LA2

Kishor

2022-09-24

```
df = read.csv('player_stats.csv')
head(df)
```

```
##    Player        Country          Team                   Agents Maps   K   D
## 1    TenZ         Canada     Sentinels ['Jett', 'Reyna', 'Raze']    9 206 139
## 2  ScreaM        Belgium   Team Liquid       ['Sage', 'Phoenix']    9 177 131
## 3 ShahZaM  United States     Sentinels          ['Sova', 'Jett']    9 172 134
## 4    L1NK United Kingdom   Team Liquid     ['Brimstone', 'Omen']    9 147 123
## 5  Jamppi        Finland   Team Liquid       ['Jett', 'Killjoy']    9 155 130
## 6   Lakia    South Korea NUTURN Gaming          ['Sova', 'Raze']   11 174 146
##    A   KD  KDA ACS.Map K.Map D.Map A.Map
## 1 55 1.48 1.87     289  22.8  15.4   6.1
## 2 56 1.35 1.77     265  19.6  14.5   6.2
## 3 52 1.28 1.67     240  19.1  14.8   5.7
## 4 57 1.19 1.65     218  16.3  13.6   6.3
## 5 32 1.19 1.43     229  17.2  14.4   3.5
## 6 62 1.19 1.61     231  15.8  13.2   5.6
```

```
library(dplyr)
```

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

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

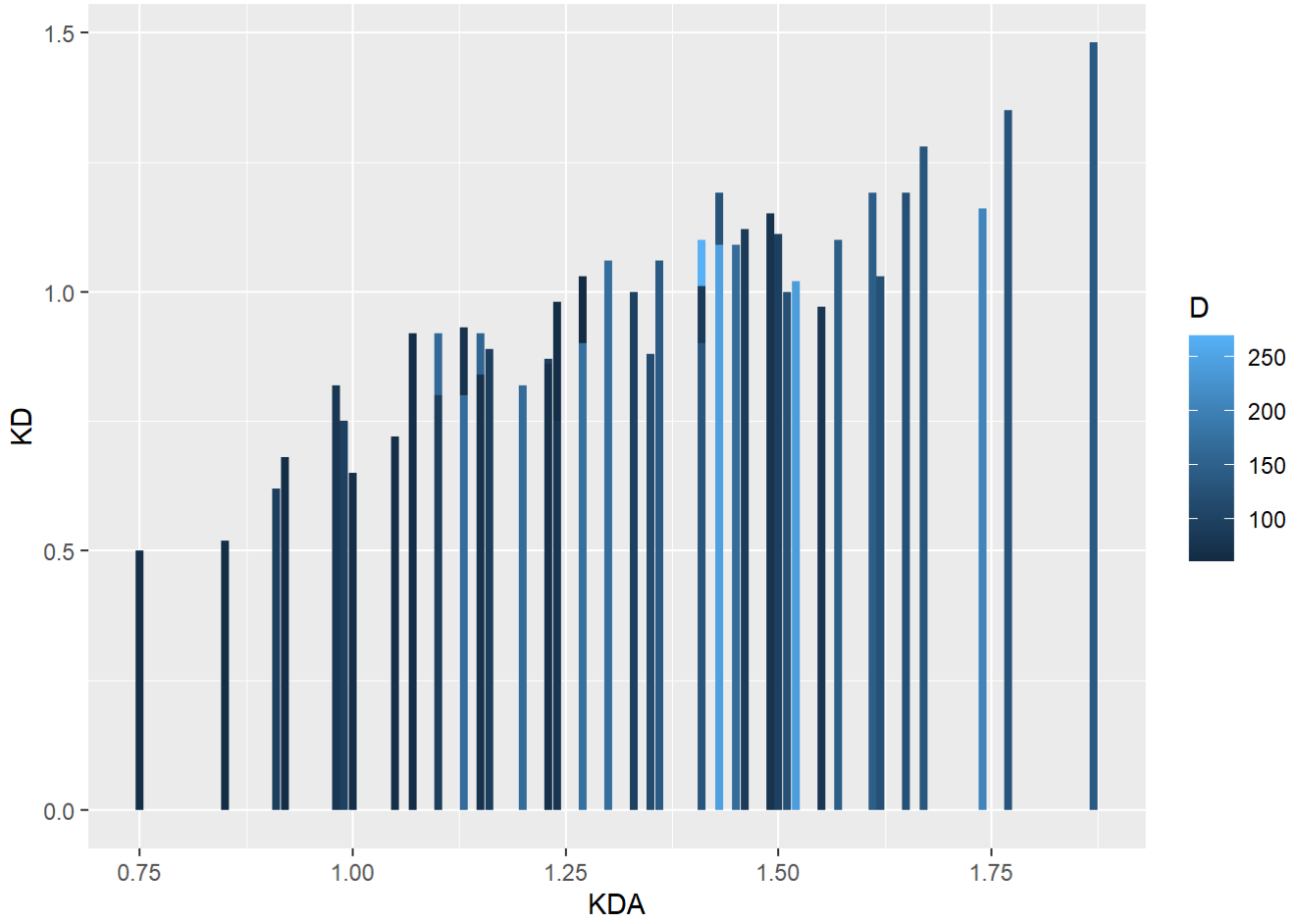
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
temp = df %>% filter(df$Country == "United Kingdom")
temp.aov = na.omit(aov(temp$Maps ~ temp$K, data = temp))
temp.aov
```

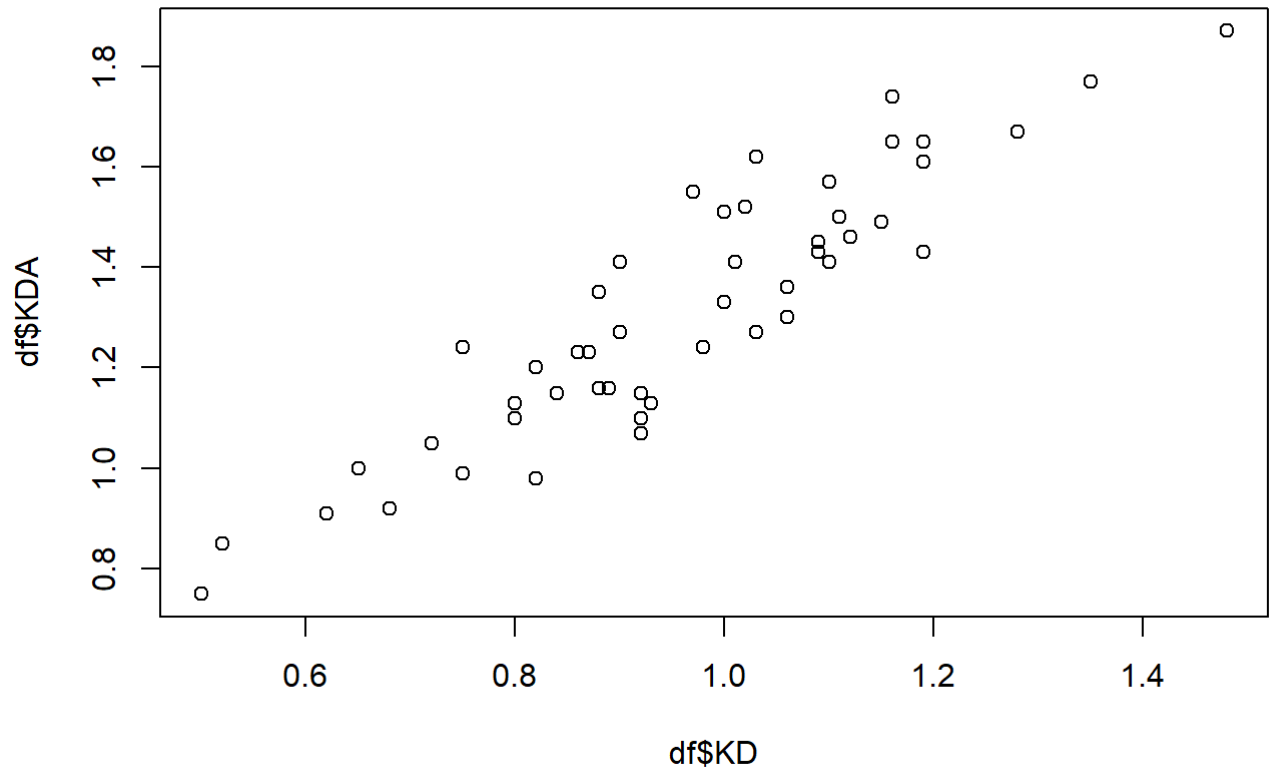
```
## Call:
##    aov(formula = temp$Maps ~ temp$K, data = temp)
##
## Terms:
##                   temp$K Residuals
## Sum of Squares  55.58306   3.21694
## Deg. of Freedom        1         3
##
## Residual standard error: 1.035525
## Estimated effects may be unbalanced
```

```
ggplot(df,aes(x=KDA, fill=D, y=KD))+geom_col(position ="dodge")
```



Creating a Scatter Plot

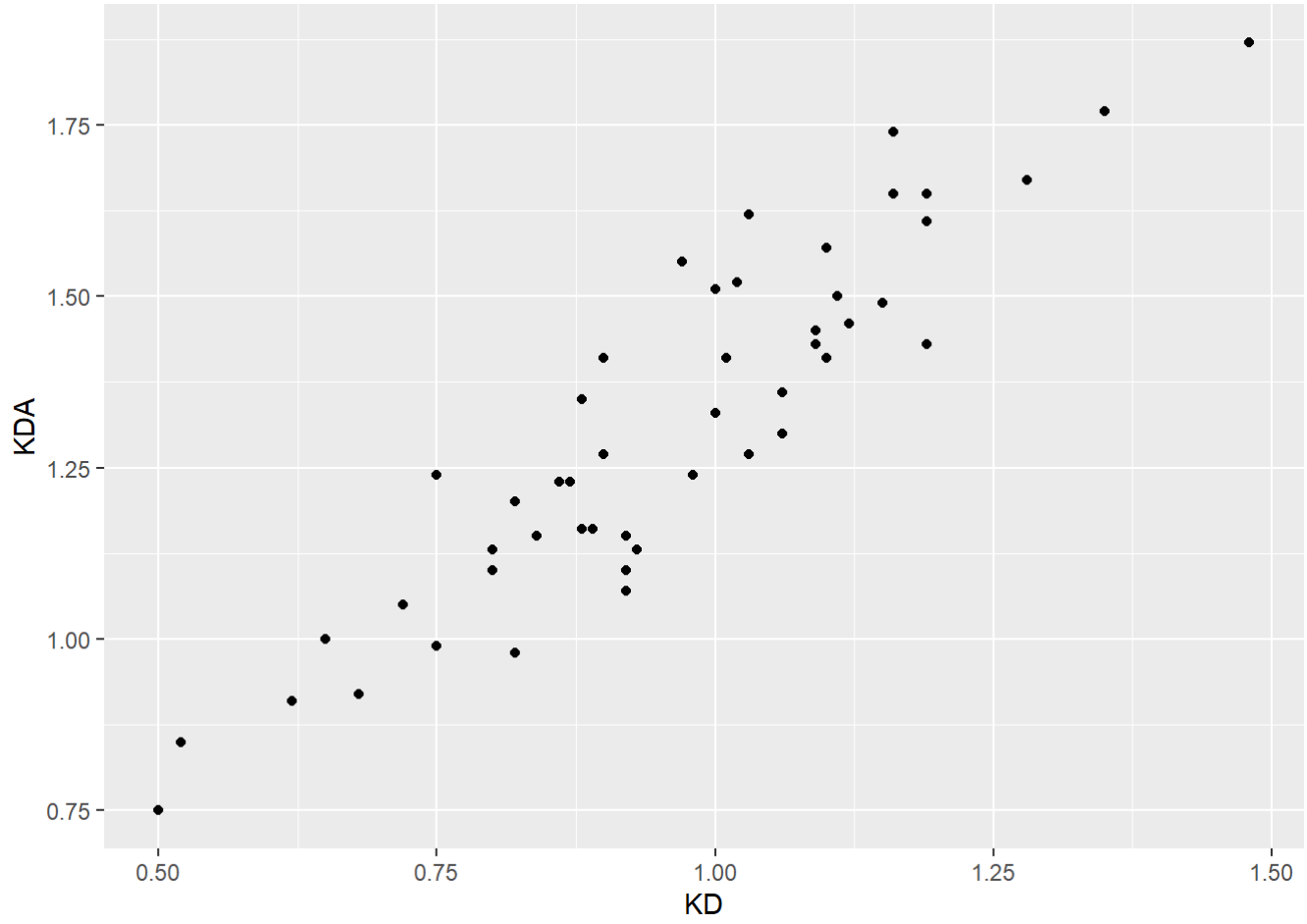
```
plot(df$KD, df$KDA)
```



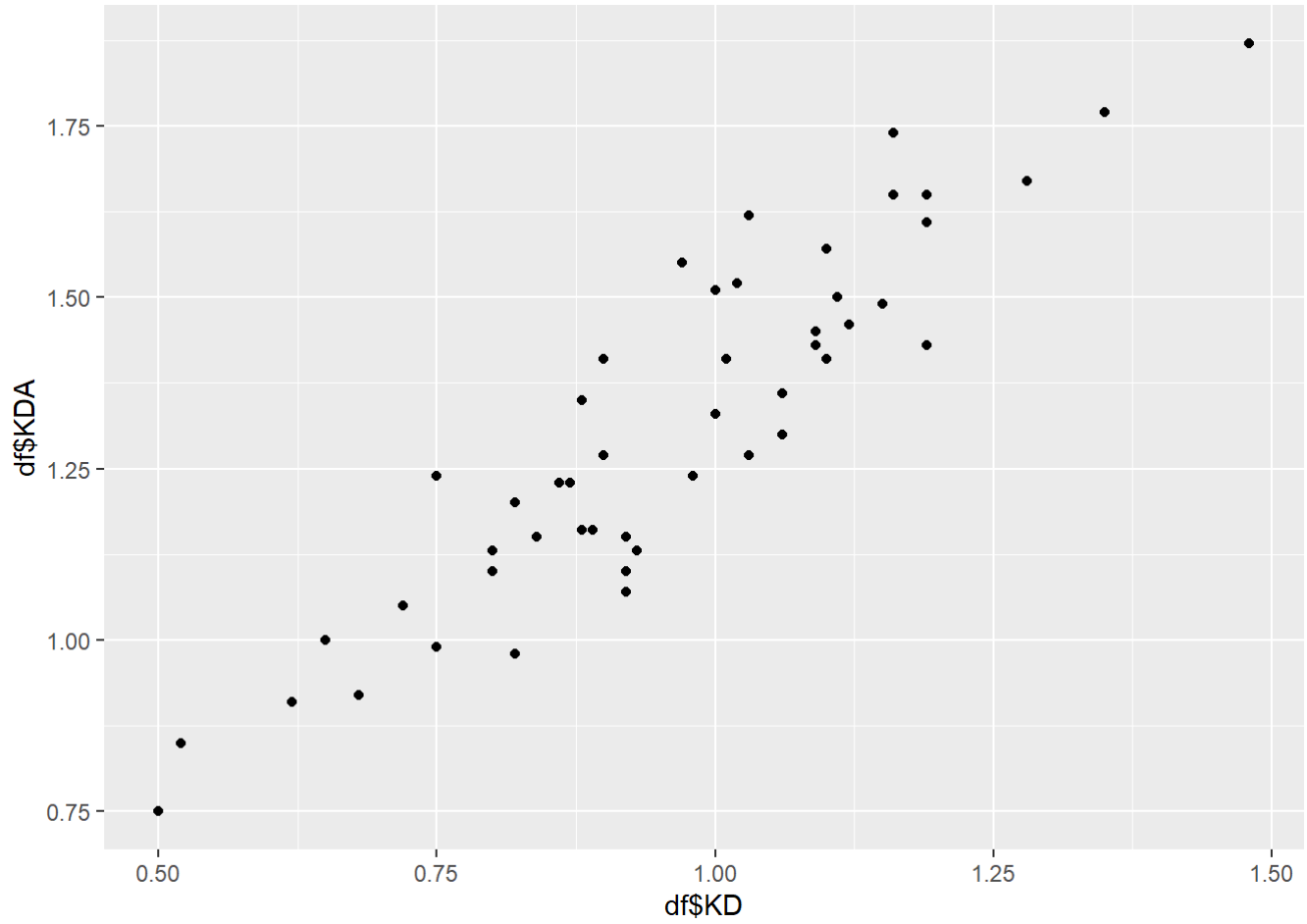

```
# Scatter plot with base graphics
```

Scatter plot with ggplot 2

```
library(ggplot2)
ggplot(df, aes(x = KD, y = KDA)) +geom_point()
```

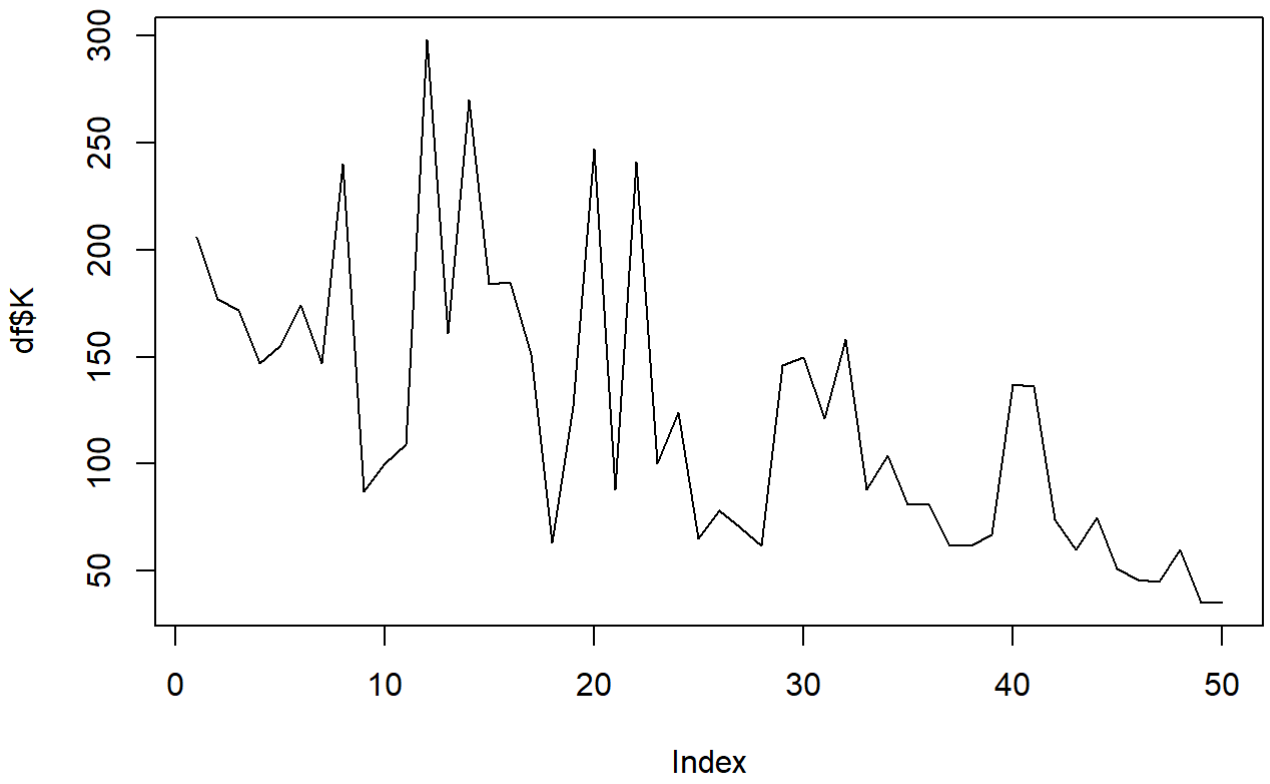



