Settlers_of_Catan

Dominik Stipić

December 28, 2019

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(tidyr)
library(stringr)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
source("../R/clean.R")
```

Priprema podataka

```
df <- read.csv(file = "../data/SettlersOfCatanStats.csv", stringsAsFactors = F)
```

```
DF <- clean.df(df)
glimpse(DF)
```

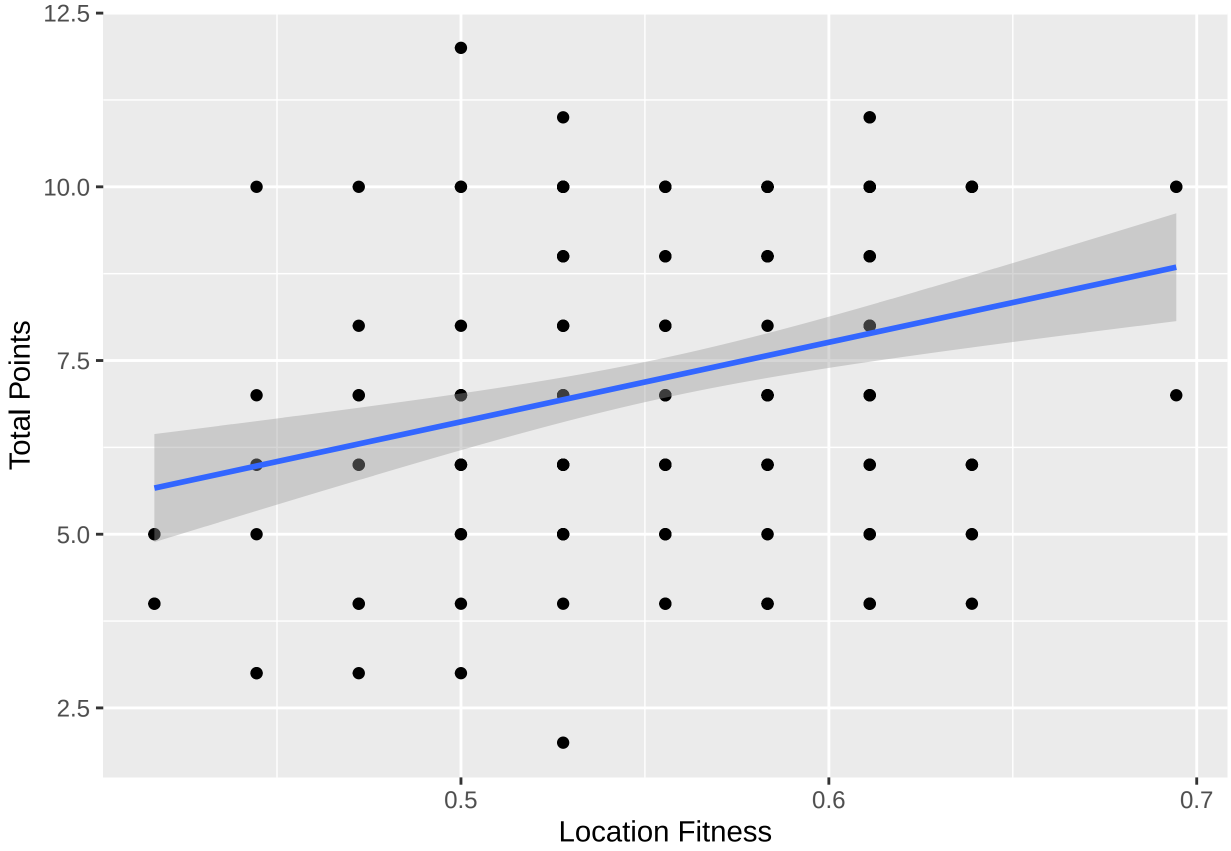
```
## Observations: 200
## Variables: 41
## $ gameNum         <int> 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4...
## $ player          <int> 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4...
## $ points          <int> 5, 9, 10, 5, 10, 6, 4, 9, 5, 10, 7, 7, 7, 10, ...
## $ X2              <int> 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1...
## $ X3              <int> 3, 3, 3, 3, 6, 6, 6, 6, 3, 3, 3, 3, 6, 6, 6, 6...
## $ X4              <int> 5, 5, 5, 5, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5...
## $ X5              <int> 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10, 10, 12, 12...
## $ X6              <int> 7, 7, 7, 7, 10, 10, 10, 10, 10, 10, 10, 10, 14...
## $ X7              <int> 10, 10, 10, 10, 8, 8, 8, 8, 4, 4, 4, 4, 20, 20...
## $ X8              <int> 6, 6, 6, 6, 14, 14, 14, 14, 5, 5, 5, 5, 12, 12...
## $ X9              <int> 7, 7, 7, 7, 9, 9, 9, 9, 5, 5, 5, 5, 11, 11, 11...
## $ X10             <int> 3, 3, 3, 3, 3, 3, 3, 3, 6, 6, 6, 6, 4, 4, 4, 4...
## $ X11             <int> 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 3, 2, 2, 2, 2...
## $ X12             <int> 1, 1, 1, 1, 3, 3, 3, 3, 1, 1, 1, 1, 3, 3, 3, 3...
## $ Value1.1        <dbl> 0.13888889, 0.11111111, 0.11111111, 0.13888889...
## $ Tile1.1         <fct> L, W, S, O, W, C, C, C, L, W, S, L, C, W, L, C...
## $ Port1.1         <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ Value1.2        <dbl> 0.05555556, 0.13888889, 0.13888889, 0.11111111...
## $ Tile1.2         <fct> C, O, S, L, O, S, W, W, L, O, W, W, L, L, C, S...
## $ Port1.2         <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ Value1.3        <dbl> 0.05555556, 0.08333333, 0.02777778, 0.05555556...
## $ Tile1.3         <fct> C, W, W, L, O, O, O, O, C, C, L, S, O, C, O, W...
## $ Port1.3         <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ Value2.1        <dbl> 0.11111111, 0.08333333, 0.13888889, 0.08333333...
## $ Tile2.1         <fct> L, L, O, L, W, W, C, L, W, L, O, O, S, S, W, S...
## $ Port2.1         <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ Value2.2        <dbl> 0.08333333, 0.11111111, 0.08333333, 0.13888889...
## $ Tile2.2         <fct> W, S, S, L, L, L, W, C, L, W, C, C, C, S, S, L...
## $ Port2.2         <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ Value2.3        <dbl> 0.05555556, 0.05555556, 0.05555556, 0.08333333...
## $ Tile2.3         <fct> O, O, C, S, L, W, L, S, S, S, G, L, G, C, W, O...
## $ Port2.3         <fct> NA, NA, NA, NA, S, NA, NA, NA, NA, NA, G, NA, ...
## $ production      <int> 38, 48, 44, 42, 60, 57, 44, 61, 44, 41, 47, 53...
## $ tradeGain       <int> 5, 8, 14, 12, 15, 12, 10, 16, 5, 4, 6, 2, 15, ...
## $ robberCardsGain <int> 2, 6, 9, 0, 16, 1, 8, 11, 5, 9, 5, 2, 12, 15, ...
## $ totalGain       <int> 45, 62, 67, 54, 91, 70, 62, 88, 54, 54, 58, 57...
## $ tradeLoss       <int> 10, 11, 24, 24, 28, 26, 18, 25, 11, 8, 10, 4, ...
## $ robberCardsLoss <int> 2, 1, 4, 6, 10, 6, 6, 6, 1, 3, 7, 4, 5, 15, 5,...
## $ tribute         <int> 4, 8, 0, 0, 0, 8, 8, 4, 9, 0, 0, 8, 12, 10, 0,...
## $ totalLoss       <int> 16, 20, 28, 30, 38, 40, 32, 35, 21, 11, 17, 16...
## $ totalAvailable  <int> 29, 42, 39, 24, 53, 30, 30, 53, 33, 43, 41, 41...
```

Kako početna konfiguracija naselja utječe na konačni broj bodova?

Definiramo dobrotu lokacije naselja (LocationFitness) kao vjerovatnost da to naselje u jednom bacanju dobije resurs. Podaci pokazuju da ako imamo pozicije koja donose jako puno resursa, to ima blagi utjecaj na konačni ishod igre.

```
DF %>% mutate(locationFitness.1 = Value1.1 + Value1.2 + Value1.3) -> DF
DF %>% mutate(locationFitness.2 = Value2.1 + Value2.2 + Value2.3) -> DF
```

```
gf <- DF %>% ggplot(aes(x = locationFitness.1 + locationFitness.2, y = points))
gf + geom_point() + labs(x = "Location Fitness", y = "Total Points") + stat_smooth(method = "lm")
```



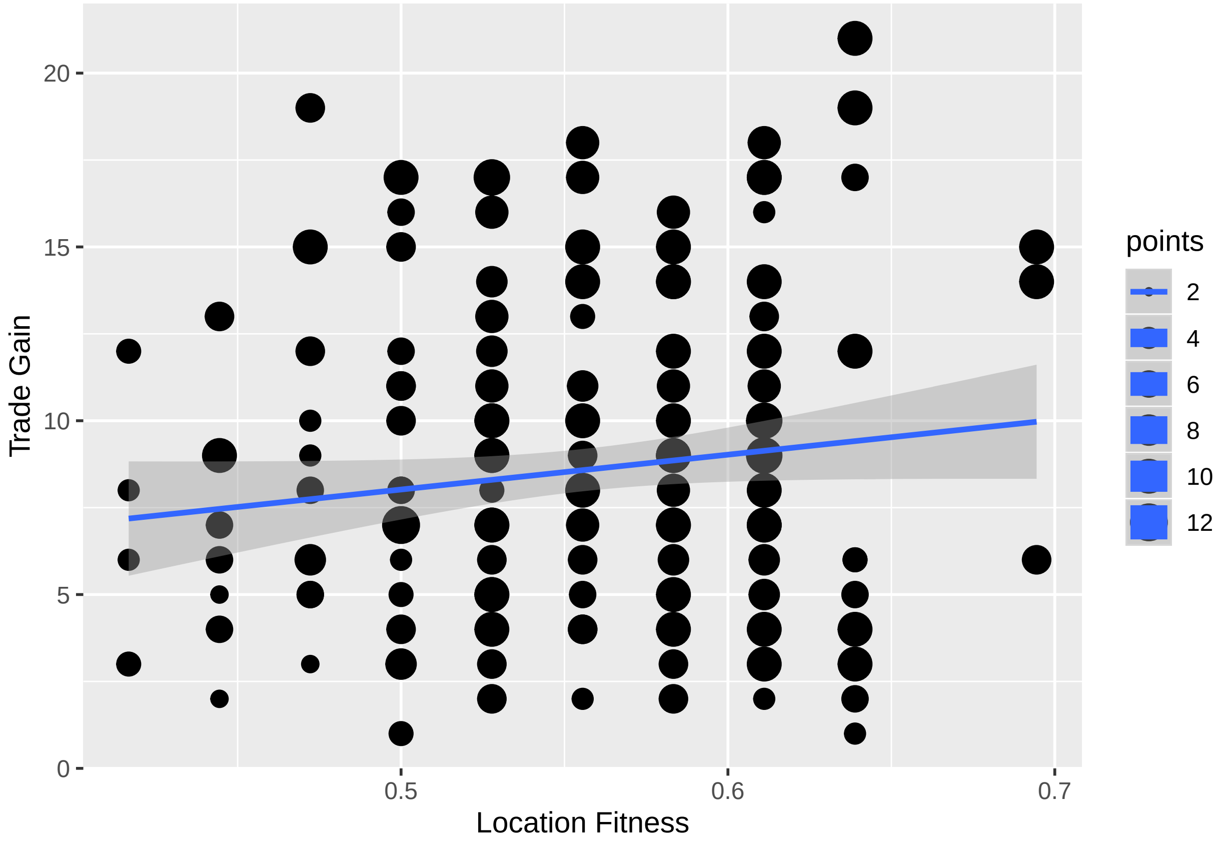
```
x <- DF$locationFitness.1 + DF$locationFitness.2
y <- DF$points

cat(str_c("Pearson correlation(Location Fitness, Points) : ", cor(x,y)))
```

```
## Pearson correlation(Location Fitness, Points) : 0.29489374689556
```

Tpična ponašanja i strategije u ovisnosti sa kvalitetom početnih lokacija

Početna konfiguracija naselja nema značajan utjecaj na daljne trgovanje igrača. Pretpostavaljamo da portefelj resursa s kojim igrač na početku igre rukuje ima veću korelaciju sa daljnjim trgovanjem. Ukoliko igrač ima mogo drva ili gline vrlo vjerovatno će ih odmah potrošiti kod izgradnje vlastitih projekata.

