Mājas darbs 2-2: dimensiju redukcija

1. Uzdevums - dimensiju redukcija ar algoritmu pēc izvēles (t-SNE)

Datu apstrāde

Algoritms t-SNE izvēlēts kā pirmais no dotajiem un tāpēc, ka pieejama realizācija R platformā.

Vispirms ielasa datus:

```
library(foreign)
data <- read.arff("ionosphere.arff")
```

Veic priekšapstrādi, lai nogrieztu klases un pēc tam varētu tās grafiski attēlot kā punktu krāsas:

```
trim <- function(df){
  df[,1:length(df[1,])-1]
}

colors <- function(df) {
  sapply(df, function(x) {
    if (x == "b") {
      "blue"
    } else {
      "red"
    }})
}

trimmed <- trim(data)
col_row <- colors(data[,length(data[1,])])
```

Kods t-SNE aprēķinam un grafiskai attēlošanai:

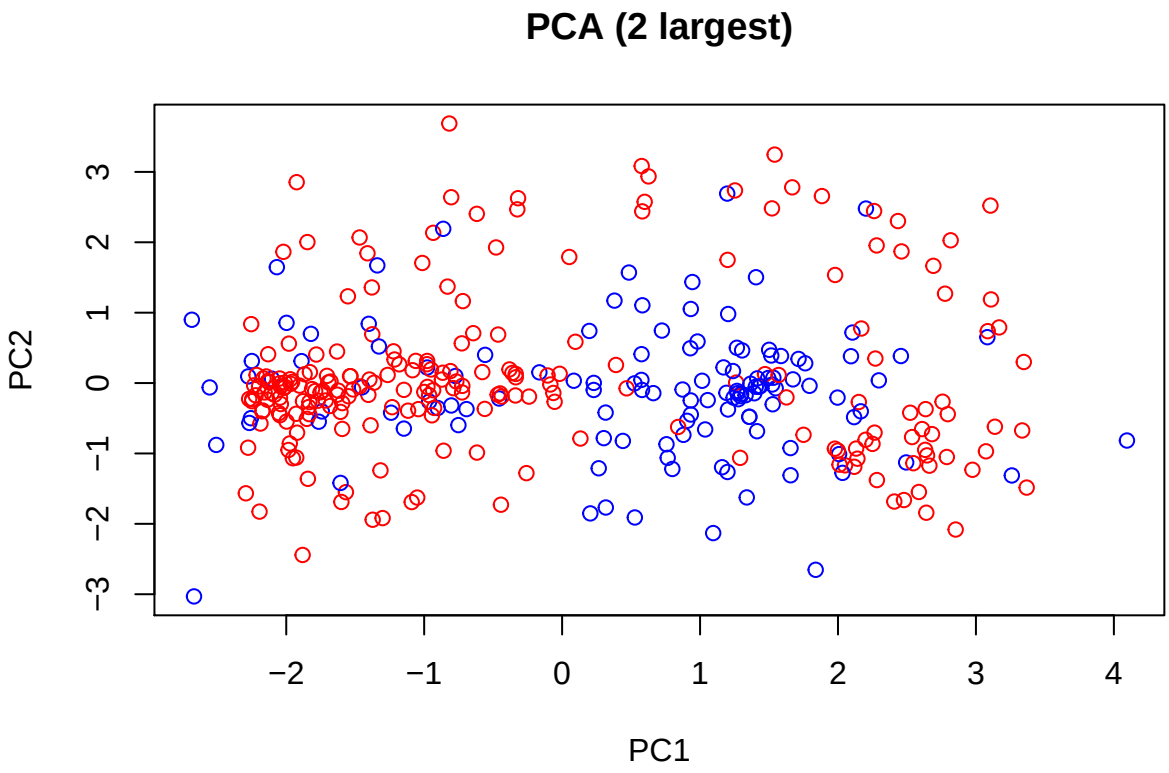
```
library(tsne)
plot.tsne <- function(df, perp=30, iter=400, class=NULL, plt=T, k=2, ret=F) {
  transformed <- tsne(df, k = k, perplexity = perp,
                      initial_dims = length(df[1,]), max_iter = iter)
  x <- transformed[,1]
  y <- transformed[,2]
  if (plt) {
    plot(x,y,col=class,xlab="comp1",
         ylab="comp2",
         main=sprintf("t-SNE perplexity = %d iter = %d", perp, iter))
  }
  if (ret) {
    transformed
  }
}
```

