LAMPIRAN

Lampiran 1 Script R

```
# # # # # # # #  Syntaks mengolah data Bioinformatika Microarray # # # # # # # #
# PACKAGES YANG DIGUNAKAN UNTUK MENGOLAH DATA BIOINFORMATIKA

library(affy)

library(GEOquery)

library(Biobase)

library(simpleaffy)

library(affyPLM)

library(hgu133plus2.db)

library(hgu133acdf)

library(hgu133a.db)

library(hgu133plus2cdf)

library(genefilter)

library(AnnotationDbi)


########################### INPUT DATA #################

## Memanggil file Cell dari GSE28160_RAW

## library(affy)

gse<- list.celfiles("C:/Users/ilham/Documents/GSE28160_RAW", full.names=T)

affy.data = ReadAffy(filenames=gse)

affy.data

## GetGEO Online GSE28160, untuk pheno data

## library(GEOquery)

gset <- getGEO(GEO="GSE28160", GSEMatrix =TRUE)

data.gse <- exprs(gset[[1]])
```

```
data.gse
##BOxplot
boxplot(data.gse)


################ MEMBACA DATA BIOINFORMATIKA ###########
# Pheno data untuk melihat deskripsi dari sample yang ada
# get pheno data dari GetGEO GSE Matrix
pheno <- pData(phenoData(gset[[1]]))
class(pheno)
row.names(pheno)
colnames(pheno)
pheno$characteristics_ch1.2
dim(pheno)
head(pheno)
setwd('E:/SKRIPSI/data data/gse 28160 Antiretroviral therapy effect on brain of patients with HIV-associated neurocognitive disorders')
#save pheno in txt file
WriteMatrixToFile <- function(tmpMatrix, tmpFileName, blnRowNames, blnColNames)
{
  output <- file(tmpFileName, "at")
  utils::write.table(tmpMatrix, output, sep = "\t", quote = FALSE,
                     row.names = blnRowNames, col.names = blnColNames)
  close(output)
}
WriteMatrixToFile(tmpMatrix=pheno, tmpFileName="phenoku.txt",
                  blnRowNames=TRUE, blnColNames=TRUE)
getwd()
```

```
## PHENO DATA
p1 <- read.AnnotatedDataFrame(file.path("phenoku.txt"), sep = "\t", header = TRUE)
class(p1)
head(p1)
phenoData(affy.data) <- p1
View(pheno)
pheno

#get pheno data dari affy data
pheno4 =        pData(phenoData(affy.data))
pheno4
varLabels(phenoData(gset[[1]]))

# Untuk melihat pheno dari characteristic 1.2 yaitu deases state
peset4 <- pheno4$characteristics_ch1.2
peset4

# pie chart yang terkena penyakit
des <-table(pheno4$characteristics_ch1.2)
percent <- round(des/sum(des)*100)
des <- as.data.frame(des)
lbls <- paste(des$Var1,'-',percent, '%', sep='')
pie(des$Freq, label= lbls, col= c('green','blue','orange','red'))

# pie chart pengobatan
des <-table(pheno4$characteristics_ch1)
percent <- round(des/sum(des)*100)
```

```
des <- as.data.frame(des)

lbls <- paste(des$Var1,'-',percent, '%', sep='')

pie(des$Freq, label= lbls, col= c('green','blue','orange','red'))


##################### PREPROCESSING #########################

# untuk function threestep preprocessing data

# default dari R untuk background RMA2, normalize quantile dan summary median polish

# threestep untuk data berbentuk affybatch, makanya digunakan data affy.data

library(affyPLM)

dchip <- threestep(affy.data)