```
DF %>% ggplot(aes(x = locationFitness.1 + locationFitness.2, y = tradeGain, size = points)) +
  geom_point()+
  stat_smooth(method = "lm") +
  labs(x = "Location Fitness", y = "Trade Gain")
```



```
x <- DF$locationFitness.1 + DF$locationFitness.2
y <- DF$tradeGain

cat(str_c("Pearson correlation : ", cor(x,y)))
```

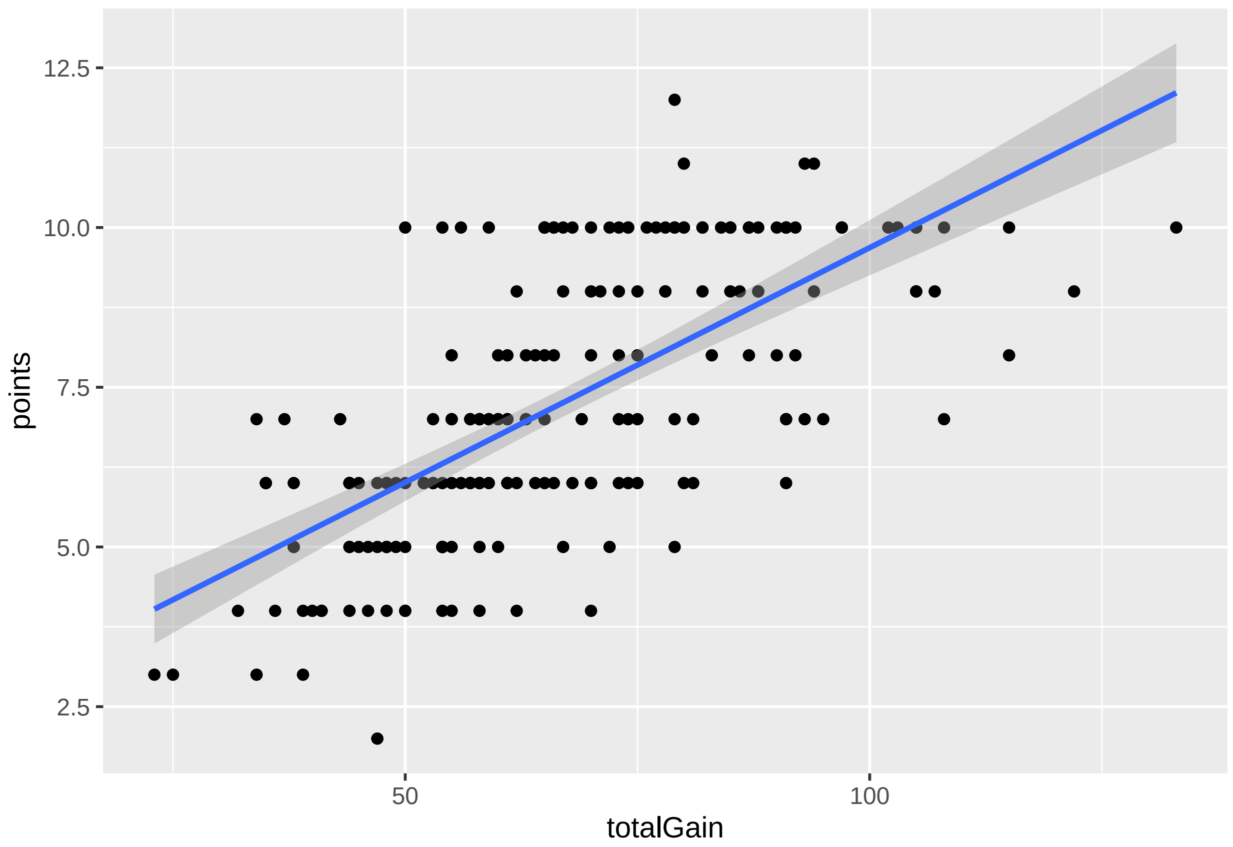
```
## Pearson correlation : 0.126847263147419
```

Analiza mogučnosti pribave resursa

Postoji vrlo jaka korelacija između mogučnosti pribavaljnja resursa igrača sa konačnim brojem bodova. Igrač može pribavaljati resurse na 3 načina:

- Proizvodnjom
- Trgovinom
- Pljačkom

```
DF %>% ggplot(aes(x = totalGain, y = points)) +
  geom_point() +
  stat_smooth(method = "lm")
```



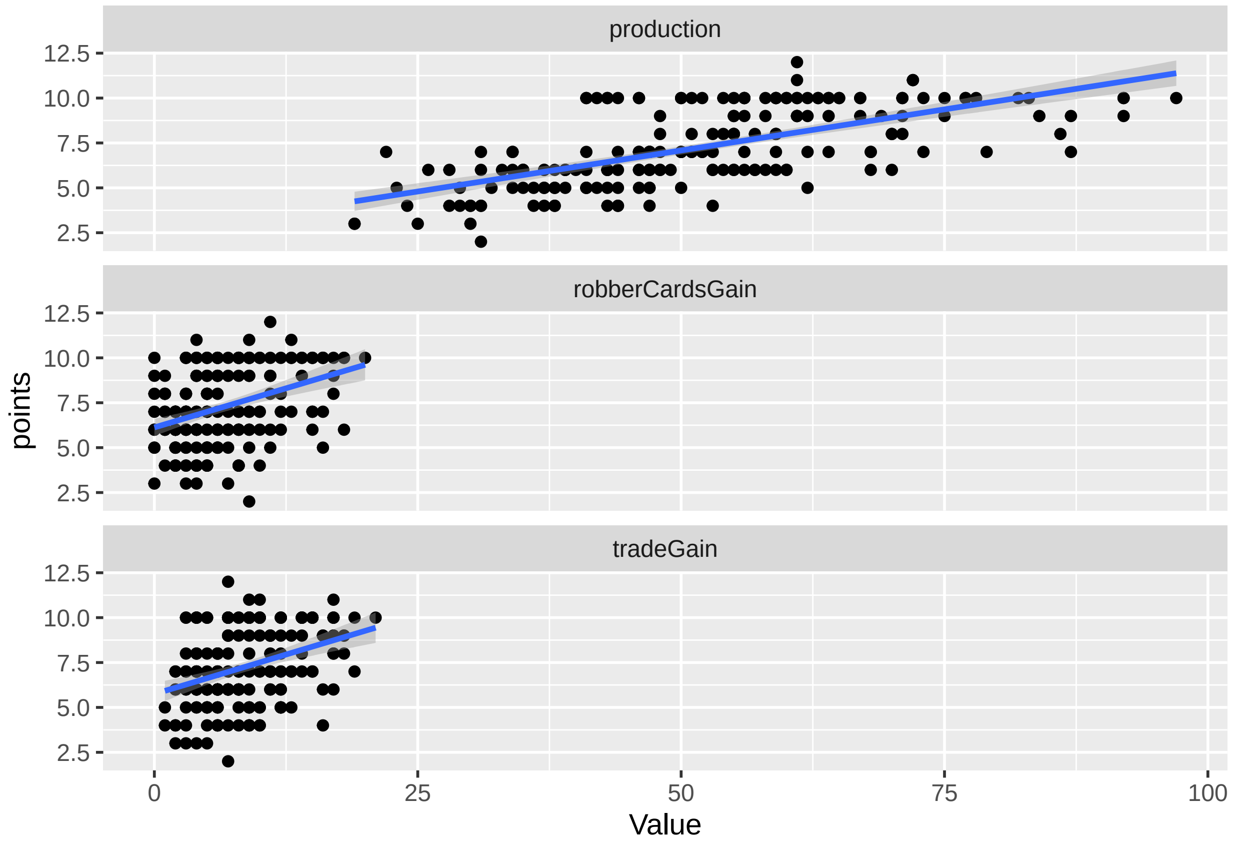
```
x <- DF$totalGain
y <- DF$points

cat(str_c("Pearson correlation : ", cor(x,y)))
```

```
## Pearson correlation : 0.676488925450187
```

Vidimo da su sve 3 ativnosti vrlo povezane sa konačnim brojem bodova, s time da je proizvodnja resursa vrlo korelirana sa konačnim brojem bodova, a ostale aktivnosti su tek blago korelirane. Pošto proizvodnja resursa najviše korelira sa pobjedom potrebno je analizirati kakva strategija pospješuje proizvodnju resursa. Varijable koje utječu na efikasniju proizvodnju resursa su sljedeće:

- Lokacije koje donose mnogo resursa
- Lokacije koje donose strateški bitne resurse, odnosno monopol nad određenim resursom je vrlo poželjna strategija
- Preferirani resursi u ranom stadiju igre su drvo i glina, a u kasnijem kamen. Važnost pojedinih resursa se tijekom igre mijenja

```
DF %>% select(points, production:robberCardsGain) -> X
X %>% gather(Type, Value, production:robberCardsGain) -> X

X %>% ggplot(aes(x = Value, y = points)) +
  geom_point() +
  stat_smooth(method = "lm") +
  facet_wrap(~Type, nrow = 3)
```



```
X %>% filter(Type == "production") %>% select(-Type) -> production
X %>% filter(Type == "tradeGain") %>% select(-Type) -> trade
X %>% filter(Type == "robberCardsGain") %>% select(-Type) -> robber

cat(str_c("Pearson correlation(Production,Points) : ", cor(production$points,production$Value)), "\n")
```

```
## Pearson correlation(Production,Points) : 0.655415712281828
```

```
cat(str_c("Pearson correlation(Trade,Points) : ", cor(trade$points, trade$Value)), "\n")
```

```
## Pearson correlation(Trade,Points) : 0.358378612578075
```

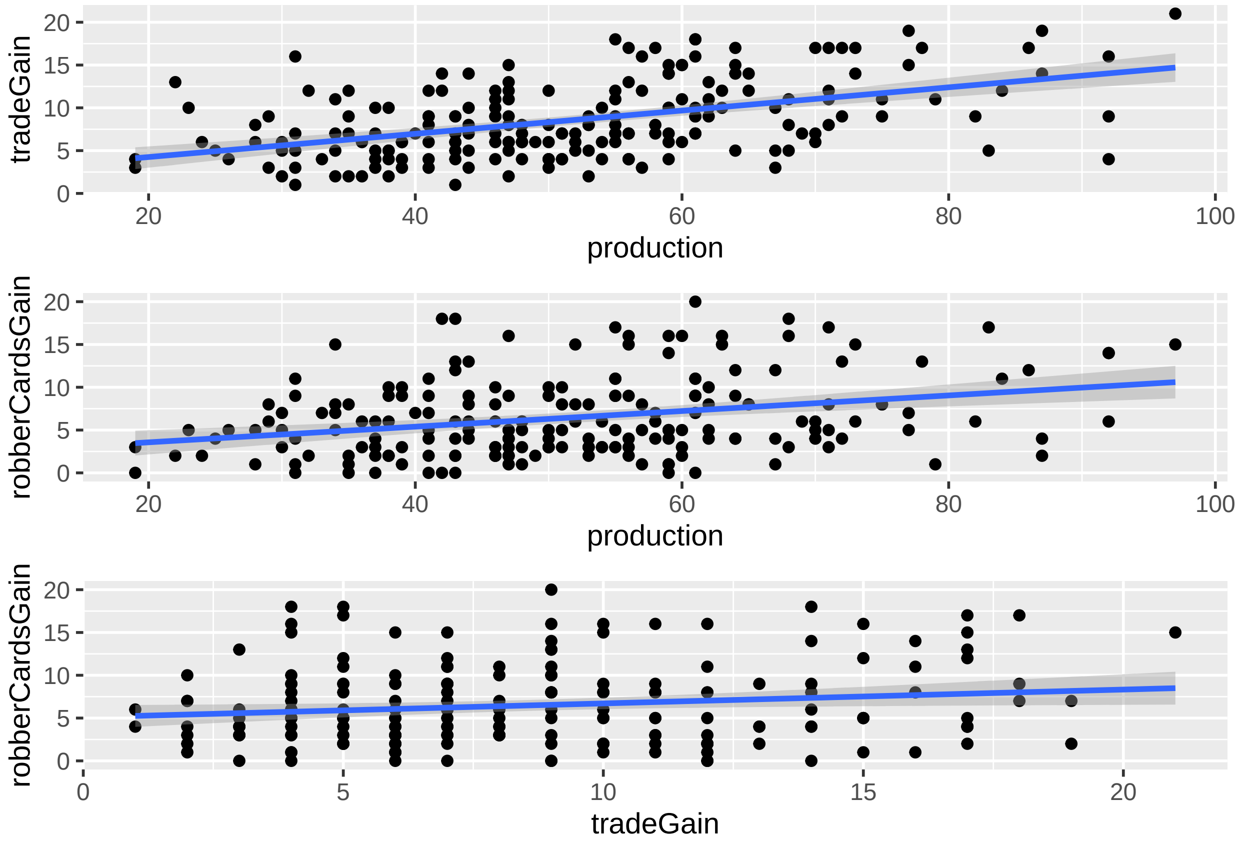
```
cat(str_c("Pearson correlation(Robber,Points) : ", cor(robber$points, robber$Value)), "\n")
```

