市区町指標
- データ読み込み
- 項目の一覧
- 8列目までを分析対象とする
- 文字化けを防ぐためカラム名変更
- 散布図で各変数間の関係を確認
- 相関行列をオブジェクトに格納
- 転出者対人口比と世帯当たり人数の相関があったのかなぜだか知りたかったので転出者対人口比がどんな数値だか見てみる
- 転出者_対人口比でソートされたカラムを作成
- 世帯あたり人数を棒グラフでみてみる
- 平行分析で因子数を見積もる
- 因子分析の実行
  - #pa 主因子法,ols 最小二乗法, ml 最尤法 # varimax 直交 promax 斜交
- 結果の表示
- 因子負荷量
- 因子得点の確認
- 因子得点をデータフレームに変換
- 行の名前を変換
- 行の名前を変換
- クラスタリングを行う
- 結果の確認
- クラスタごとの数を確認
- 色ラベルの配列を作るためにクラスタ番号の配列をコピー
- factor(カテゴリ変数)に変換。カテゴリ変数にはlevelがある
- levelに色をつける
  - factorの実体は整数型なので文字列に変換
- 色分けして因子得点をプロット
- 元のデータに因子得点とクラスタ番号を付加
- 最後の列名の名前変更

マンション取引価格情報
- 価格データの読み込み
- 中古マンションに絞る
- stringrのインポート
- 取引時点から取引年を抽出
- 取引年をlistからintegerに変換
- stringiのインポート
- 取引時点から取引四半期を抽出
- 取引四半期をlistからintegerに変換
- 間取りを半角英数字に変換
- 建築年を和暦から西暦に変換
- 面積をintegerに変換
- 必要なカラムのみに絞る
- カラム名変更

駅別乗降客数
- データ読み込み
- マージキーの作成
- 中古マンション取引データと駅別乗降客数のマージ
- マージキーの削除
- GYOSEIをマージキーとして作成し、マージ
- 重複するカラムを削除
- ヒストグラム
- 常用対数ヒストグラム
- 面積と取引価格の散布図
- 取引物件の築年数のヒストグラム

市区町指標

データ読み込み

```
DFCity <- read.table("20200519_Presentation/data/TokyoSTAT.csv", sep=",",
header=TRUE, stringsAsFactors = FALSE, na.strings = "", fill=TRUE)
```

項目の一覧

```
str(DFCity)
```

8列目までを分析対象とする

```
DFCity <- DFCity[,1:8]
```

文字化けを防ぐためカラム名変更

```
colnames(DFCity) <- c("市町村","行政CD","世帯あたり人
数","Under15Ratio","Over65Ratio", "転入者_対人口比", "転出者_対人口比",
"DayPopulationRatio")
```

散布図で各変数間の関係を確認

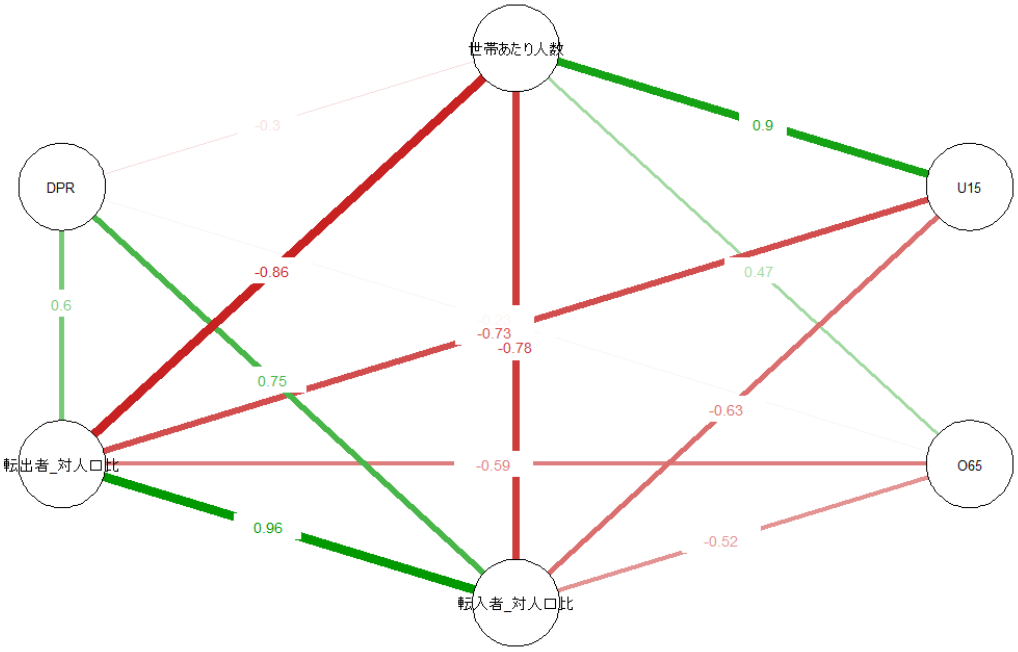
```
pairs(DFCity[,-c(1:2)])
```

相関行列をオブジェクトに格納

```
COR <- cor(DFCity[,-c(1:2)])
```

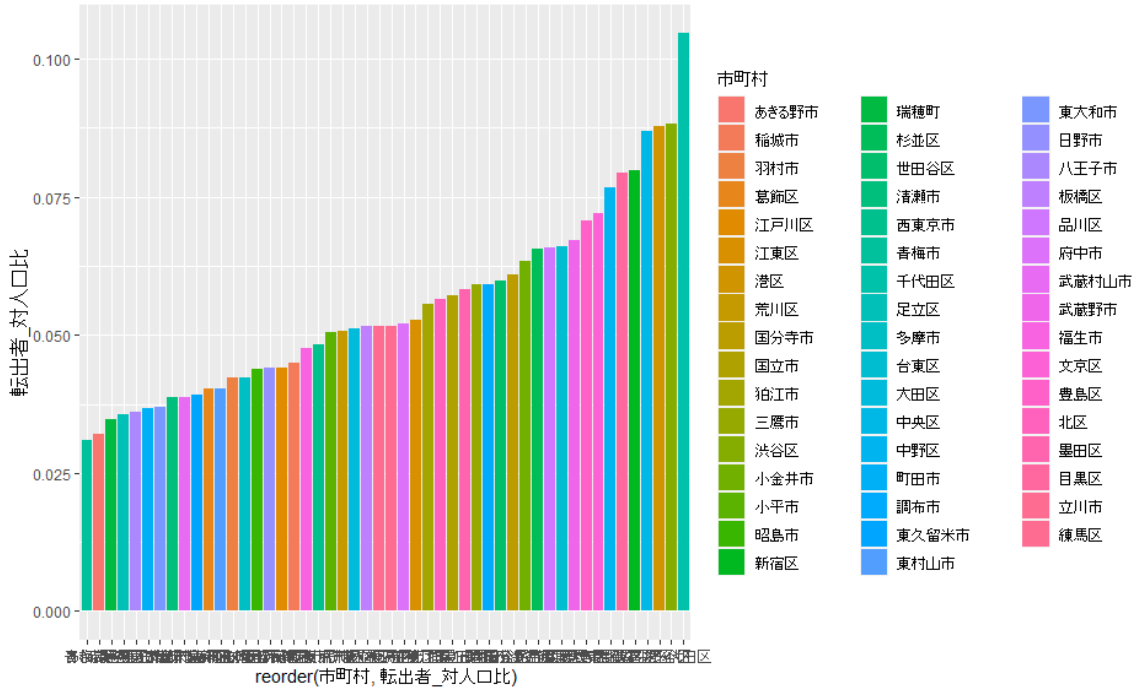
```
library(qgraph)
```

```
qgraph(COR,minimum=.20,edge.labels=T,label.scale=F,label.cex=0.8,edge.label.cex=
1.4)
```



転出者対人口比と世帯当たり人数の相関があったのかなぜだか知りたかったので転出者対人口比がどんな数値だか見てみる

```
ggplot(DFCity, aes(x=reorder(市町村,転出者_対人口比), y=転出者_対人口比, fill=市町村))
+ geom_bar(stat="identity")
```