# exprs digunakan untuk merubah data affybatch menjadi data matrix/data frame

dchip4 <-exprs(dchip)

class(dchip)

boxplot(dchip4)

head(dchip)

dim(dchip)


###################### FILTERING DATA ######################

# Filtering menggunaakn fungsi nsFilter

# Defaul dari R untuk entrez, dupentrez, varian/sd

# nsFilter digunakan untuk data berbentuk expression set yaitu data affybatch yang sudah di exprs

## library(genefilter)

filter4 <- nsFilter(dchip, require.entrez =T, remove.dupEntrez = T,var.func = IQR,

                    feature.exclude = "^AFFX")

filter4

log4 <-filter4$filter.log

eset4 <- filter4$eset
```

```
class(eset4)

## dimensi menjadi 10093 x 35

## Membuat data microarray menjadi matrix/data frame dengan exprs

## Untuk dilanjutkan ketahap Feature selection dengan multtest

data4 <- exprs(eset4)

dim(data4)

class(data4)

head(data4)


# vector yg terdiri dari klasifikasi binary Conrol NO  HIV dan HIV (PUNYA ILHAM)

data4cl <- c(0,0,0,0,0,0,0,0,0,

            1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

length(data4cl)

class(data4cl)

data4fac <- factor(data4cl,levels=0:1, labels= c("NO HIV","HIV"))

data4fac


########################## 5. FEATURE SELECTION ##################################

## data yang digunakan adalah data matrix atau data frame

## baris berupa nama gen dan kolom merupakan nama sampel

## Anova untuk lebih 2 kelas, ttest untuk 2 kelas

## Filtering dengan t.test

library(multtest)

data4ttest <- mt.teststat(data4,data4cl,test="t")

class(data4ttest)

length(data4ttest)

qqnorm(data4ttest)
```

```
qqline(data4ttest)

# Adjusting p-value (untuk melihat p-value yg sesuai dan tidak sesuai)
rawp4 = 2 * (1 - pnorm(abs(data4ttest)))
procedure4 = c("Bonferroni", "Holm", "Hochberg", "BH", "BY")
adjusted4 = mt.rawp2adjp(rawp4, procedure4)
p4adjp <- adjusted4$adjp[,]
data4adjp <- p4adjp[order(adjusted4$index), ]
class(data4adjp)
dim(data4adjp)
head(data4adjp)

# mengambil data p-value adjp (p-value adjusted)
ffs4 <- data4adjp[,1]
ffs4
class(ffs4)
length(ffs4)

#bainding data hasil filtering dengan p-value adjusted
data4pa <- data.frame(data4, ffs4)
row.names(data4pa) <- row.names(data4)
class(data4pa)
head(data4pa)
dim(data4pa)

library(dplyr)
#datadatarawpfilter <- filter(datarawp, ffs < 0.0000001)
#class(datarawpfilter)
```

```
## dimensi menjadi 26 x 41
data4analisis <- subset(data4pa, ffs4 < 0.0000001)
rownames(data4analisis)
class(data4analisis)
dim(data4analisis)
head(data4analisis)
## mendefinisikan data baru setelah filter
data4new <- data4analisis[,1:35]
head(data4new)
dim(data4new)
colnames(data4new)

########################## 6. DATA SIAP ANALISIS ##############################
library(e1071)
library(pROC)
data4analisisfinal   = as.data.frame (t((data4new)))
head(data4analisisfinal)
data4Y = as.factor(data4cl)
length(data4Y)
data4use = as.data.frame(cbind(data4analisisfinal,data4Y))
dim(data4use)

######## SVM kernel Linier ########
set.seed(978)