```
## Pearson correlation(Robber,Points) : 0.373866091994192
```

Igrači koji imaju jaku proizvodnju češće trguju nego ostali igrači, također se u manjoj mjeri može uočiti da je proizvodnja povezana sa pljačkom

```
DF %>% ggplot(aes(x = production, y = tradeGain)) +
  geom_point() +
  stat_smooth(method = "lm") -> g1
DF %>% ggplot(aes(x = production, y = robberCardsGain)) +
  geom_point() +
  stat_smooth(method = "lm") -> g2
DF %>% ggplot(aes(x = tradeGain, y = robberCardsGain)) +
  geom_point() +
  stat_smooth(method = "lm") -> g3
```

```
grid.arrange(g1,g2,g3,nrow = 3,ncol = 1)
```



```
cat(str_c("Pearson correlation(Production,Trade) : ",  cor(production$Value, trade$Value)), "\n")
```

```
## Pearson correlation(Production,Trade) : 0.47761386317312
```

```
cat(str_c("Pearson correlation(Production,Robber) : ", cor(production$Value, robber$Value)), "\n")
```

```
## Pearson correlation(Production,Robber) : 0.305213977569689
```

```
cat(str_c("Pearson correlation(Robber,Trade) : ",      cor(robber$Value, trade$Value)), "\n")
```

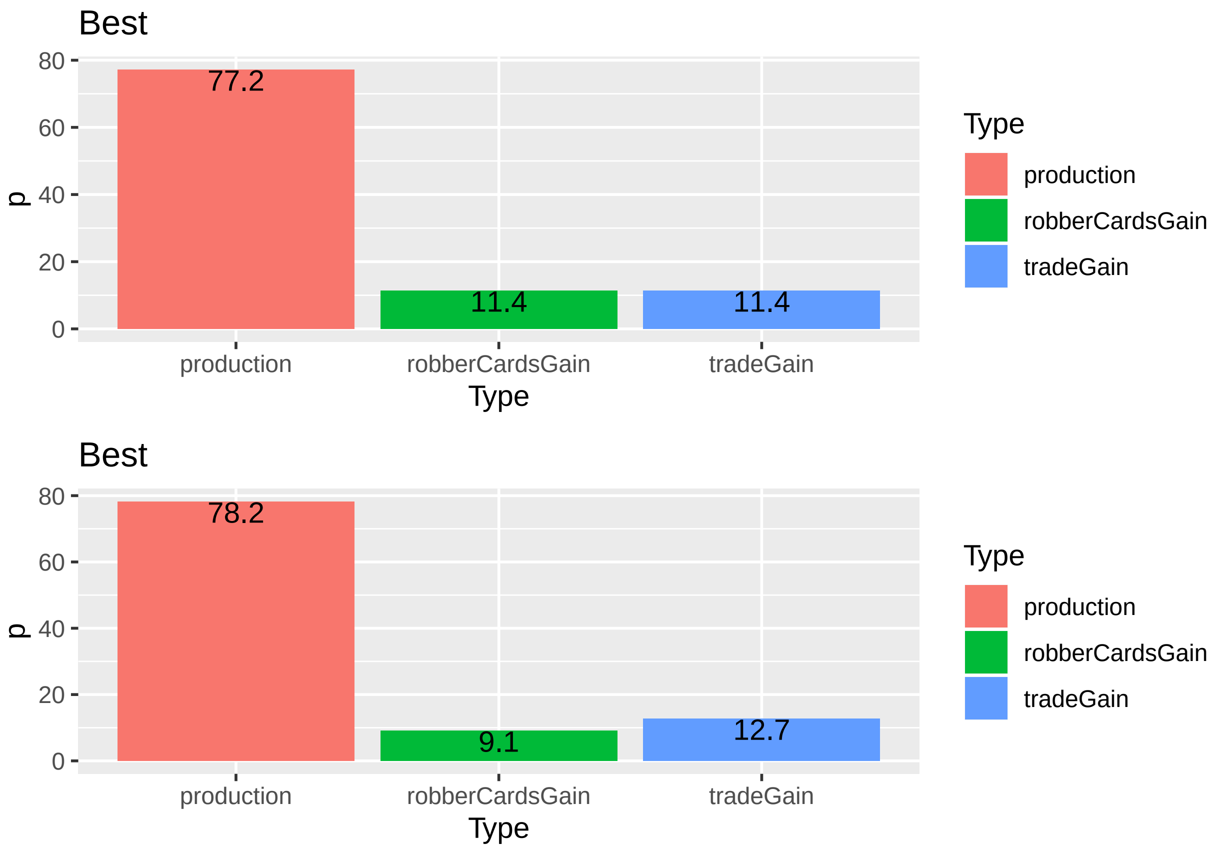
```
## Pearson correlation(Robber,Trade) : 0.154019561834999
```

Analiza jakih i slabih igrača

```
DF %>% select(gameNum, points, production) %>% group_by(gameNum) %>% summarise(points = max(points)) %>

DF %>% select(gameNum, points, production) %>% group_by(gameNum) %>% summarise(points = min(points)) %>
```

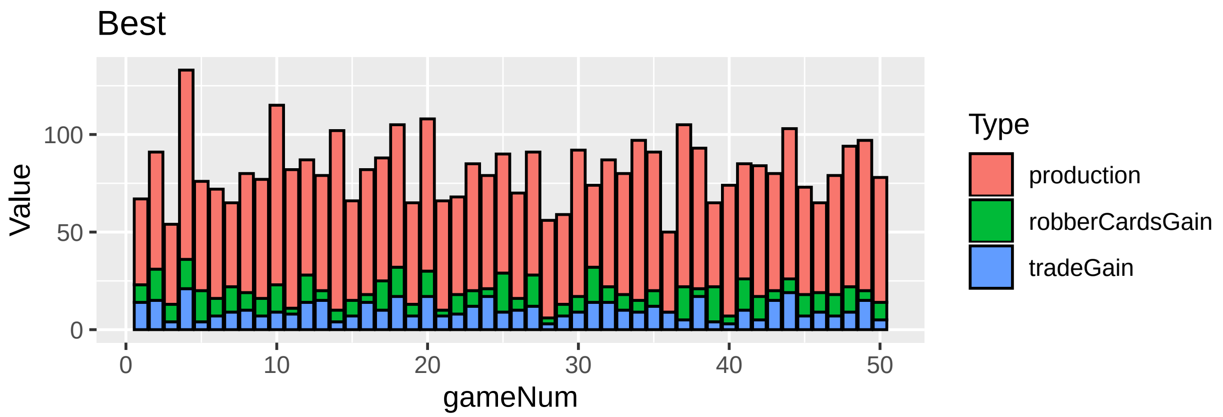
Uprosječene su sve aktivnosti te je dobiven tortni prikaza distribucija aktivnosti između boljih i lošijih igrača. Pokazalo se da bolji igrači sudjeluju ipak malo više u trgovini sa ostalim igračima, ali možemo zaključiti da alokacija vremena po aktivnostima nije od presudne važnosti.

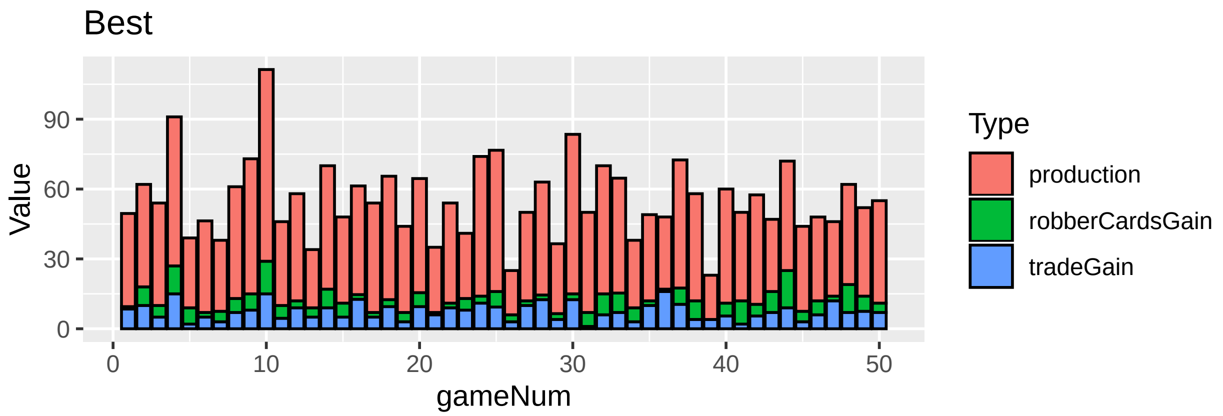
```
bestPlayers %>% select(gameNum, production:robberCardsGain) %>% gather(Type, Value, production:robberCa
X %>% group_by(Type) %>% summarise(Value = median(Value)) %>% mutate(p = round(Value / sum(Value),3)*10
X %>% ggplot(aes(x = Type, y = p, fill = Type)) +
  geom_bar(stat = "identity") +
  labs(title = "Best") +
  geom_text(aes(label = p), vjust = 1) -> g1


worstPlayers %>% select(gameNum, production:robberCardsGain) %>%
  group_by(gameNum) %>%
  summarise(production = mean(production), tradeGain = mean(tradeGain), robberCardsGain = mean(robberCa
  gather(Type, Value, production:robberCardsGain) -> X
X %>% group_by(Type) %>% summarise(Value = median(Value)) %>% mutate(p = round(Value / sum(Value),3)*10

X %>% ggplot(aes(x = Type, y = p, fill = Type)) +
  geom_bar(stat = "identity") +
  labs(title = "Best")  +
  geom_text(aes(label = p), vjust = 1) -> g2

grid.arrange(g1, g2, nrow = 2, ncol = 1 )
```



Zaključili smo da igrači podjednako sudjeluju u svim aktivnostima, no iz ovog grafa je jasno vidljivo da od presudne važnosti ima ukupna količina proizvodnje odnosno efikasnost. Bolji igrači u prosjeku imaju gotovo 30 kartica više od lošijih igrača.