```
ggplot(data = NULL, aes(x =df$KD, y=df$KDA)) +
geom_point()
```

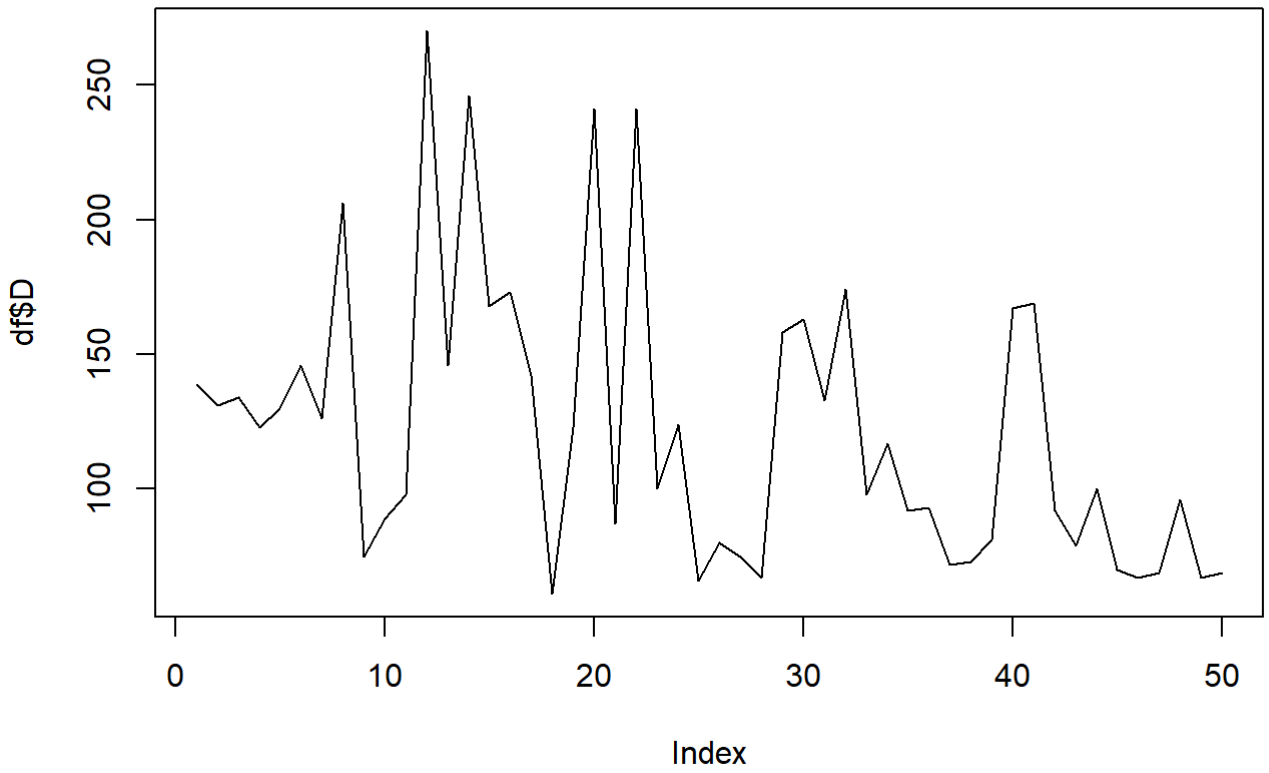


Creating a Line Graph

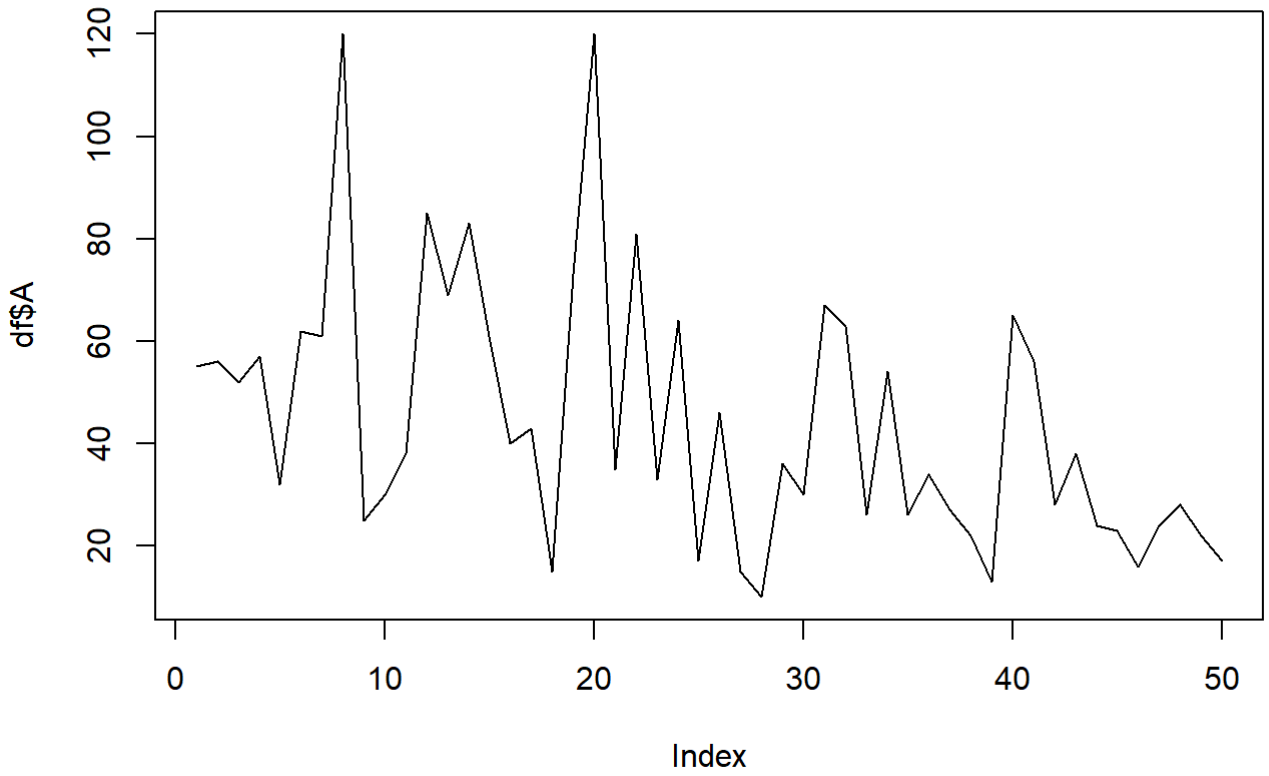
```
plot(df$K,type = "l")
```



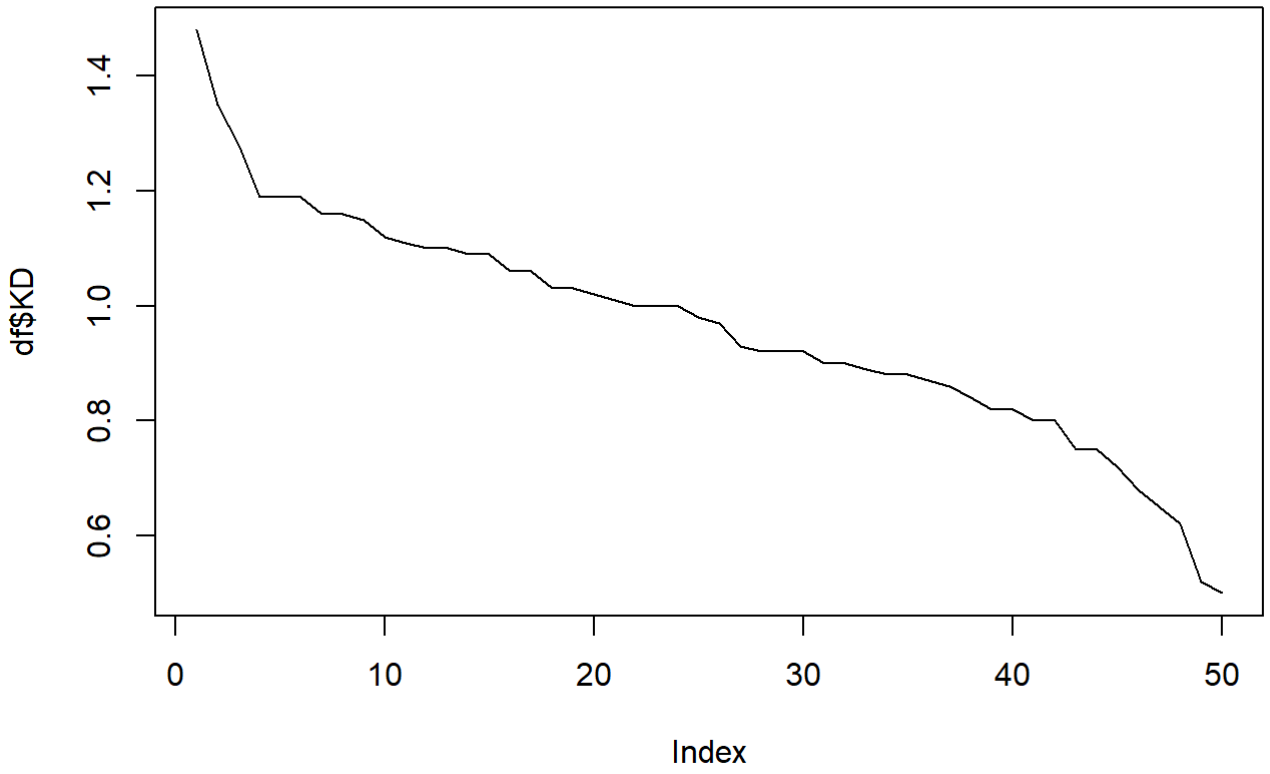

```
plot(df$D,type = "l")
```



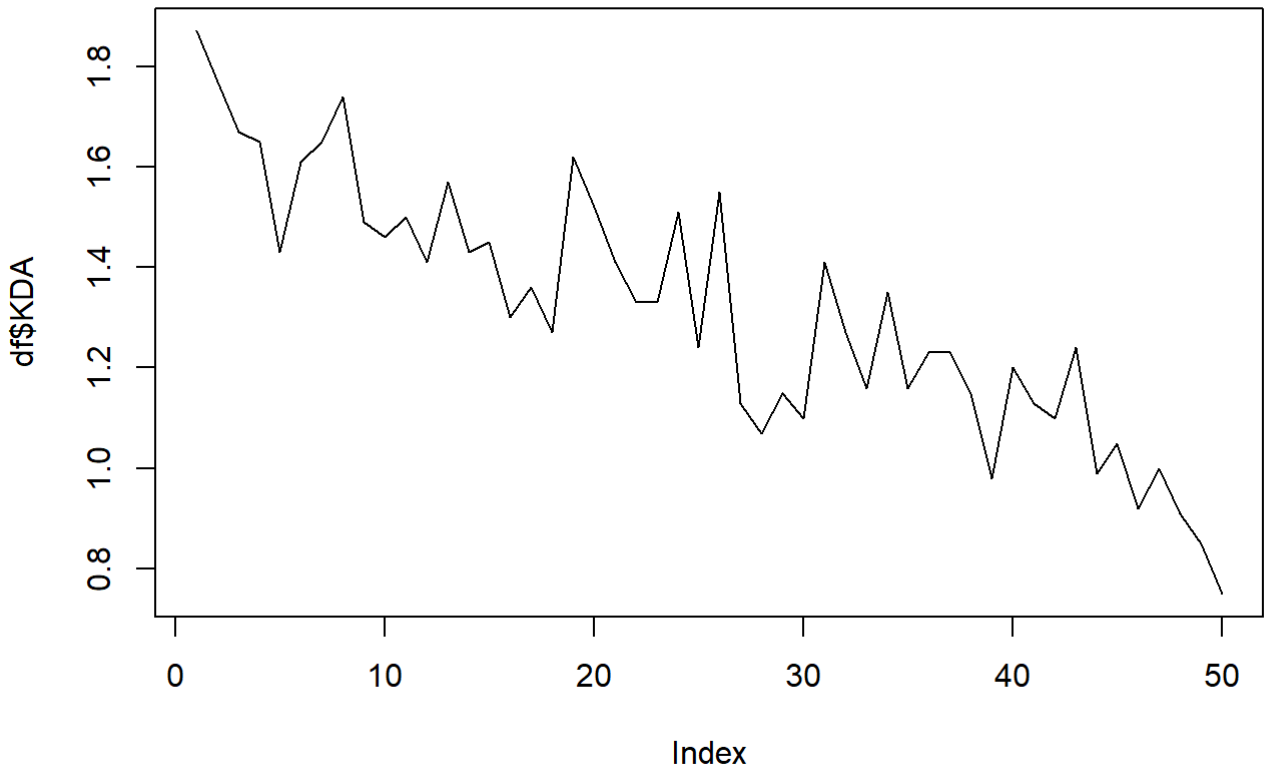

```
plot(df$A,type = "l")
```



```
plot(df$KD,type = "l")
```

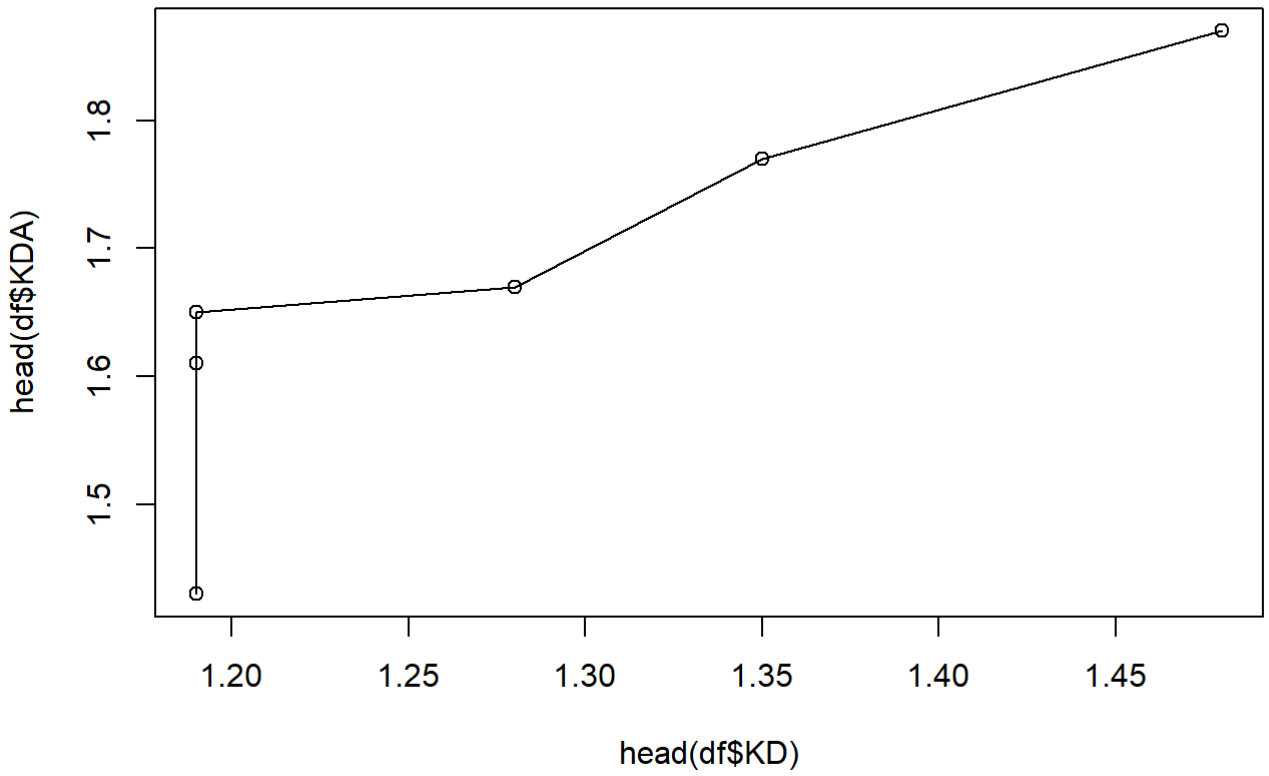



```
plot(df$KDA,type = "l")
```



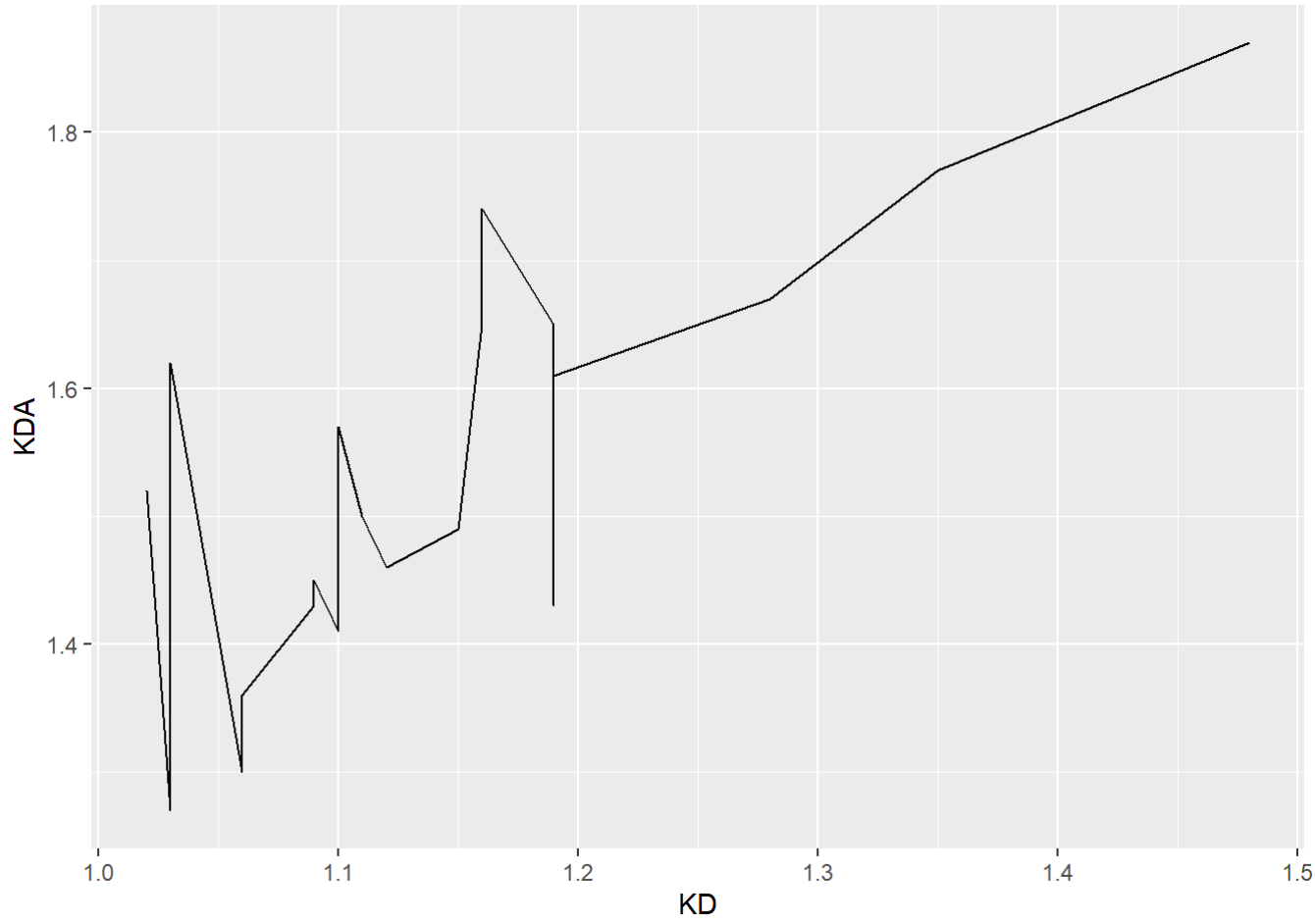
Line graph with base graphics

```
plot(head(df$KD),head(df$KDA),  type = "l")
points(head(df$KD),head(df$KDA))
```

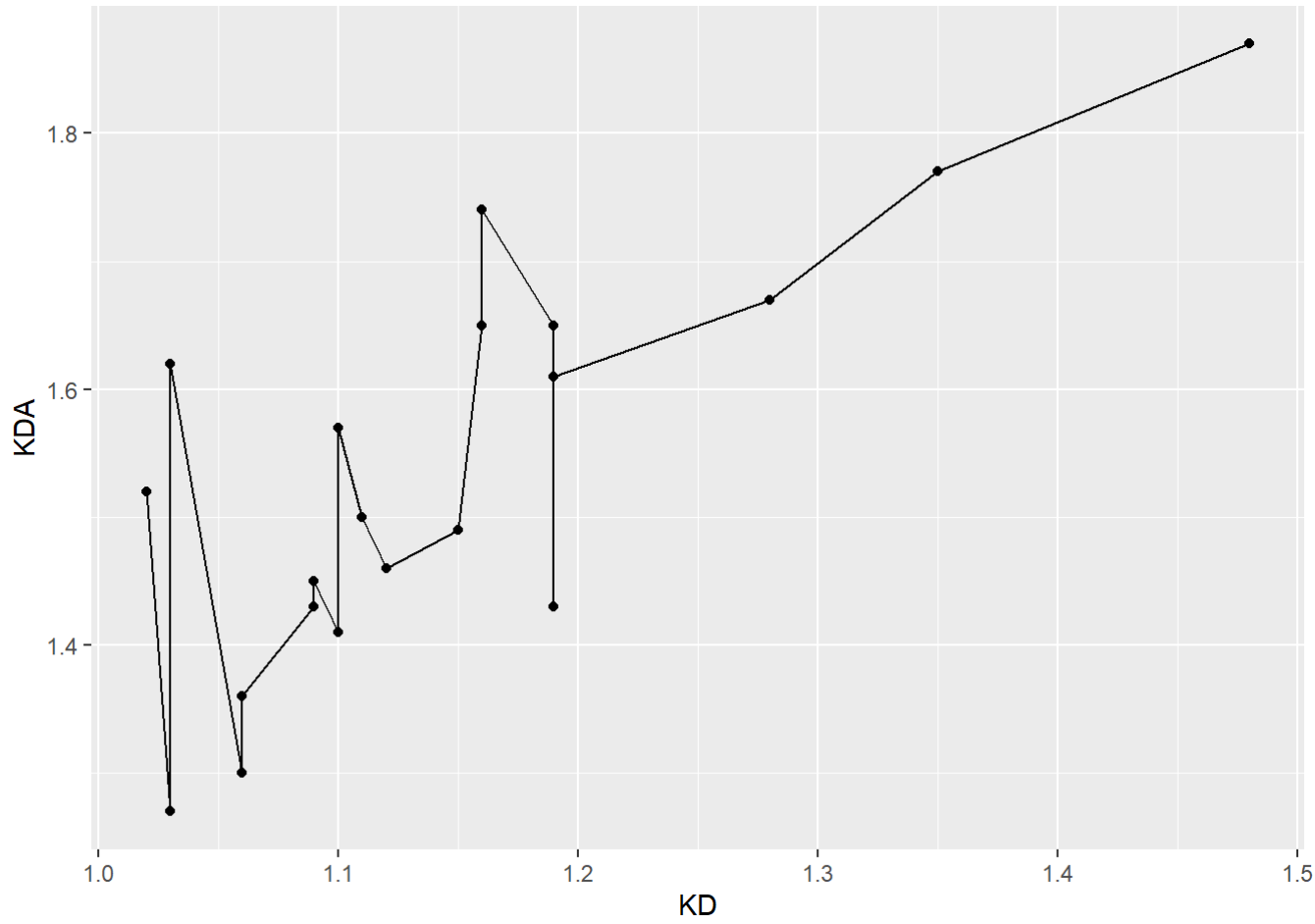


Line graph with ggplot() and With points added to ggplot()

```
ggplot(head(df,n=20), aes(x = KD, y = KDA)) +geom_line()
```

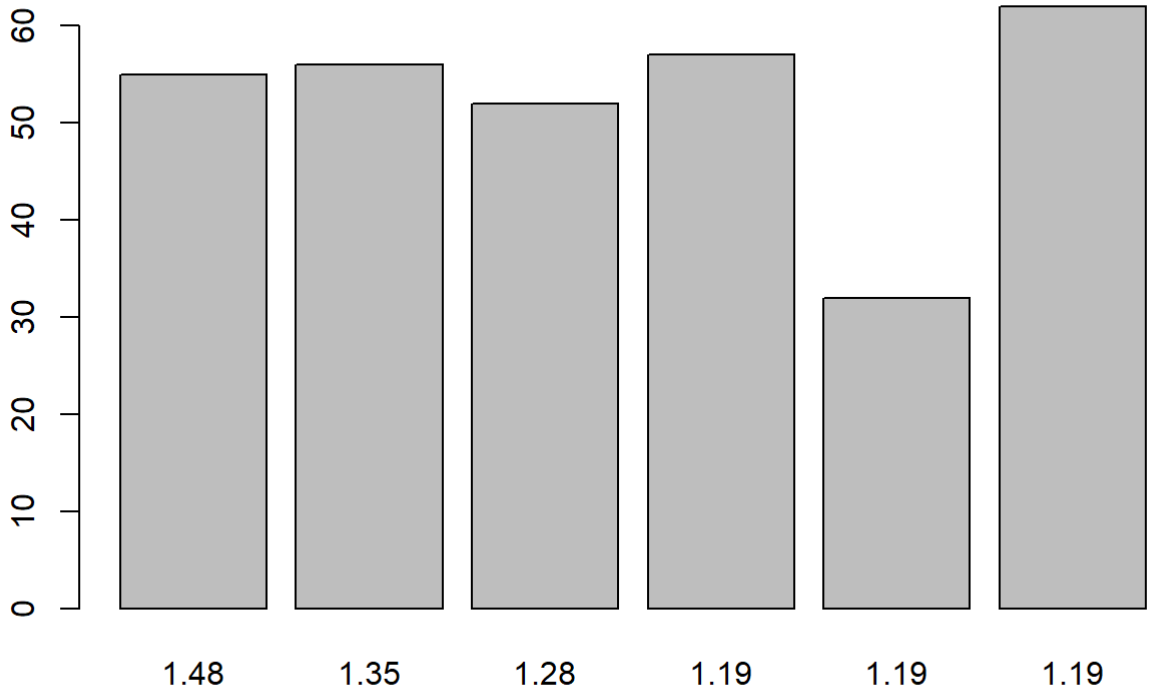



```
ggplot(head(df,n=20), aes(x = KD, y = KDA)) +geom_line()+geom_point()
```



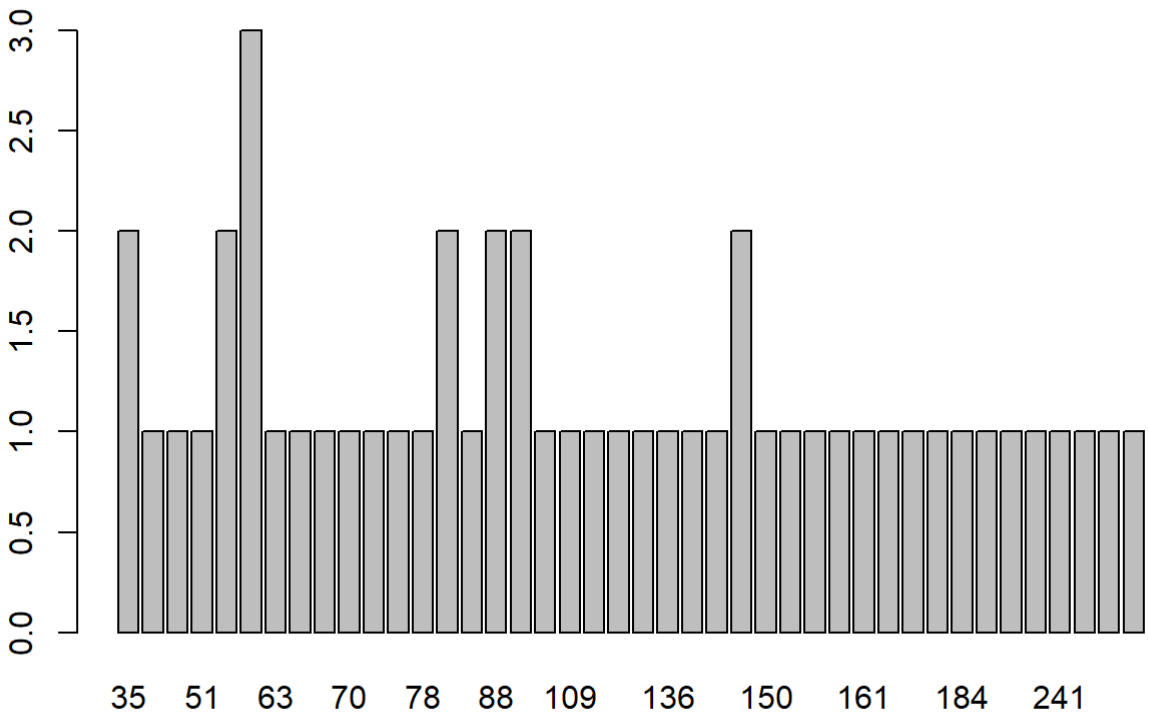
Creating a Bar Graph

