*  weights = 70
##                  82) ORB > 0.2
##                    84)*  weights = 112
##                81) AST > 1.3
##                  85)*  weights = 161
##              80) Wt > 97
##                86)*  weights = 441
##          39) Wt > 104
##            87) Ht <= 196; criterion = 0.997, statistic = 17.646
##              88)*  weights = 14
##            87) Ht > 196
##              89) X3PA <= 0; criterion = 0.996, statistic = 17.116
##                90)*  weights = 21
##              89) X3PA > 0
##                91)*  weights = 42
##        38) Ht > 198
##          92) Wt <= 102; criterion = 1, statistic = 918.712
##            93) Ht <= 203; criterion = 1, statistic = 368.971
##              94) Wt <= 87; criterion = 1, statistic = 195.122
##                95) X3PA <= 0.4; criterion = 0.998, statistic = 15.799
##                  96)*  weights = 21
##                95) X3PA > 0.4
##                  97)*  weights = 24
##              94) Wt > 87
##                98) Wt <= 96; criterion = 1, statistic = 137.481
##                  99) BLK <= 0.8; criterion = 1, statistic = 46.384
##                    100) AST <= 3.8; criterion = 1, statistic = 31.955
##                      101) AST <= 0.7; criterion = 1, statistic = 33.156
##                        102) Ht <= 201; criterion = 0.999, statistic = 20.354
```