```
bestPlayers %>% select(gameNum, production:robberCardsGain) %>% gather(Type, Value, production:robberCa
X %>% ggplot(aes(x = gameNum, y = Value, fill = Type)) +
  geom_bar(stat = "identity", color = "Black") +
  labs(title = "Best") -> g1

worstPlayers %>% select(gameNum, production:robberCardsGain) %>%
  group_by(gameNum) %>%
  summarise(production = mean(production), tradeGain = mean(tradeGain), robberCardsGain = mean(robberCa
  gather(Type, Value, production:robberCardsGain) -> X
X %>% ggplot(aes(x = gameNum, y = Value, fill = Type)) +
  geom_bar(stat = "identity", color = "Black") +
  labs(title = "Best") -> g2

grid.arrange(g1, g2, nrow = 2, ncol = 1)
```





```
best.total.gain <- mean(bestPlayers$totalGain)
worst.total.gain <- mean(worstPlayers$totalGain)
best.production <- mean(bestPlayers$production)
worst.production <- mean(worstPlayers$production)
best.trade <- mean(bestPlayers$tradeGain)
worst.trade <- mean(worstPlayers$tradeGain)
best.robber <- mean(bestPlayers$robberCardsGain)
worst.robber <- mean(worstPlayers$robberCardsGain)

type <- c("Best", "Worst")
production <- c(best.production, worst.production)
trading <- c(best.trade, worst.trade)
steals <- c(best.robber, worst.robber)
total <- c(best.total.gain, worst.total.gain)
```

```
tmp <- data.frame(type,production,trading, steals, total)

knitr::kable(
  tmp, caption = 'Prosječne karakteristike za najbolje i najgore igrače'
)
```

Table 1: Prosječne karakteristike za najbolje i najgore igrače

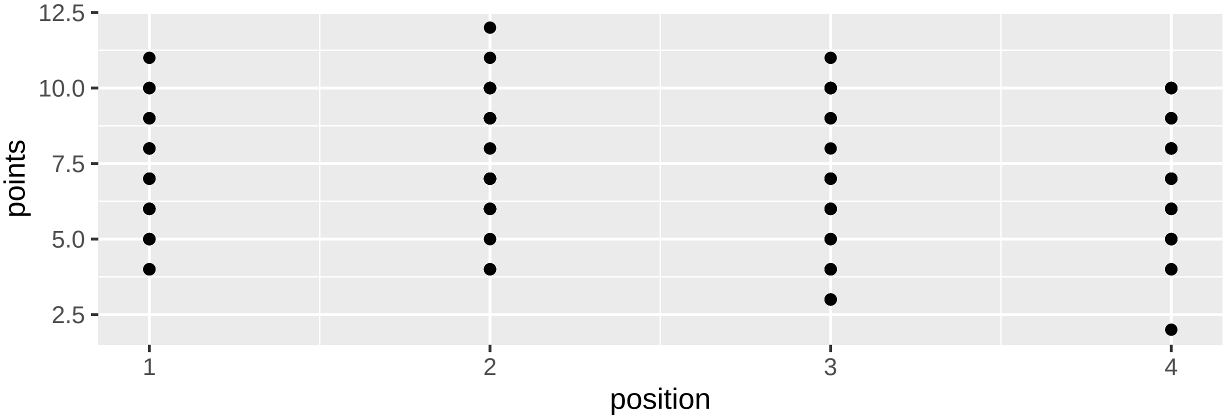
type	production	trading	steals	total
Best	62.34000	10.100000	9.640000	82.08000
Worst	44.89041	7.712329	5.328767	57.93151

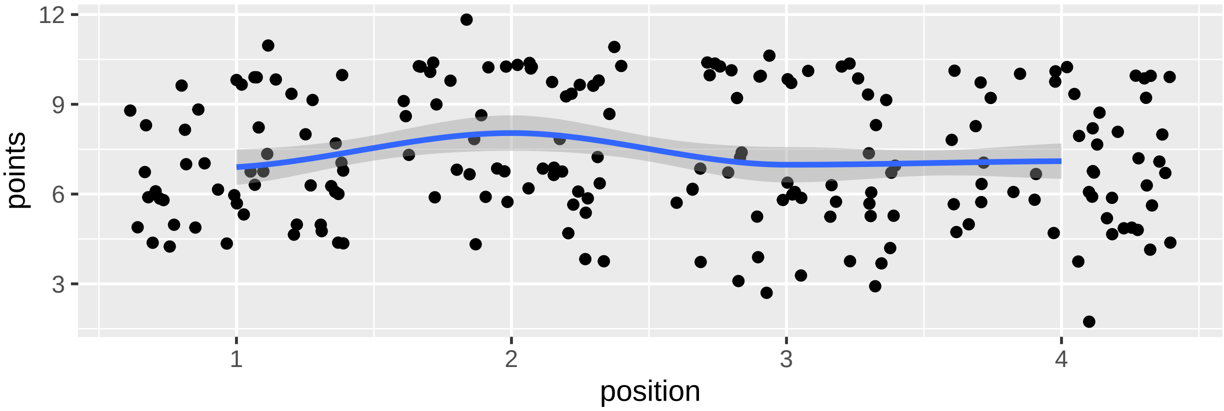
Utjecaj redoslijeda igranja

Postoji blaga prednost prilikom igranja na drugoj poziciji

```
DF %>% select(gameNum, points)  %>% group_by(gameNum) %>% mutate(position = order(gameNum), Rank = dense

X %>% ggplot(aes(x = position, y = points)) +
  geom_point() +
  geom_smooth(method = "loees") -> g1

X %>% ggplot(aes(x = position, y = points)) +
  geom_jitter(width = 0.4, height = 0.4) +
  geom_smooth(method = "loess") -> g2

grid.arrange(g1,g2,nrow = 2,ncol = 1)
```

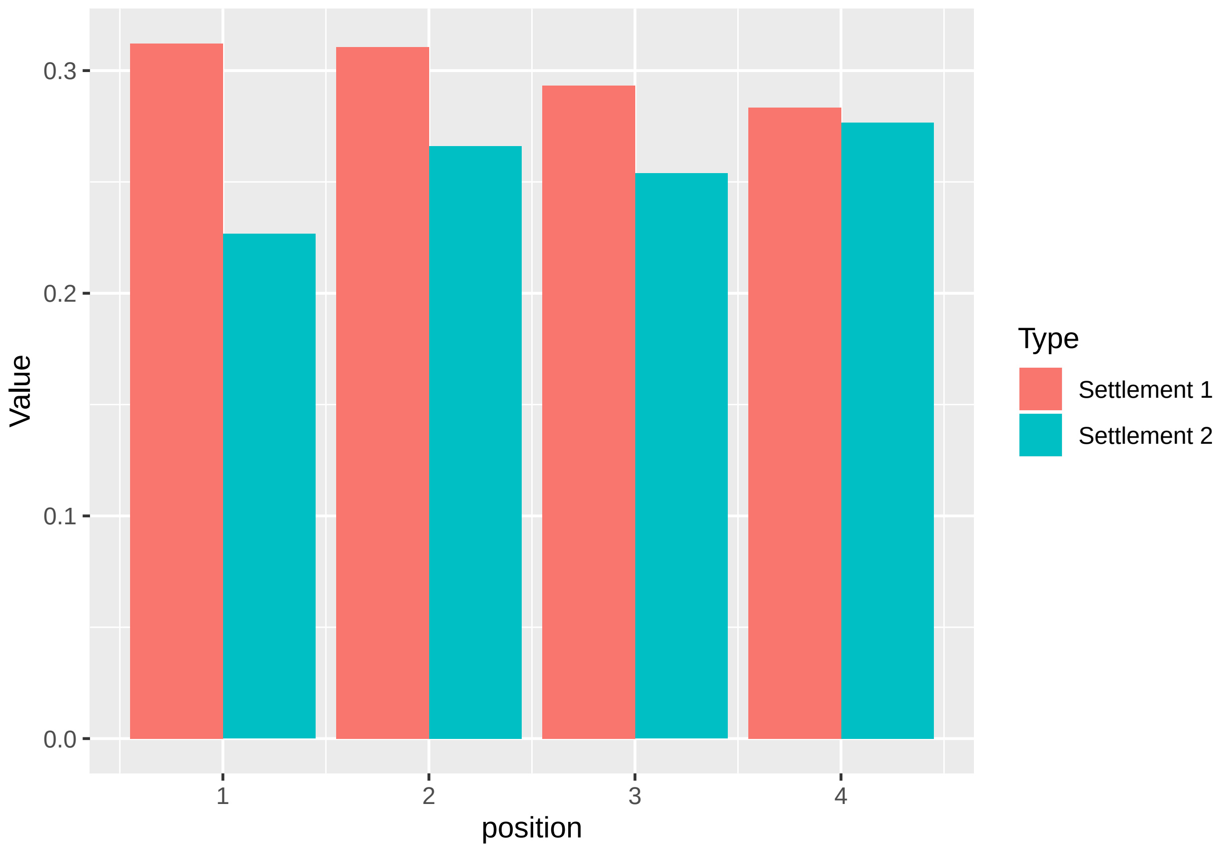
```
## Warning: Computation failed in `stat_smooth()`:
## object 'loees' of mode 'function' was not found

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.985

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 2.015

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 1.0159e-16

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.0602

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
## at 0.985

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 2.015

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 1.0159e-16

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 4.0602
```





Igrači koji počinju prvi u pravilu odabiru najkvalitetniju lokaciju, ali zato im je drugo naselje lošije pozicionirano jer su iscrpljene sve dobre lokacije. Iz grafa se vidi da za razliku od prvog igrača, drugi igrač u pravilu odabire podjednako dobru lokaciju za prvo naselje, ali odabire mnogo bolju lokaciju za drugo naselje

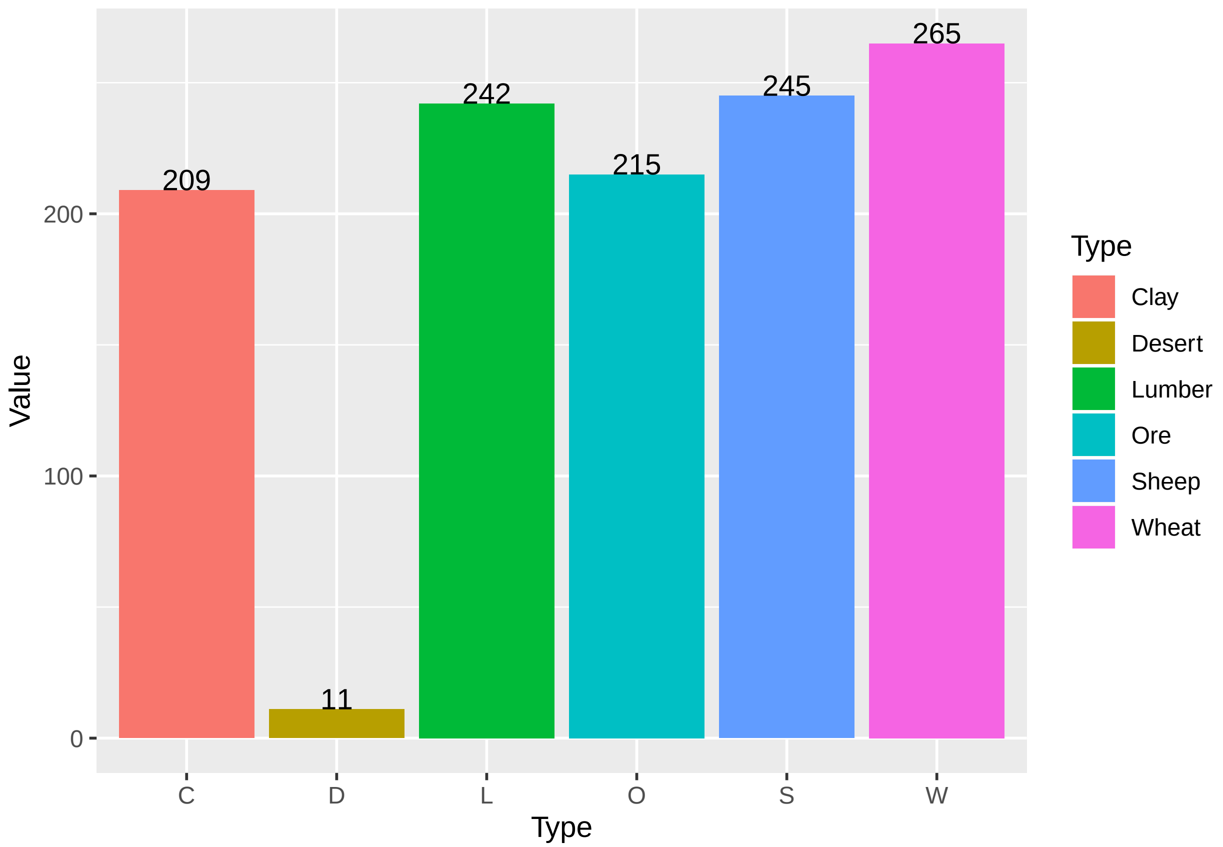
```
DF  %>% select(gameNum, points,production)  %>% group_by(gameNum) %>% mutate(position = order(gameNum))