```
barplot(head(df$A), names.arg = head(df$KD))
```

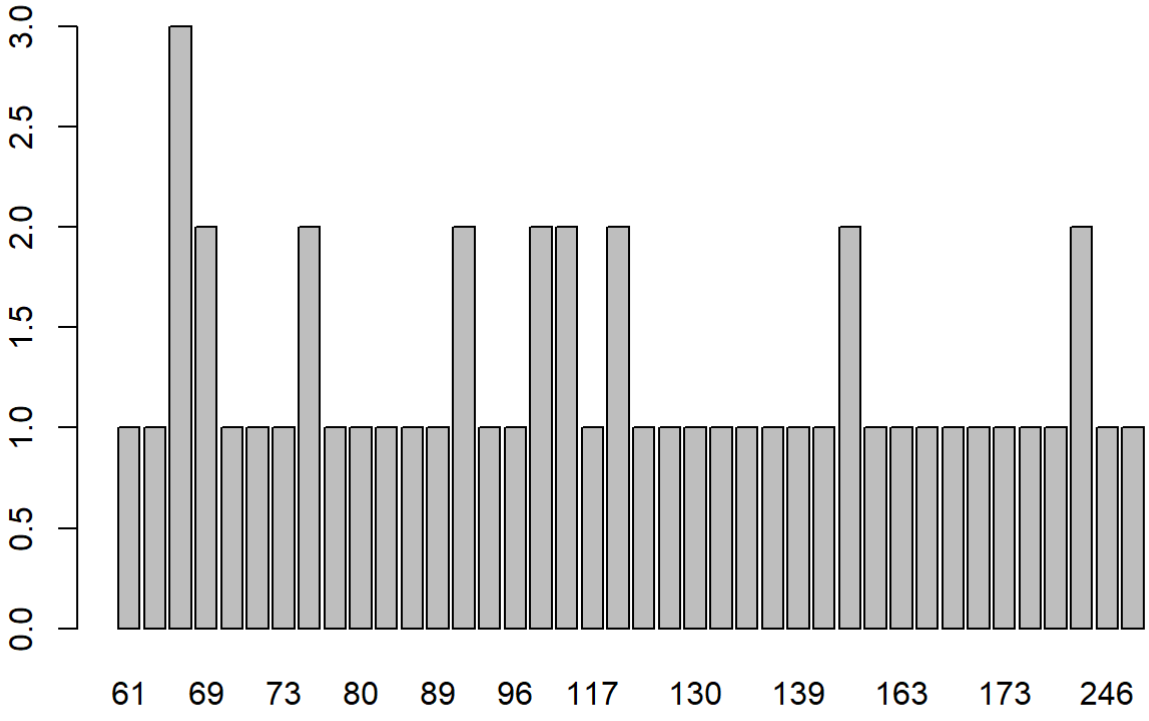


Generate a table of counts

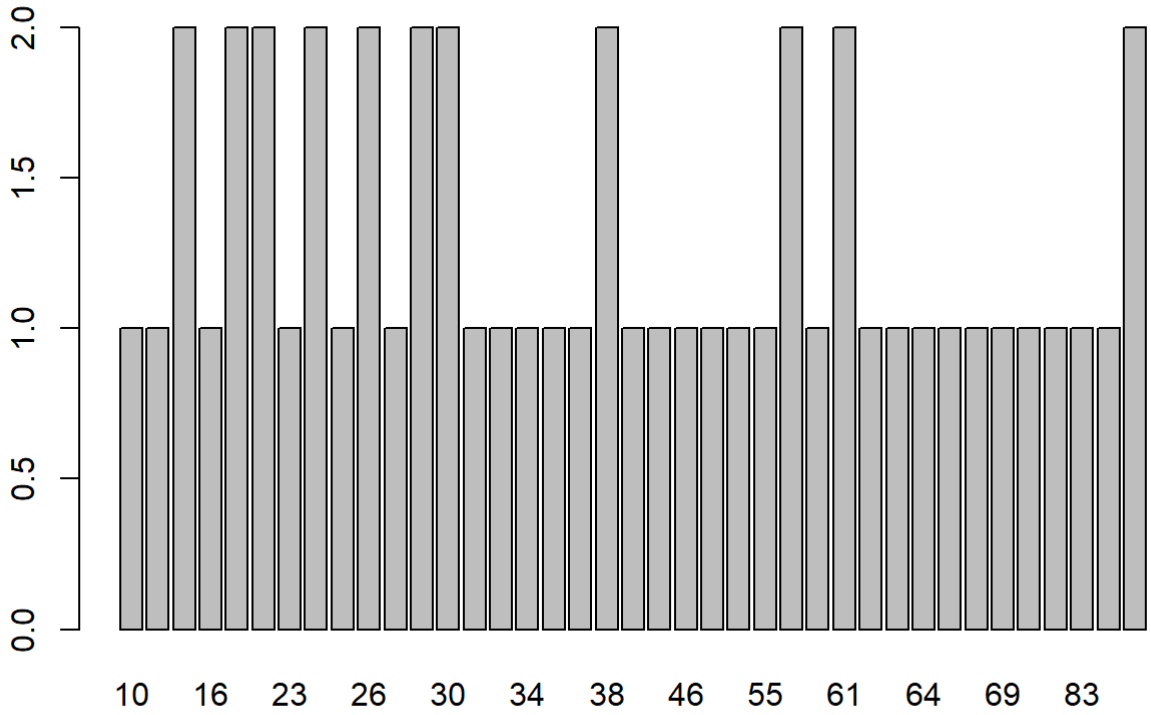
```
barplot(table(df$K))
```



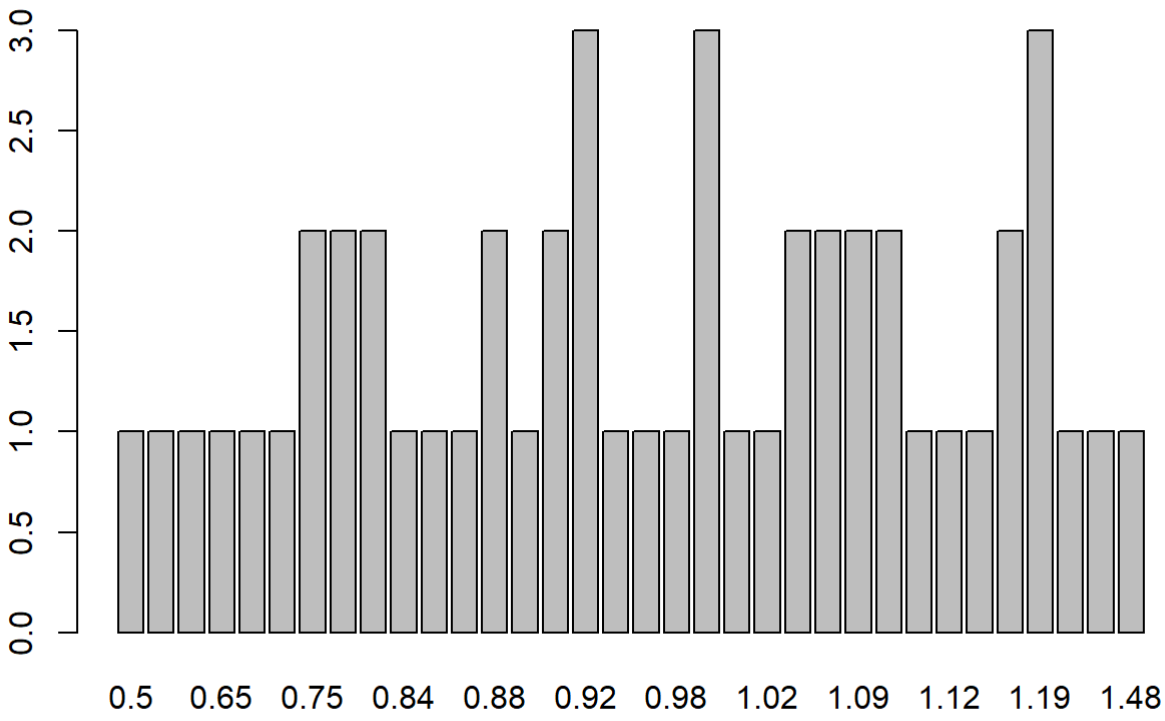

```
barplot(table(df$D))
```



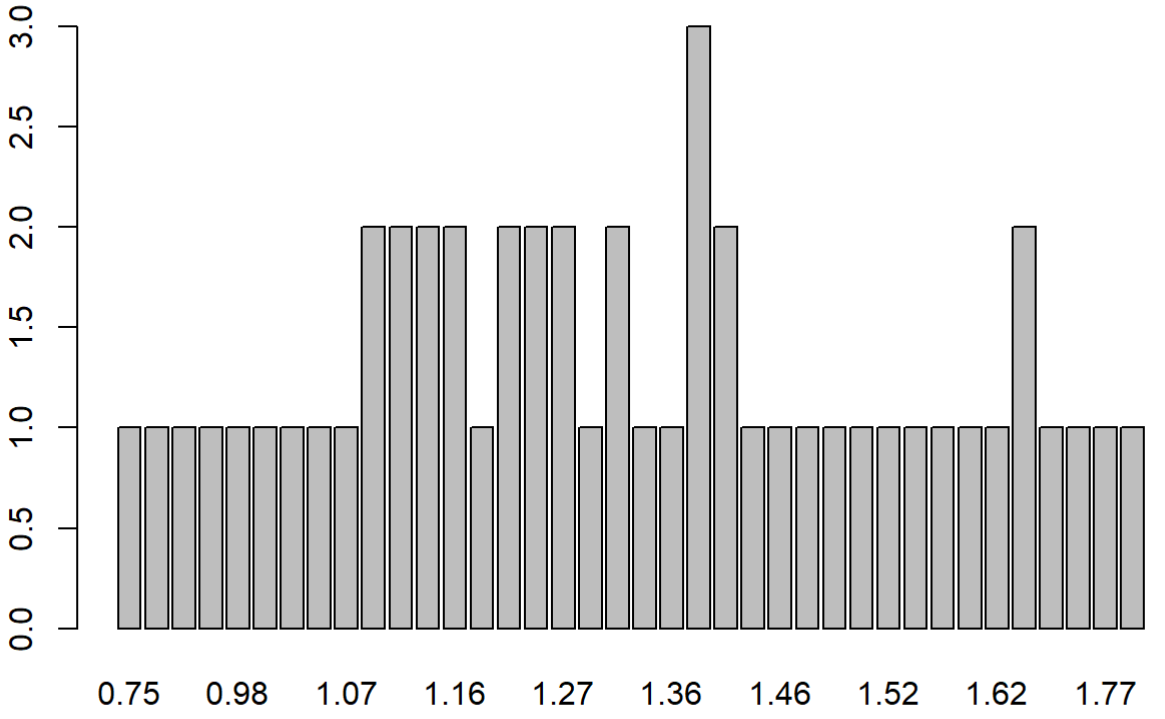
```
barplot(table(df$A))
```




```
barplot(table(df$KD))
```




```
barplot(table(df$KDA))
```



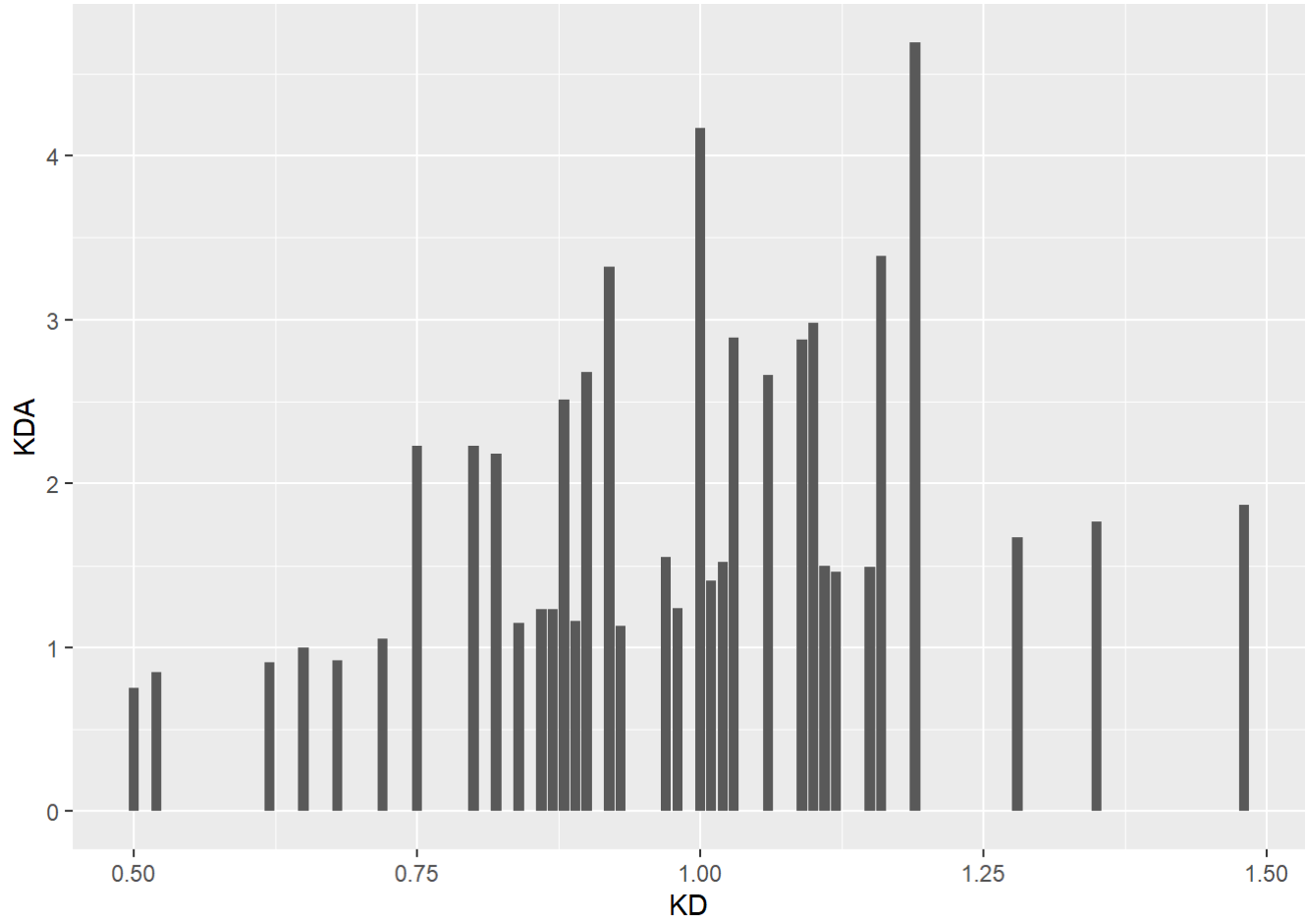
Laoding ggplot2() package

```
library(ggplot2)
```

Bar graph of values. This uses the dataset data frame, with the

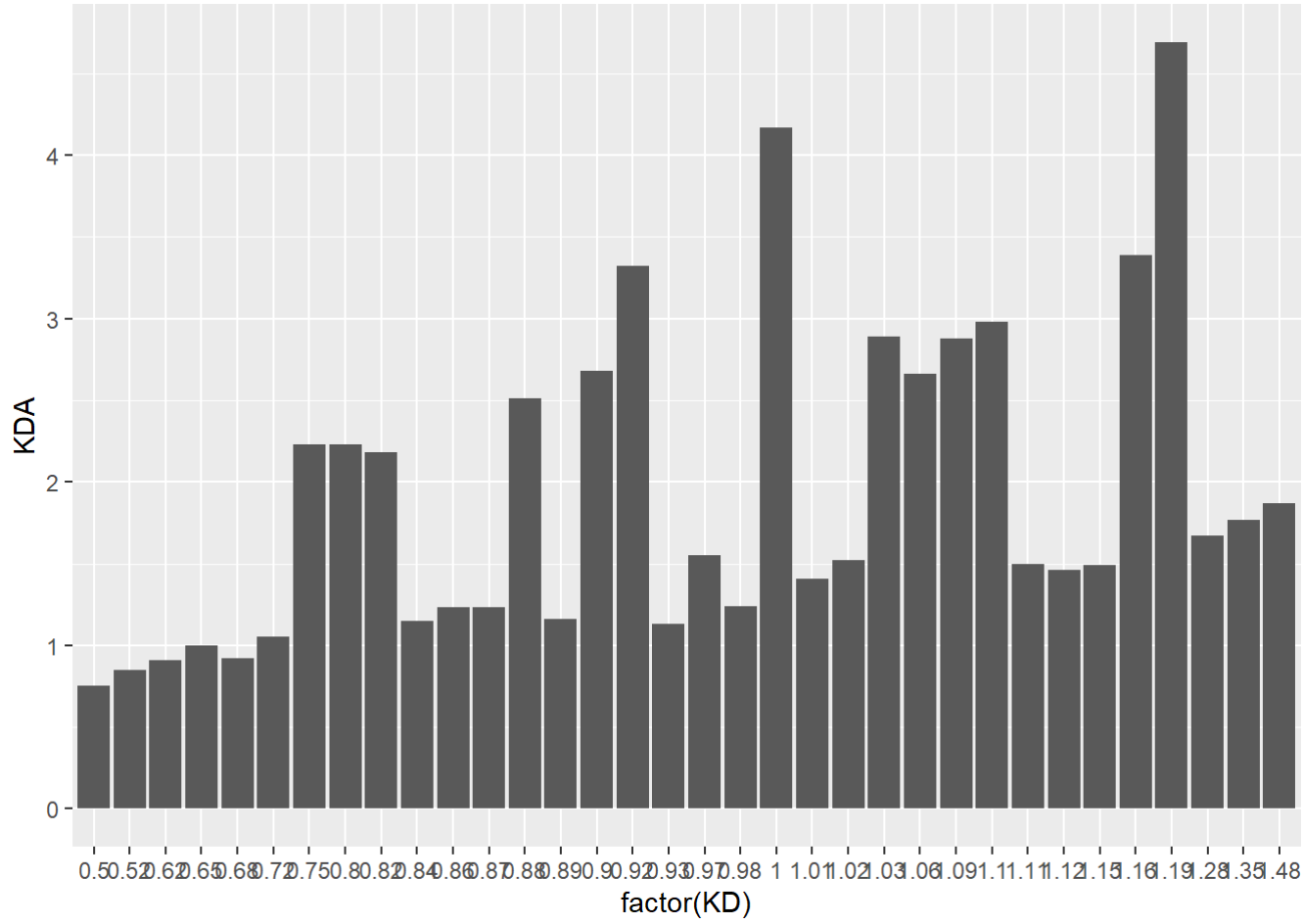
“KD” column for x values and the “KDA” column for y values.

```
ggplot(df, aes(x = KD, y = KDA)) +geom_col()
```



Convert the x variable to a factor, so that it is treated as discrete

```
ggplot(df, aes(x = factor(KD), y = KDA)) +geom_col()
```

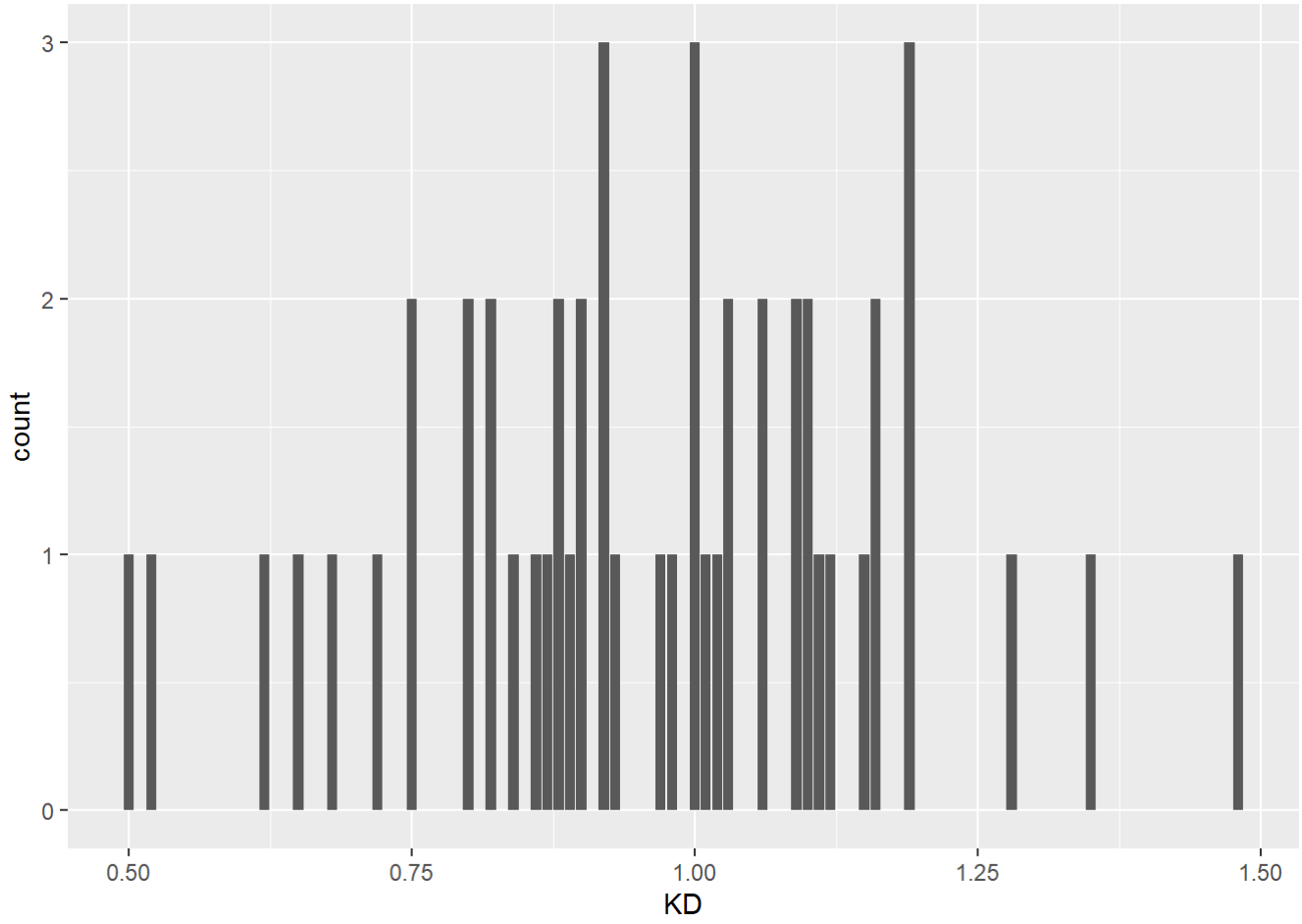


Bar graph of counts. This uses the dataset data frame, with the “KD” column for

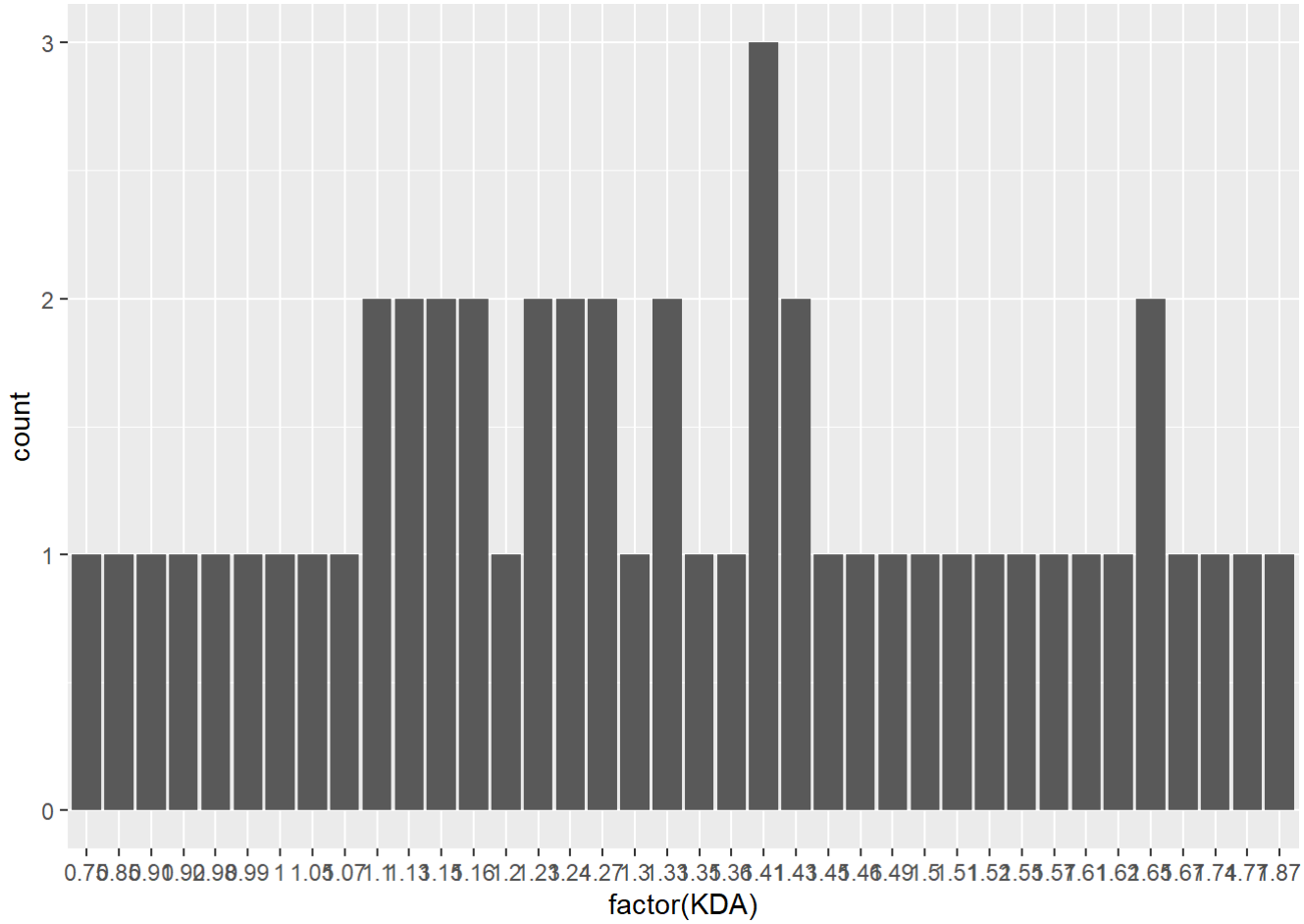
x position. The y position is calculated by counting the number of rows for

each value of pretest.