Kods PCA aprēķinam un attēlošanai (salīdzināšanas mērķiem):

```
plot.pca <- function(df, class){
  pca <- prcomp(df)
  plot(pca$x[,1:2],col=class,main="PCA (2 largest)")
  s <- summary(pca)
  s$importance[2,1:2]
}
```

PCA ar divām galvenajām komponentēm, 2 galveno komponenšu dispersijas proporcija:

```
v<-plot.pca(trimmed, col_row)
```




```
v
```

```
##     PC1     PC2
## 0.31344 0.12272
```

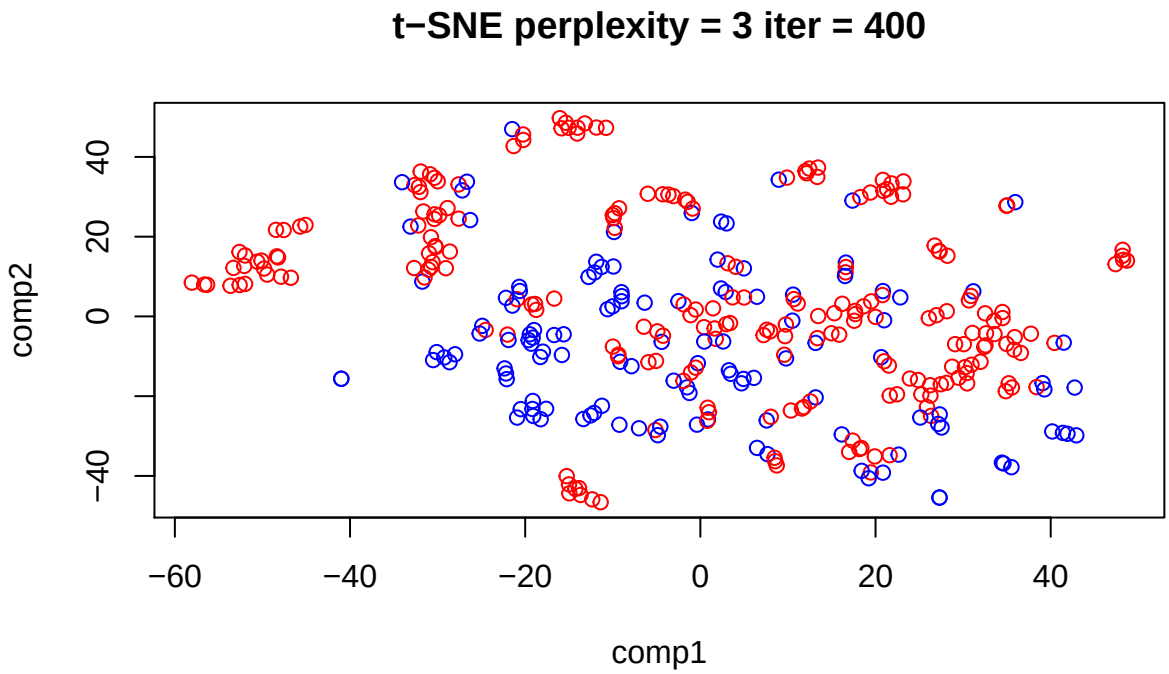
```
sum(v)
```

```
## [1] 0.43616
```

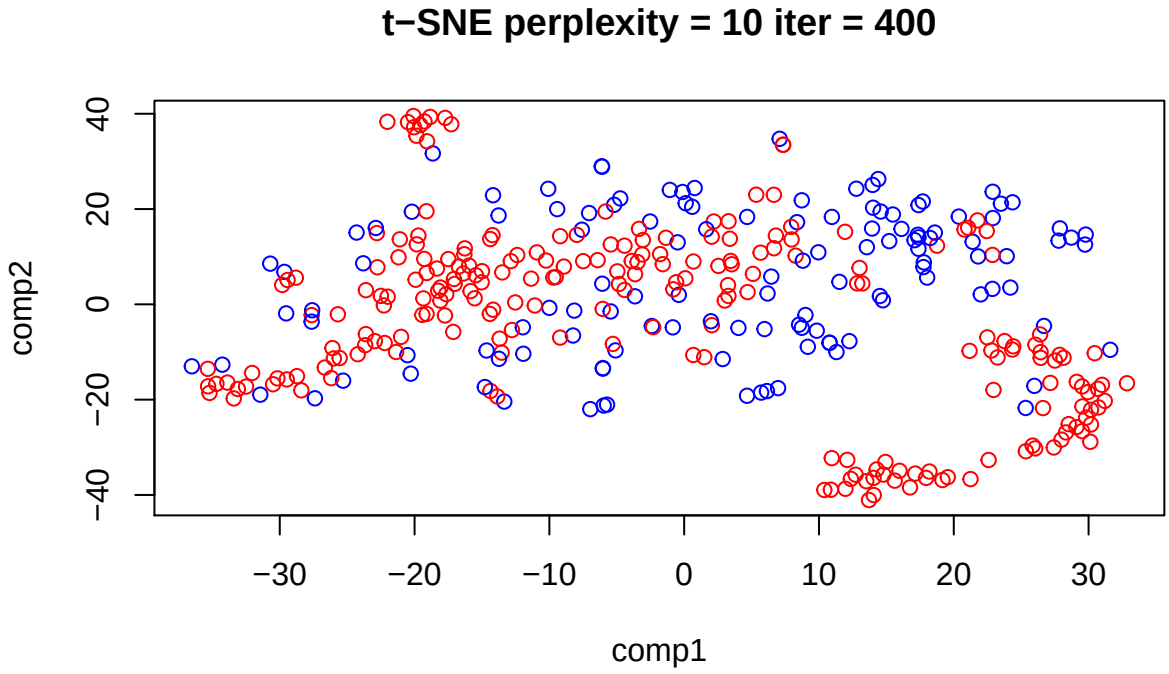
Šai t-SNE realizācijai pieejami divi galvenie hiperparametri - *perplexity* (ļoti aptuveni ekvivalents tuvāko kaimiņu skaitam) un *max_iter* - gradientu optimizācijas algoritma iterāciju skaits. Funkcija automātiski drukā *loss* vērtību pēc katrām 100 optimizācijas algoritma iterācijām un uzreiz redzams, ka visām izvēlētām *perplexity* vērtībām *loss* vērtība ir faktiski minimizēta jau pēc 200 iterācijām. Pēc noklusējuma iterāciju skaits ir 1000, šeit atstāts 400 un sīkāk nav pētīts, jo lielas izmaiņas nav novērotas. Ir pieejams arī *min_cost* parametrs, kas ļauj pārtraukt darbu pie izvēlētas *loss* vērtības, taču atkarībā no *perplexity* izskatās, ka tām ir dažādas asimptotes - tāpēc mainot *perplexity* šis parametrs nav īpaši noderīgs.