DF %>% select(starts_with("Value"), points, gameNum) %>%
  transmute(Value1 = Value1.1 + Value1.2 + Value1.3, Value2 = Value2.1 + Value2.2 + Value2.3, gameNum, p
  group_by(gameNum) %>%
  mutate(position = order(gameNum)) -> X

gather(X, Type, Value, Value1:Value2) -> X
X %>% group_by(position,Type) %>% summarise(Value = mean(Value)) -> X

X %>% ggplot(aes(x = position, y = Value, fill = Type)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_discrete(labels = c("Settlement 1", "Settlement 2"))
```



Analiza portefelja

Analizom portefelja pokazuje se da igrači najčešće zauzimaju žito i ovce. U ranim stadijima igre najvažniji resursi su glina i drvo, što ova analiza ne pokazuje. Mana ovakvog oblika analiza je ta što ne uzimamo u obzir kvalitetu resursa, odnosno vjerovatnost da igrač dobije taj resurs.

```
DF %>% select(starts_with("Tile")) -> X
get.portofolio.table(X) %>% select(-starts_with("Tile")) %>% gather(Type, Value, L:S) %>% group_by(Type

X %>% ggplot(aes(x = Type, y = Value, fill = Type)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Value), vjust = 0) +
  scale_fill_discrete(labels = c("Clay", "Desert", "Lumber", "Ore", "Sheep", "Wheat"))
```

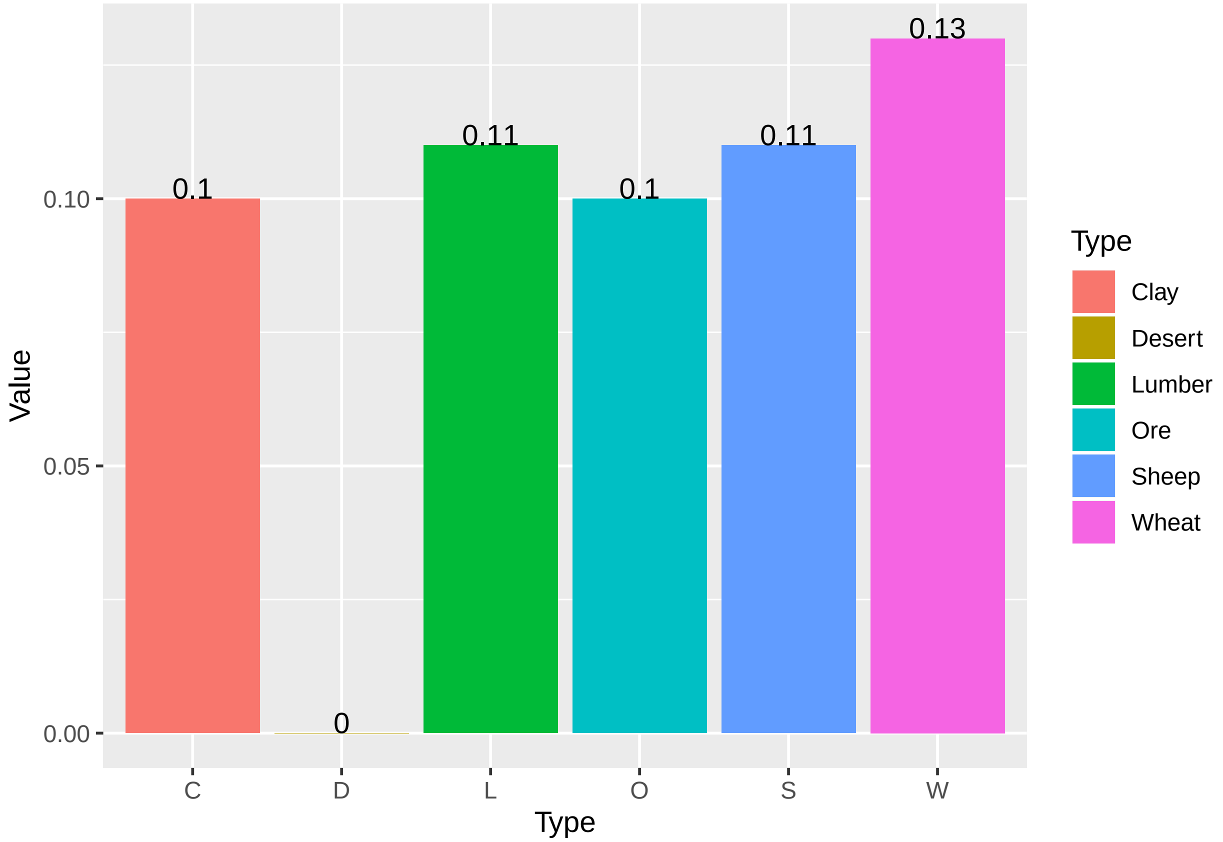


Ovaj graf prikazuje portefelj igrača, ali u ovom sluaju uzeli smo u obzir kvalitetu resursa. Za svaki resurs izračunali smo statistiku R_i kao težinsku sumu pojave resursa i, težine w_i predstavljaju vjerovatnost prikupljanja tog resursa,a f_i predstavljaju frekvenciju biranja resursa:

$$R_i = \frac{\sum_i w_i f_i}{N}$$

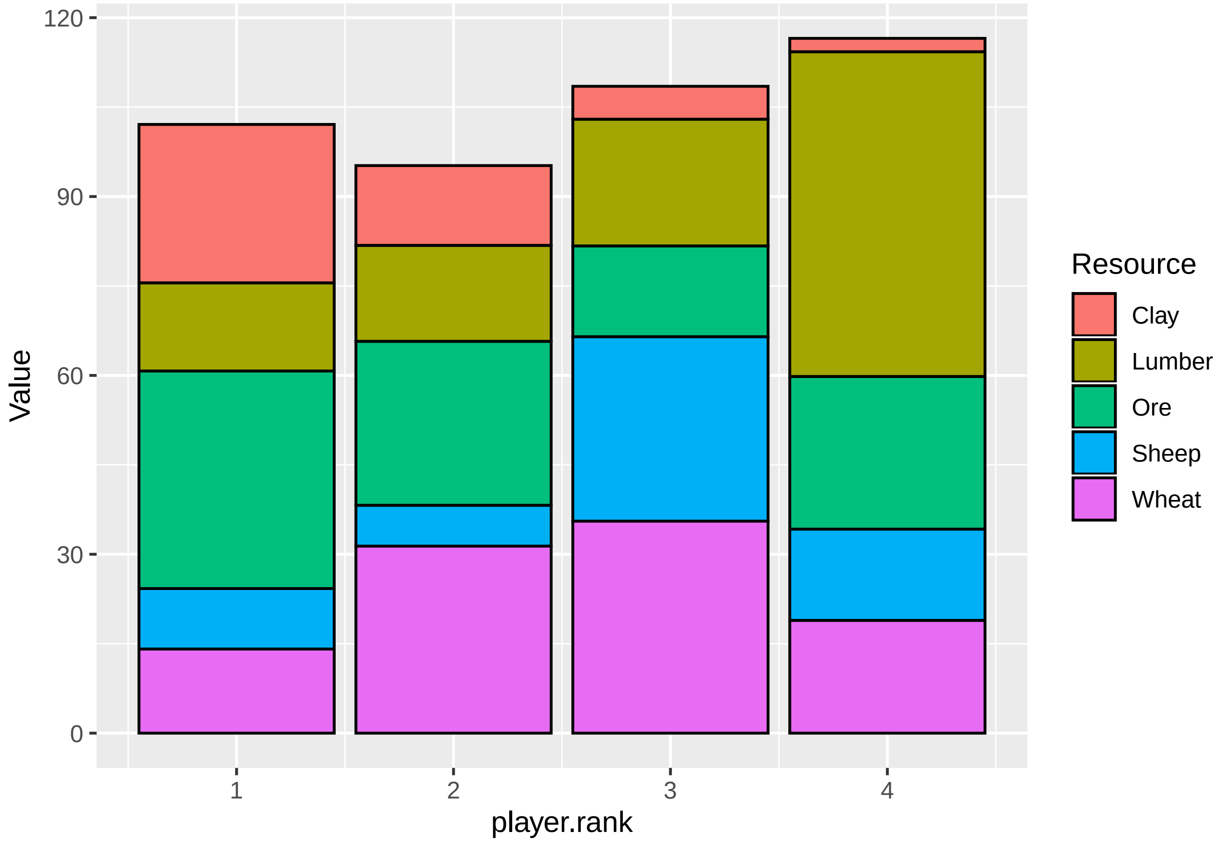
Time smo svakom igraču pridjelili distribuciju koja svakom resursu pridjeljuje vjerovatnost prikupljanja tog resursa. Svaki igrač će imati različitu distribuciju resursa, a nama je u interesu naći onu optimlanu. Na ovom grafu je prikazana prosječna distribucija u ovih 50 partija.

```r
DF %>% select(starts_with("Tile"),starts_with("Value")) -> X
df <- get.weigthened.portofolio(X)
df[is.na(df)] <- 0
df %>% summarise(C=mean(C) %>% round(2),D=mean(D)%>% round(2), L=mean(L)%>% round(2), O=mean(O)%>% roun