```
ggplot(df, aes(x = KD)) +
geom_bar()
```

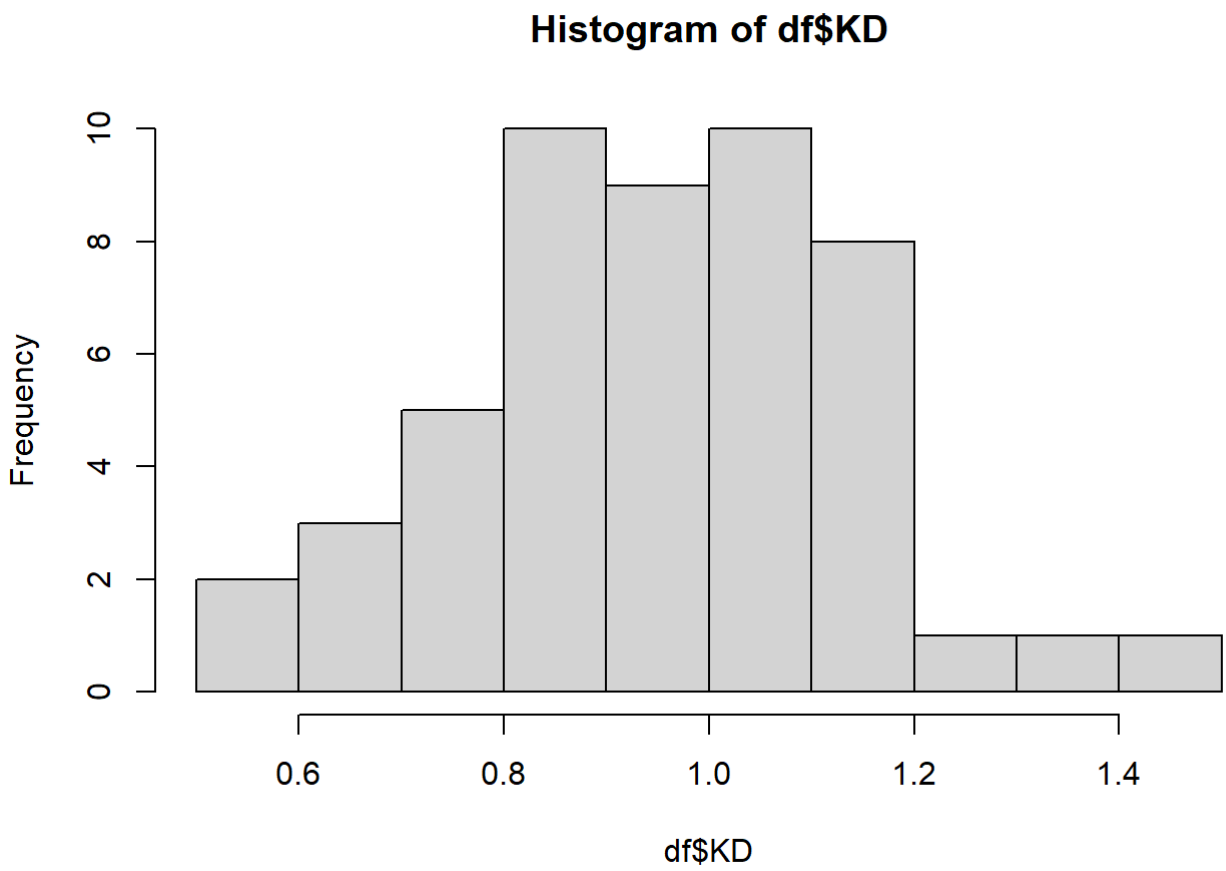



```
# Bar graph of counts
ggplot(df, aes(x = factor(KDA))) +
geom_bar()
```

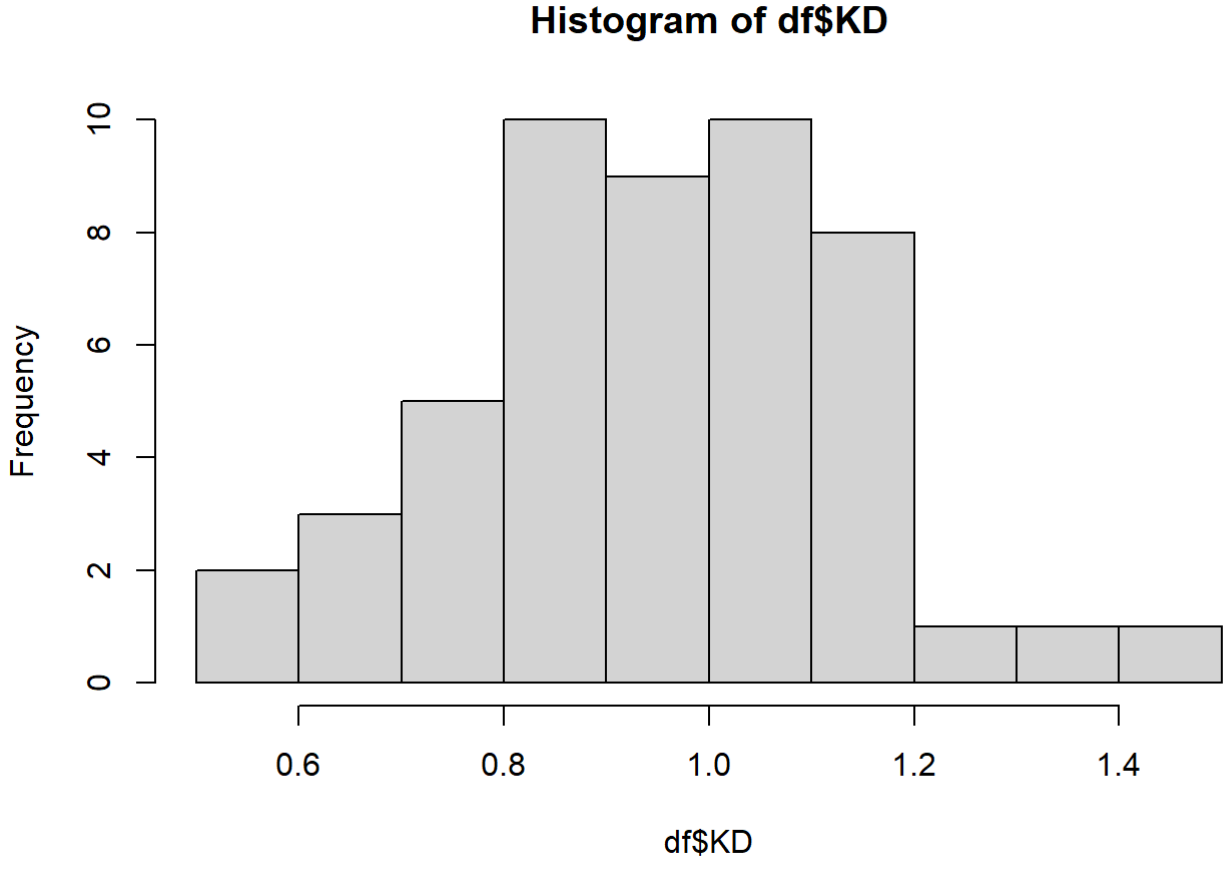


Creating a Histogram

```
hist(df$KD)
```



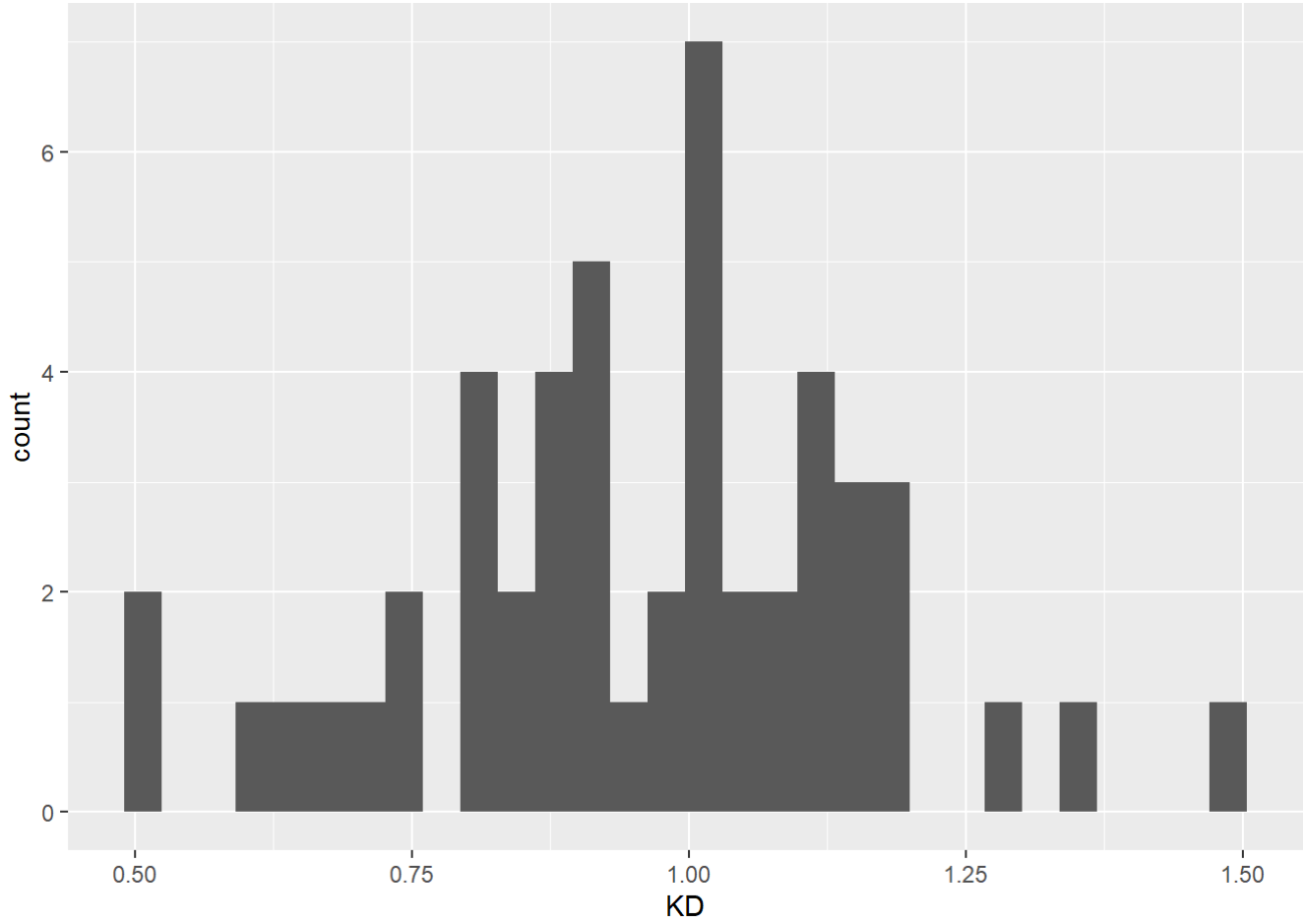

```
# Specify approximate number of bins with breaks
hist(df$KD, breaks = 10)
```



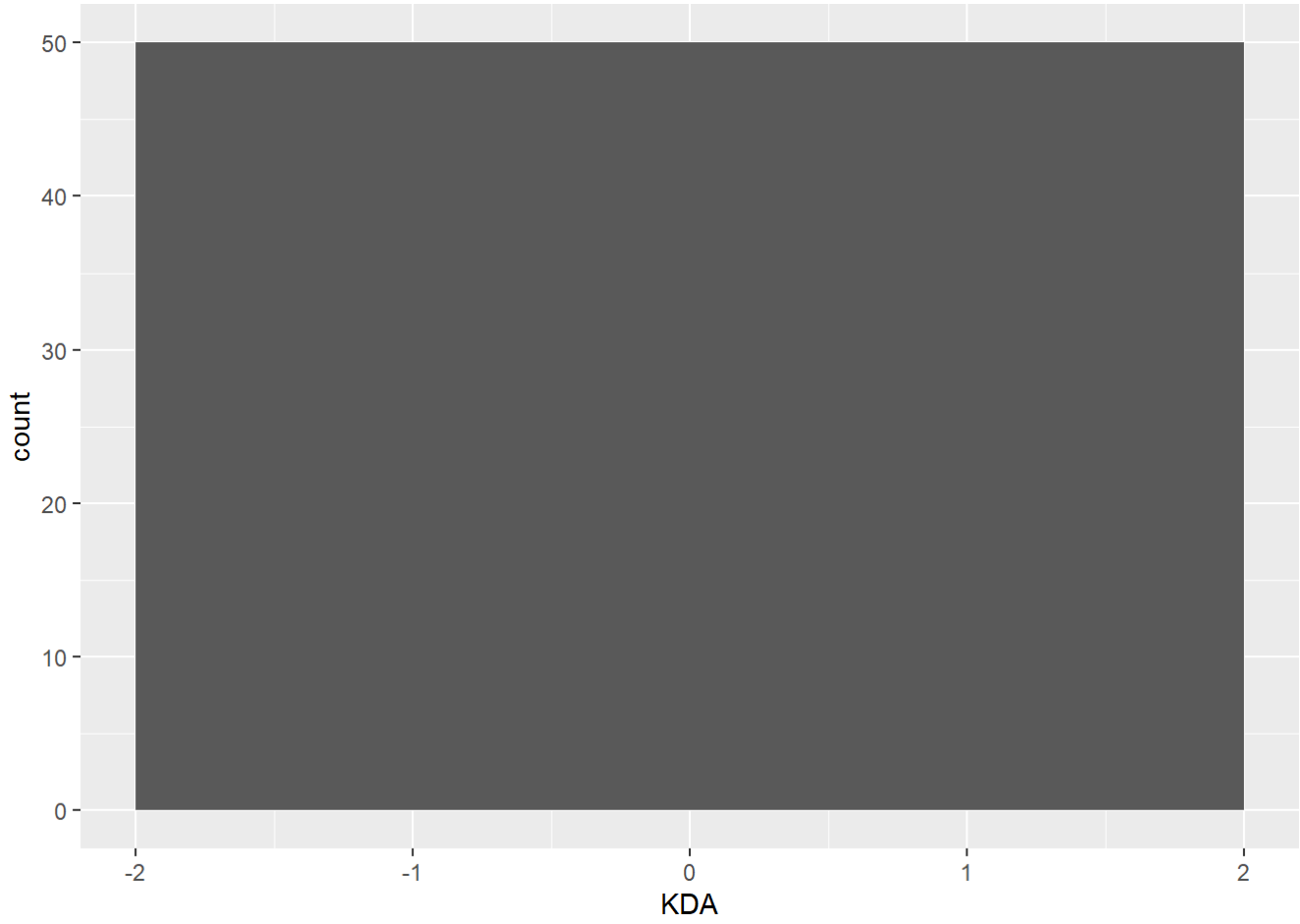
#ggplot2 histogram with default bin width (left); With wider bins (right)

```
ggplot(df, aes(x = KD)) +
geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

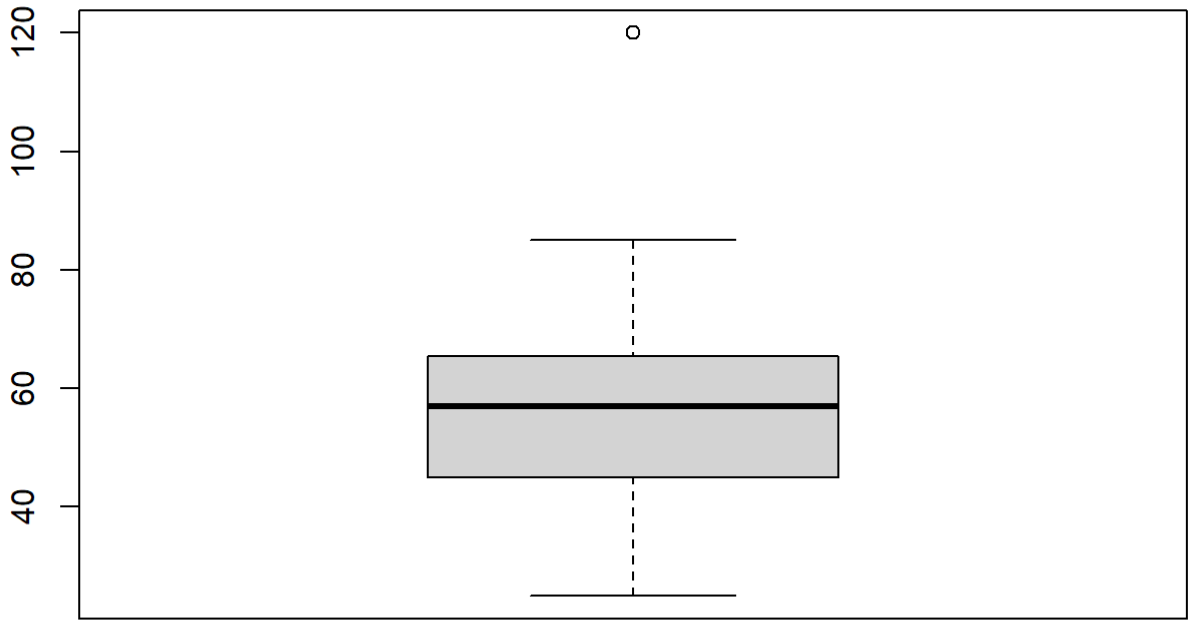



```
ggplot(df, aes(x = KDA)) +
geom_histogram(binwidth = 4)
```