```
##                     103)*  weights = 75
##                   102) Ht > 201
##                     104)*  weights = 34
##                 101) AST > 0.7
##                   105) BLK <= 0.3; criterion = 1, statistic = 24.532
##                     106) Wt <= 88; criterion = 1, statistic = 22.303
##                       107)*  weights = 17
##                     106) Wt > 88
##                       108) ORB <= 0.9; criterion = 0.996, statistic = 14.622
##                         109)*  weights = 108
##                       108) ORB > 0.9
##                         110)*  weights = 16
##                   105) BLK > 0.3
##                     111) AST <= 2.2; criterion = 0.986, statistic = 12.058
##                       112)*  weights = 49
##                     111) AST > 2.2
##                       113)*  weights = 11
##               100) AST > 3.8
##                 114)*  weights = 20
##             99) BLK > 0.8
##               115)*  weights = 26
##           98) Wt > 96
##             116) ORB <= 1.7; criterion = 1, statistic = 51.812
##               117) Ht <= 201; criterion = 1, statistic = 27.155
##                 118)*  weights = 263
##               117) Ht > 201
##                 119) X3PA <= 0; criterion = 0.972, statistic = 12.965
##                   120)*  weights = 17
##                 119) X3PA > 0
##                   121) ORB <= 1.1; criterion = 0.981, statistic = 13.749
##                     122)*  weights = 196
##                   121) ORB > 1.1
##                     123)*  weights = 51
##             116) ORB > 1.7
##               124)*  weights = 60
##       93) Ht > 203
##         125) Wt <= 96; criterion = 1, statistic = 51.412
##           126) AST <= 1.3; criterion = 0.973, statistic = 13.056
##             127)*  weights = 68
##           126) AST > 1.3
##             128)*  weights = 34
##         125) Wt > 96
##           129) ORB <= 0.6; criterion = 1, statistic = 32.637
##             130)*  weights = 77
##           129) ORB > 0.6
##             131) X3PA <= 0.4; criterion = 1, statistic = 21.651
##               132) ORB <= 0.8; criterion = 0.979, statistic = 13.632
##                 133)*  weights = 14
##               132) ORB > 0.8
##                 134)*  weights = 95
##             131) X3PA > 0.4
##               135)*  weights = 67
```

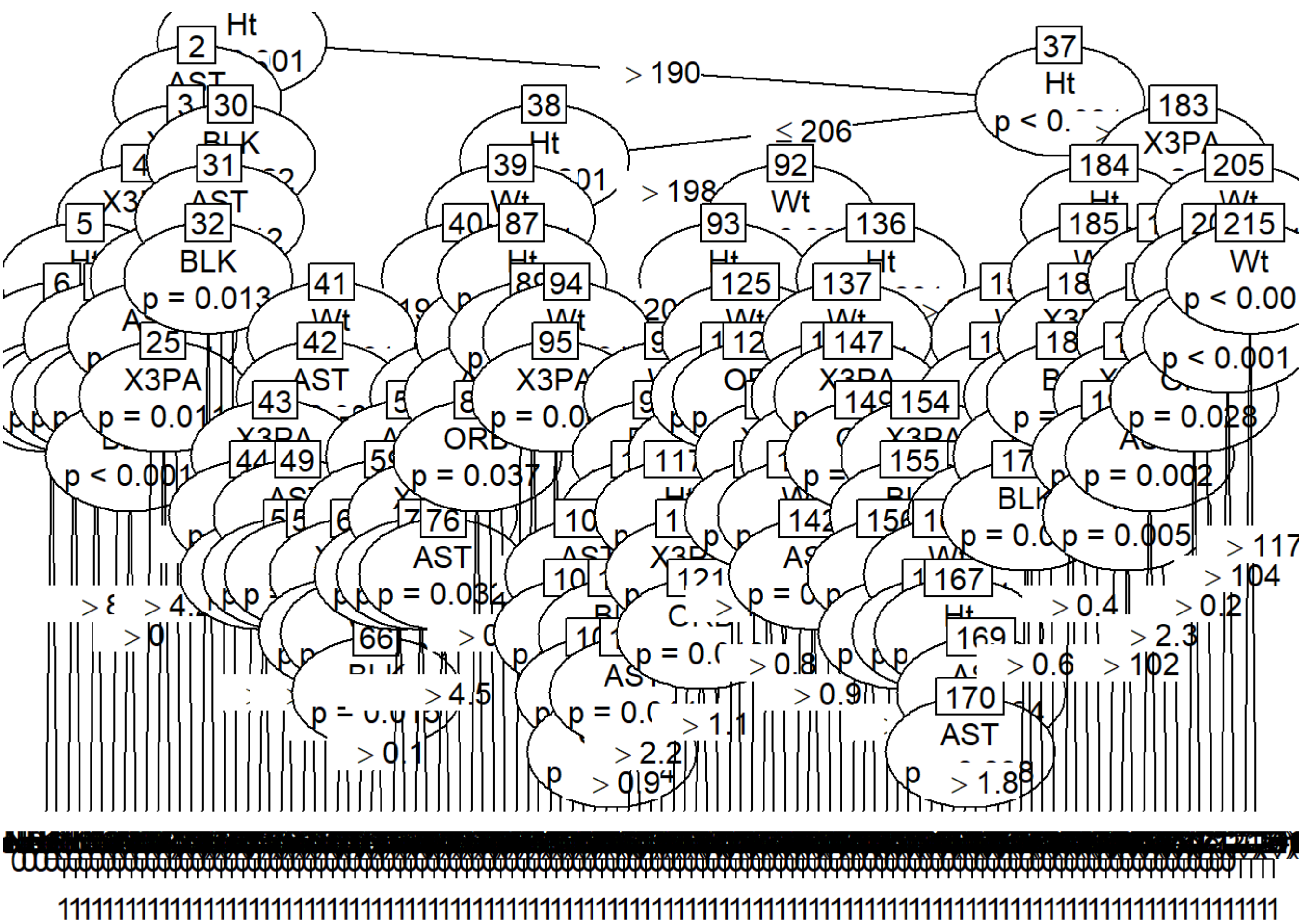
```
##        92) Wt > 102
##          136) Ht <= 201; criterion = 1, statistic = 276.942
##            137) Wt <= 111; criterion = 1, statistic = 74.766
##              138) AST <= 4.2; criterion = 1, statistic = 48.032
##                139) ORB <= 2.3; criterion = 1, statistic = 21.95
##                  140) Wt <= 107; criterion = 0.981, statistic = 13.788
##                    141)*  weights = 114
##                  140) Wt > 107
##                    142) AST <= 0.9; criterion = 0.995, statistic = 16.721
##                      143)*  weights = 20
##                    142) AST > 0.9
##                      144)*  weights = 55
##                139) ORB > 2.3
##                  145)*  weights = 7
##              138) AST > 4.2
##                146)*  weights = 18
##            137) Wt > 111
##              147) X3PA <= 0.6; criterion = 1, statistic = 42.493
##                148)*  weights = 52
##              147) X3PA > 0.6
##                149) ORB <= 1.5; criterion = 0.996, statistic = 11.691
##                  150)*  weights = 19
##                149) ORB > 1.5
##                  151)*  weights = 9
##          136) Ht > 201
##            152) Wt <= 114; criterion = 1, statistic = 137.676
##              153) AST <= 6.8; criterion = 1, statistic = 110.104
##                154) X3PA <= 0.7; criterion = 1, statistic = 65.439
##                  155) BLK <= 0.9; criterion = 1, statistic = 54.049
##                    156) Ht <= 203; criterion = 1, statistic = 21.929
##                      157)*  weights = 152
##                    156) Ht > 203
##                      158) BLK <= 0.4; criterion = 0.989, statistic = 12.544
##                        159)*  weights = 203
##                      158) BLK > 0.4
##                        160)*  weights = 101
##                  155) BLK > 0.9
##                    161)*  weights = 51
##                154) X3PA > 0.7
##                  162) AST <= 5.4; criterion = 1, statistic = 31.308
##                    163) Wt <= 104; criterion = 0.999, statistic = 21.233
##                      164) ORB <= 1.4; criterion = 0.983, statistic = 14.046
##                        165)*  weights = 74
##                      164) ORB > 1.4
##                        166)*  weights = 13
##                    163) Wt > 104
##                      167) Ht <= 203; criterion = 0.991, statistic = 15.431
##                        168)*  weights = 161
##                      167) Ht > 203
##                        169) AST <= 3.7; criterion = 0.996, statistic = 17.256
##                          170) AST <= 1.8; criterion = 0.962, statistic = 12.281
##                            171)*  weights = 123
```