これだとlegendがソートされない。そこで

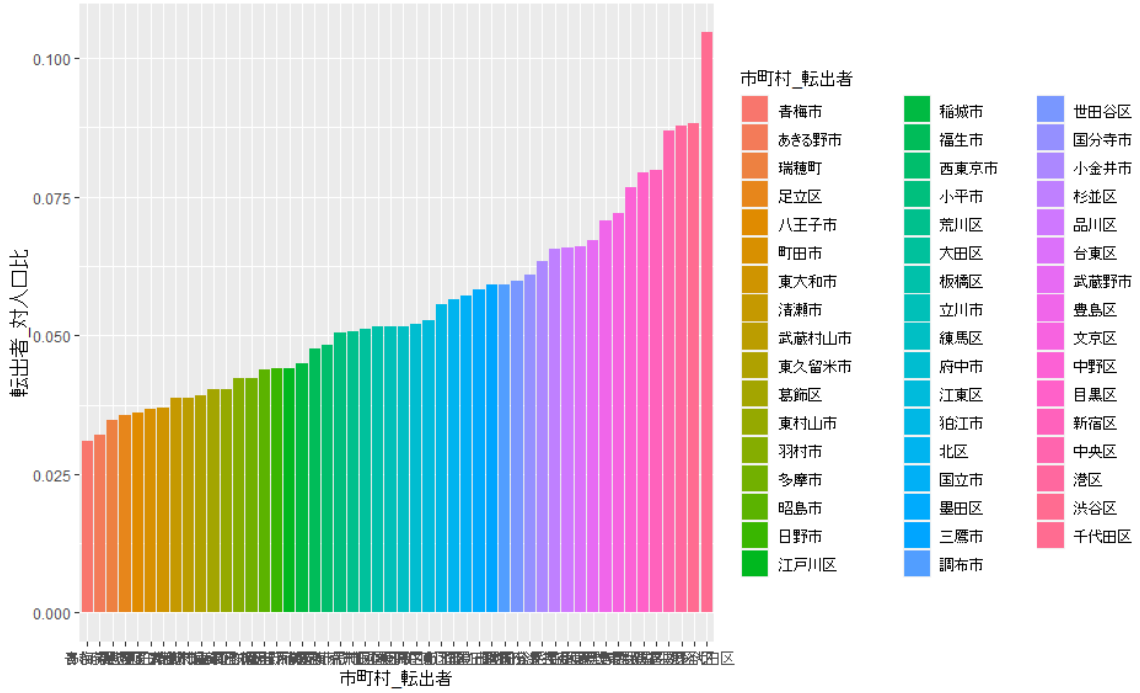
転出者_対人口比でソートされたカラムを作成

cf)[How to reorder a legend in ggplot2?](https://stackoverflow.com/questions/26872905/how-to-reorder-a-legend-in-ggplot2)

```
DFCity$市町村_転出者 <- with(DFCity, reorder(市町村, 転出者_対人口比))
```

```
ggplot(DFCity, aes(x=市町村_転出者, y=転出者_対人口比, fill=市町村_転出者)) +
geom_bar(stat="identity")
```

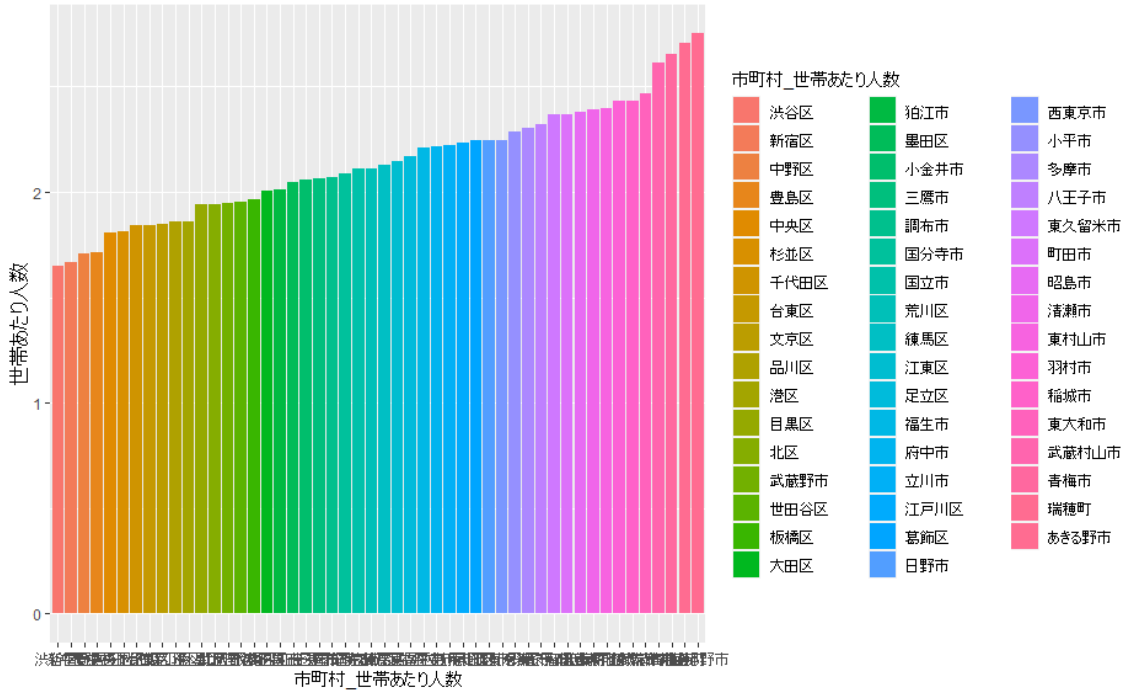
ヘルプをみるとわかるが、stat="identity"をつけないとケース数になってしまう。geom_barの代わりにgeom_col()を用いてもいい



世帯あたり人数を棒グラフでみてみる

```
DFCity$市町村_世帯あたり人数 <- with(DFCity, reorder(市町村, 世帯あたり人数))
```

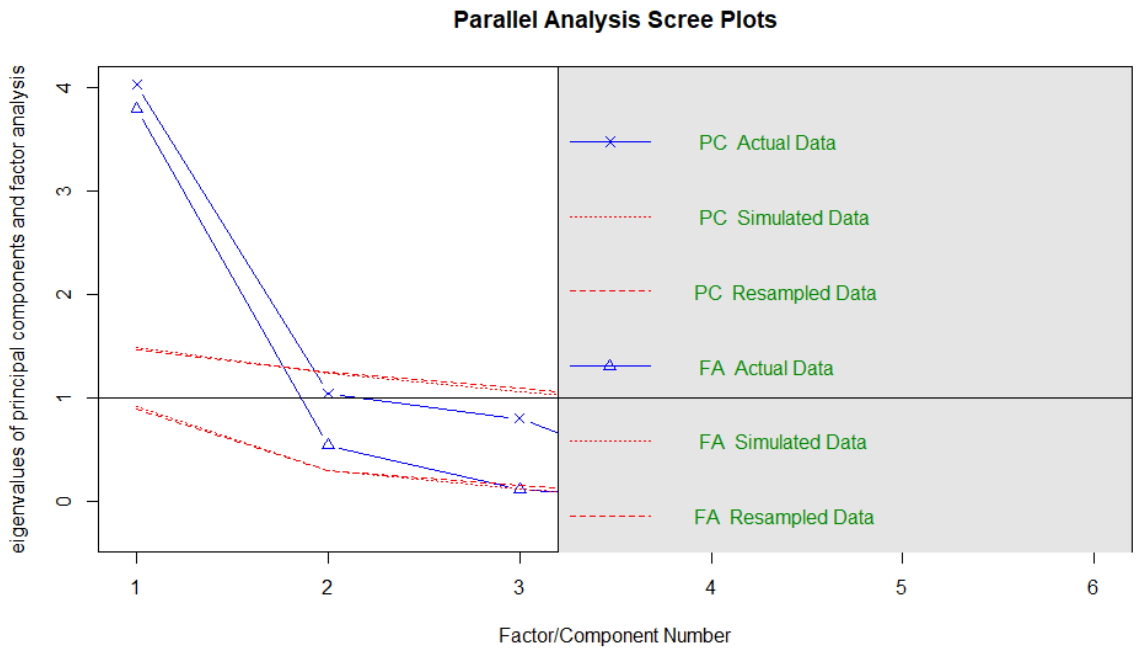
```
ggplot(DFCity, aes(x=市町村_世帯あたり人数, y=世帯あたり人数, fill=市町村_世帯あたり人数)) + geom_bar(stat="identity")
```



平行分析で因子数を見積もる

```
library(psych)
```

```
fa.parallel(DFCity[,-c(1:2)])
```



因子分析の実行

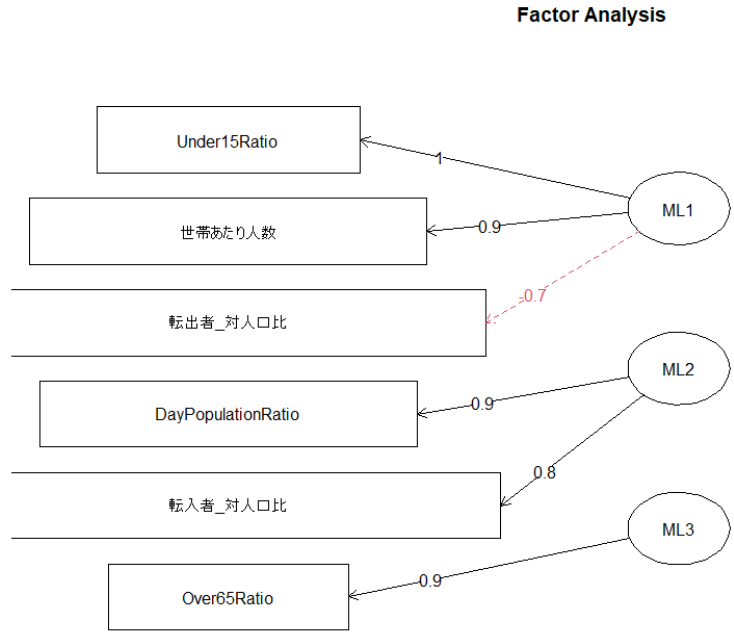
#pa 主因子法,ols 最小二乗法, ml 最尤法 # varimax 直交 promax 斜交

```
result.fa <- fa(DFCity[,-c(1:2)],nfactors=3,fm="ml",rotate="varimax")
```