Creating a Box Plot

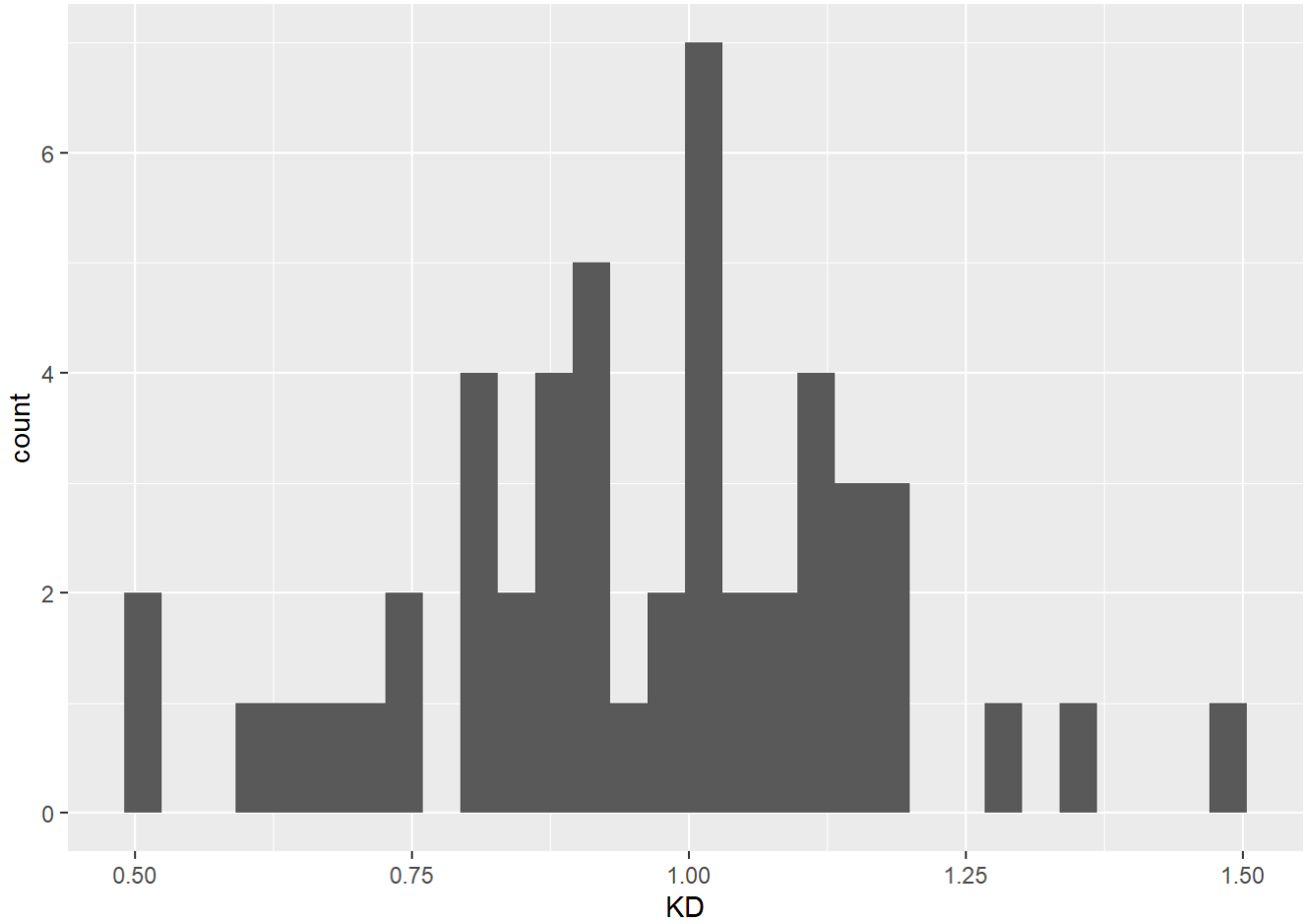
```
boxplot(head(df$A,n=15))
```



Making a Basic Histogram

```
ggplot(df, aes(x = KD)) +geom_histogram()
```

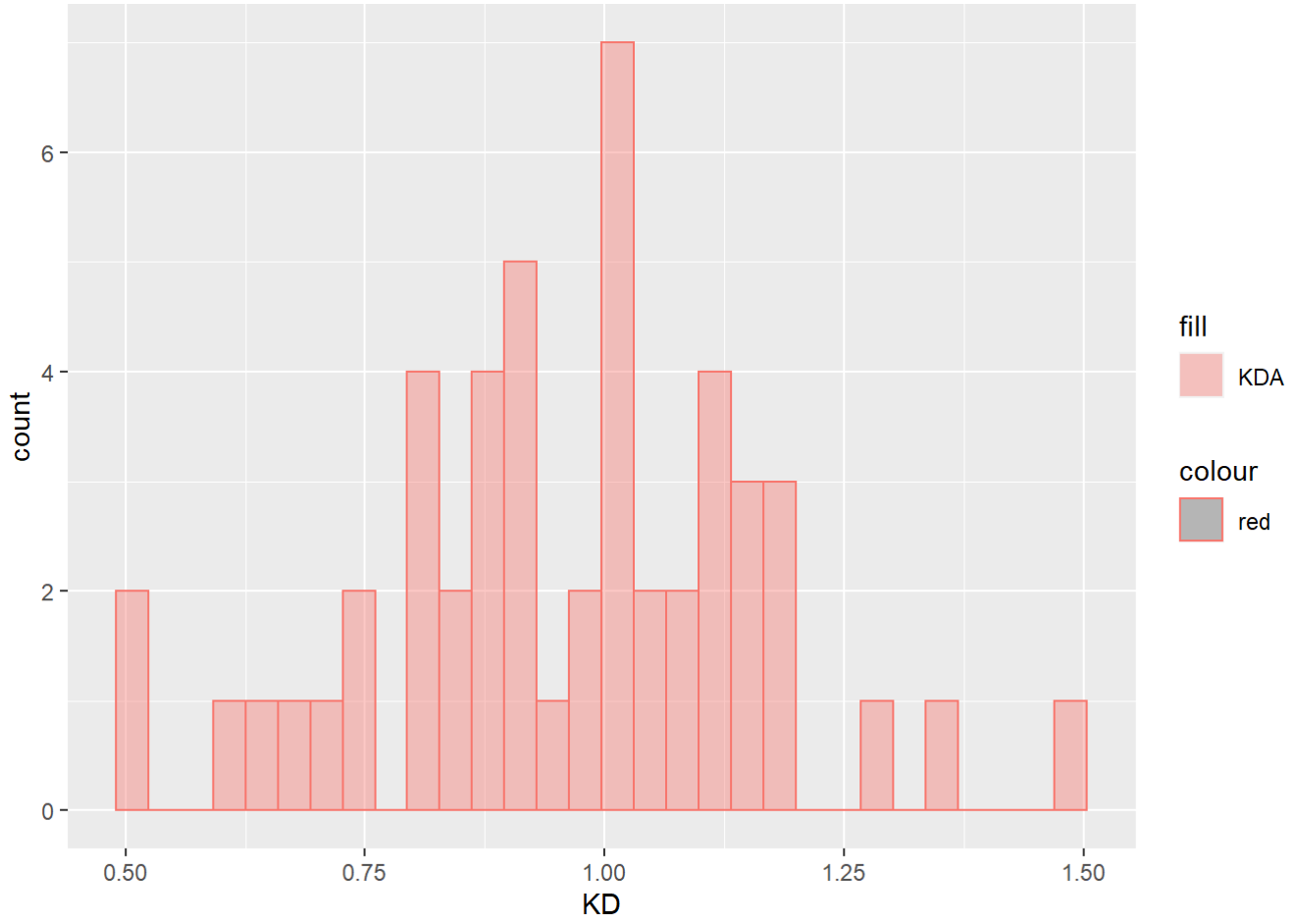
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Map pretest to fill, make the bars NOT stacked, and make them semitransparent

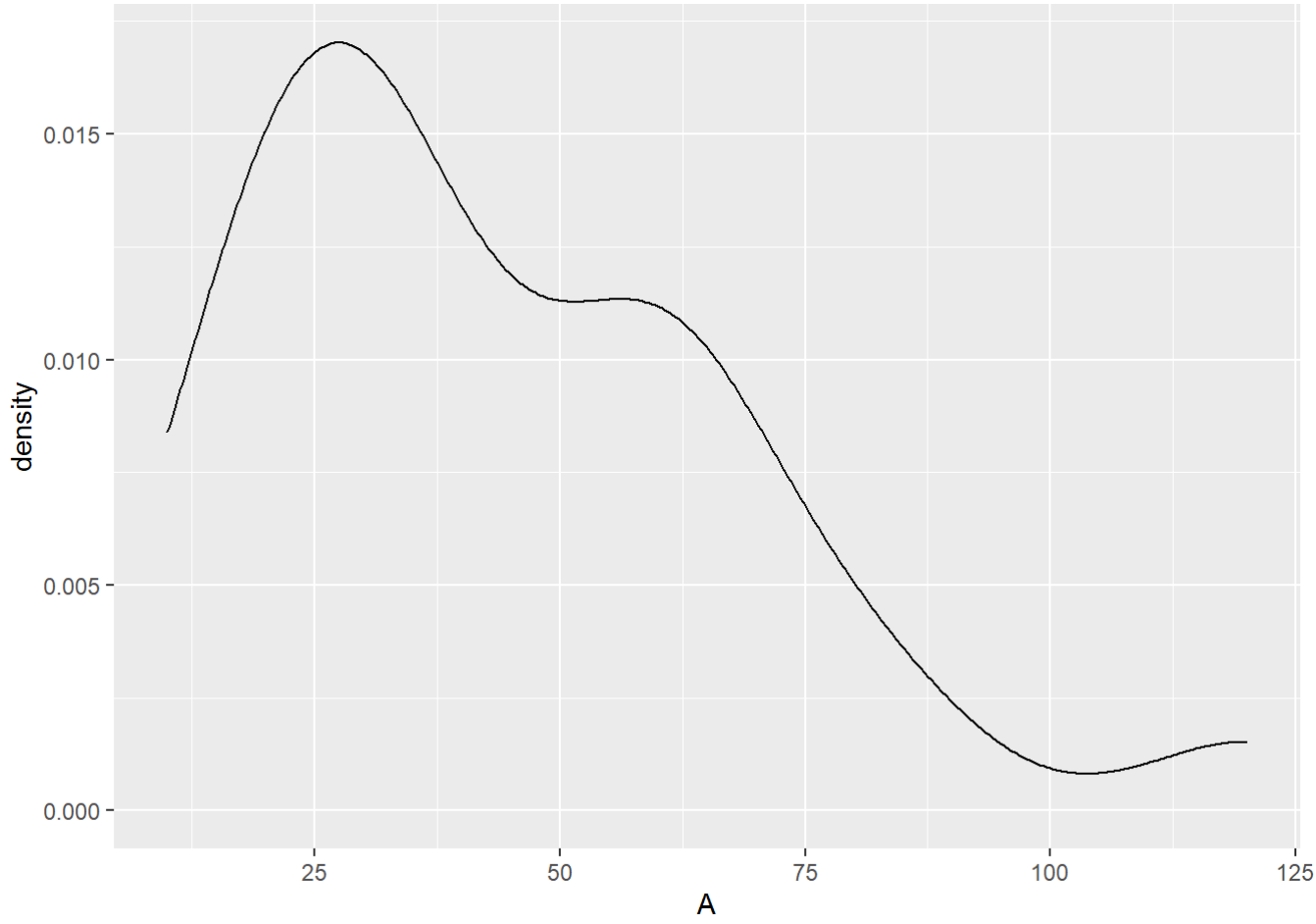
```
ggplot(df, aes(x = KD, fill ='KDA',colour='red' )) +
geom_histogram(position = "identity", alpha = 0.4)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

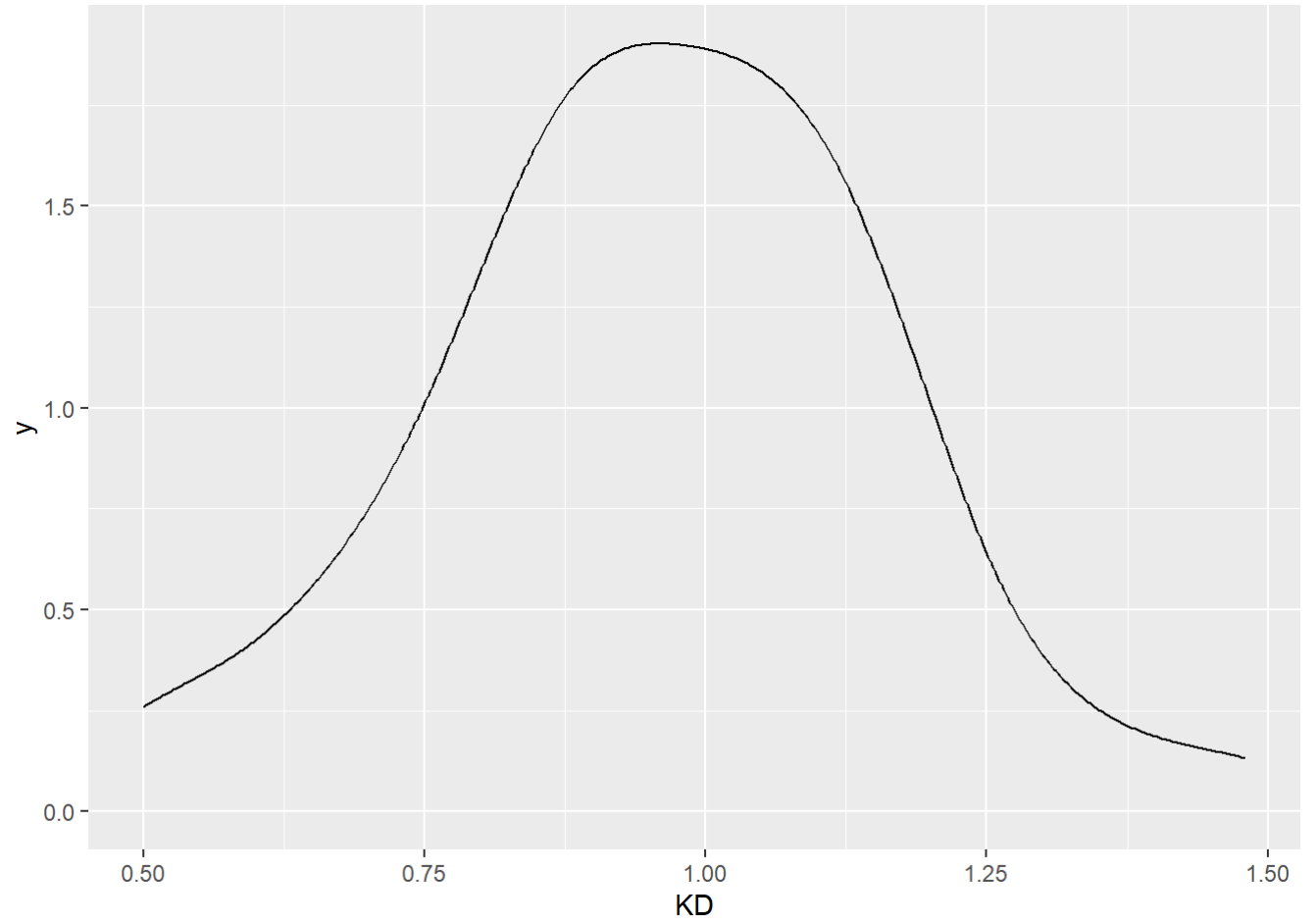


Making a Density Curve

```
ggplot(df, aes(x = A)) +geom_density()
```

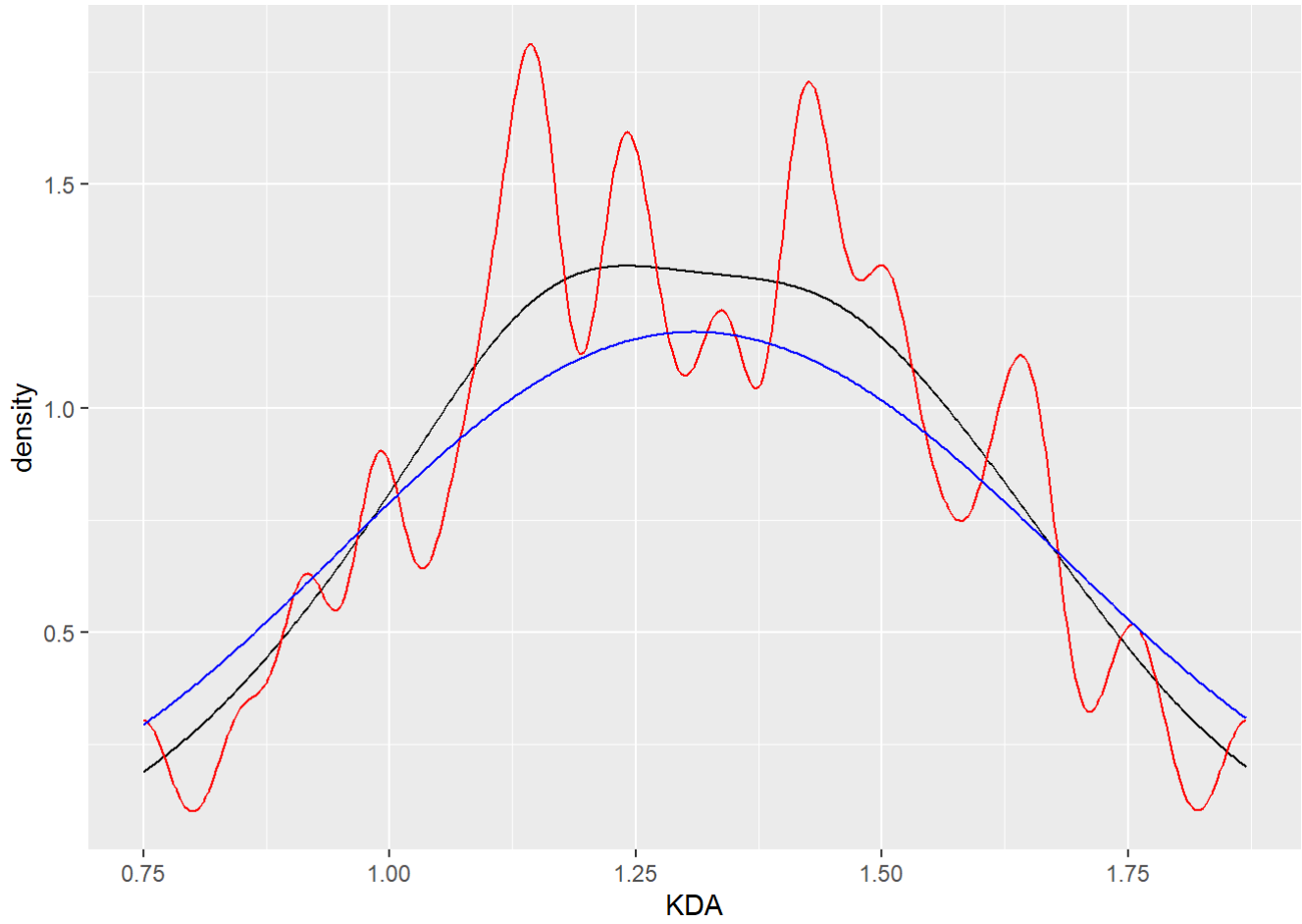



```
# expand_limits() increases the y range to include the value 0
ggplot(df, aes(x = KD)) +geom_line(stat = "density") +expand_limits(y = 0)
```



Density curve with a smaller and larger value of adjust:

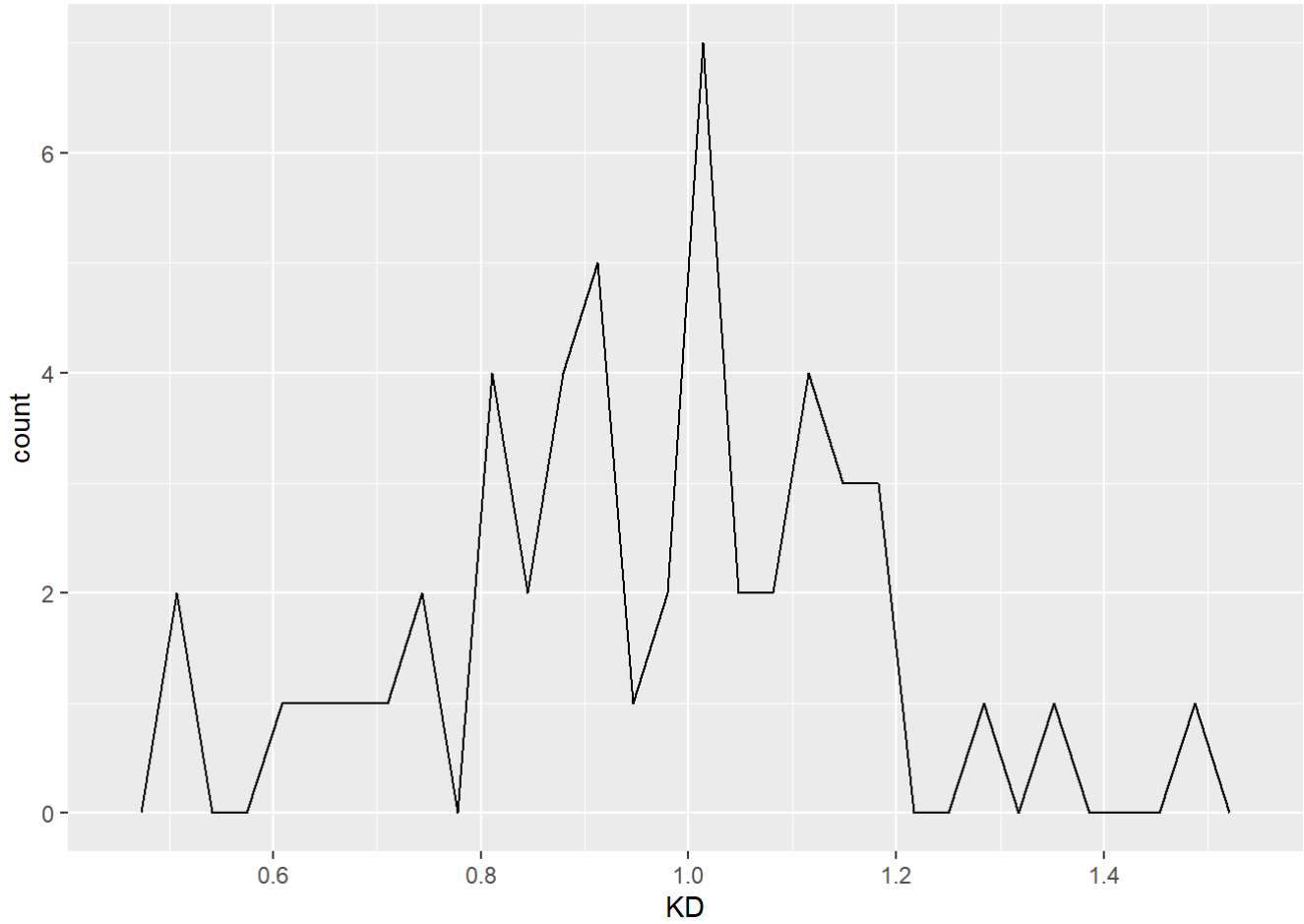
```
ggplot(df, aes(x = KDA)) +geom_line(stat = "density") +
geom_line(stat = "density", adjust = .25, colour = "red") +
geom_line(stat = "density", adjust = 2, colour = "blue")
```



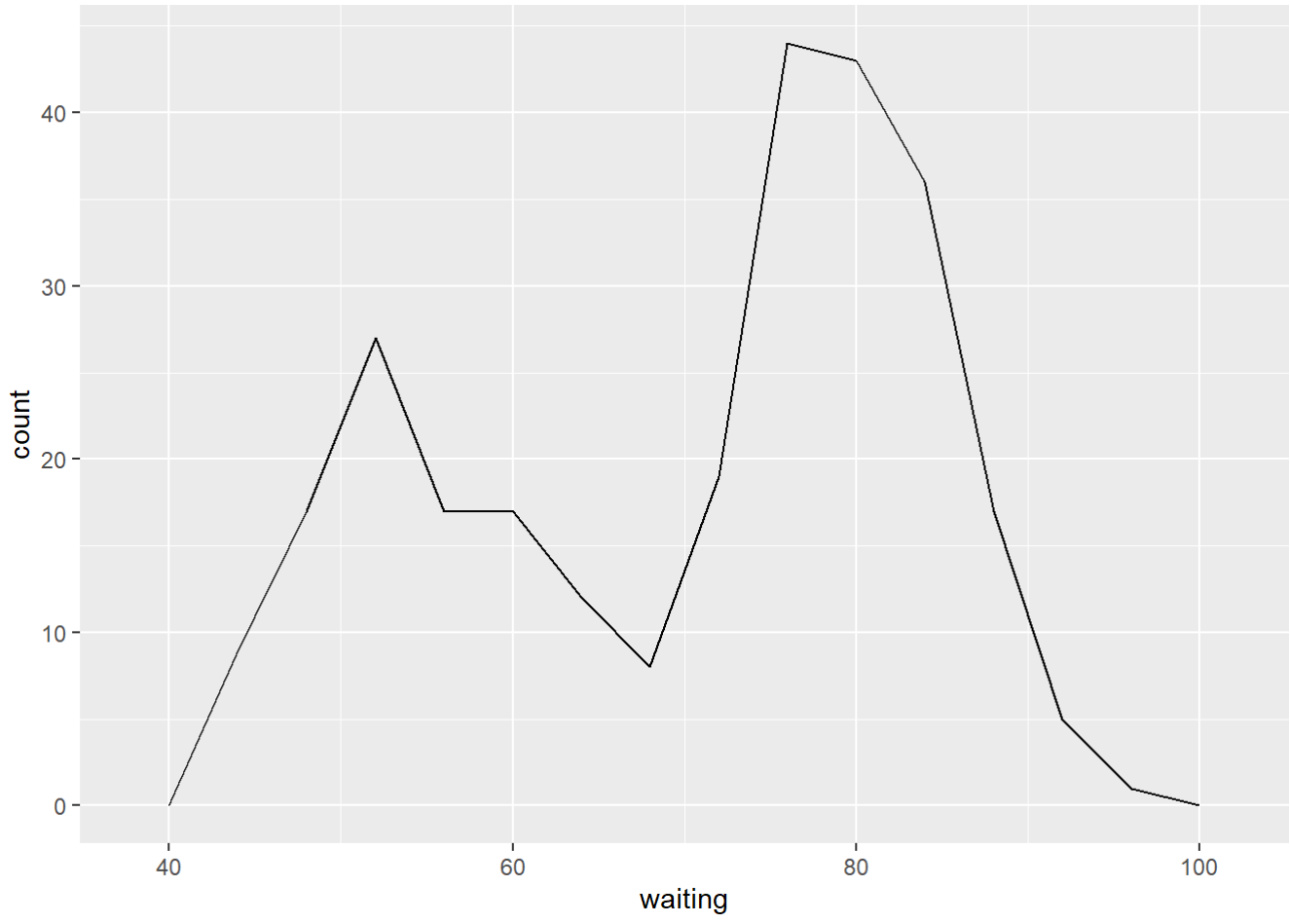
Making a Frequency Polygon