X %>% ggplot(aes(x = Type, y = Value, fill = Type)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Value), vjust = 0) +
  scale_fill_discrete(labels = c("Clay", "Desert", "Lumber", "Ore", "Sheep", "Wheat"))
```



Tada smo analizirali kako se razlikuje portefelj igrača u odnosu na završnu poziciju u igri. Podaci pokazuju da najbolji igrači ipak imaju otprilike podjednako jake pozicije na drvu i glini, ali ono što ih dosta razlikuje od ostalih pozicija jest najjača pozcija na kamenu

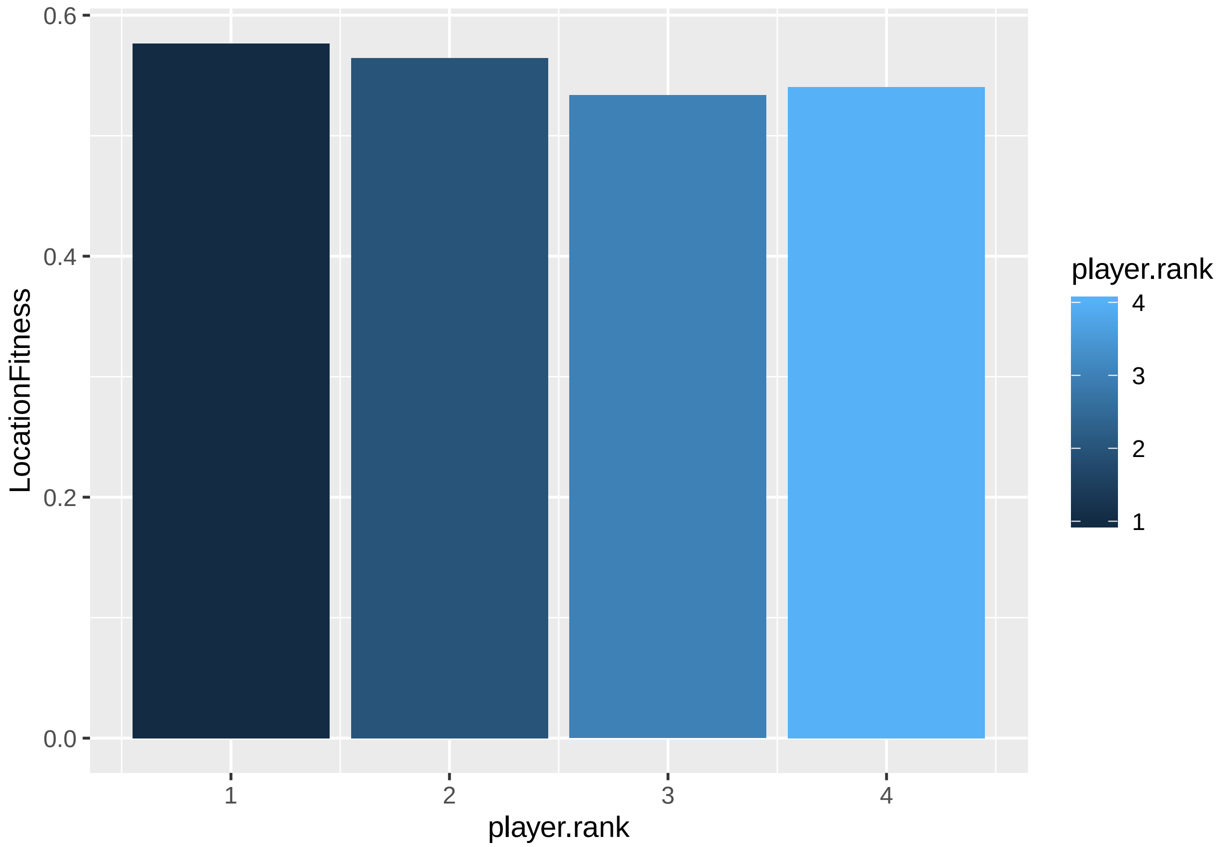
```
DF %>% group_by(gameNum) %>% mutate(player.rank = dense_rank(-points))  %>% select(gameNum, player.rank
DF$id <- 1:nrow(DF)
X$id <- 1:nrow(X)
X$gameNum <- NULL
inner_join(DF,X, by = c("id" = "id")) %>% select(starts_with("Tile"),starts_with("Value"),player.rank,i

df <- get.weigthened.portofolio(X)
df[is.na(df)] <- 0
df$id <- 1:nrow(df)
inner_join(df,X,by=c("id"="id")) %>% select(C:W,player.rank) -> X

X %>% group_by(player.rank) %>%
  summarise(C=mean(C) %>% round(2),D=mean(D)%>% round(2), L=mean(L)%>% round(2), O=mean(O)%>% round(2),
  gather(Resource, Value, C:W) %>%
  filter(Resource != "D") -> Y

Y %>% ggplot(aes(x = player.rank, y = Value, fill = Resource)) +
  geom_bar(stat = "identity",color = "black") +
  scale_fill_discrete(labels = c("Clay", "Lumber", "Ore", "Sheep", "Wheat"))
```



Pozicije igrača i vrijednosti gradova

Iznenađujuće vidimo da najbolji igrači nemaju nužno početne lokacije koje donose najviše resursa. Možemo ustvrditi da je bitnija stvar imati jak i raznovrsan portefelj proizašao iz početnih lokacija negoli pozicije koje donose najviše resursa.

```
DF %>% group_by(gameNum) %>% mutate(player.rank = dense_rank(-points))  %>% select(gameNum, player.rank
DF$id <- 1:nrow(DF)
X$id <- 1:nrow(X)
X$gameNum <- NULL
inner_join(DF,X, by = c("id" = "id")) %>% select(player.rank, starts_with("location")) %>% transmute(Lo
X %>% group_by(player.rank) %>% summarise(LocationFitness = mean(LocationFitness)) -> X

X %>% ggplot(aes(player.rank, LocationFitness, fill = player.rank)) +
  geom_bar(stat = "identity")
```



Prediktivno modeliranje

U ovom dijelu pokušat čemo iskoristiti neke od postojećih metoda strojnog učenja da bismo predvidjeli dali će igrač pobjediti. Kao ciljnu labelu izabrali smo poziciju igrača na kraju igre. Zbog toga naš zadatak se svodi na klasifikaciju u 4 klase. Kao značajke modela koristimo:

- Portefelj igračevih resursa
- Načini na koje je pribavio resurse
- Načini na koje je izgubio resurse

```
# Get features for machine learning : player.rank is target variable
get.dataset(DF) -> data
glimpse(data)
```

```
## Observations: 200
## Variables: 12
## $ player.rank     <fct> 3, 2, 1, 3, 1, 3, 4, 2, 3, 1, 2, 2, 3, 1, 2, 2...
## $ production      <int> 38, 48, 44, 42, 60, 57, 44, 61, 44, 41, 47, 53...
## $ tradeGain       <int> 5, 8, 14, 12, 15, 12, 10, 16, 5, 4, 6, 2, 15, ...
## $ robberCardsGain <int> 2, 6, 9, 0, 16, 1, 8, 11, 5, 9, 5, 2, 12, 15, ...
## $ tradeLoss       <int> 10, 11, 24, 24, 28, 26, 18, 25, 11, 8, 10, 4, ...
## $ robberCardsLoss <int> 2, 1, 4, 6, 10, 6, 6, 6, 1, 3, 7, 4, 5, 15, 5,...
## $ tribute         <int> 4, 8, 0, 0, 0, 8, 8, 4, 9, 0, 0, 8, 12, 10, 0,...
## $ C               <dbl> 0.11111111, 0.00000000, 0.05555556, 0.00000000...
## $ L               <dbl> 0.25000000, 0.08333333, 0.00000000, 0.38888889...
## $ O               <dbl> 0.05555556, 0.19444444, 0.13888889, 0.13888889...
## $ S               <dbl> 0.00000000, 0.11111111, 0.33333333, 0.08333333...
## $ W               <dbl> 0.08333333, 0.19444444, 0.02777778, 0.00000000...
```

Za problem klasifikacije koristiti ćemo logističku regresiju.

```
# 5-fold CV
ctrl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 2)

train <- data %>% sample_frac(0.7)
test <- data %>% setdiff(train)

nn <- train(player.rank ~ .,
            data = train,
            method = "monmlp",
            trControl = ctrl,
            preProcess = c("center","scale"))
```

```
## ** Ensemble 1
## 0.8118878
## ** 0.8118878
##
## ** Ensemble 1
## 0.6157993
## ** 0.6157993
##
## ** Ensemble 1
## 0.4715811
## ** 0.4715811
##
## ** Ensemble 1
## 0.8140221
## ** 0.8140221
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6933869  :
##  [1]  1.049823e+04  3.580137e+03  5.307269e+03 -5.803623e+03  2.955607e+02
##  [6] -4.576014e+03  8.491988e+02  4.962297e+02  5.961725e+02 -1.385669e+02
## [11]  2.227193e+03 -2.062940e+03 -5.922441e+03  3.126787e+02  3.043282e+02
## [16] -4.220044e+02  1.419725e+03  7.836808e+03  2.446863e+03 -3.266400e+03
## [21] -7.509717e+01  2.239551e+03 -7.143209e+02 -2.095766e+03 -3.426943e+03
## [26] -3.094180e+03 -2.926611e+02  3.350828e+03  3.103034e+03  3.757753e+01
## [31] -4.649702e+03 -3.410687e+03 -3.109023e+03  2.309842e+03 -8.434106e+02
## [36] -4.631021e+03  7.192049e-01  1.478445e-01 -7.333638e-02  1.725446e-01
## [41] -1.976990e-01 -3.015996e-01 -2.412043e-01 -2.278524e-01 -2.990156e-01
## [46] -8.173374e-02  5.264358e-01  1.661861e-01 -2.332728e-01  3.579744e-01
## [51] -3.056989e-01 -1.344828e-01
## 0.6933869
## ** 0.6933869
##
## ** Ensemble 1
## 0.4180136
## ** 0.4180136
```