Zīmējot grafikus ar *perplexity* vērtībām {3,10,20,30,50,100}:

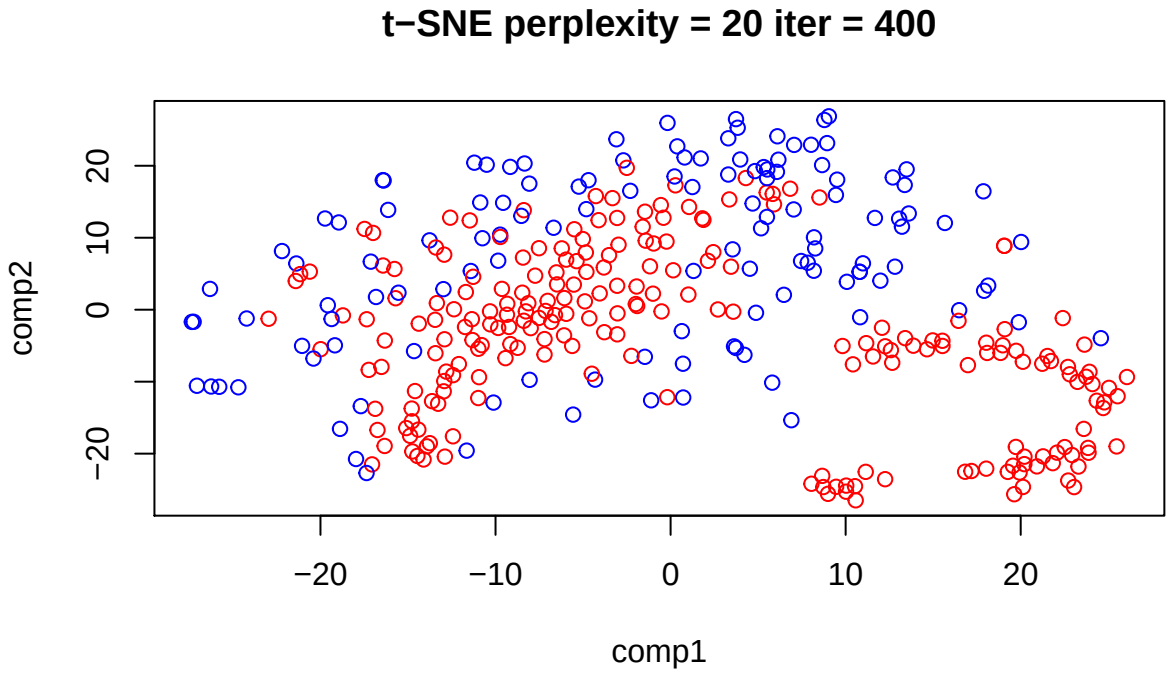
```
plot.tsne(trimmed,3,class=col_row)
```



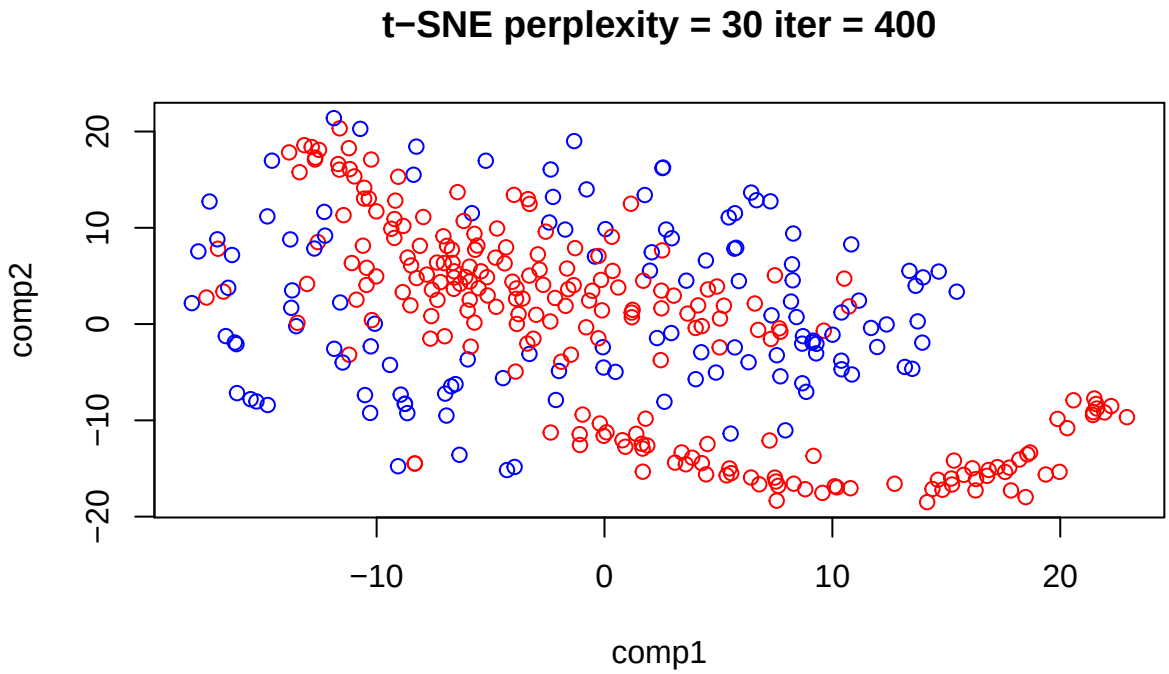

```
plot.tsne(trimmed,10,class=col_row)
```