```
ggplot(df, aes(x=KD)) +
geom_freqpoly()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

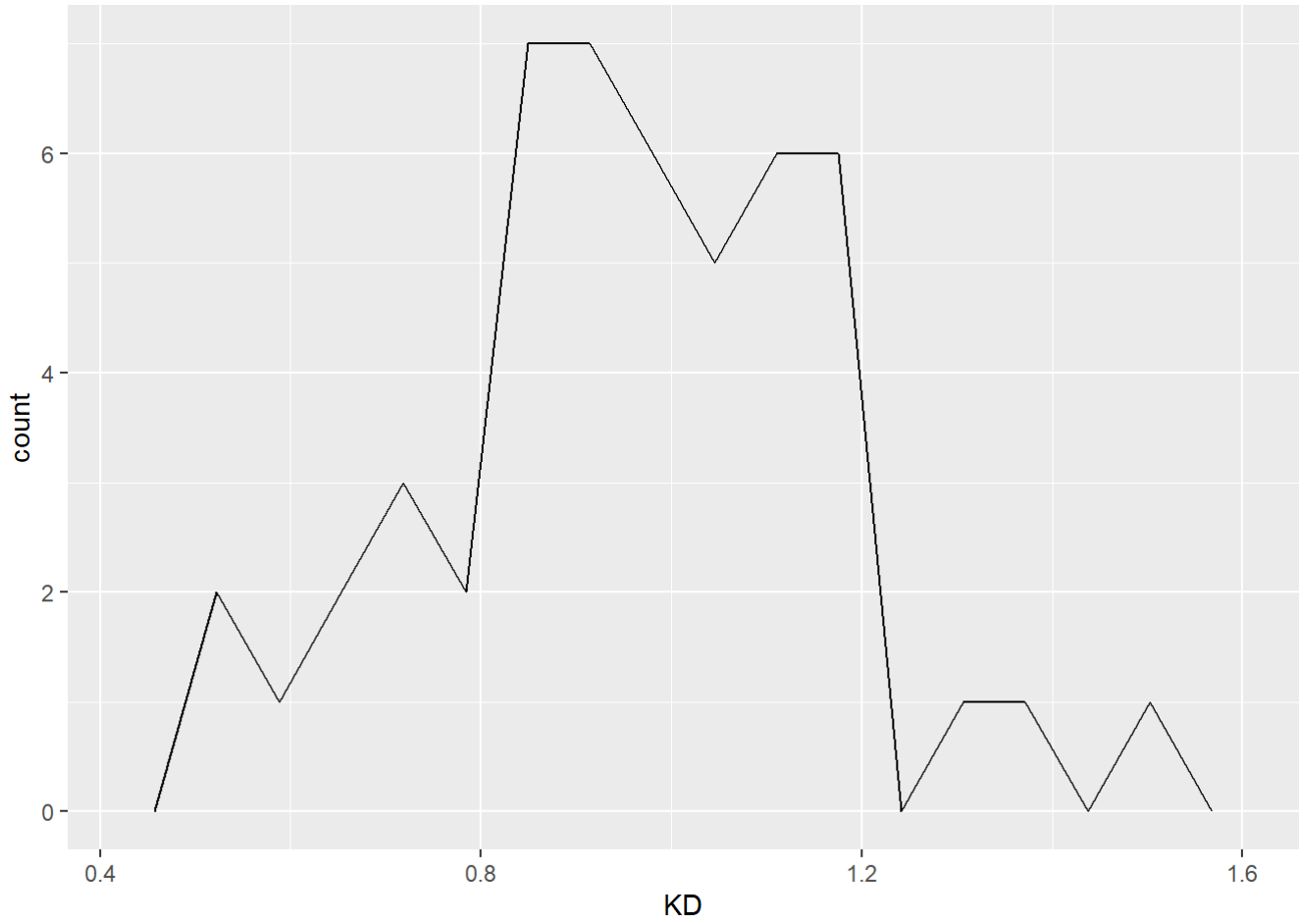



```
ggplot(faithful, aes(x = waiting)) +
geom_freqpoly(binwidth = 4)           #controlling bin width
```

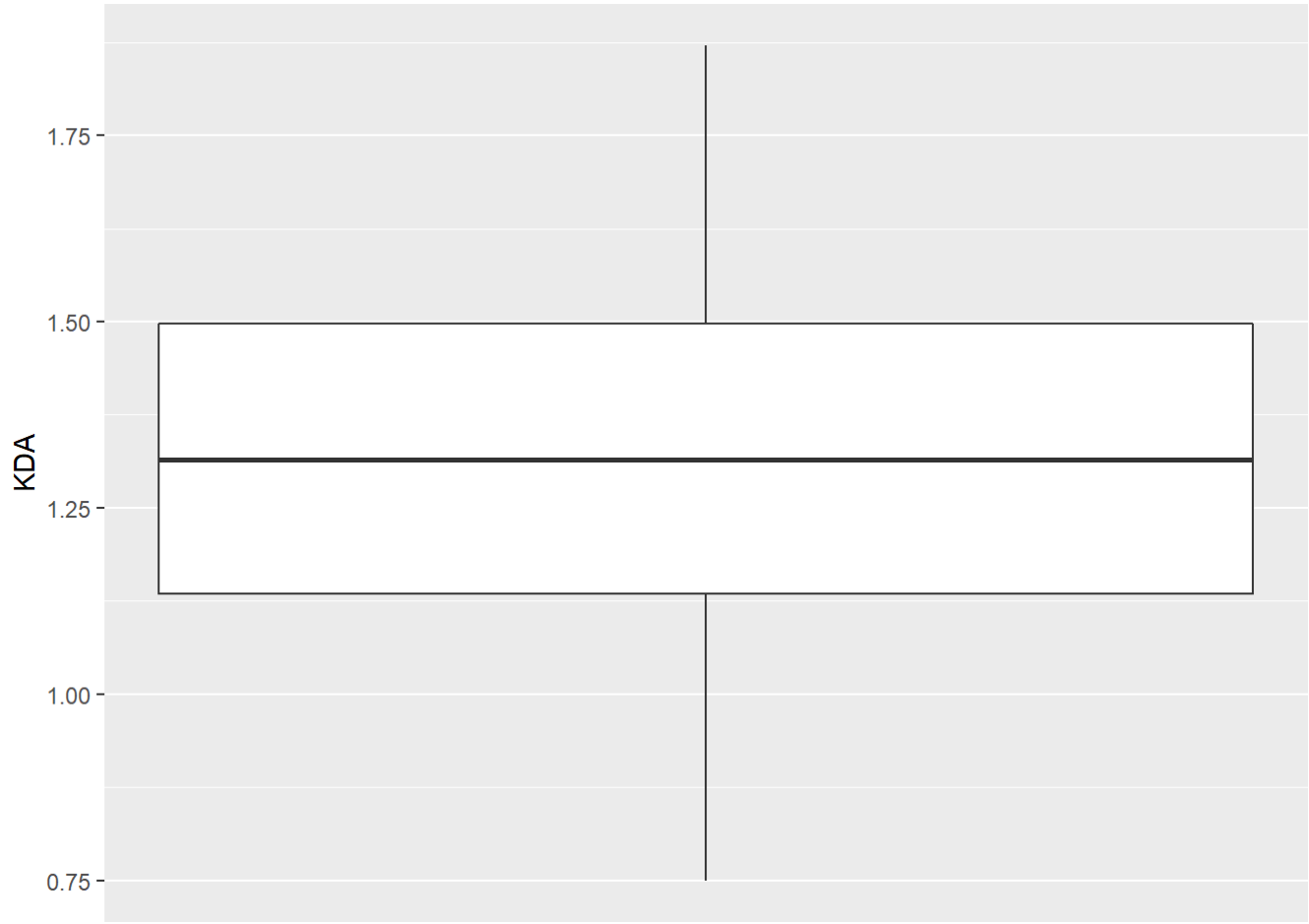


Divide the x-axis range into 15 bins

```
binsize <- diff(range(df$KD))/15
ggplot(df, aes(x = KD)) +
geom_freqpoly(binwidth = binsize)
```



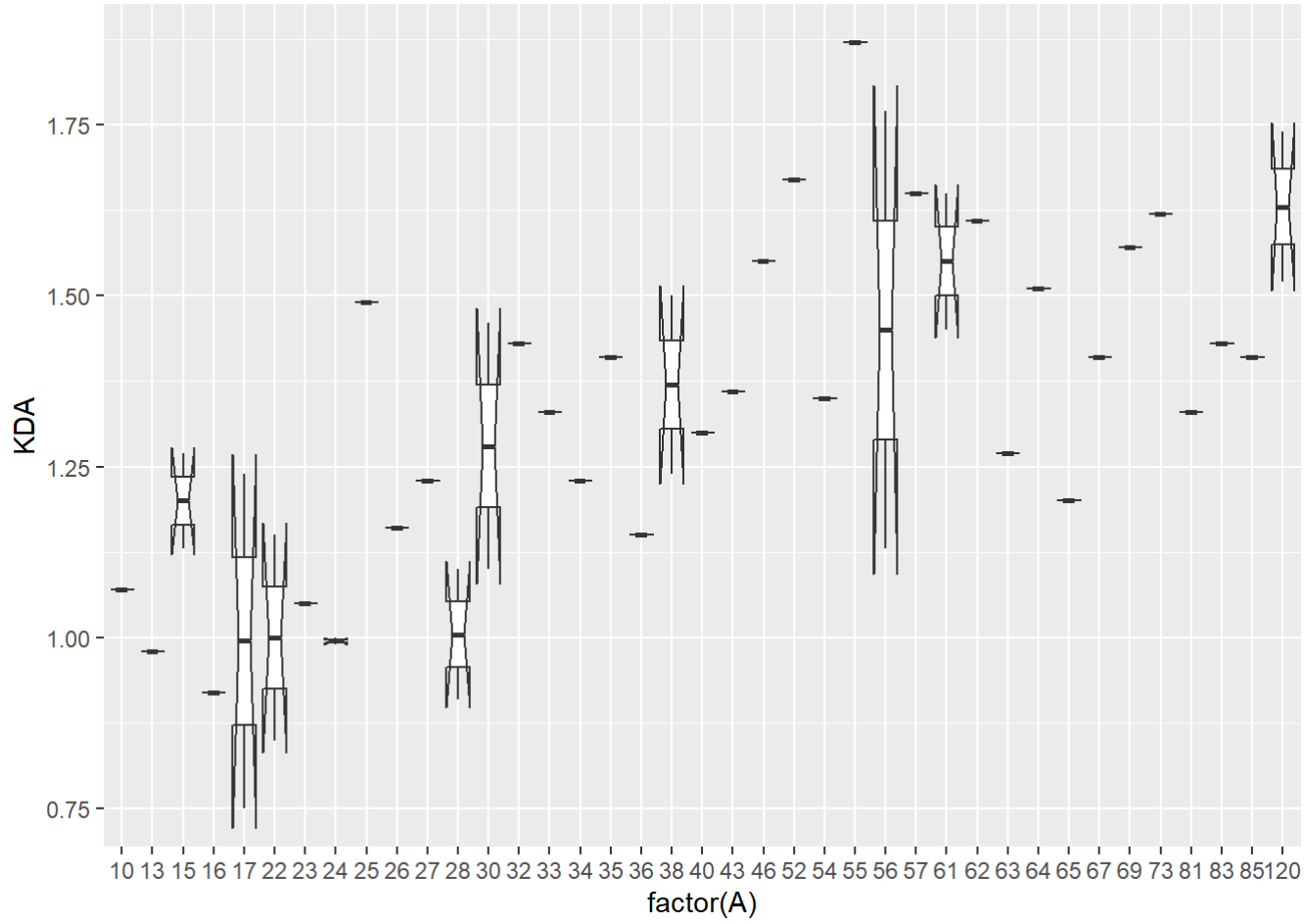

```
ggplot(df, aes(x = 1, y = KDA)) +geom_boxplot() +scale_x_continuous(breaks = NULL) +theme(axi
s.title.x = element_blank())
```



Adding notches to a box plot to assess whether the medians are different.

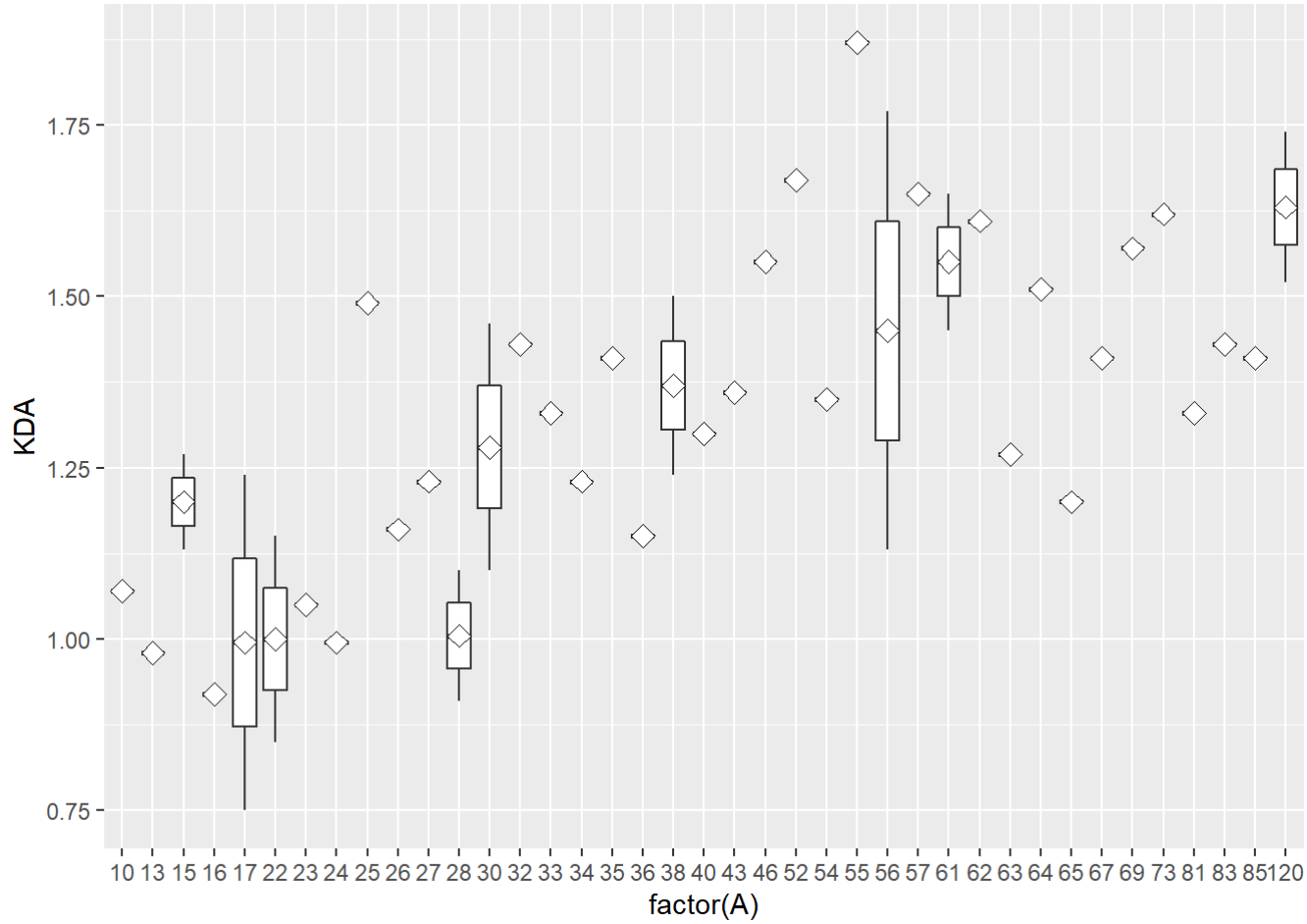
```
ggplot(df, aes(x = factor(A), y = KDA)) +geom_boxplot(notch = TRUE)
```

```
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
```



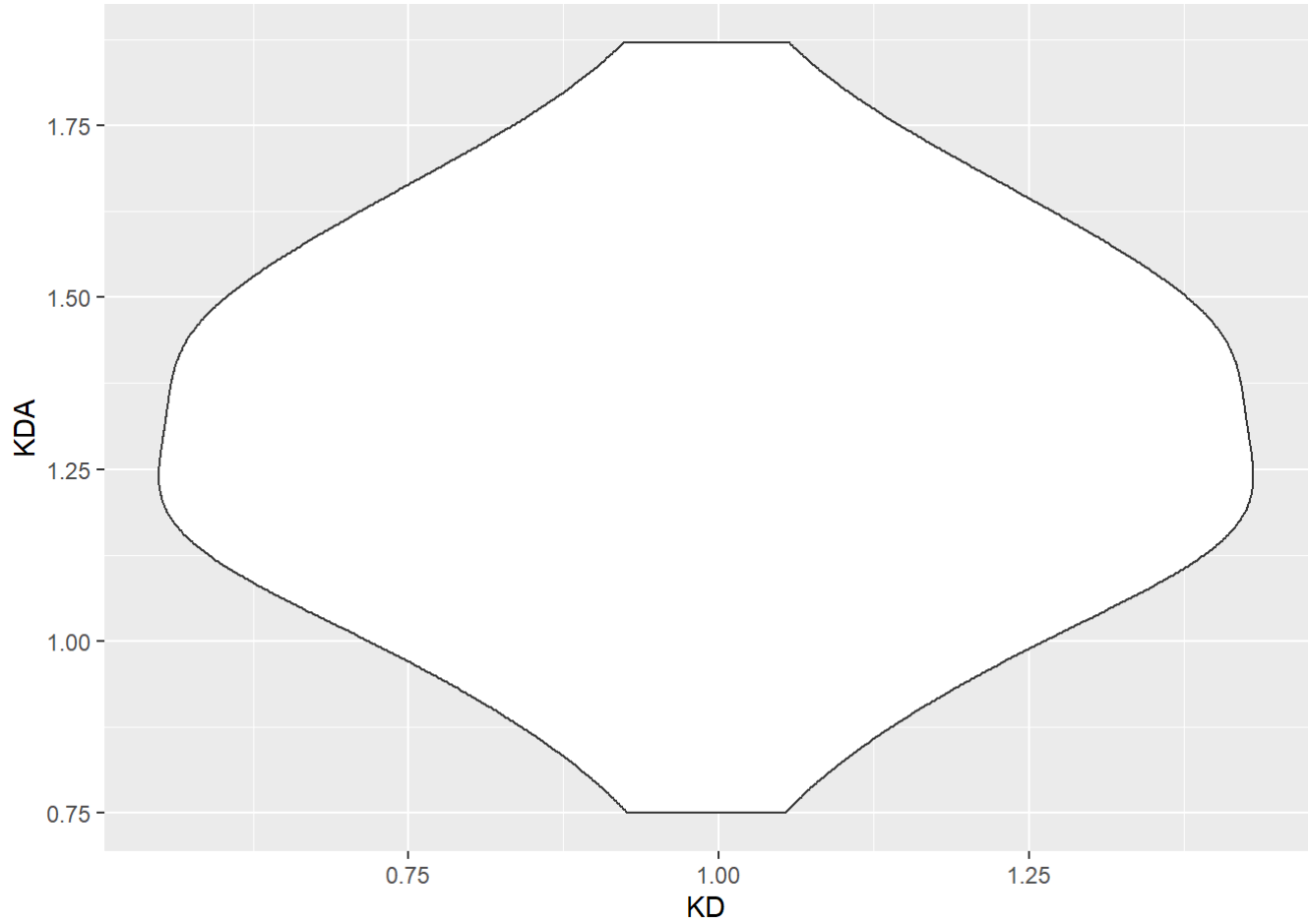
Adding means to box plot

```
ggplot(df, aes(x = factor(A), y = KDA)) +geom_boxplot() +stat_summary(fun = "mean", geom = "p
oint", shape = 23, size = 3,
fill = "white")
```



Making a Violin Plot

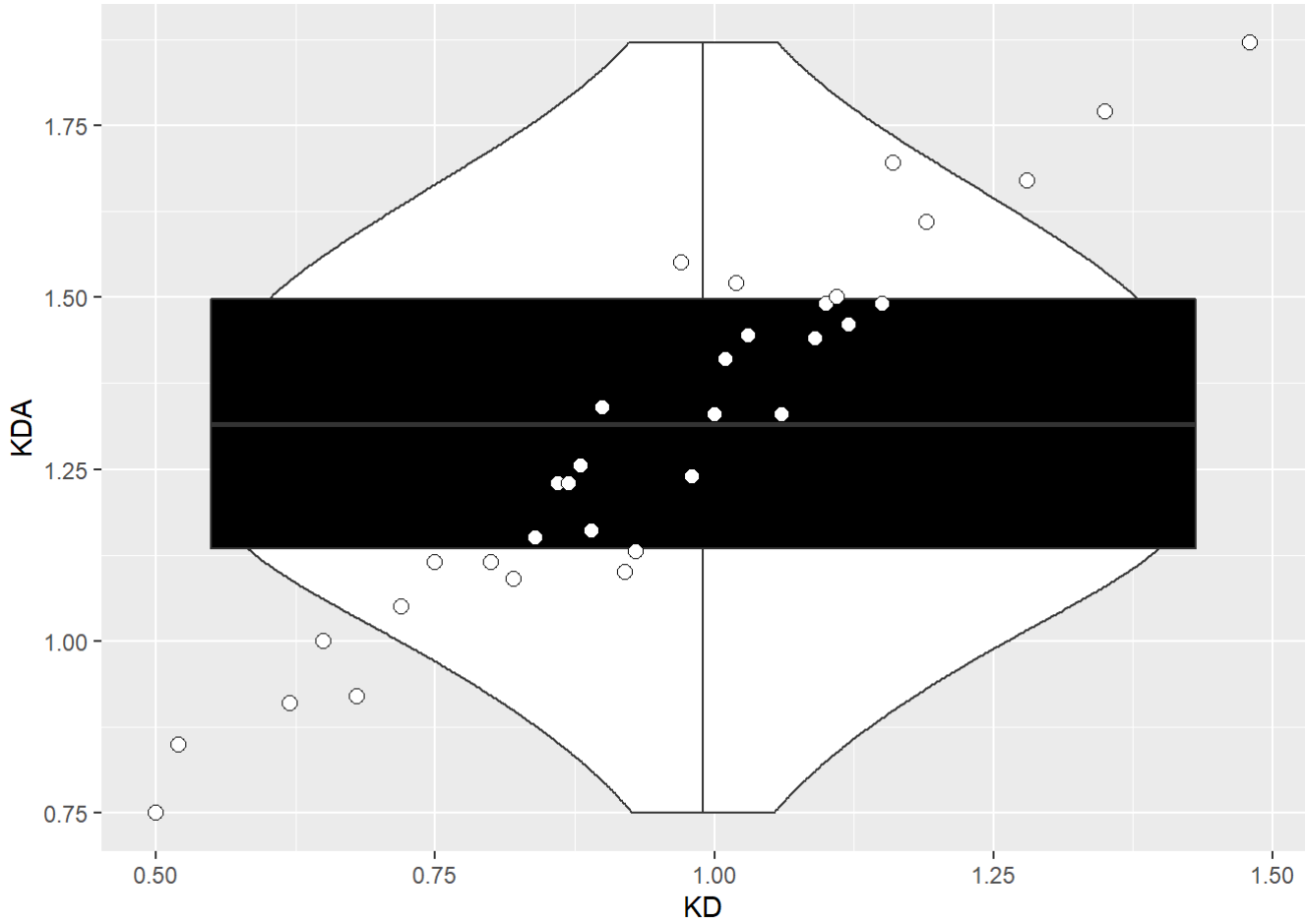
```
data6 <- ggplot(df, aes(x = KD, y = KDA))
data6+geom_violin()
```