```
##                            170) AST > 1.8
##                              172)*  weights = 19
##                          169) AST > 3.7
##                            173)*  weights = 7
##                    162) AST > 5.4
##                      174)*  weights = 10
##                153) AST > 6.8
##                  175)*  weights = 11
##              152) Wt > 114
##                176) X3PA <= 0.3; criterion = 0.999, statistic = 17.506
##                  177) AST <= 1.2; criterion = 0.997, statistic = 15.423
##                    178) BLK <= 0.6; criterion = 0.998, statistic = 16.217
##                      179)*  weights = 82
##                    178) BLK > 0.6
##                      180)*  weights = 47
##                  177) AST > 1.2
##                    181)*  weights = 43
##                176) X3PA > 0.3
##                  182)*  weights = 16
##     37) Ht > 206
##       183) X3PA <= 0.2; criterion = 1, statistic = 514.261
##         184) Ht <= 208; criterion = 1, statistic = 139.654
##           185) Wt <= 117; criterion = 1, statistic = 32.45
##             186) X3PA <= 0; criterion = 0.999, statistic = 17.304
##               187) BLK <= 0.4; criterion = 0.96, statistic = 9.979
##                 188)*  weights = 131
##               187) BLK > 0.4
##                 189)*  weights = 145
##             186) X3PA > 0
##               190)*  weights = 106
##           185) Wt > 117
##             191)*  weights = 112
##         184) Ht > 208
##           192) AST <= 3.6; criterion = 1, statistic = 95.845
##             193) Wt <= 113; criterion = 1, statistic = 50.803
##               194) X3PA <= 0; criterion = 1, statistic = 21.475
##                 195) Ht <= 211; criterion = 0.998, statistic = 15.764
##                   196)*  weights = 234
##                 195) Ht > 211
##                   197) Wt <= 102; criterion = 0.995, statistic = 11.062
##                     198)*  weights = 17
##                   197) Wt > 102
##                     199)*  weights = 189
##               194) X3PA > 0
##                 200) AST <= 2.3; criterion = 0.998, statistic = 16.126
##                   201)*  weights = 136
##                 200) AST > 2.3
##                   202)*  weights = 27
##             193) Wt > 113
##               203)*  weights = 490
##           192) AST > 3.6
##             204)*  weights = 13
```