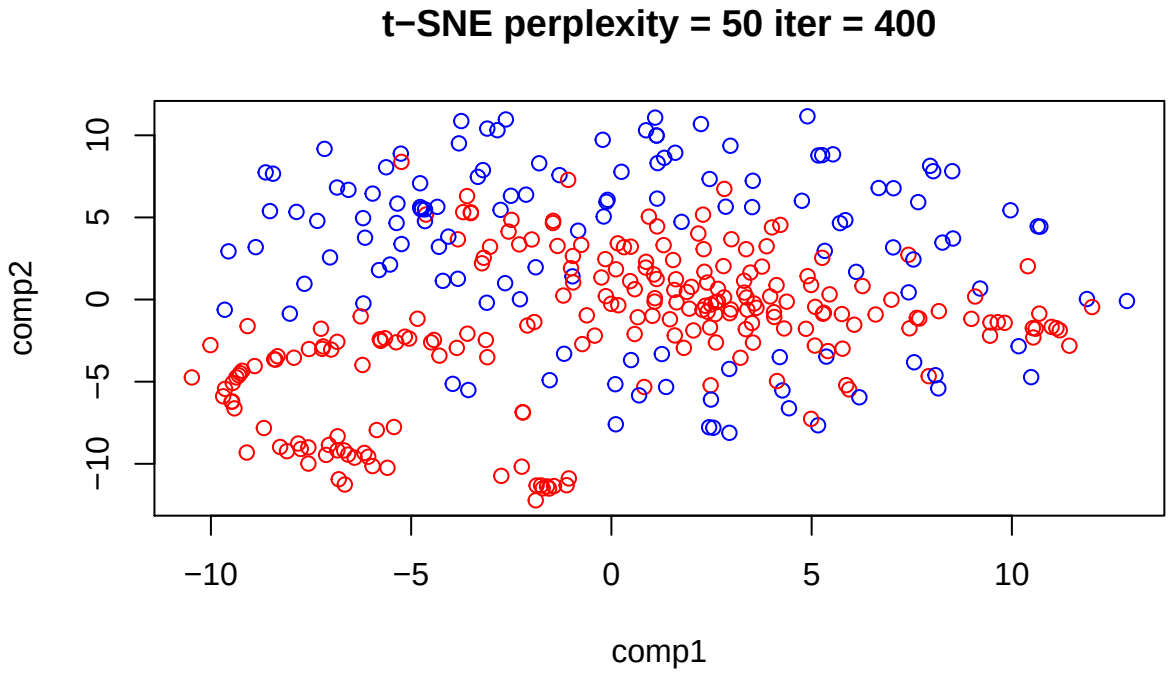
```
plot.tsne(trimmed,20,class=col_row)
```