A violin plot with box plot overlaid on it

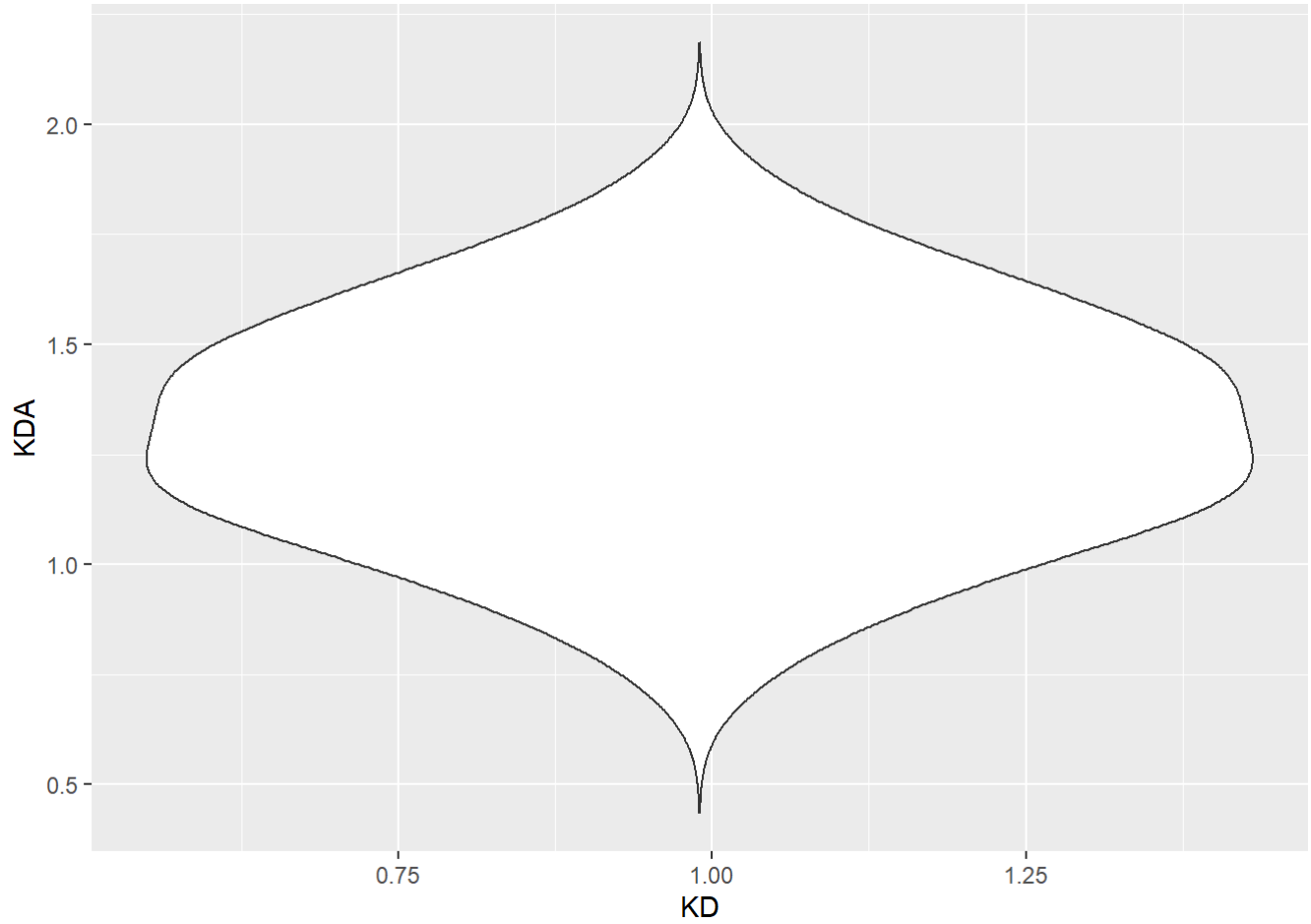
```
data6+geom_violin() +geom_boxplot(width = .1, fill = "black", outlier.colour = NA) +
stat_summary(fun= median, geom = "point", fill = "white", shape = 21,
size = 2.5)
```

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```



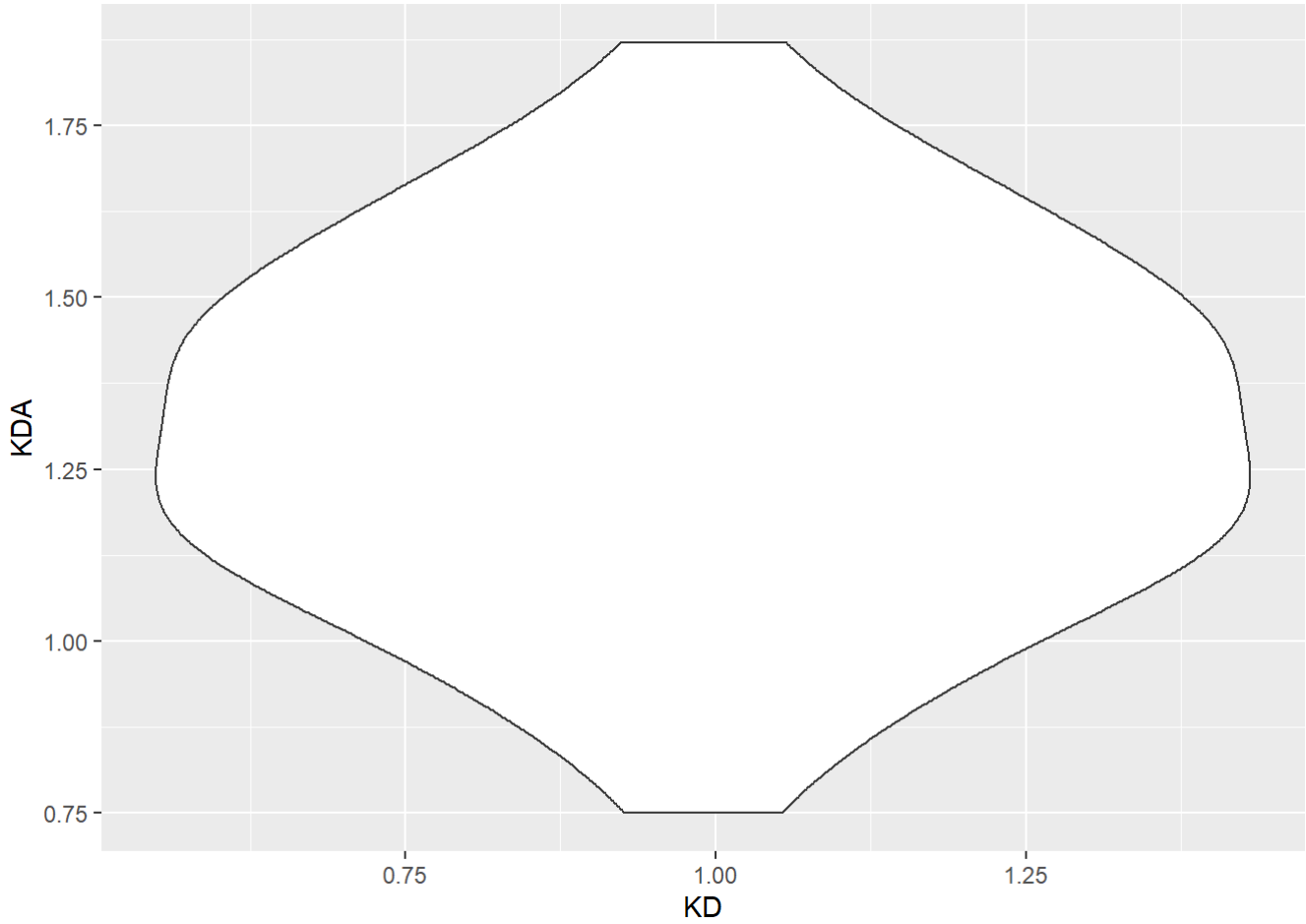
A violin plot with tails

```
data6+geom_violin(trim = FALSE)
```



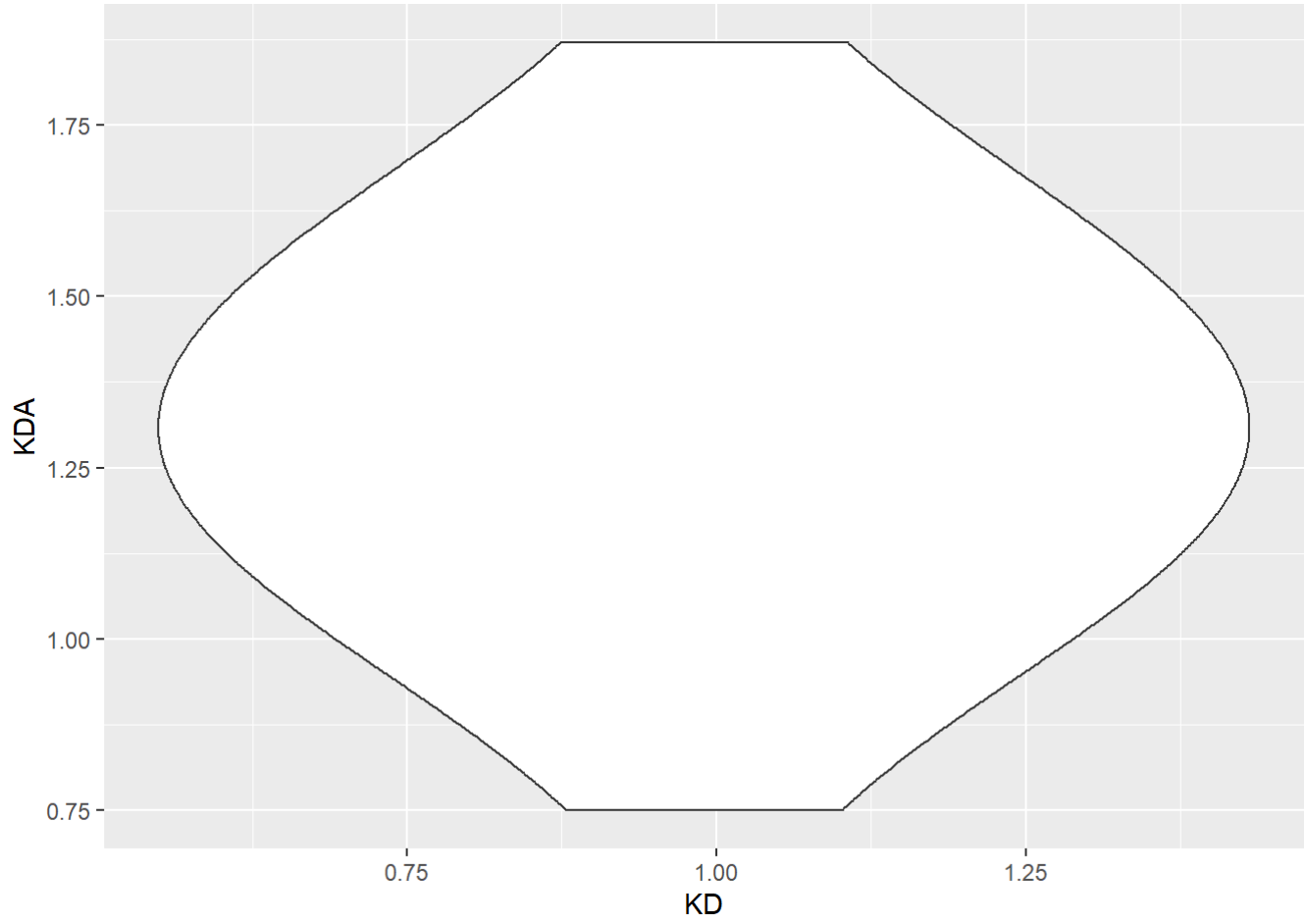
Violin plot with area proportional to number of observations

```
data6 +geom_violin(scale = "count")
```

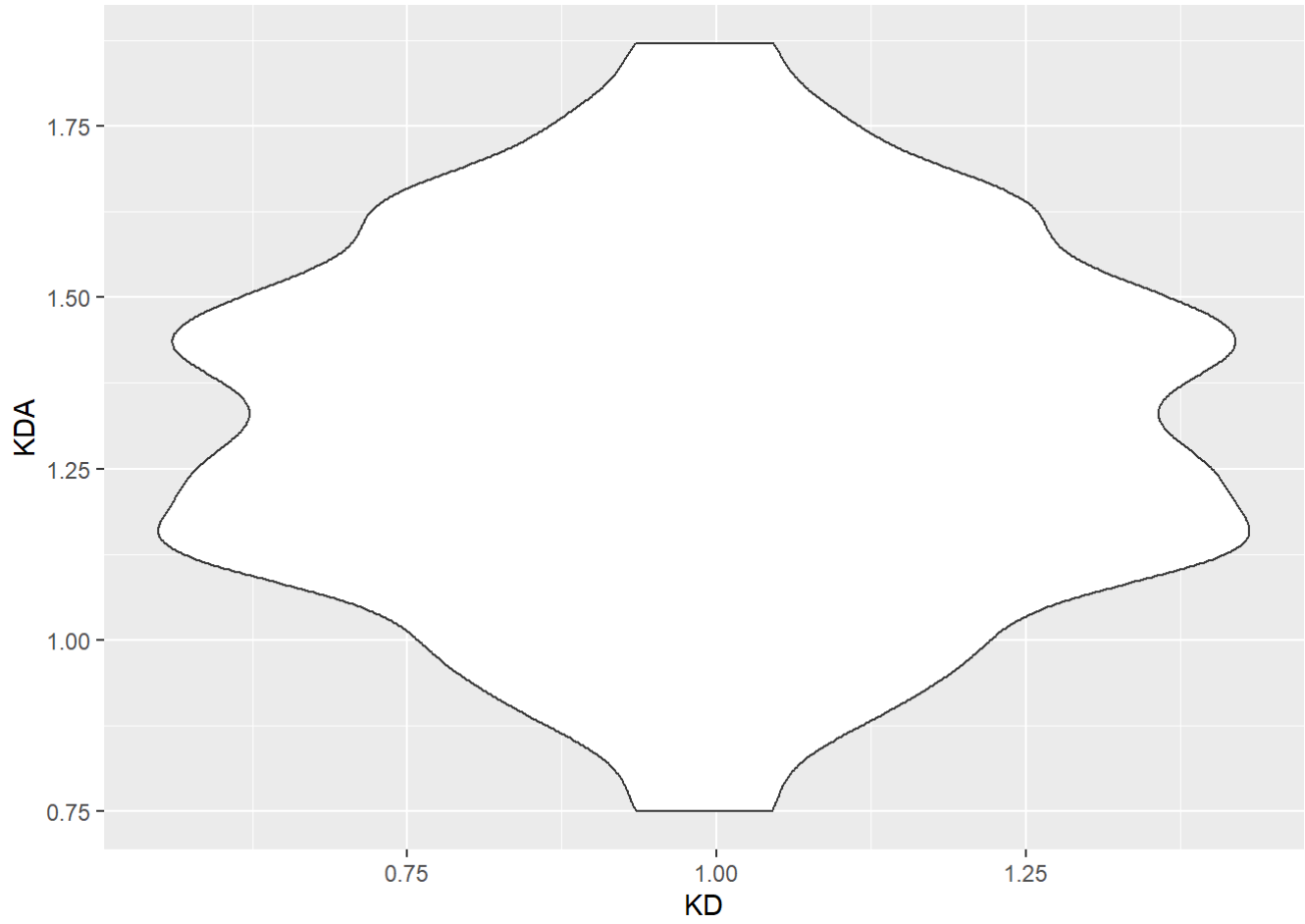


Violin plot with

```
# More smoothing
data6+geom_violin(adjust = 2)
```

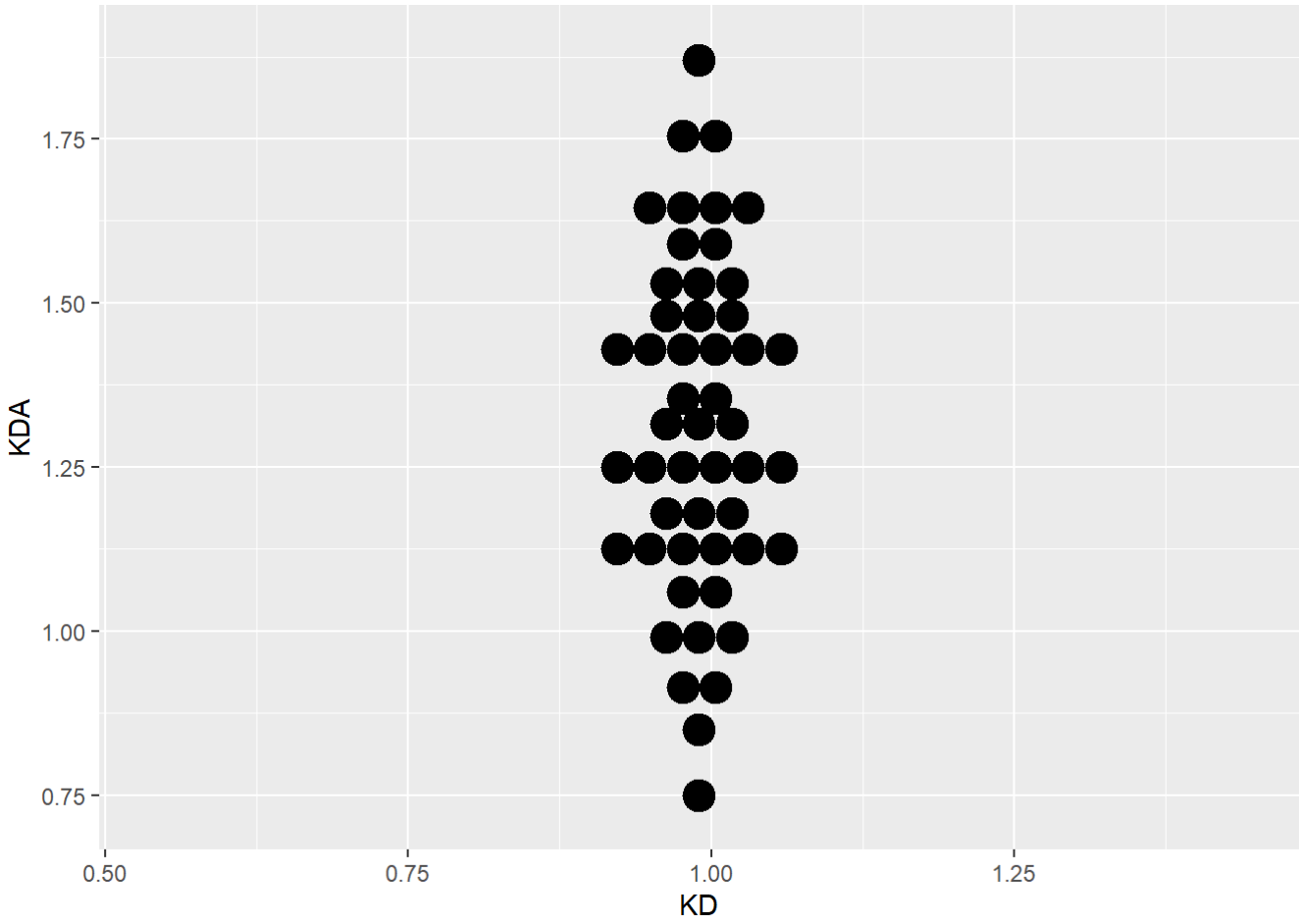



```
# Less smoothing
data6 +geom_violin(adjust = .5)
```



Making Multiple Dot Plots for Grouped Data

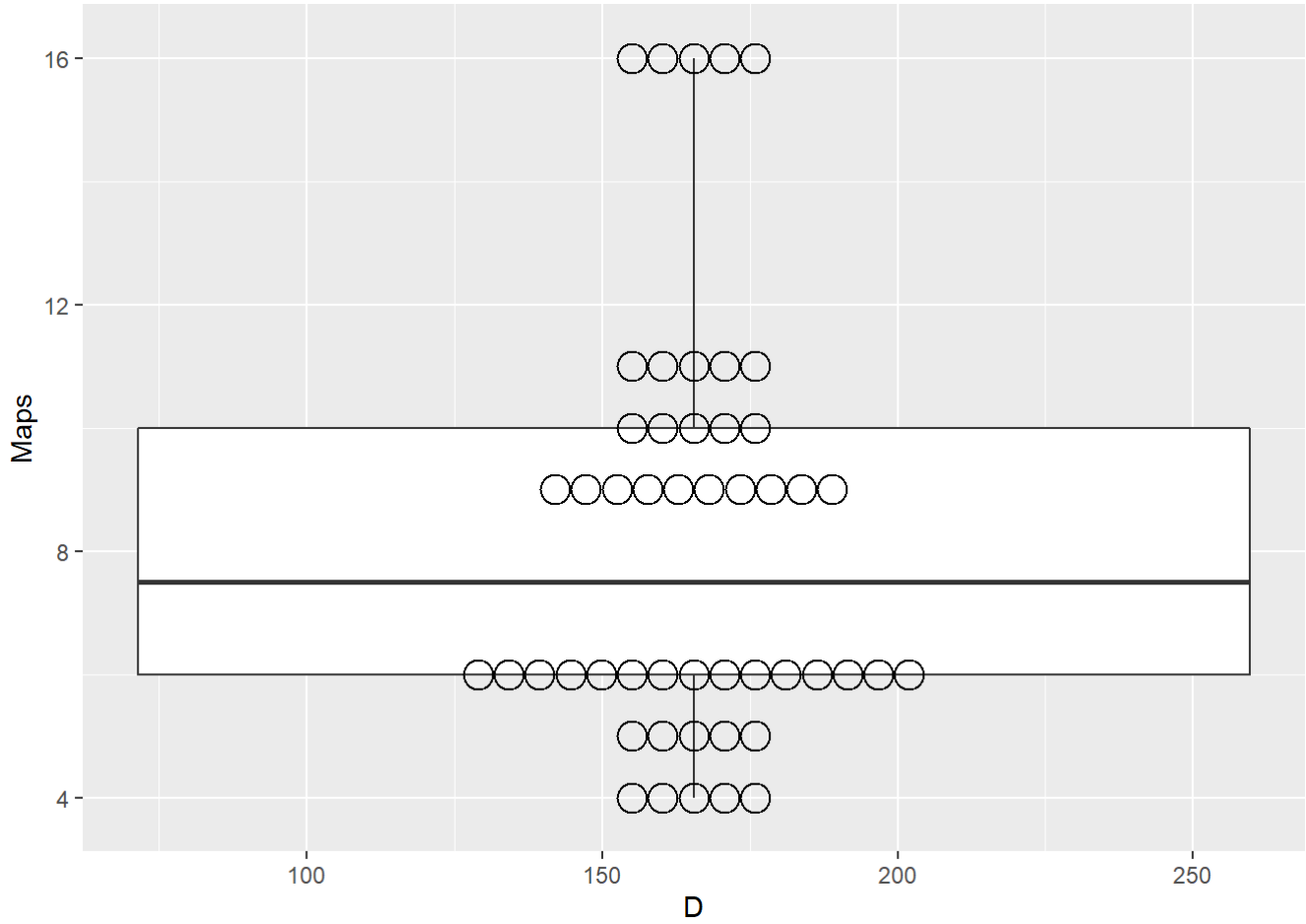
```
ggplot(df, aes(x = KD, y = KDA)) +
geom_dotplot(binaxis = "y", binwidth = .05, stackdir = "center")
```