```
##
## ** Ensemble 1
## 0.8079885
## ** 0.8079885
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6450671  :
##  [1] -9.815504e+02 -2.990245e+02 -5.394242e+02  6.985323e+02  3.761173e+02
##  [6]  3.802225e+02 -1.597101e+02 -6.035314e+02 -3.398221e+02 -2.852688e+02
## [11] -1.583577e+02  6.394433e+02 -1.044933e+03 -1.044463e+02 -6.853911e+02
## [16]  6.048058e+02  1.135550e+03  1.044493e+03 -1.307290e+03 -9.913153e+02
## [21] -1.307606e+03 -5.564297e+02 -1.295874e+03 -3.266285e+02 -2.072156e+03
## [26] -9.583833e+01  8.333774e+01 -5.328814e+01 -7.017993e+02  4.457239e+02
## [31] -1.597097e+02 -8.319349e+02 -1.006349e+03 -7.571397e+02 -4.799047e+02
## [36] -6.739885e+02 -8.707303e-01  5.646696e-02 -5.108214e-02  5.192419e-01
## [41]  5.953219e-01 -4.882929e-01 -4.447905e-01 -5.044948e-01  2.506056e-01
## [46]  3.331481e-01  1.773192e-01 -8.000580e-02 -7.159366e-02  1.521790e-01
## [51]  4.449088e-01  1.565470e-01
## 0.6450671
## ** 0.6450671
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.5261486  :
##  [1]  171.72490500   97.55656871  118.49664687 -160.63550461  -48.19282861
##  [6]  -64.72972963  -11.07180567   95.31362072   37.52554391   35.77608864
## [11]   -4.84718168 -139.52250966  -78.68287621   53.60023471  349.55056378
## [16]   75.55075320  -74.56085776  310.06105403  -66.22258670   43.19181827
## [21] -151.25357235  -59.58351069 -123.51924444  335.00376587 -210.67133583
## [26]   34.32047567   78.64814423   56.82947970   18.73703660  232.75469842
## [31]  -69.97890019 -160.73804762   -8.78803509   60.16011676  -29.54097289
## [36]   40.00932746 -126.00364167  -97.58134177 -208.74792712  -91.90788198
## [41]   79.83471841 -119.09019574   32.46852808  -31.61121810  -83.32044426
## [46]  -52.01016506  -56.55627930  -88.37752294    3.41506946  144.21001179
## [51]   18.63115606   21.33008747  187.02180244   73.14582693 -203.32469472
## [56] -157.33443092  130.92008517  137.01574914   33.54940450 -109.75471489
## [61]    0.66132581    0.21269209   -0.13088542    0.07433073    0.33346206
## [66]    0.38766707   -0.29756032   -0.65923224   -0.32505969   -0.72455175
## [71]   -0.32023334    0.01883224   -0.40805038    0.70646260    0.10087958
## [76]    0.84785231    0.49345249   -0.44661388    0.13934627   -0.32543849
## [81]    0.48478704   -0.25276973   -0.66110901    0.09777277
## 0.5261486
## ** 0.5261486
##
## ** Ensemble 1
## 0.7992874
## ** 0.7992874
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6576418  :
##  [1] -1.211083e+03 -2.573694e+02 -9.620189e+01  1.607831e+02  4.613665e+01
##  [6]  3.881948e+02  4.479528e+02 -1.113552e+02 -1.658851e+01  4.038185e+02
## [11] -6.474294e+01 -7.598280e+02  1.482639e+03  1.078879e+03  1.034998e+03
## [16] -1.331841e+03 -3.518156e+02 -4.358026e+02  3.196713e+02  4.000107e+02
## [21]  1.861865e+02 -6.181532e+01  2.308329e+02 -3.902437e+02  1.382077e+03
## [26]  1.043718e+02  8.249471e+02 -2.840837e+02 -3.528840e+02 -9.436170e+02
## [31]  5.488184e+02  6.434325e+02  8.209950e+02  1.693645e+02  6.966862e+02
## [36]  4.321363e+02 -1.427651e-01  6.567211e-01 -6.589048e-04  1.311169e-01
## [41] -1.810181e-01 -5.698458e-01  5.298766e-01 -3.439875e-01 -4.133525e-02
## [46]  7.792614e-02 -6.281999e-01  1.354976e-01  5.091278e-01 -1.512570e-01
## [51]  1.372438e-01  1.305011e-01
## 0.6576418
## ** 0.6576418
##
## ** Ensemble 1
## 0.5689597
## ** 0.5689597
##
## ** Ensemble 1
## 0.7934857
## ** 0.7934857
##
## ** Ensemble 1
## 0.6510232
## ** 0.6510232
##
## ** Ensemble 1
## 0.4193767
## ** 0.4193767
##
## ** Ensemble 1
## 0.8077532
## ** 0.8077532
##
## ** Ensemble 1
## 0.5614164
## ** 0.5614164
##
## ** Ensemble 1
## 0.5108567
## ** 0.5108567
##
## ** Ensemble 1
## 0.7946423
## ** 0.7946423
##
## ** Ensemble 1
## 0.6443729
## ** 0.6443729
##
## ** Ensemble 1
## 0.4795495
## ** 0.4795495
##
## ** Ensemble 1
## 0.808538
## ** 0.808538
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6925473  :
##  [1]  9.203989e+02  5.506230e+02  7.338233e+02 -1.091573e+02 -2.680115e+02
##  [6]  2.069568e+02  3.791376e+02  1.159619e+01  2.206702e+02  9.834205e+01
## [11] -1.633803e+02 -6.920698e+02 -2.080905e+02  4.345225e+01 -4.626984e+02
## [16] -2.757692e+02  1.262582e+02  3.608653e+02 -9.742822e+02 -3.355480e+02
## [21] -7.057116e+02 -6.941298e+01 -8.883905e+02 -2.009134e+02  1.488019e+02
## [26] -9.502467e+01 -9.699487e+02  2.512122e+02  9.472958e+01  2.427654e+02
## [31] -1.113317e+03  3.154470e+02 -1.446654e+02  3.824068e+02 -2.222365e+02
## [36] -1.292961e+02  6.692192e-01 -7.023506e-02 -2.218535e-01  2.621330e-01
## [41] -2.681581e-01 -5.769682e-01  4.113293e-01 -1.761508e-01 -2.078969e-01
## [46]  3.888350e-01  6.104809e-02 -3.409328e-02 -1.806765e-01  3.722519e-01
## [51] -3.981406e-01 -3.528071e-02
## 0.6925473
## ** 0.6925473
##
## ** Ensemble 1
## 0.519827
## ** 0.519827
##
## ** Ensemble 1
## 0.8112724
## ** 0.8112724
##
## ** Ensemble 1
## 0.7106744
## ** 0.7106744
##
## ** Ensemble 1
## 0.4755776
## ** 0.4755776
##
## ** Ensemble 1
## 0.7960045
## ** 0.7960045
##
## ** Ensemble 1
## 0.6797383
## ** 0.6797383
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.4781104  :
##  [1]  4.295020e+03 -1.043592e+02  2.866932e+03  1.679597e+03  1.681484e+03
##  [6]  4.690476e+02  1.486041e+03  2.047737e+03  1.601652e+03 -1.446136e+03
## [11] -5.548659e+02  3.813070e+03 -3.444202e+03  1.036832e+02  2.149869e+03
## [16]  2.083715e+03  2.293988e+02  2.582026e+02 -7.806468e+02 -1.347249e+03
## [21] -1.567724e+02 -1.045940e+03 -2.890036e+03  1.909957e+03 -5.070435e+02
## [26]  6.817341e+02  2.430504e+03  7.342410e+02  1.140256e+03  9.682660e+02
## [31]  2.315941e+03  4.622639e+02  1.642878e+03 -4.643939e+02 -4.527289e+02
## [36]  2.841470e+03  3.313529e+03  4.048310e+02  1.740863e+03 -1.902734e+03
## [41] -1.121167e+03 -1.565349e+03 -1.160432e+02  5.829875e+02 -8.952296e+01
## [46] -3.075849e+02  7.208251e+02 -8.576695e+02  2.645335e+03  1.754233e+03
## [51]  3.029521e+03 -2.423548e+03 -1.236844e+03 -2.500095e+03  7.009067e+02
## [56]  1.313563e+03  8.344880e+02  5.890624e+02  5.598304e+02 -3.641035e+01
## [61]  2.308286e-01  2.898554e-01  5.576944e-02  6.909284e-01 -1.830318e-01
## [66] -5.859751e-02  2.205376e-01 -7.305805e-01 -8.927291e-02 -8.813104e-01
## [71]  8.336960e-01  1.854394e-01  2.144635e-01  4.397966e-01 -6.196456e-01
## [76]  1.907826e-01 -5.183831e-01  5.009078e-02 -9.278933e-01  4.756113e-02
## [81]  9.393733e-01  9.334710e-02 -2.250317e-01 -2.624367e-01
## 0.4781104
## ** 0.4781104
##
## ** Ensemble 1
## 0.817778
## ** 0.817778
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.688637  :
##  [1] -199.43789067 -119.28008145  -23.73775446   63.67622156    4.05667999
##  [6]   64.99211174  -71.53400715  -42.16245268 -129.11755341  -21.34333073
## [11] -110.51051477  -63.92261819 -137.70046977   12.68388063 -122.17194728
## [16]   64.97225872   67.42325302  110.96661921  -50.60226174 -154.13893516
## [21]   10.78702002  -51.21347042  -49.32153532   -0.64186330  -99.50734342
## [26]  -21.27385705   28.96954578  -68.96003793   90.75743869  243.15022743
## [31] -158.13364729 -148.96956445  -30.89979170  -44.43947223 -138.10394465
## [36]   11.03900077   -0.43764071   -0.65915390    0.50296045   -0.08272826
## [41]   -0.26167822    0.31992098   -0.49973958   -0.06813213    0.45034371
## [46]    0.11113594    0.11028892    0.09989209    0.30929005    0.24120689
## [51]   -0.09220165    0.06442477
## 0.688637
## ** 0.688637
##
## ** Ensemble 1
## 0.4131427
## ** 0.4131427
##
## ** Ensemble 1
## 0.8133929
## ** 0.8133929
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.7199899  :
##  [1] -354.75630478 -342.86521594 -269.91900517  397.03683256   24.06394200
##  [6]  308.59803099 -128.01228693 -137.13106716  -44.20373995  -51.26916798
## [11]   -1.64473253  -50.42130227 -170.21044918 -120.34725011  306.09684908
## [16] -199.97972313   25.35880383  546.65099809   20.18956181 -303.96804173
## [21]  -30.77213495   52.93254764  -99.34464799  -40.25346581   22.61642973
## [26]   46.49202611  230.16447343  -16.10270284   17.92794436  -68.20376513
## [31]  331.61822538   71.12203265   -0.71157101  -94.34731104   50.12963351
## [36]  128.01252317   -0.68822593    0.30123190   -0.15813505    0.05494111
## [41]   -0.22722619   -0.25809366   -0.14538075    0.02949623    0.42014089
## [46]   -0.10579715   -0.09977084    0.01005183    0.60453724    0.13298125
## [51]    0.54570530   -0.12534395
## 0.7199899
## ** 0.7199899
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.4908256  :
##  [1]  1.529860e+03  7.466009e+02  2.308248e+02 -1.035057e+03  6.340431e+01
##  [6] -9.947377e+02  1.913153e+02  1.953674e+02  1.725894e+02  3.975785e+01
## [11]  4.231980e+02 -1.867563e+01 -1.068054e+02  2.270362e+02 -1.793427e+02
## [16]  3.187181e+02  3.298207e+02  9.240198e+01  2.863697e+02 -2.270341e+02
## [21]  1.930837e+02 -5.820003e+02 -3.630451e+02  3.021477e+02 -8.507154e+02
## [26] -5.106775e+02  6.355476e+02  5.495770e+02  4.103045e+01  8.797765e+02
## [31] -1.459437e+02  6.314637e+01  3.649414e+01 -7.164611e+00 -1.975338e+02
## [36]  4.018633e+02 -1.253040e+03 -5.680516e+02  2.468190e+02 -1.572006e+02
## [41] -3.650379e+02  5.214777e+02  1.543434e+02 -1.257309e+02  1.123136e+02
## [46]  5.410732e+02  4.352924e+02 -6.096916e+02 -2.687149e+02 -3.481183e+02
## [51] -4.116377e+02 -5.614054e+02 -1.272594e+02  9.701576e+01  9.868920e+01
## [56] -6.342901e+02 -7.012951e+02  6.106855e+02  7.862594e+02 -9.933087e+02
## [61]  7.254340e-01  1.473122e-01  6.590777e-01 -1.148720e-01  1.201320e-01
## [66] -2.322141e-01  2.836791e-02 -2.348720e-01 -7.626659e-01  7.632034e-01
## [71] -9.208348e-01  1.592642e-01 -7.203016e-01 -2.639274e-01  1.318644e-01
## [76] -1.024970e+00  5.736996e-01 -1.123616e-04  5.032811e-02  5.159771e-01
## [81]  6.281358e-02  4.981891e-01  3.502542e-01  6.817165e-02
## 0.4908256
## ** 0.4908256
##
## ** Ensemble 1
## 0.7941833
## ** 0.7941833
##
## ** Ensemble 1
## 0.6463804
## ** 0.6463804
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.513163  :
##  [1]  191.24052595   27.62226052 -566.17710808  221.74636959  -18.35253895
##  [6]  -75.89123692  -43.69413230  184.75062813  -82.21238043  -12.00741082
## [11]  -23.56049988 -125.25280790  136.90128929  -18.11423070  -88.90002758
## [16]  -88.17216489   10.57968897 -125.10413465   27.32881025  111.22467167
## [21]   -3.70227012  -29.46883436  -28.52918280  -40.43743590 -187.32905194
## [26]   89.11369619 -222.62234406  173.31488564   64.65624681  316.17842003
## [31] -348.86507563 -272.31513751  -38.63751064   30.96974900 -148.26182320
## [36] -362.40177153  456.47888421 -101.58327634  254.71557324 -134.34082415
## [41] -185.09877676 -369.41928078  258.04094965  198.82936313  286.16228633
## [46]   32.59431082  300.05438397  209.64333847 -368.24500969 -137.14030344
## [51] -356.83303484   50.08064846   90.54850087  -36.14839423  321.25553172
## [56]  142.78800371 -356.09526110 -109.84645842 -184.42326858  244.67203666
## [61]   -0.11294338    0.25345802   -0.59773276   -0.18764406   -0.87628691
## [66]   -0.02293015    0.60654721   -0.56039124    0.21517777    0.73142835
## [71]    0.53299518   -0.29371573   -0.47335528    0.63660573    0.66619641
## [76]   -0.35164911    0.09611487    0.51605143   -0.05297151   -0.43109198
## [81]   -0.47510767   -0.30870454    0.24453073   -0.28126934
## 0.513163
## ** 0.513163
##
## ** Ensemble 1
## 0.8024846
## ** 0.8024846
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6039148  :
##  [1] -114.14888156    2.16868629  -71.84543462   10.74939889   35.64504081
##  [6]   82.37883796  -44.44047822  -65.08586407 -110.55708883  -28.68974225
## [11]  -72.32956594  -11.50575921   41.79485841   70.54333137  144.12102203
## [16] -122.20741469  -27.10158219  -68.72629691  114.30279133   99.87061008
## [21]  -34.67108709  -16.50115805  -26.12867914  125.32472679   79.36652750
## [26]   26.02185385   34.83080698   38.71080644   17.83895221   -6.78621401
## [31] -219.31033049 -117.21753326  129.95342713   32.40655841   75.75886401
## [36]    5.56658104   -0.01199180    0.83959548    0.76408830   -0.33618069
## [41]   -0.74710658   -0.72607315   -0.81428822    0.28934565    0.55875462
## [46]   -0.17120963    0.14278595    0.07252633    0.29348618    0.19227212
## [51]   -0.03211740   -0.08069123
## 0.6039148
## ** 0.6039148
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.5520207  :
##  [1]   89.28905313   21.41943945   33.74834114  -42.30114094 -111.36758300
##  [6]   33.78594161  180.80874468  346.65741536   14.50566976   20.79200515
## [11]  151.97418982  121.10510215  178.45707374   87.77190518   80.20683099
## [16]   96.60341528  -26.62156828  293.79770915   62.02717212  172.63037942
## [21] -312.34652659  -91.11982597  -79.81277335   -1.57483029  -16.60440404
## [26]   18.27020173 -299.58161785 -114.40129901 -114.32019507  -68.63124572
## [31] -171.39229749   17.77047149  -38.35902553   38.39037309   28.86958366
## [36] -244.34434837  279.71473496  137.01205464  213.85155203    1.87501219
## [41]  -26.77883941   66.41956091  -84.65057869 -161.02127458  -89.52627063
## [46] -134.65766698 -145.76588582  -60.33698198 -275.02899350  -92.06957874
## [51]  335.38371324   94.68567664  -70.82894908  165.96632012 -215.12626602
## [56]  -56.35338047   91.10029757   12.21223116 -272.81270588  191.53800272
## [61]    0.35063405   -0.21353258    0.02427571    0.72602347    0.33326074
## [66]   -0.05260795    0.38228854   -0.53448457   -0.03238173    0.23750988
## [71]   -0.46351923   -0.06881145   -0.65196382    0.69053154    0.40044048
## [76]   -0.43886106    0.12171628    0.31088238   -0.07710371    0.06156300
## [81]   -0.56067122   -0.66768284    0.05736284   -0.27813137
## 0.5520207
## ** 0.5520207
##
## ** Ensemble 1
## 0.8195349
## ** 0.8195349
##
## ** Ensemble 1
## 0.6192643
## ** 0.6192643
##
## ** Ensemble 1
## 0.4825654
## ** 0.4825654
##
## ** Ensemble 1
## 0.8205496
## ** 0.8205496
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6567845  :
##  [1]  1.504398e+03  6.012137e+02 -8.306047e+02 -5.035708e+02 -6.527044e+02
##  [6] -1.265599e+03  5.449559e+02 -6.981295e+01  3.459581e+02 -3.195018e+02
## [11]  4.107546e+02 -9.405987e+02 -3.550442e+02 -6.684553e+02  8.839553e+02
## [16]  9.473773e+02  5.395889e+02  2.413728e+02  1.149338e+03  3.354795e+02
## [21]  6.581214e+02  2.438882e+02  3.764847e+02  1.194632e+03 -2.238078e+03
## [26] -5.074230e+02 -1.188955e+03  1.131433e+03 -5.000778e+01  1.024632e+03
## [31] -3.073074e+02 -1.443882e+02 -3.869358e+02  2.187840e+02 -4.666292e+02
## [36]  5.957737e+02 -2.630410e-01 -7.474293e-02 -7.743965e-01  1.351225e-01
## [41]  5.867231e-01  3.939910e-01  2.406830e-01 -1.421685e-02 -2.082675e-01
## [46] -5.130535e-01  2.625215e-01  1.196520e-01 -2.032914e-01  2.522811e-01
## [51]  2.761632e-01 -3.165987e-01
## 0.6567845
## ** 0.6567845
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.5166798  :
##  [1] -1.839680e+03 -6.257212e+02 -1.167230e+03  1.023515e+03  2.785063e+02
##  [6]  6.992128e+02 -1.772737e+02 -2.036246e+02  2.749855e+01  4.278095e+02
## [11] -2.824788e+02  5.152902e+02 -2.952447e+02  5.020506e+02 -5.696070e+01
## [16] -1.001258e+03 -1.055147e+02  2.452500e+02  3.419956e+02 -7.146080e+02
## [21] -2.697210e+02  6.042765e+01 -2.966426e+01 -5.278462e+02 -7.192586e+02
## [26]  4.219410e+02  4.354822e+02  1.727906e+02 -3.463837e+02  8.622888e+02
## [31]  5.755531e+02 -4.922867e+02 -2.802478e+02  7.748718e+02  1.111502e+03
## [36]  3.270179e+02 -1.206097e+03 -6.465746e+02 -6.519936e+02  7.930821e+02
## [41] -1.584726e+02  1.697769e+03 -7.370837e+02 -3.643477e+02 -1.048175e+03
## [46]  6.900399e+02 -4.640536e+02 -1.447404e+03 -2.391797e+02  7.158034e+02
## [51] -9.466792e+02 -5.426813e+02 -6.673891e+01  6.298680e+02 -7.126611e+02
## [56] -9.253050e+02 -5.888214e+02  2.946348e+02 -1.949923e+01 -2.097011e+03
## [61] -7.753856e-01  8.760344e-02 -1.238546e-01  8.217828e-04  2.859560e-02
## [66]  3.031632e-01  4.137357e-01 -5.489165e-01  4.612474e-01 -4.152304e-01
## [71] -8.573182e-02 -6.138998e-01  2.168412e-01  2.831162e-01 -3.892043e-01
## [76] -1.887212e-01  9.179340e-01  6.543872e-01  9.751926e-02  2.680198e-01
## [81]  4.807690e-02  8.444502e-01 -1.188994e+00 -4.279910e-01
## 0.5166798
## ** 0.5166798
##
## ** Ensemble 1
## 0.802791
## ** 0.802791
##
```