結果の表示

```
print(result.fa, digits = 2, sort = TRUE)
```

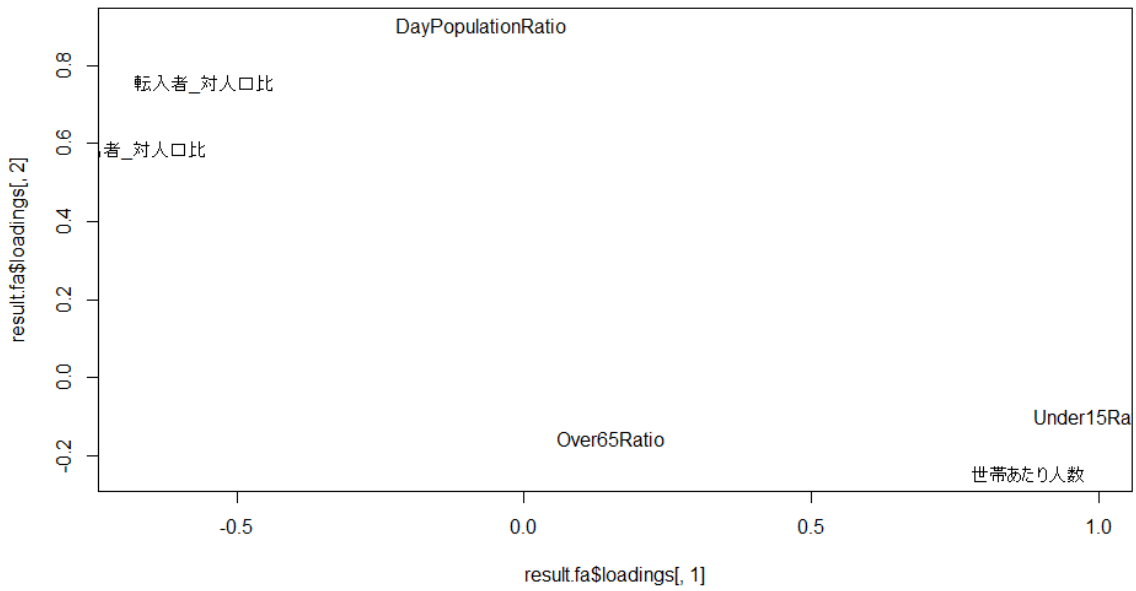
```
fa.diagram(result.fa)
```



因子負荷量

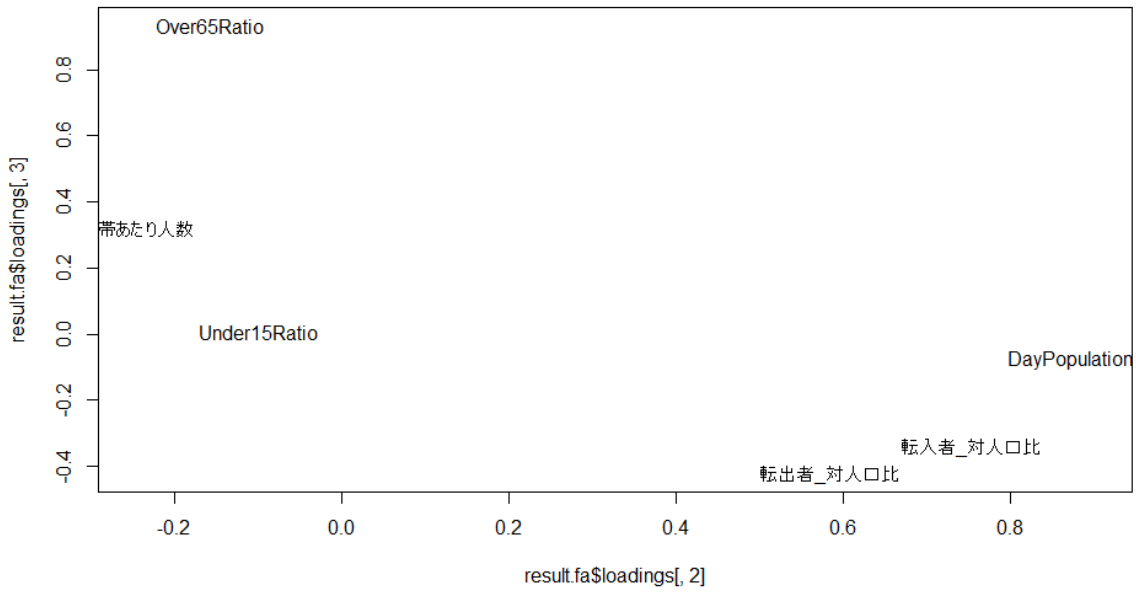
```
plot(result.fa$loadings[,1], result.fa$loadings[,2], type="n")
```

```
text(result.fa$loadings[,1], result.fa$loadings[,2],
rownames(result.fa$loadings))
```



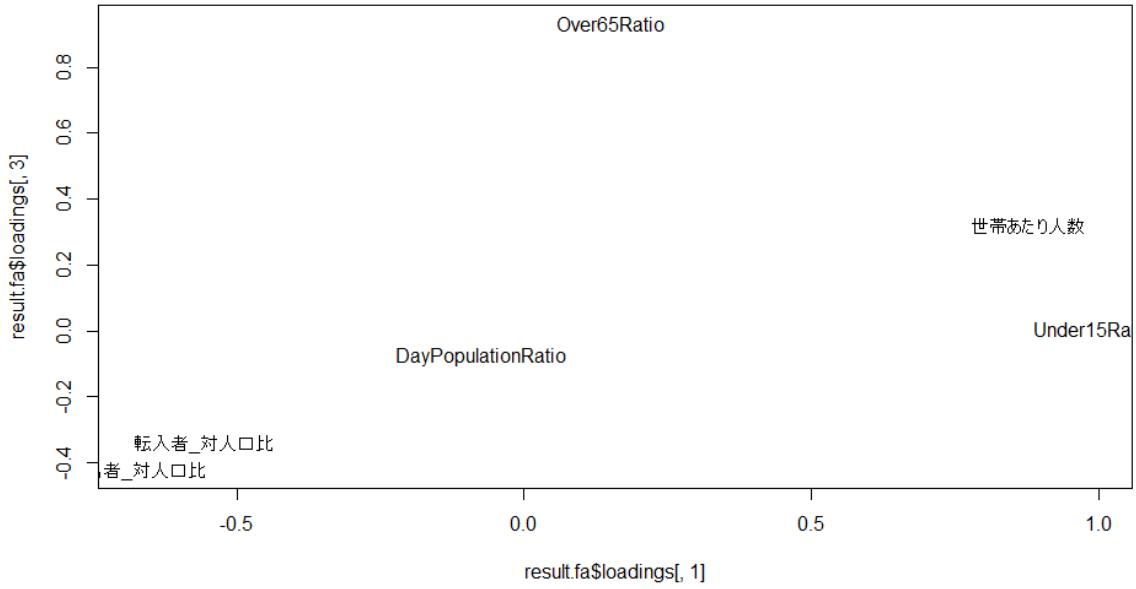

```
plot(result.fa$loadings[,2], result.fa$loadings[,3], type="n")
```

```
text(result.fa$loadings[,2], result.fa$loadings[,3],
rownames(result.fa$loadings))
```




```
plot(result.fa$loadings[,1], result.fa$loadings[,3], type="n")
```

```
text(result.fa$loadings[,1], result.fa$loadings[,3],
rownames(result.fa$loadings))
```



因子得点の確認

```
head(result.fa$scores)
```

因子得点をデータフレームに変換

```
DFfa <- as.data.frame(result.fa$scores)
```

行の名前を変換

```
rownames(DFfa) <- DFCity$市町村
```

行の名前を変換

```
names(DFfa) = c("郊外生活度","経済活性度","高齢化度")
```

namesとcolnamesの違い

https://stackoverflow.com/questions/24799153/what-is-the-difference-between-names-and-colnames#:~:text=Not%20quite%2D%2Dthe%20big,and%20an%20error%20for%20setting)..)

colnamesは行列とデータフレームに適用可能。namesはデータフレームだけ

colnamesはできないが、namesはvectorをset/getできる。（names(DF) = c("A", "B") colnames(DF) <- c("A", B))

クラスタリングを行う

```
num.km <- kmeans(DFfa, 3, iter.max = 10)
```

結果の確認

```
head(num.km$cluster)
```

クラスタごとの数を確認

```
table(num.km$cluster)
```

```
 1  2  3
15 16 19
```

色ラベルの配列を作るためにクラスタ番号の配列をコピー

```
color.km <- num.km$cluster
```

```
head(color.km)
```