ratio = 8/10
train = sample(length(data4Y), size = floor(ratio*length(data4Y)))
datatrain<- data4use[train,]
```

```
datatest <- data4use[-train,]

dim(datatrain)

dim(datatest)

tunaslin <- tune(svm, data4Y~. , data = datatrain, kernel="linear",types = "C-clasification",ranges= list( cost = c(0.1,0.01, 0.001,1 , 10 , 100)))

summary(tunaslin)

plot(tunaslin)

model <- svm(data4Y~. , datatrain, kernel = "linear", cost=1,scale=F, types = "C-clasification",decision.value=T)

predictions <- predict(model, datatest)

table(predictions,datatest$data4Y)

mean(predictions == datatest$data4Y)

predictions <- predict(model, datatrain)

table(predictions,datatrain$data4Y)

mean(predictions == datatrain$data4Y)

#plot(model,datatrain, datatrain$gender~datatrain$ethnic, slice = list(ethnic = 4, gender = 4))


############ gen probe dengan weighted dari kernel linear #############

w <- t(model$coefs) %*% model$SV

w <- apply(w, 2, function(v){sqrt(sum(v^2))})

w <- sort(w, decreasing = T)

x<- as.data.frame(w)

View(x)

all(colnames(w)==colnames(datatrain))


#ROC

forauc <- as.numeric(predict(model,datatest))

auc <- multiclass.roc(datatest$data4Y,forauc)
```

```
auc$auc
plot.roc(auc[["rocs"]][[1]],print.auc = T,col="blue",print.auc.x = .3,
        print.auc.y = .6,legacy.axes = TRUE,lty=4,main="ROC curve")

######### melihat gennya saja ##############
probe4 <-substring(as.character(head(rownames(x),n=10)),2)
View(probe4)



############## kernel Polynomial ##################
set.seed(978)
ratio = 8/10
train = sample(length(data4Y), size = floor(ratio*length(data4Y)))
datatrain<- data4use[train,]
dim(datatrain)
datatest <- data4use[-train,]
dim(datatest)

#tuning (best parameter)
tuning <- tune(svm, data4Y~. , data = datatrain, kernel="polynomial",
            types = "C-clasification",ranges= list( cost = c(100,200,300,400,500)))
summary(tuning)

#membangung model dengan best parameter
model <- svm(data4Y~. , datatrain, kernel = "polynomial", degree=2,cost=100,
types = "C-clasification",decision.value=T)
prediksi <- predict(model, datatest)
table(prediksi,datatest$data4Y)
```

```
mean(prediksi == datatest$data4Y)

prediksi <- predict(model, datatrain)

table(prediksi,datatrain$data4Y)

mean(prediksi == datatrain$data4Y)

#plot(model,datatrain, datatrain$gender~datatrain$ethnic, slice = list(ethnic = 4,
gender = 4))

#weighted

w <- t(model$coefs) %*% model$SV

w <- apply(w, 2, function(v){sqrt(sum(v^2))})

w <- sort(w, decreasing = T)

x<- as.data.frame(w)

View(x)

all(colnames(w)==colnames(datatrain))

#ROC

forauc <- as.numeric(predict(model,datatest))

auc <- multiclass.roc(datatest$data4Y,forauc)

auc$auc


plot.roc(auc[["rocs"]][[1]],print.auc = T,col="blue",print.auc.x = .3,

        print.auc.y = .6,legacy.axes = TRUE,lty=4,main="ROC curve")


############# SVM kernel RBF #################

set.seed(978)

ratio = 8/10

train = sample(length(data4Y), size = floor(ratio*length(data4Y)))