```
##     183) X3PA > 0.2
##       205) Wt <= 111; criterion = 1, statistic = 215.008
##         206) Ht <= 208; criterion = 1, statistic = 79.898
##           207) X3PA <= 1.9; criterion = 1, statistic = 45.211
##             208) ORB <= 0.2; criterion = 0.972, statistic = 10.697
##               209)*  weights = 16
##             208) ORB > 0.2
##               210)*  weights = 114
##           207) X3PA > 1.9
##             211)*  weights = 172
##         206) Ht > 208
##           212) Wt <= 104; criterion = 1, statistic = 51.009
##             213)*  weights = 70
##           212) Wt > 104
##             214)*  weights = 163
##       205) Wt > 111
##         215) Wt <= 117; criterion = 0.999, statistic = 18.314
##           216)*  weights = 109
##         215) Wt > 117
##           217)*  weights = 62
```

```
population.tree <- ctree(Pos~., data=data)
```

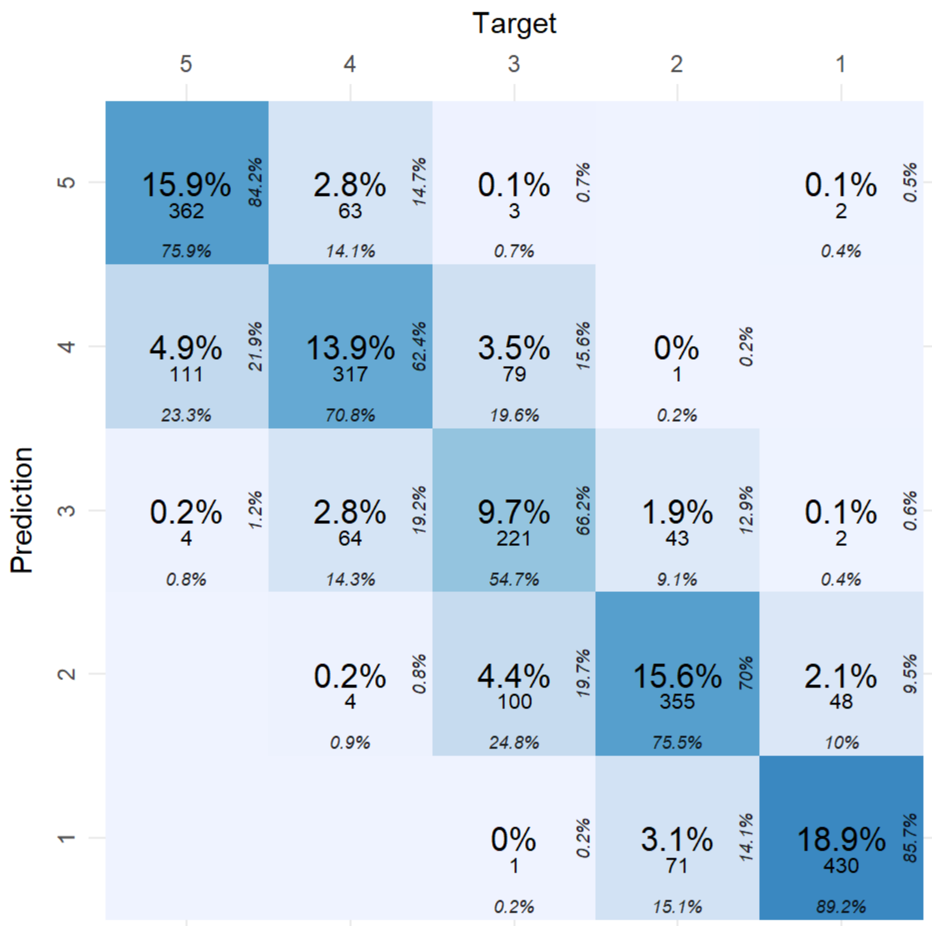
```
plot(tree)
```



Damn -_-