factor(カテゴリ変数)に変換。カテゴリ変数にはlevelがある

```
color.km <- as.factor(color.km)
```

levelに色をつける

```
levels(color.km) <- c("blue", "red", "green")
```

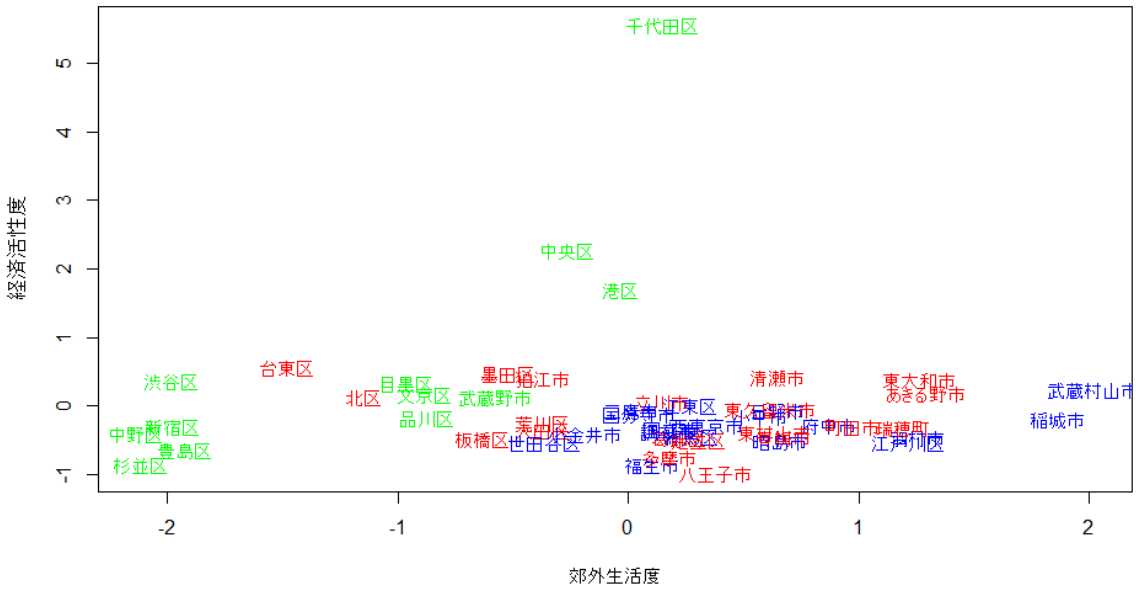
factorの実体は整数型なので文字列に変換

```
color.km <- as.character(color.km)
```

色分けして因子得点をプロット

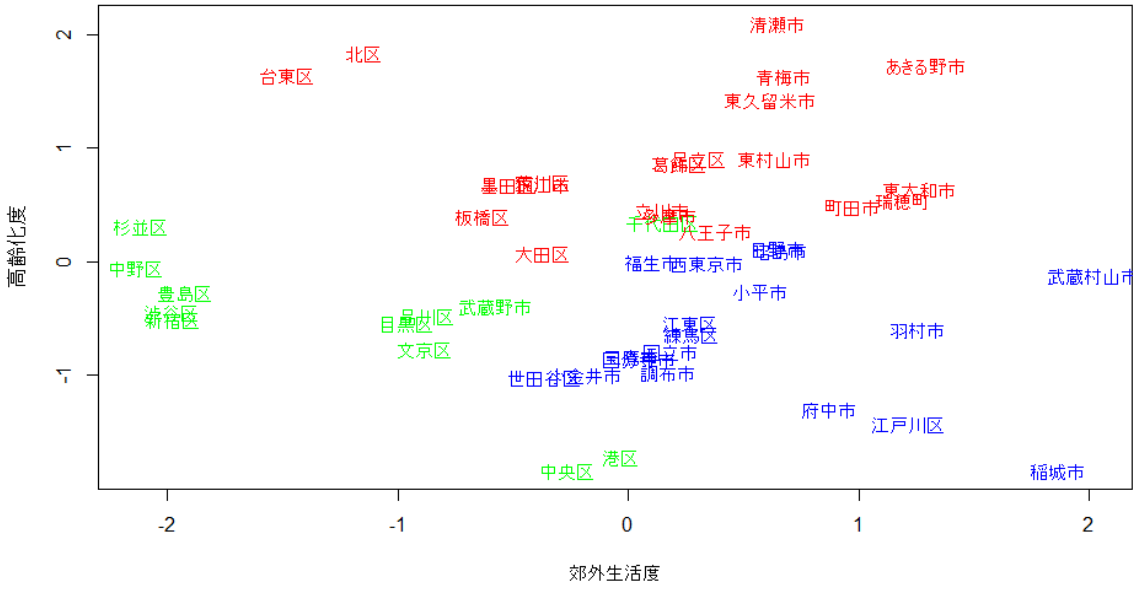
```
plot(DFfa[, 1], DFfa[, 2], type="n", xlab="郊外生活度", ylab="経済活性度")
```

```
text(DFfa[, 1], DFfa[, 2], rownames(DFfa), col=color.km)
```



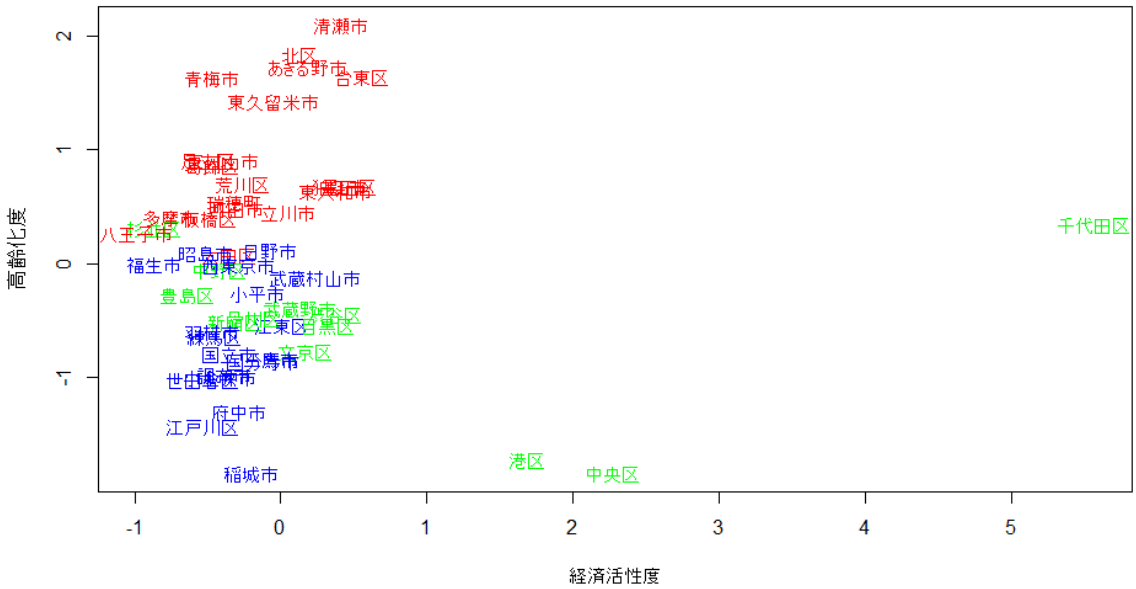

```
plot(DFfa[, 1], DFfa[, 3], type="n", xlab="郊外生活度", ylab="高齢化度")
```

```
text(DFfa[, 1], DFfa[, 3], rownames(DFfa), col=color.km)
```



```
plot(DFfa[, 2], DFfa[, 3], type="n", xlab="経済活性度", ylab="高齢化度")
```

```
text(DFfa[, 2], DFfa[, 3], rownames(DFfa), col=color.km)
```



元のデータに因子得点とクラスタ番号を付加

```
DFCity <- cbind(DFCity, DFfa)
```

```
DFCity <- cbind(DFCity, num.km$cluster)
```

最後の列名の名前変更

```
colnames(DFCity)[ncol(DFCity)] <- "clusterNo"
```

マンション取引価格情報

価格データの読み込み

```
DFProperty =
read.table("20200519_Presentation/data/13_Tokyo_20131_20144.csv",sep=",",header=
TRUE,stringsAsFactors = FALSE, na.strings = "", fill=TRUE)
```

中古マンションに絞る

```
DFProperty <- DFProperty[DFProperty$種類=="中古マンション等",]
```

stringrのインポート

```
library("stringr")
```

取引時点から取引年を抽出

```
ex.year <- function(x) {
    if (is.na(x)) {
        return(NA)
    } else {
        year <- str_match(x, "([0-9]{1,})年.*")[2]
        return(as.integer(year))
    }
}
```

```
DFProperty$取引年 <- lapply(DFProperty$取引時点, ex.year)
```

取引年をlistからintegerに変換

```
DFProperty$取引年 <- as.integer(DFProperty$取引年)
```

stringiのインポート

```
library("stringi")
```

取引時点から取引四半期を抽出

```
ex.quater <- function(x) {
    zenkaku <- str_match(x,pattern=".*第(.*)四半期")[2]
    return(as.integer(stri_trans_nfkc(zenkaku)))
}
```

```
DFProperty$取引四半期 <- lapply(DFProperty$取引時点, ex.quater)
```

取引四半期をlistからintegerに変換

```
DFProperty$取引四半期 <- as.integer(DFProperty$取引四半期)
```

間取りを半角英数字に変換

```
DFProperty$間取り <- lapply(DFProperty$間取り, stri_trans_nfkc)
```

建築年を和暦から西暦に変換

```
cv.wareki <- function(x) {
    if (is.na(x)) {
        return(NA)
    } else if (str_detect(x,pattern="昭和")) {
        year <- str_match(x, "昭和([0-9]{1,})年")[2]
        return(as.integer(year) + 1925)
    } else if (str_detect(x,pattern="平成")) {
        year <- str_match(x, "平成([0-9]{1,})年")[2]
        return(as.integer(year) -12 + 2000)
    } else if (str_detect(x, pattern="戦前")){
        return(1941)
    }
}
```

```
DFProperty$建築年 <- lapply(DFProperty$建築年,cv.wareki)
```

```
DFProperty$建築年 <- as.integer(DFProperty$建築年)
```