datatrain<- data4use[train,]

dim(datatrain)

datatest <- data4use[-train,]
```

```
dim(datatest)


#(best parameter)
tun <- tune(svm, data4Y~. , data = datatrain,
            kernel="radial",types = "C-clasification",
            ranges= list( cost = c(0.1,0.01, 0.001,1 , 10 , 100), gamma=c(1,2,3,4,5)))
summary(tun)
#membangung model dengan best parameter
model <- svm(data4Y~. , datatrain, kernel = "radial",
             ost=0.1, gamma=1)
predictions <- predict(model, datatest)
table(predictions,datatest$data4Y)
mean(predictions == datatest$data4Y)
predictions <- predict(model, datatrain)
table(predictions,datatrain$data4Y)
mean(predictions == datatrain$data4Y)
#plot(model,datatrain, datatrain$gender~datatrain$ethnic, slice = list(ethnic = 4,
gender = 4))
#weighted
w <- t(model$coefs) %*% model$SV
w <- apply(w, 2, function(v){sqrt(sum(v^2))})
w <- sort(w, decreasing = T)
x<- as.data.frame(w)
View(x)
all(colnames(w)==colnames(datatrain))
#ROC
forauc <- as.numeric(predict(model,datatest))
auc <- multiclass.roc(datatest$data4Y,forauc)
```

```
auc$auc
plot.roc(auc[["rocs"]][[1]],print.auc = T,col="blue",print.auc.x = .3,
        print.auc.y = .6,legacy.axes = TRUE,lty=4,main="ROC curve")

############### SVM sigmoid #################
set.seed(978)
ratio = 8/10
train = sample(length(data4Y), size = floor(ratio*length(data4Y)))
datatrain<- data4use[train,]
dim(datatrain)
datatest <- data4use[-train,]
dim(datatest)

#(best parameter)
tun <- tune(svm, data4Y~. , data = datatrain,
           kernel="sigmoid",types = "C-clasification",
           ranges= list( cost = c(0.1,0.01, 0.001,1 , 10 , 100), gamma=c(1,2,3,4,5)))
summary(tun)
#membangung model dengan best parameter
model <- svm(data4Y~. , datatrain, kernel = "sigmoid",
            ost=0.1, gamma=1)
predictions <- predict(model, datatest)
table(predictions,datatest$data4Y)
mean(predictions == datatest$data4Y)
predictions <- predict(model, datatrain)
table(predictions,datatrain$data4Y)
mean(predictions == datatrain$data4Y)
```

```
#ROC
forauc <- as.numeric(predict(model,datatest))
auc <- multiclass.roc(datatest$data4Y,forauc)
auc
auc$auc
plot.roc(auc[["rocs"]][[1]],print.auc = T,col="blue",print.auc.x = .3,
        print.auc.y = .6,legacy.axes = TRUE,lty=4,main="ROC curve")
```

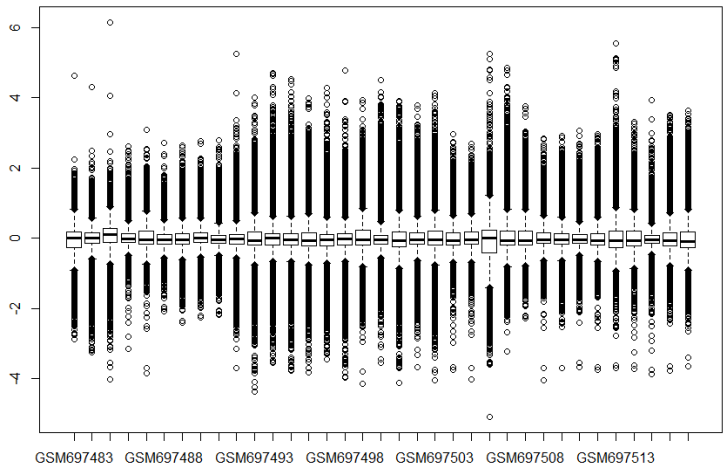
Lampiran 2 Data

Pasien/Sampel	1007_s_at	1053_at	…	1552278_a_at
GSM697483	-1,0478544	0.134922030	…	0.142282490
GSM697484	0.085256580	0.058036804	…	0.124458310
GSM697485	0.351787570	-0.054499150	…	0.103277680
GSM697486	0.002227783	-0.146765710	…	-0.091307640
GSM697487	0.009019852	0.276632800	…	-0.001266003
GSM697488	0.396997450	-0.344458580	…	-0.117162230
GSM697489	0.505466460	0.230651860	…	-0.101543430
GSM697490	0.156991960	-0.097144130	…	-0.057471275
GSM697491	0.243683820	-0.057373047	…	-0.001266003
GSM697492	0.517208100	0.354259970	…	-0.090759280
GSM697493	0.954639430	0.234135630	…	-0.094734670