```
tree.pred <- predict(tree, test.X)
population.pred <- predict(population.tree, test.X)
```

```
plot_confusion_matrix(confusion_matrix(targets=test.Y$Pos, prediction=tree.pred))
```



```
stats <- ml_test(tree.pred, test.Y$Pos, output.as.table = FALSE)
tree.results <- data.frame(
    Accuracy = round(rep(stats$accuracy, 5),4),
    Specificity = round(stats$specificity,4),
    Precision = round(stats$precision,4),
    Sensitivity = round(stats$recall,4),
    Bias = rep(mean(abs(as.integer(population.pred) - as.integer(tree.pred))), 5),
    Variance = rep(var(as.integer(population.pred), as.integer(tree.pred)), 5)
)
message("Accuracy: ", round(stats$accuracy,4)*100, "%\n")
```

```
## Accuracy: 73.87%
```

```
tree.results
```

	Accuracy	Specificity	Precision	Sensitivity	Bias	Variance
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.7387	0.9457	0.8566	0.8921	0.1363437	1.983488
2	0.7387	0.8974	0.7002	0.7553	0.1363437	1.983488
3	0.7387	0.9283	0.6617	0.5470	0.1363437	1.983488
4	0.7387	0.8775	0.6240	0.7076	0.1363437	1.983488
5	0.7387	0.9511	0.8419	0.7589	0.1363437	1.983488

5 rows

SVM

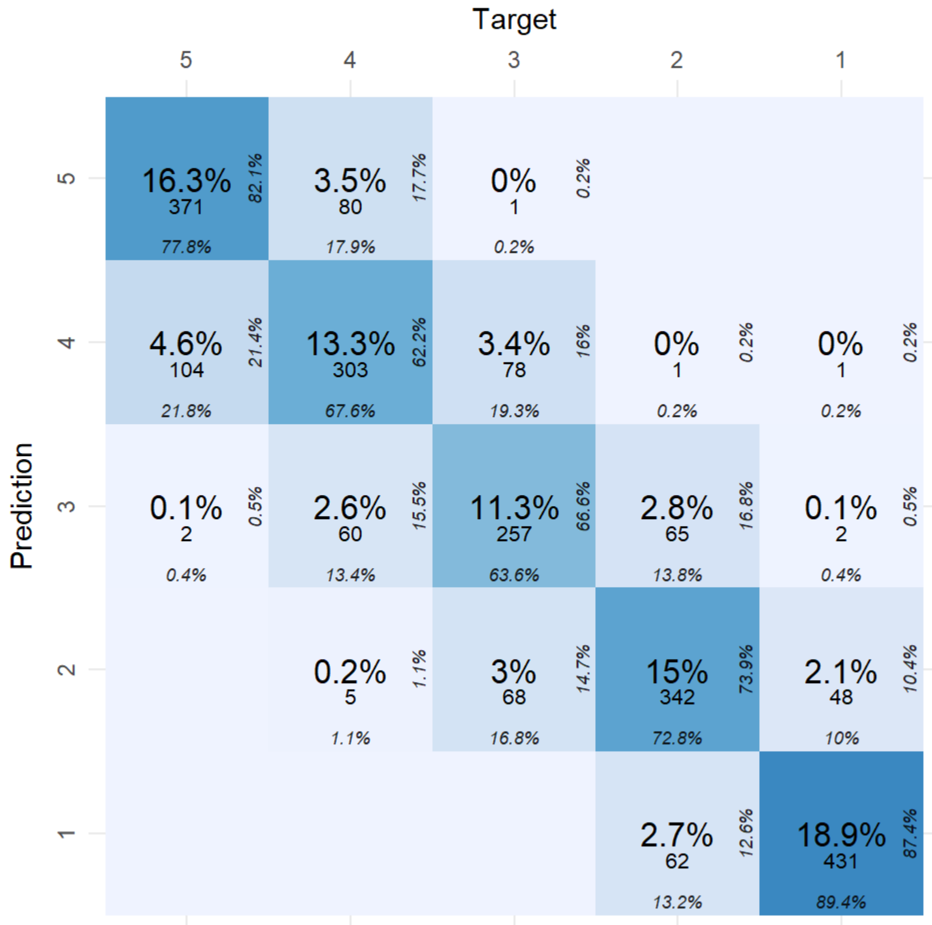
```
svm <- svm(Pos~., data=train,type = 'C-classification',kernel = 'linear')
svm
```