面積をintegerに変換

```
DFProperty$面積 <- as.integer(DFProperty$面積)
```

必要なカラムのみに絞る

```
columnList <- c("No","種類","市区町村コード","都道府県名","市区町村名","地区名","最寄駅.名称","最寄駅.距離.分.","取引価格.総額.","間取り","面積","建築年","建物の構造","用途","今後の利用目的","都市計画","建ぺい率...","容積率...","取引年","取引四半期","改装")
```

```
DFProperty <- DFProperty[, columnList]
```

カラム名変更

```
columnList2 <- c("No","種類","市区町村CD","都道府県","市区町村","地区","最寄駅","最寄駅分","取引価格","間取り","面積m2","建築年","構造","現状用途","今後用途","計画地域","建ぺい率","容積率","取引年","取引四半期","備考")
```

```
colnames(DFProperty) <- columnList2
```

駅別乗降客数

データ読み込み

```
DFStation <- read.table("20200519_Presentation/data/TokyoSTATION.csv", sep=",", header = TRUE, stringsAsFactors = FALSE, na.strings = "", quote = "\"", fill = TRUE, fileEncoding = "UTF-8")
```

マージキーの作成

```
DFProperty$Station <- DFProperty$最寄駅
```

中古マンション取引データと駅別乗降客数のマージ

```
DFPSt <- merge(DFProperty, DFStation, by="Station")
```

マージキーの削除

```
DFPSt <- DFPSt[,-1]
```

GYOSEIをマージキーとして作成し、マージ

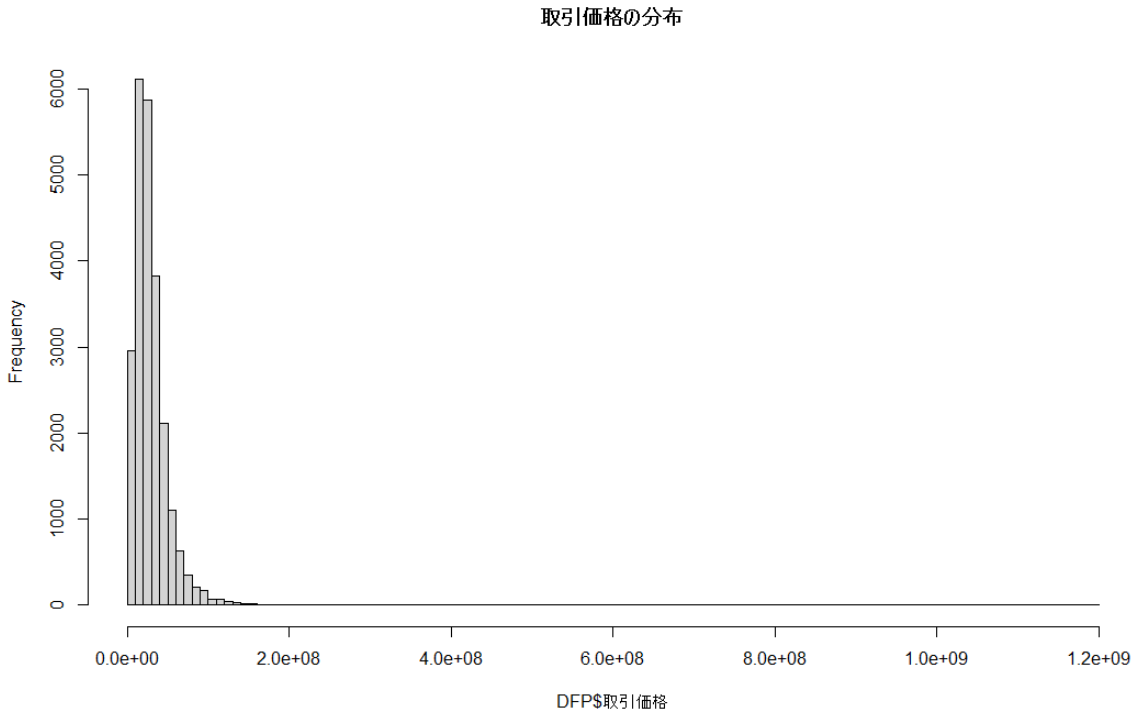
```
DFPSt$GYOSEI <- DFPSt$市区町村CD
DFCity$GYOSEI <- DFCity$行政CD
DFAll <- merge(DFPSt, DFCity, by="GYOSEI")
DFAll <- DFAll[,-1]
```

重複するカラムを削除

```
DFAll$市町村 <- NULL
DFAll$行政CD <- NULL
```

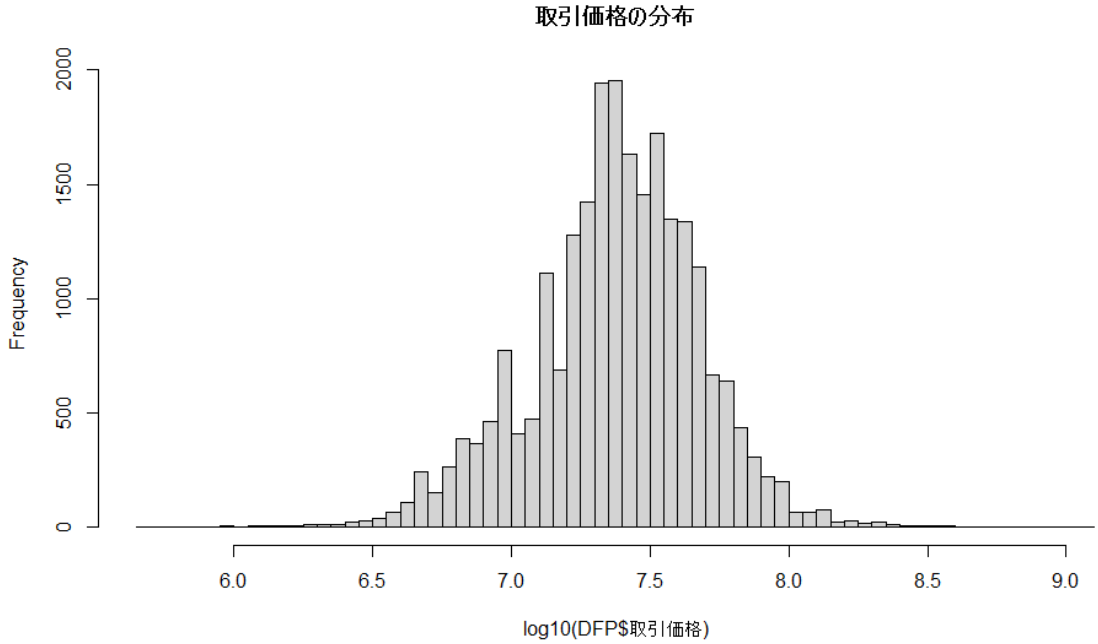
ヒストグラム

```
hist(DFAll$取引価格,breaks=100, main="取引価格の分布")
```



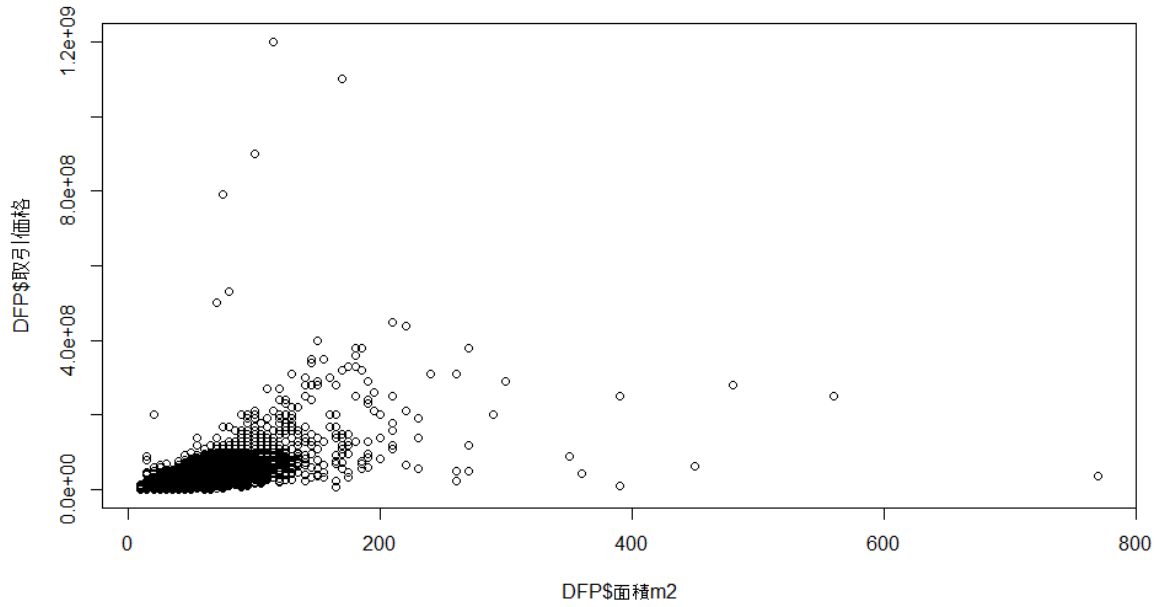
常用対数ヒストグラム

```
hist(log10(DFAll$取引価格),breaks=100, main="取引価格の分布")
```