GSM697494	0.408341400	0.020513535	…	-0.001266003
GSM697495	0.395303730	0.041858673	…	-0.153636460
GSM697496	0.618742940	0.439848900	…	-0.039492130
GSM697497	1,4749203	0.206989770	…	0.128879070
GSM697498	0.763892200	0.086241245	…	0.044453144
GSM697499	1,2126408	0.586079100	…	0.156159880
GSM697500	0.128031730	0.209784510	…	0.217408660
GSM697501	0.462210660	0.404117100	…	-0.178968900
GSM697502	0.670137400	0.817574000	…	0.034820080
GSM697503	1,02265	0.427543160	…	-0.057911396
GSM697504	0.455203060	0.138401030	…	-0.031758310
GSM697505	0.217674260	0.321174620	…	0.261491300
GSM697506	-1,4722023	-0.111159325	…	0.184110160
GSM697507	1,3765421	0.719270200	…	-0.051078320
GSM697508	1,1801825	0.570490840	…	-0.167957300
GSM697509	0.673688900	0.354915620	…	0.180250640
GSM697510	0.374373440	-0.000222000	…	0.031305313
GSM697511	0.442090030	-0.330330850	…	-0.040653230
GSM697512	0.026637077	0.190008160	…	0.025387287
GSM697513	0.815703400	0.479506500	…	-0.097828390
GSM697514	0.496749880	0.351761820	…	-0.155004020
GSM697515	0.295758250	0.335460200	…	-0.025104523
GSM697516	0.428554530	0.155141830	…	0.037294388
GSM697517	0.585118300	0.288979530	…	0.073505880

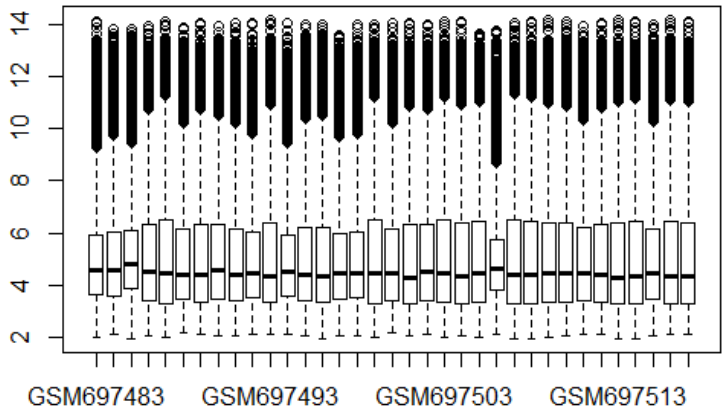
Lampiran 3 Pheno Data

```
 [1] "title"                  "geo_accession"            "status"
 [4] "submission_date"        "last_update_date"         "type"
 [7] "channel_count"          "source_name_ch1"          "organism_ch1"
[10] "characteristics_ch1"    "characteristics_ch1.1"    "characteristics_ch1.2"
[13] "molecule_ch1"           "extract_protocol_ch1"     "label_ch1"
[16] "label_protocol_ch1"     "taxid_ch1"                "hyb_protocol"
[19] "scan_protocol"          "description"              "data_processing"
[22] "platform_id"            "contact_name"             "contact_email"
[25] "contact_phone"          "contact_laboratory"       "contact_department"
[28] "contact_institute"      "contact_address"          "contact_city"
[31] "contact_state"          "contact_zip/postal_code"  "contact_country"
[34] "supplementary_file"     "data_row_count"           "disease state:ch1"
[37] "tissue:ch1"             "treatment:ch1"
```

Lampiran 4 Boxplot sebelum *pre-processing*



Lampiran 5 Boxplot sesudah *pre-processing*



Lampiran 6 Hasil analisis SVM

```
> summary(tunaslin)

Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost
    1

- best performance: 0.03333333

- Detailed performance results:
   cost      error dispersion
1 1e-01 0.06666667  0.1405457
2 1e-02 0.11666667  0.1932503
3 1e-03 0.25000000  0.2389535
4 1e+00 0.03333333  0.1054093
5 1e+01 0.03333333  0.1054093
6 1e+02 0.03333333  0.1054093

> plot(tunaslin)
> model <- svm(data4Y~. , datatrain, kernel = "linear", cost=1,scale=F, types = "C-clasificati
on",decision.value=T)
> predictions <- predict(model, datatest)
> table(predictions,datatest$data4Y)

predictions 0 1
          0 2 1
          1 0 4
> mean(predictions == datatest$data4Y)
[1] 0.8571429
```