Dot plot overlaid on box plot

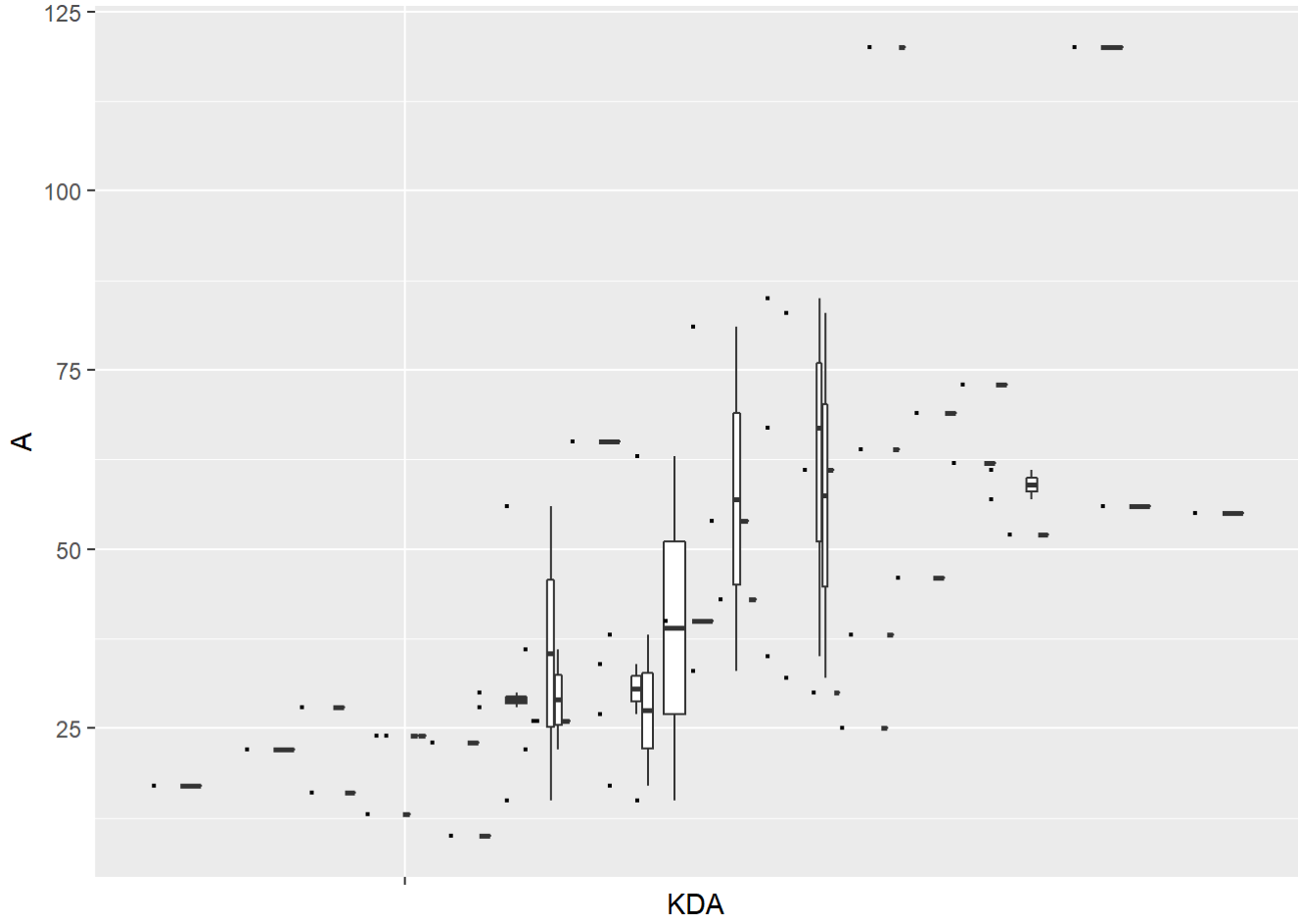
```
ggplot(df, aes(x = D, y = Maps)) +
geom_boxplot(outlier.colour = NA, width = .4) +
geom_dotplot(binaxis = "y", binwidth = .5, stackdir = "center", fill = NA)
```

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```



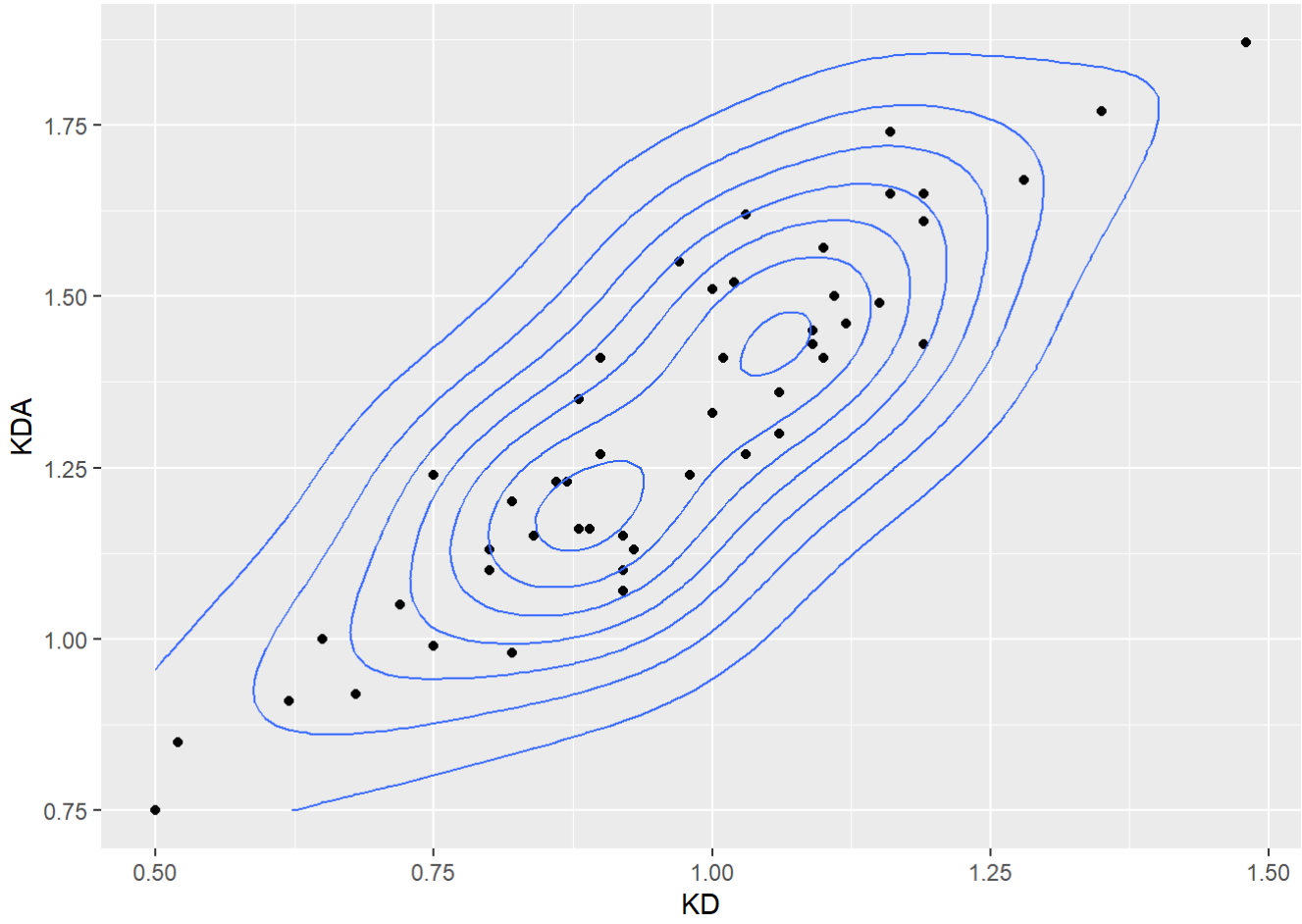
Dot plot next to box plot

```
ggplot(df, aes(x = KDA, y = A)) +
geom_boxplot(aes(x = as.numeric(KDA) + .02, group = KDA), width = .025) +geom_dotplot(aes(x =
as.numeric(KDA) - .02, group = KDA),binaxis = "y",binwidth = .5,stackdir = "center") +scale_x
_continuous(breaks = 1:nlevels(df$pretest),labels=levels(df$KDA)
)
```

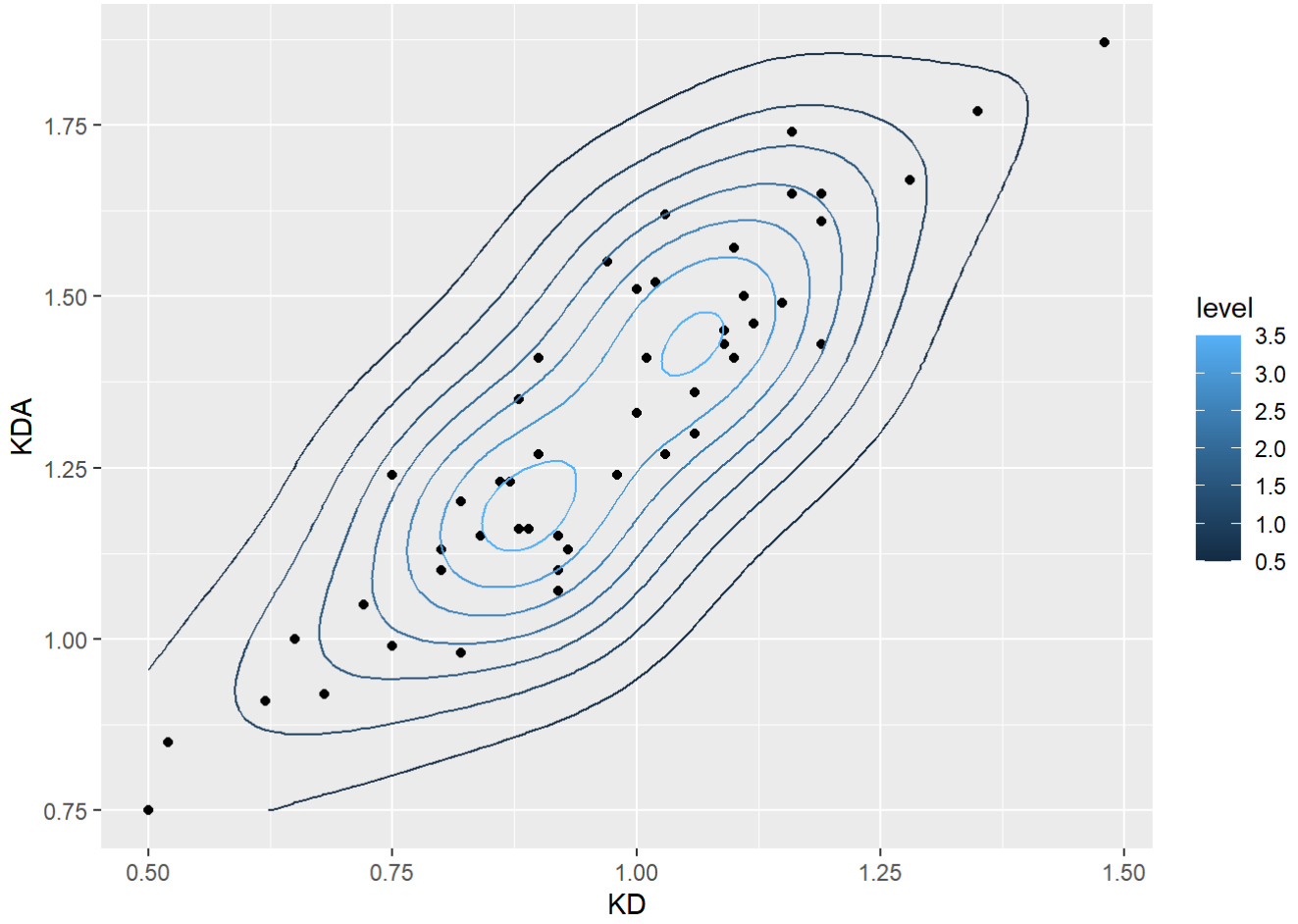


Making a Density Plot of Two-Dimensional Data

```
# Save a base plot object
 ggplot(df, aes(x = KD, y = KDA))+geom_point() +stat_density2d()
```



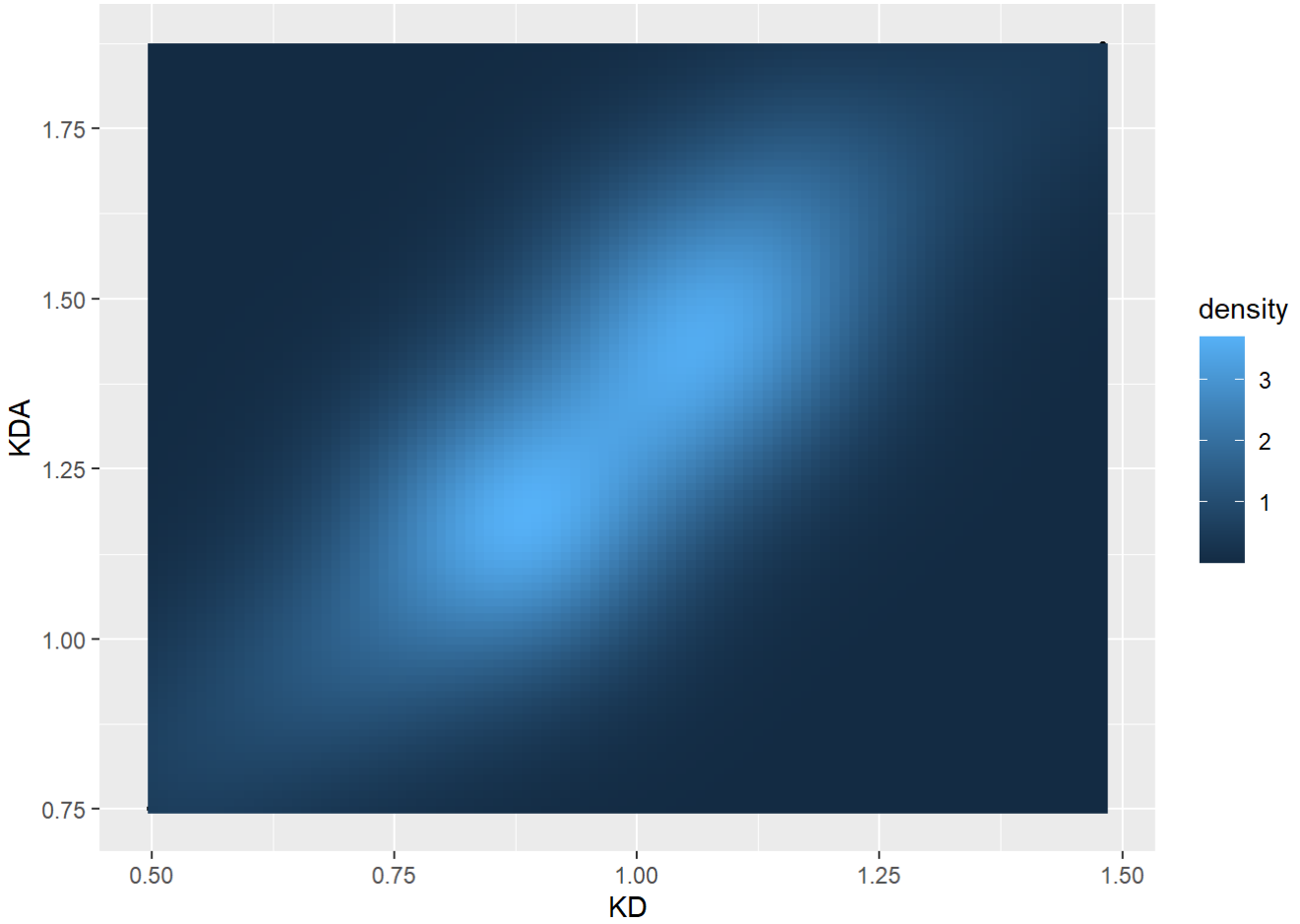

```
# Contour lines, with "height" mapped to color
ggplot(df, aes(x = KD, y = KDA))+geom_point() +stat_density2d()+stat_density2d(aes(colour =
 ..level..))
```



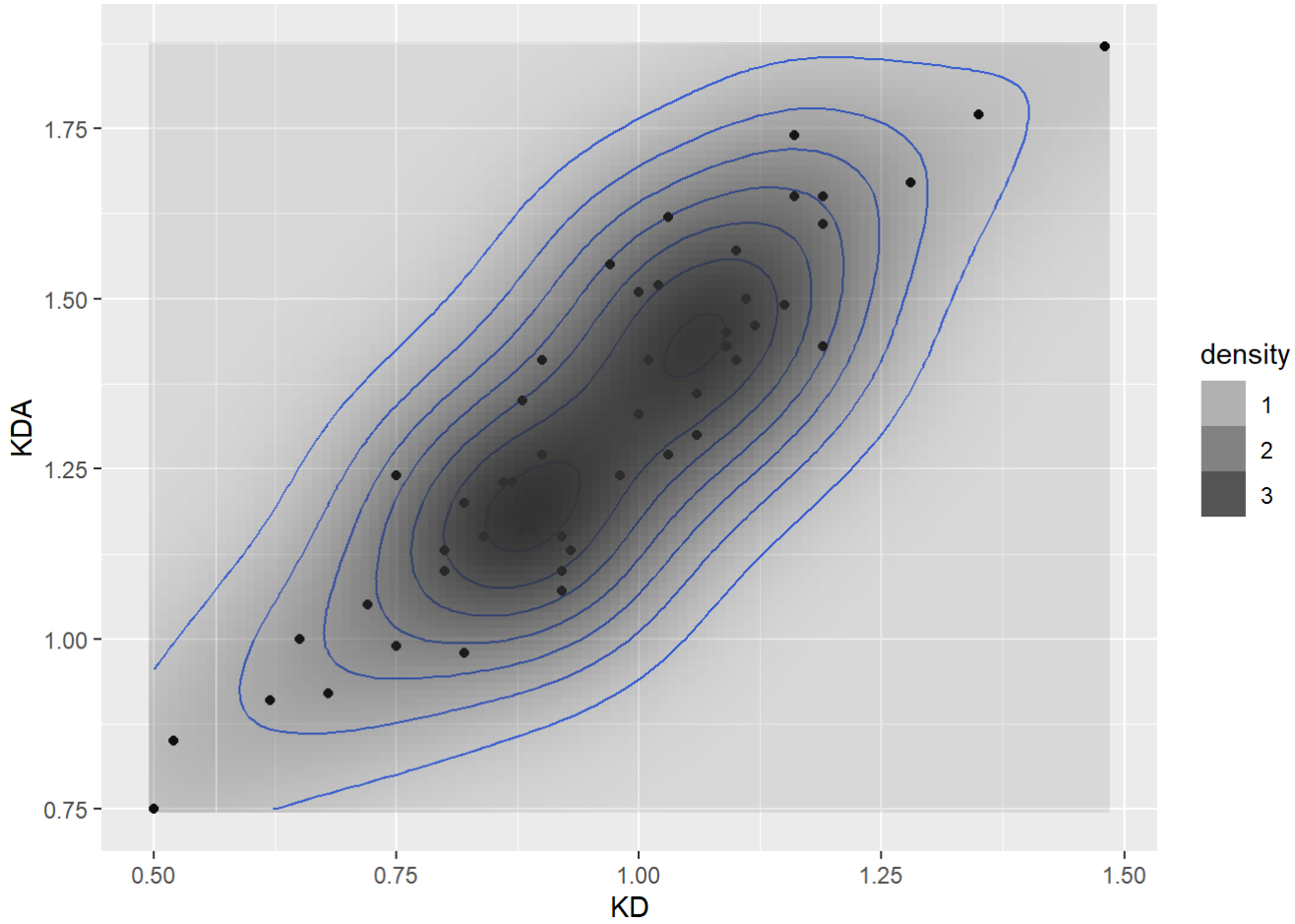
With ..density.. mapped to fill (1)

With points, and ..density.. mapped to alpha(2)

```
# Map density estimate to fill color
ggplot(df, aes(x = KD, y = KDA))+geom_point() +stat_density2d()+stat_density2d(aes(fill = ..d
ensity..), geom = "raster", contour = FALSE)
```



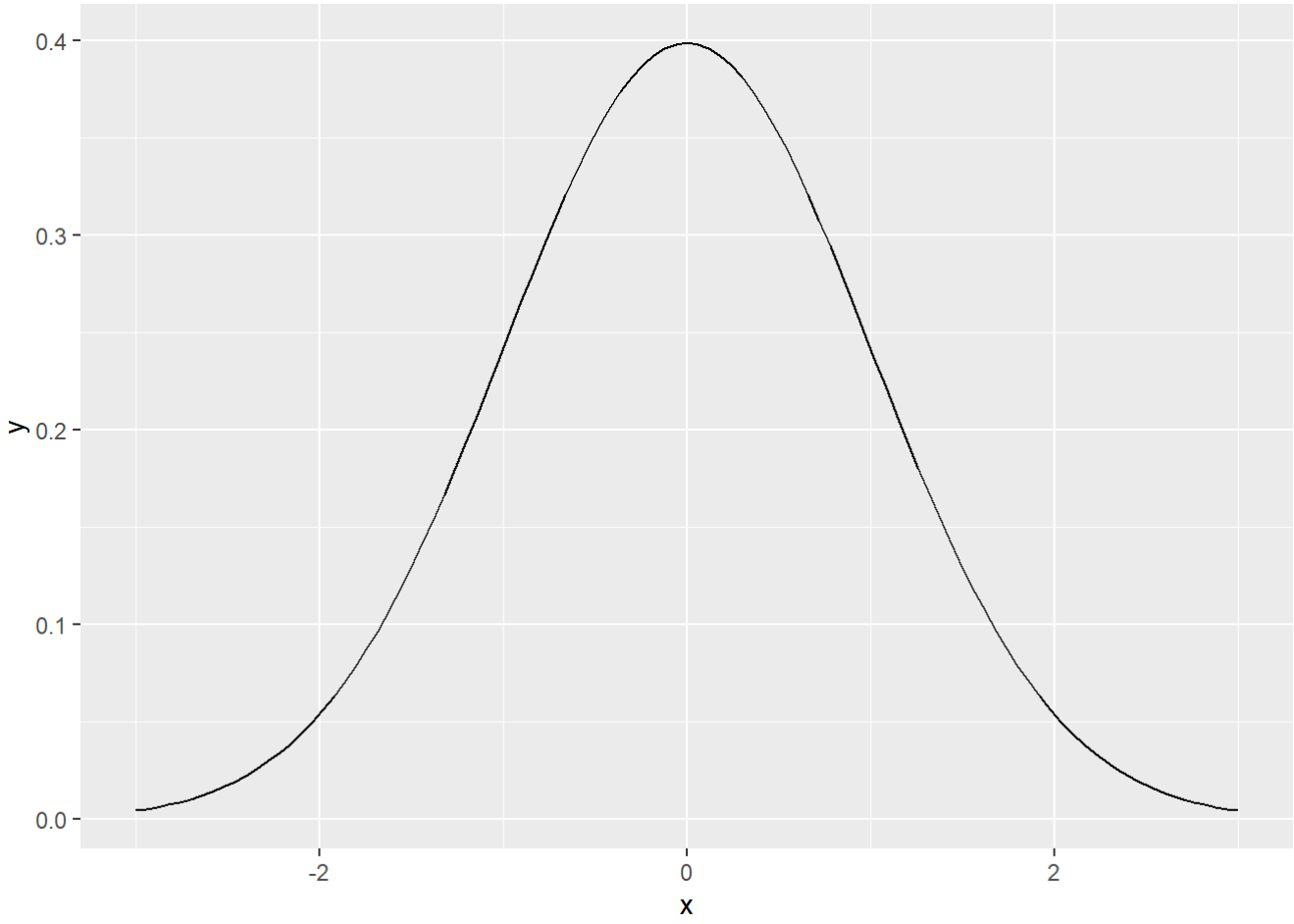

```
# With points, and map density estimate to alpha
ggplot(df, aes(x = KD, y = KDA))+geom_point() +stat_density2d()+geom_point() +stat_density2d
(aes(alpha = ..density..), geom = "tile", contour = FALSE)
```



Plotting a Function

The data frame is only used for setting the range

```
# The normal distribution
ggplot(data.frame(x = c(-3, 3)), aes(x = x))  + stat_function(fun = dnorm)
```




```
# The t-distribution with df=2
ggplot(data.frame(x = c(-3, 3)), aes(x = x)) + stat_function(fun = dt, args = list(df = 2))
```