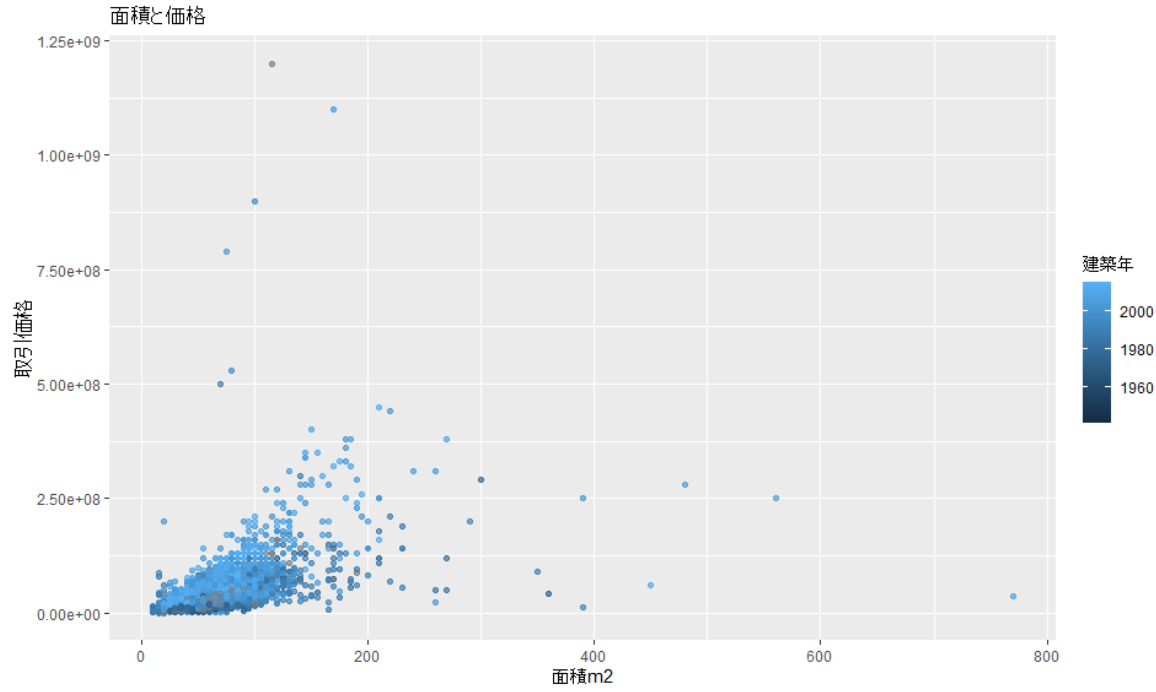
面積と取引価格の散布図

```
plot(DFAll$面積m2, DFAll$取引価格)
```

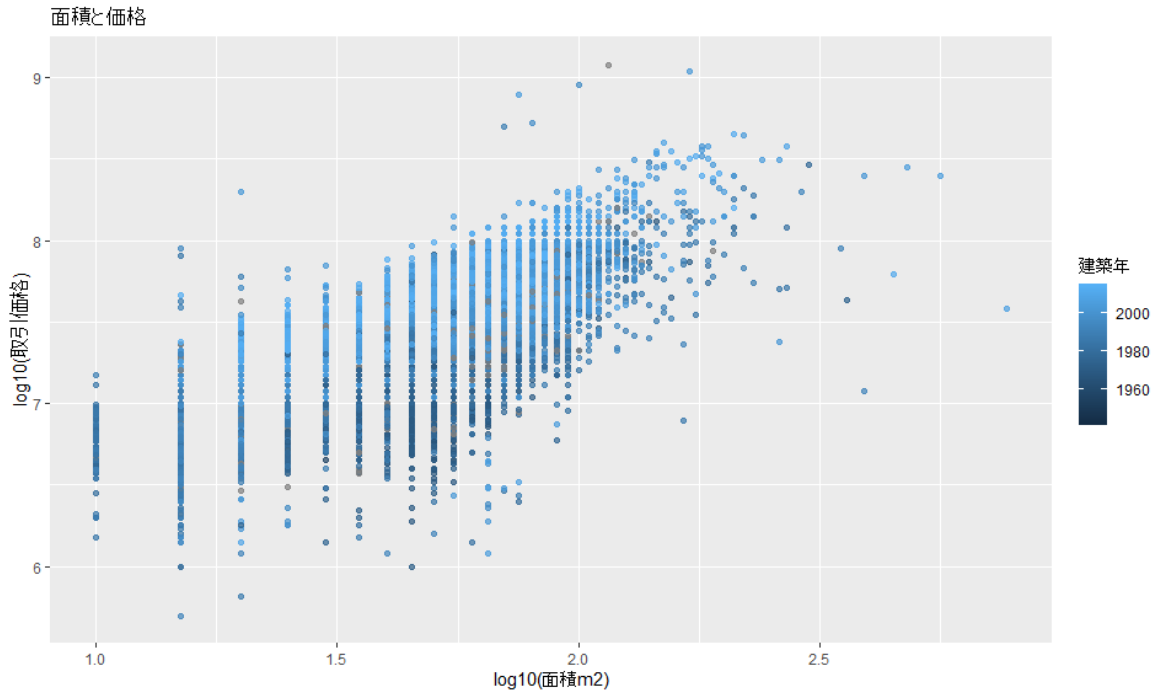



```
library(ggplot2)
```

```
ggplot(DFAll, aes(x = 面積m2, y = 取引価格)) + geom_point(aes(colour=建築年),
alpha=0.7) + labs(colour="建築年") + ggtitle("面積と価格")
```

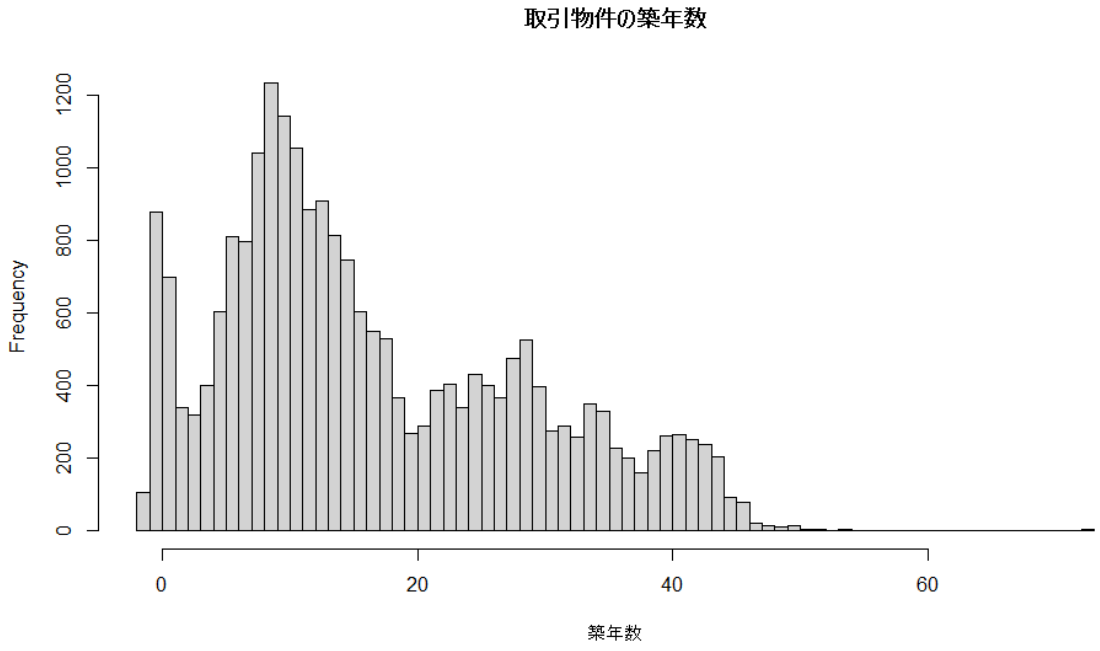



```
ggplot(DFAll, aes(x = log10(面積m2), y = log10(取引価格))) +
geom_point(aes(colour=建築年), alpha=0.7) + labs(colour="建築年") + ggtitle("面積と
価格")
```



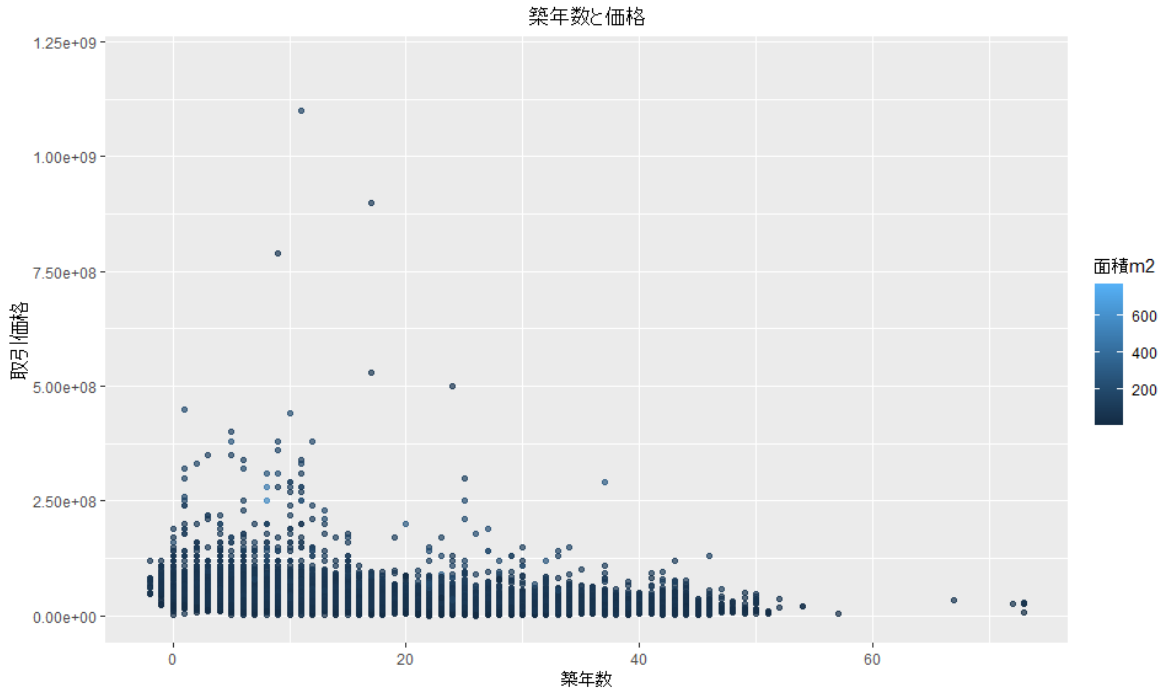
取引物件の築年数のヒストグラム

```
DFAll$築年数 <- DFAll$取引年 - DFAll$建築年
hist(DFAll$築年数, breaks=100, main="取引物件の築年数", xlab="築年数")
```