```
##
## Call:
## svm(formula = Pos ~ ., data = train, type = "C-classification", kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  5789
```

```
population.svm <-svm(Pos~., data=data, type = 'C-classification',kernel = 'linear')
```

```
svm.pred <- predict(svm, test.X)
population.pred <- predict(population.svm, test.X)
```

```
plot_confusion_matrix(confusion_matrix(targets=test.Y$Pos, prediction=svm.pred))
```




```
stats <- ml_test(svm.pred, test.Y$Pos, output.as.table = FALSE)
svm.results <- data.frame(
    Accuracy = round(rep(stats$accuracy, 5),4),
    Specificity = round(stats$specificity,4),
    Precision = round(stats$precision,4),
    Sensitivity = round(stats$recall,4),
    Bias = rep(mean(abs(as.integer(population.pred) - as.integer(svm.pred))), 5),
    Variance = rep(var(as.integer(population.pred), as.integer(svm.pred)), 5)
)
message("Accuracy: ", round(stats$accuracy,4)*100, "%\n")
```

```
## Accuracy: 74.7%
```

```
svm.results
```

	Accuracy <dbl>	Specificity <dbl>	Precision <dbl>	Sensitivity <dbl>	Bias <dbl>	Variance <dbl>
1	0.747	0.9536	0.8742	0.8942	0.007014467	2.069484
2	0.747	0.9184	0.7387	0.7277	0.007014467	2.069484
3	0.747	0.9181	0.6658	0.6361	0.007014467	2.069484

	Accuracy <dbl>	Specificity <dbl>	Precision <dbl>	Sensitivity <dbl>	Bias <dbl>	Variance <dbl>
4	0.747	0.8839	0.6222	0.6763	0.007014467	2.069484
5	0.747	0.9427	0.8208	0.7778	0.007014467	2.069484

5 rows

Results

For this data KNN (k=7) shows the best results, but generally I think there are possible cases, where other models will lead. Anyways I am glad to get such results, since after running ANOVA in my EDA I was afraid to only one prediction into the model.

Since the accuracies are close, so the variances.

There is no reason to commend Specificity and Sensitivity since my classes are not ordered, and I am not afraid to get prediction of +- 1 class. Besides, many players switch positions from season to season. So errors here are not crucial, it is not some heart decease data.

```
results <- data.frame(
    Model = c("Logit", "KNN", "Decision Tree", "SVM"),
    Accuracy = c(logit.results$Accuracy[1], knn.results$Accuracy[1], tree.results$Accuracy[1], s
vm.results$Accuracy[1]),
    Bias = c(logit.results$Bias[1], knn.results$Bias[1], tree.results$Bias[1], svm.results$Bias[
1]),
    Variance = c(logit.results$Variance[1], knn.results$Variance[1], tree.results$Variance[1], s
vm.results$Variance[1])
)
results
```

Model <chr>	Accuracy <dbl>	Bias <dbl>	Variance <dbl>
Logit	0.7488	0.008768084	2.064605
KNN	0.7619	0.081543183	2.028007
Decision Tree	0.7387	0.136343709	1.983488
SVM	0.7470	0.007014467	2.069484

4 rows