```
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.6816855  :
##  [1]   531.8074208     4.7550403   341.5528490    63.0805463  -771.0765566
##  [6]  -626.7868032  1524.0013232  1136.0749806  1320.6972761   262.1509240
## [11]  1354.8491451   283.9129536   862.4231099   110.3999262  -995.3394679
## [16]   458.8769937    16.1854628  -622.8783777  -392.9260682 -1084.6917098
## [21]  -369.5844845  -536.0180411   369.6761531 -1581.4972041 -1403.2877532
## [26] -1305.6227351  -571.5244914  1244.0630375    43.0699638   654.1217514
## [31]  -406.7891643  -322.5421408  -515.5648319  -101.6652308  -495.2325456
## [36]   359.9654644    -0.0302713    -0.4895641    -0.7560137    -0.1869916
## [41]     0.3588113     0.4444748     0.1952044     0.1622457    -0.4530422
## [46]     0.1088478     0.2830871     0.1381182     0.1673629    -0.1539357
## [51]     0.3079947    -0.1856221
## 0.6816855
## ** 0.6816855
##
## ** Ensemble 1
## 0.4695355
## ** 0.4695355
##
## ** Ensemble 1
## 0.8071322
## ** 0.8071322
##
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.5919265  :
##  [1] -239.27706629 -400.11840346 -903.32839982  246.51343064  135.84257561
##  [6]   52.74562758 -373.80057114 -246.76964683 -305.57944851 -100.51710400
## [11] -126.06905643  -83.50800273  628.52613684  141.42029496   40.88712926
## [16] -427.23911468  -50.76949569 -275.28505768 -189.71796931  -93.97010391
## [21]  -95.30929995   74.59217290    2.53564556  766.58218440  -46.93210518
## [26]   59.55240415  895.21683524  -69.03758806 -165.21504342  279.50646068
## [31] -335.46624173 -245.03657263 -379.25106065 -313.44078430 -511.16286381
## [36]  698.72602469   -0.56706920    0.21344562   -0.01416322   -0.13251467
## [41]   -0.35547066    0.28077100   -0.74714764    0.19298414    0.77291079
## [46]    0.50089212    0.79174569   -0.96374137    0.14354950   -1.40615590
## [51]   -0.03205747    1.27663908
## 0.5919265
## ** 0.5919265
##
## ** Ensemble 1
## 0.5305157
## ** 0.5305157
##
## ** Ensemble 1
## 0.7991041
## ** 0.7991041
##
## ** Ensemble 1
## 0.6683815
## ** 0.6683815
##
```

```
## ** Ensemble 1
## 0.4698101
## ** 0.4698101
## 
## ** Ensemble 1
## 0.7964563
## ** 0.7964563
## 
## ** Ensemble 1
## 0.6354812
## ** 0.6354812
## 
## ** Ensemble 1
## Complex eigenvalues found for method = BFGS
## coefficients for function value 0.5595042  :
##  [1] -495.85749194 -243.65007488 -477.30706600  335.43157577  192.59125168
##  [6] -359.93943778   71.60407245 -238.73130104 -160.99025651  114.78025817
## [11]  279.62966358  302.86802053 -533.63109138   15.77563522 -213.18842211
## [16]  234.68194271   92.26326553   57.48515862  -16.05678852 -230.44831745
## [21]  133.65551327  115.13398561  115.94010140  -18.61851238 -434.05453028
## [26] -442.19504189 -286.60505629  449.91384467  151.79520571  137.95684735
## [31]  441.75131614  279.59211076  -91.24693342  -35.06972725   21.00068594
## [36]   34.29447955  -76.93842429 -188.08568999  103.81059011   10.73873919
## [41]   -2.09702854   16.94992056  316.47004463    5.29126889   91.91847654
## [46]  -49.90976527  -47.35948229 -190.63124829 -945.44871746 -131.01382125
## [51] -467.66462052  288.12743530   16.20869276  764.01025178 -337.40981229
## [56] -402.34192287 -549.96266176 -273.22206763 -472.30151798 -327.40459082
## [61]   -0.33296160   -0.10334347   -0.32541108    0.23010142   -0.26366416
## [66]    0.20202419   -0.07693557    0.34524351    0.48546717   -0.35400883
## [71]   -0.60103493   -0.27661819    0.54528944   -0.49986635   -0.35626178
## [76]   -0.05646511    0.74223360   -0.04369051   -0.22864617    0.34554902
## [81]    0.23116781    0.28335229    0.14534195    0.19231078
## 0.5595042
## ** 0.5595042
## 
## ** Ensemble 1
## 0.8085567
## ** 0.8085567
```

```
lr <- train(player.rank ~ .,
                data = train,
                method = "LogitBoost",
                trControl = ctrl)
```

```
test$pred <- predict(nn, test %>% select(-player.rank))
confusionMatrix(test$pred, test$player.rank)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  1  2  3  4
##          1 11 11  3  1
##          2  0  0  0  0
##          3  6 11 11  6
##          4  0  0  0  0
```

```
##
## Overall Statistics
##
##                Accuracy : 0.3667
##                  95% CI : (0.2459, 0.501)
##     No Information Rate : 0.3667
##     P-Value [Acc > NIR] : 0.5485
##
##                   Kappa : 0.1499
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.6471   0.0000   0.7857   0.0000
## Specificity            0.6512   1.0000   0.5000   1.0000
## Pos Pred Value         0.4231      NaN   0.3235      NaN
## Neg Pred Value         0.8235   0.6333   0.8846   0.8833
## Prevalence             0.2833   0.3667   0.2333   0.1167
## Detection Rate         0.1833   0.0000   0.1833   0.0000
## Detection Prevalence   0.4333   0.0000   0.5667   0.0000
## Balanced Accuracy      0.6491   0.5000   0.6429   0.5000
```

```r
test$pred <- predict(lr, test %>% select(-player.rank))
confusionMatrix(test$pred, test$player.rank)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 1 2 3 4
##          1 6 5 1 2
##          2 2 3 4 1
##          3 2 4 4 1
##          4 0 0 0 1
##
## Overall Statistics
##
##                Accuracy : 0.3889
##                  95% CI : (0.2314, 0.5654)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 0.2933
##
##                   Kappa : 0.1502
##
##  Mcnemar's Test P-Value : 0.4672
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.6000  0.25000   0.4444  0.20000
## Specificity            0.6923  0.70833   0.7407  1.00000
## Pos Pred Value         0.4286  0.30000   0.3636  1.00000
## Neg Pred Value         0.8182  0.65385   0.8000  0.88571
## Prevalence             0.2778  0.33333   0.2500  0.13889
## Detection Rate         0.1667  0.08333   0.1111  0.02778
## Detection Prevalence   0.3889  0.27778   0.3056  0.02778
## Balanced Accuracy      0.6462  0.47917   0.5926  0.60000
```