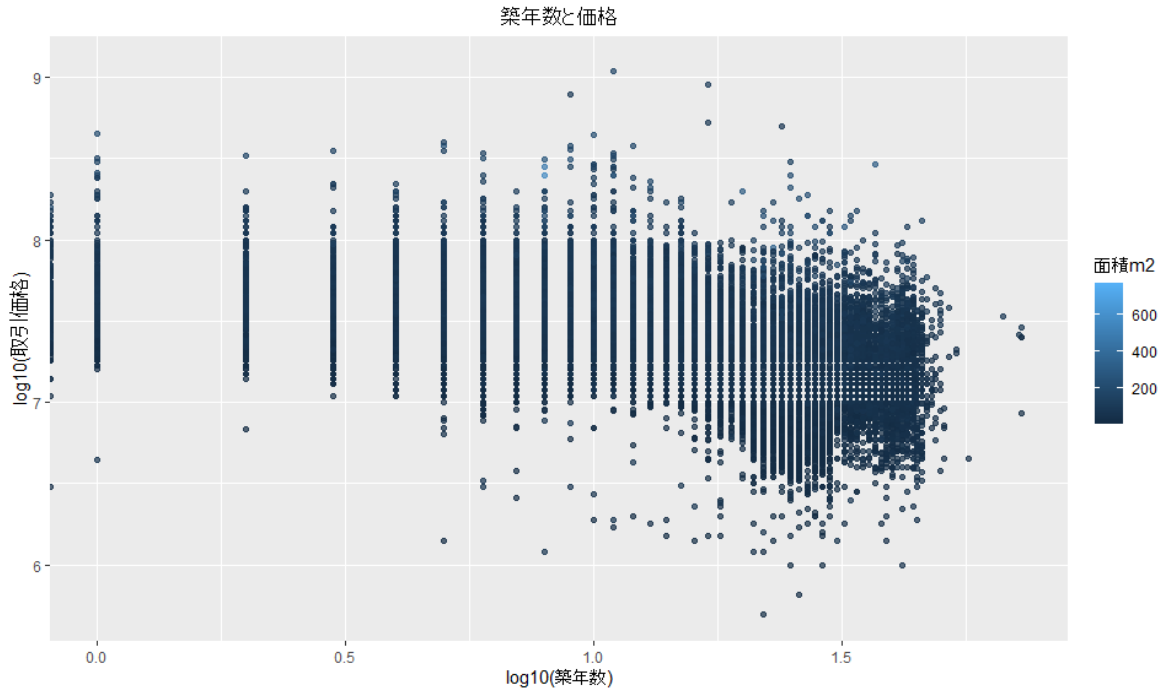


築年数と価格

```
ggplot(DFAll, aes(x = 築年数, y = 取引価格)) + geom_point(aes(colour=面積m2),
alpha=0.7) + ggtitle("築年数と価格") + theme(plot.title = element_text(hjust =
0.5))
```




```
ggplot(DFAll, aes(x = log10(築年数), y = log10(取引価格))) +
geom_point(aes(colour=面積m2), alpha=0.7) + ggtitle("築年数と価格") +
theme(plot.title = element_text(hjust = 0.5))
```

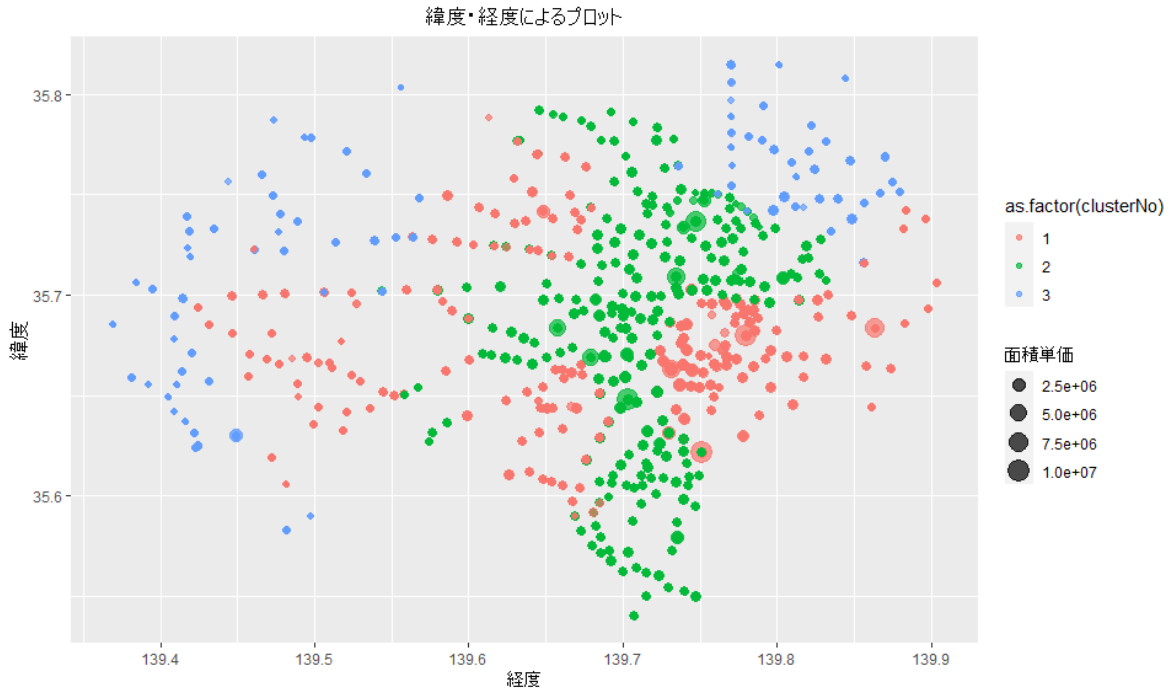


面積単価を求める

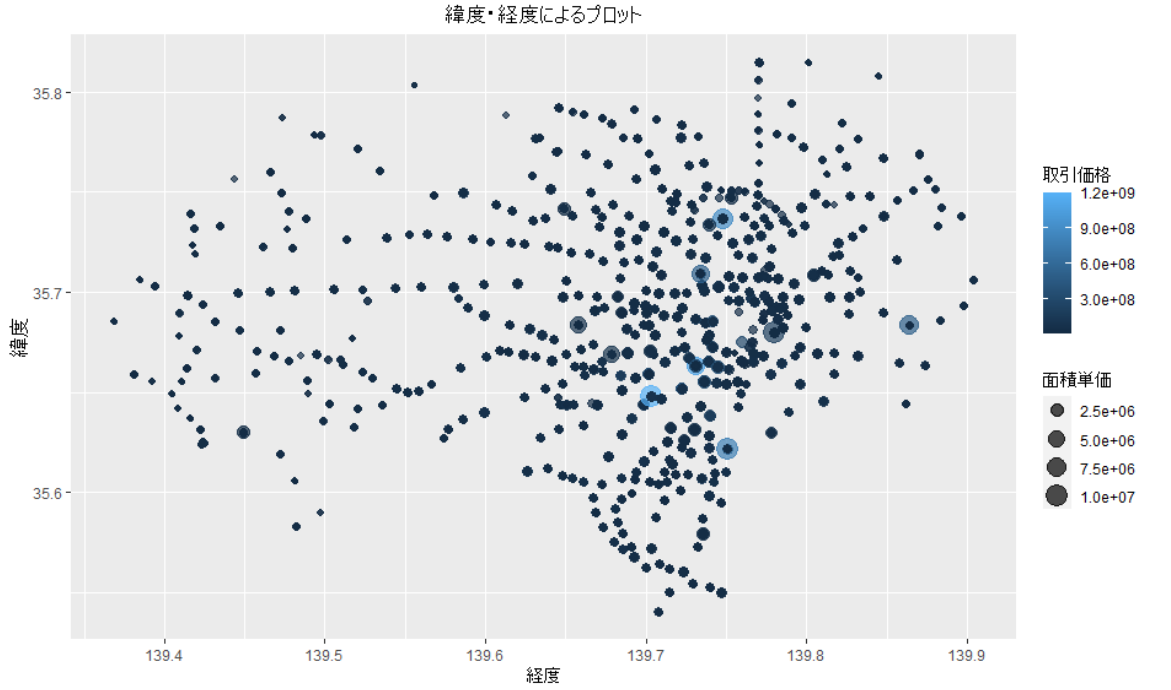
```
DFAll$面積単価 <- DFAll$取引価格/DFAll$面積m2
```