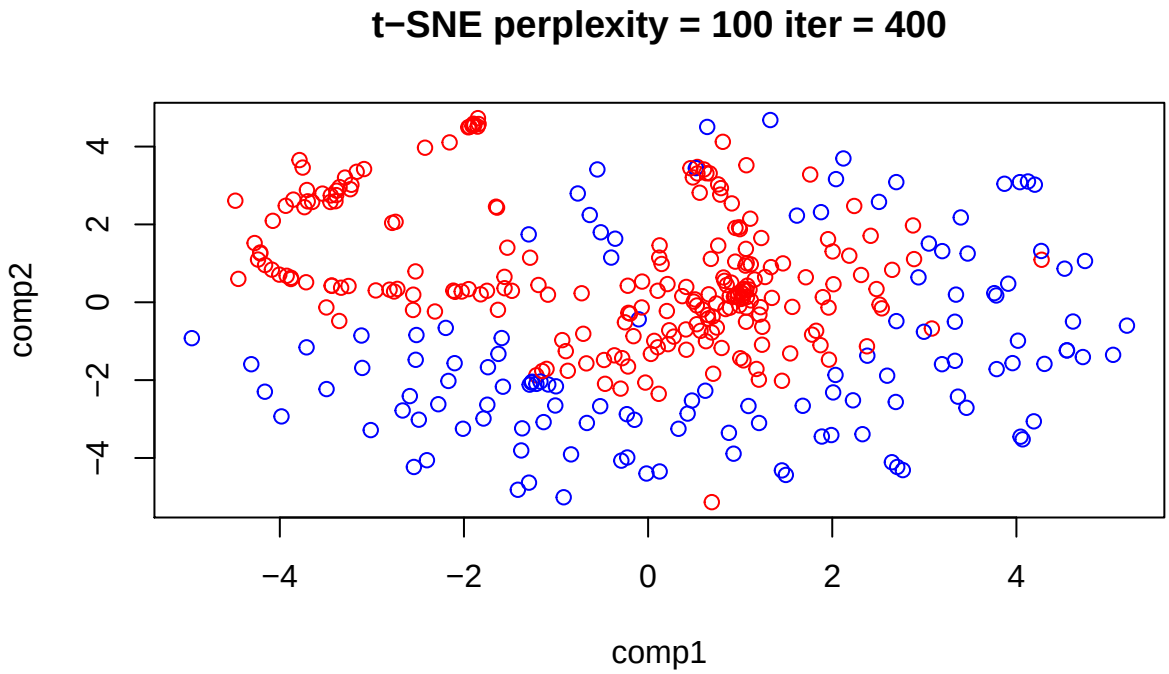

```
plot.tsne(trimmed,30,class=col_row)
```



```
plot.tsne(trimmed,50,class=col_row)
```




```
plot.tsne(trimmed,100,class=col_row)
```



Secinājumi

Ar visām *perplexity* vērtībām, izņemot 3 un 100, var novērot, ka sarkanā klase veido divus klasterus - vienu izstieptu un ārpus zilo punktu mākoņa, otru bez nekādas izteiktas formas taču ar zilās klases punktiem visapkārt. Pie vērtības 3 pēc dimensiju redukcijas iegūtais punktu mākonis veido tikai nelielus lokālus klasterus, kas škietami nejauši izkaisīti plaknē. Pie vērtības 100, abi sarkanās klases klasteri apvienojušies izliektas "T" formas struktūrā, kam apkārt ir zilās klases punkti. Zīmīgi, ka lokāli klases nav sevišķi sajauktas, taču izdarīt kaut kādu nozīmīgus spriedumus par struktūru bez zināšanām par datu kopu nav iespējams - un bez klašu anotācijas apšaubāmi, vai kāds klasterizācijas algoritms varētu konstatēt divu klašu klātbūtni.

Ar PCA iegūtais attēls ietver tikai 43% kopējās dispersijas, taču arī tajā novērojams kas līdzīgs t-SNE atrastajām struktūrām.

2. Uzdevums - dimensiju redukcija ar algoritmu pēc izvēles (???)

```
library(reticulate)
use_python("/usr/bin/python3")
```

```
print(["test" for x in range(5)])
```

```
## ['test', 'test', 'test', 'test', 'test']
```