緯度・経度によるプロット

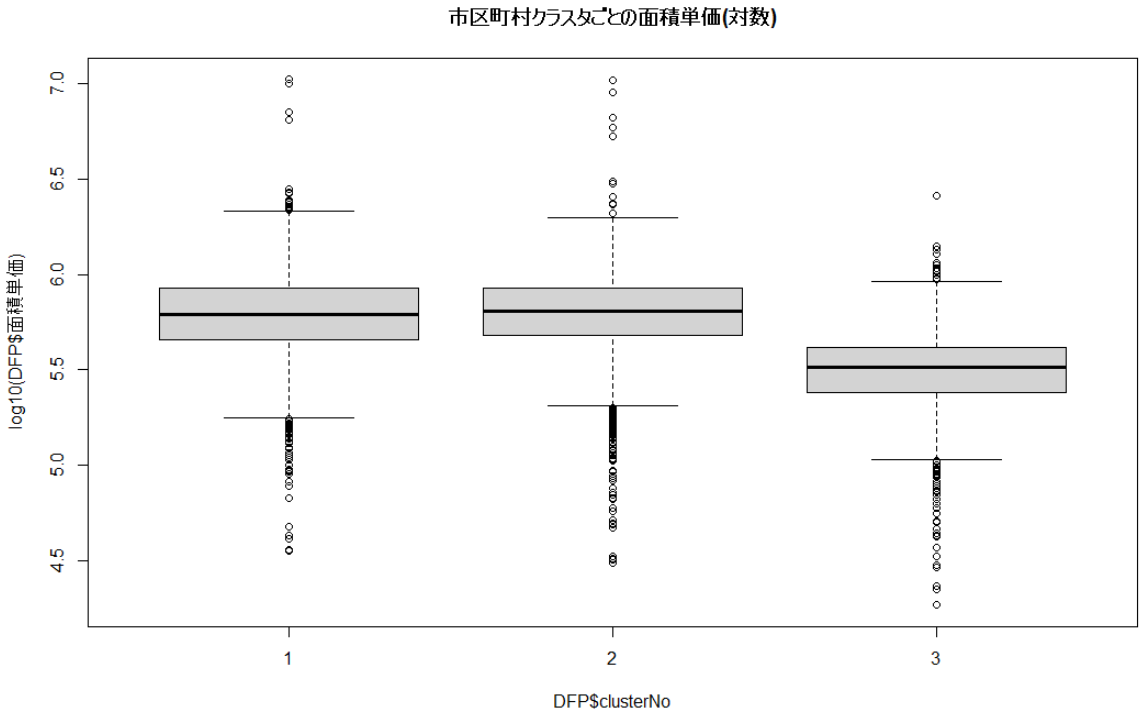
```
ggplot(DFAll, aes(x = Longit, y = Latit)) +
geom_point(aes(colour=as.factor(clusterNo), size=面積単価), alpha=0.7) +
ggtitle("緯度・経度によるプロット") + xlab("経度") + ylab("緯度") + theme(plot.title
= element_text(hjust = 0.5))
```




```
ggplot(DFAll, aes(x = Longit, y = Latit)) + geom_point(aes(colour=取引価格, size=面積単価), alpha=0.7) + ggtitle("緯度・経度によるプロット") + xlab("経度") + ylab("緯度") + theme(plot.title = element_text(hjust = 0.5))
```

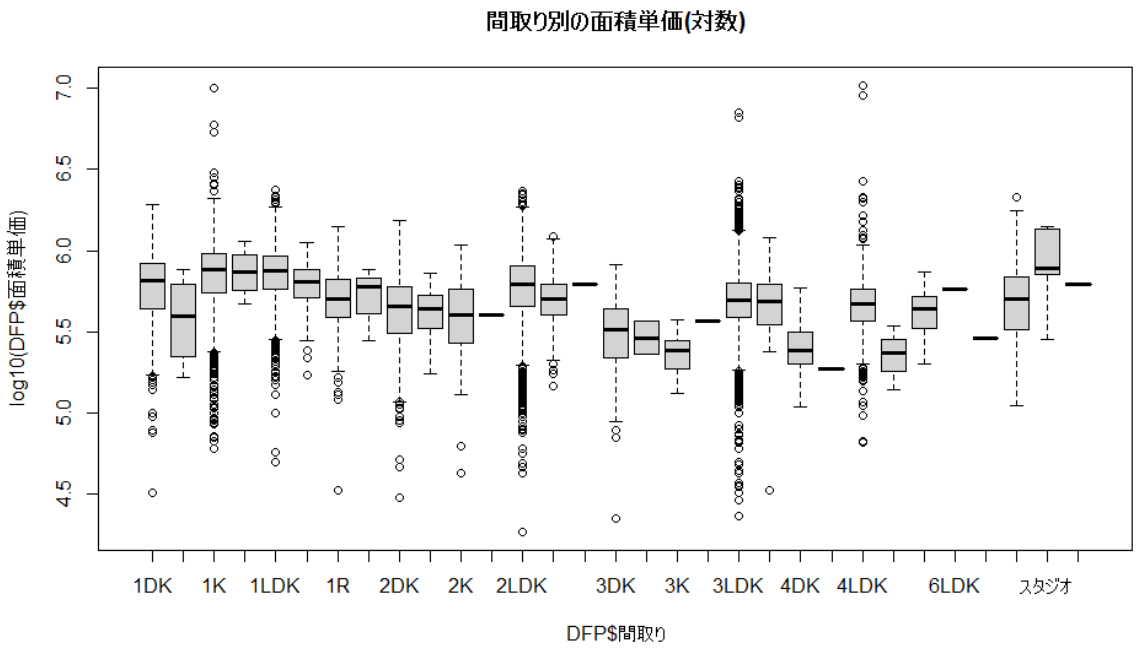



```
boxplot(log10(DFAll$面積単価) ~ DFAll$clusterNo, main="市区町村クラスタごとの面積単価(対数)")
```

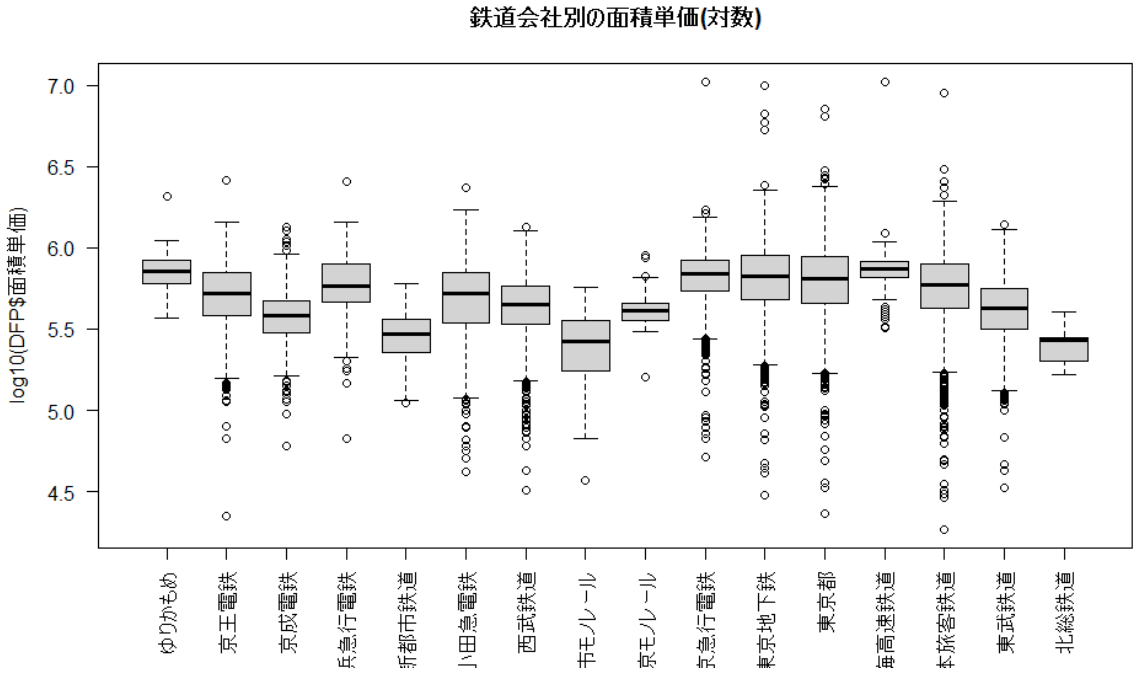



```
DFAll$間取り <- as.character(DFAll$間取り)
```

```
boxplot(log10(DFAll$面積単価) ~ DFAll$間取り, main="間取り別の面積単価(対数)")
```




```
boxplot(log10(DFAll$面積単価) ~ DFAll$RailCo, main="鉄道会社別の面積単価(対数)",
xlab="", las=2)
```


鉄道会社別の面積単価(対数)
log10(DFP$面積単価)
7.0
6.5
6.0
5.5
5.0
4.5
ゆりかもめ
京王電鉄
京成電鉄
京急行電鉄
新都市鉄道
小田急電鉄
西武鉄道
モノレール
京モノレール
京急行電鉄
東京地下鉄
東京都
首高速鉄道
東旅客鉄道
東武鉄道
北総鉄道