Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Tugas 1 :

Screen shot code program setelah line 34:

```
34 #confusion matrix
35 table(predict=ypred, truth=testdat$y)
36
37 tbl = table(predict=ypred, truth=testdat$y)
38
39 tp = as.vector(tbl[4])
40 tp
41 fn = as.vector(tbl[3])
42 fn
43 fp = as.vector(tbl[2])
44 fp
45 tn = as.vector(tbl[1])
46 tn
47
48 sum_all = tn + fn + fp + tp
49 sum_all
50
51 accuracy = (tp + tn )/sum_all
52 accuracy
53 error_rate = (fp + fn)/sum_all
54 error_rate
55 sensitivity = tp/(tp + fn)
56 sensitivity
57 specificity = tn/(fp + tn)
58 specificity
59 precision = tp/(tp + fp)
60 precision
61 recall = tp/(tp + fn)
62 recall
63 f_measure = (2*precision*recall)/(precision+recall)
64 f_measure
```

Berikut ini adalah penjelasan dari setiap kode program.

Membangun dataset yang tergolong ke dalam dua kelas. Didefinisikan 20 data dengan 10 data berkelas positif (1) dan 10 data berkelas negatif (-1).

```
1 # Tugas 1
2 # Construct a linearly seperable dataset on 2-D plane
3 set.seed(100)
4 x=matrix(rnorm(20*2), ncol=2)
5 y=c(rep(-1,10),rep(1,10))
6 x[y==1,]=x[y==1,]+1
```

Plot menunjukkan bahwa data yang kita *generate* di *step* sebelumnya tidak terpisah secara linear.

```
8 plot(x, col=(3-y))
```

Selanjutnya, kita simpan data tersebut dalam *data frame* yang secara sederhana kita sebut *data matrix.*

```
10 dat=data.frame(x=x,y=as.factor(y))
```

Memanggil *library* e1071 yang merupakan *library* libsvm di R.

```
14 library('e1071')
```

Memanggil fungsi **svm()** dan melakukan pemilihan *kernel function* dan *error parameter* C (*cost function*). Argumen **scale=FALSE** berfungsi untuk memberitahu fungsi **svm()** untuk tidak menskalakan masing-masing fitur supaya memiliki mean nol atau standar deviasi satu.

```
16 svmfit=svm(y ~ ., data=dat, kernel='linear', cost=10, scale=FALSE)
17 plot(svmfit,dat)
```

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Mengidentifikasi identitas support vector

```
18 svmfit$index
```

Hasilnya:

```
[1]  4  8 13 15 19
```

Menampilkan informasi ringkas mengenai model yang telah dibuat dengan menggunakan summary().

```
19 summary(svmfit)
```

Hasilnya:

```
Call:
svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10
      gamma:  0.5

Number of Support Vectors:  5

 ( 2 3 )


Number of Classes:  2

Levels:
 -1 1
```

Menemukan *optimal tuning parameter*

```
22 set.seed (1)
```

Menerapkan *cross-validation* dengan menggunakan fungsi *tune()* untuk melakukan *10-fold cross-validation* pada sebuah *set model* dengan cara membandingkan beberapa *model* yang diperoleh dengan nilai *cost* yang berbedabeda

```
23 tune.out=tune(svm,y ~ .,data=dat, kernel="linear", ranges = list(cost=c(0.001, 0.01, 0.1, 1,5, 10, 100)))
```

Memilih dan menyimpan model yang terbaik:

```
24 bestmod=tune.out$best.model
```

Menampilkan informasi ringkas mengenai model yang telah dibuat dengan menggunakan summary().

```
25 summary(bestmod)
```

Hasilnya:

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

```
Call:
best.tune(method = svm, train.x = y ~ ., data = dat, ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  5
      gamma:  0.5

Number of Support Vectors:  7

 ( 3 4 )


Number of Classes:  2

Levels:
 -1 1
```

Mendefininisikan *testing set menggunakan 20 testing set*

```
28  xtest=matrix(rnorm(20*2), ncol=2)
29  ytest=sample(c(-1,1), 20, rep=TRUE)
30  xtest[ytest==1,]=xtest[ytest==1,]+1
31  testdat=data.frame(x=xtest, y=as.factor(ytest))
```

Menggunakan fungsi predict untuk memprediksi label kelas masing masing testing set.

```
33  ypred=predict(bestmod, testdat)
```

Menampilkan hasil prediksi terhadap 20 testing data dalam confusion matrix.

```
35  table(predict=ypred, truth=testdat$y)
```

Hasilnya :

```
       truth
predict -1  1
     -1 11  1
     1   0  8
```

Mendefenisikan *variable* tbl sebagai *confusion matrix.*
Mendefenisikan *variable* tp untuk memperoleh nilai *true positive* diperoleh dari nilai *truth* 1, dan *predict* 1 yaitu nilai pada kotak ke-4.
Mendefenisikan *variable* fn untuk memperoleh nilai *false negative* diperoleh dari nilai *truth* 1, dan *predict* -1 yaitu nilai pada kotak ke-3.
Mendefenisikan *variable* tp untuk memperoleh nilai *false positive* diperoleh dari nilai *truth* -1, dan *predict* 1 yaitu nilai pada kotak ke-2.
Mendefenisikan *variable* tp untuk memperoleh nilai *true neagative* diperoleh dari nilai *truth* 1, dan *predict* 1 yaitu nilai pada kotak ke-1.
Mendefenisikan *variable* sum_all untuk memperoleh nilai dari penjumlahan tp + tn + fp + fn.

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

```
tbl = table(predict=ypred, truth=testdat$y)

tp = as.vector(tbl[4])
tp

fn = as.vector(tbl[3])
fn

fp = as.vector(tbl[2])
fp

tn = as.vector(tbl[1])
tn

sum_all = tn + fn + fp + tp
sum_all
```

Hasilnya:

```
> tbl = table(predict=ypred, truth=testdat$y)
>
> tp = as.vector(tbl[4])
> tp
[1] 8
>
> fn = as.vector(tbl[3])
> fn
[1] 1
>
> fp = as.vector(tbl[2])
> fp
[1] 0
>
> tn = as.vector(tbl[1])
> tn
[1] 11
>
> sum_all = tn + fn + fp + tp
> sum_all
[1] 20
```

Menghitung *accuracy*, *error rate*, *sensitivity*, *specificity*, *precision*, *recall*, dan *f_measure:*

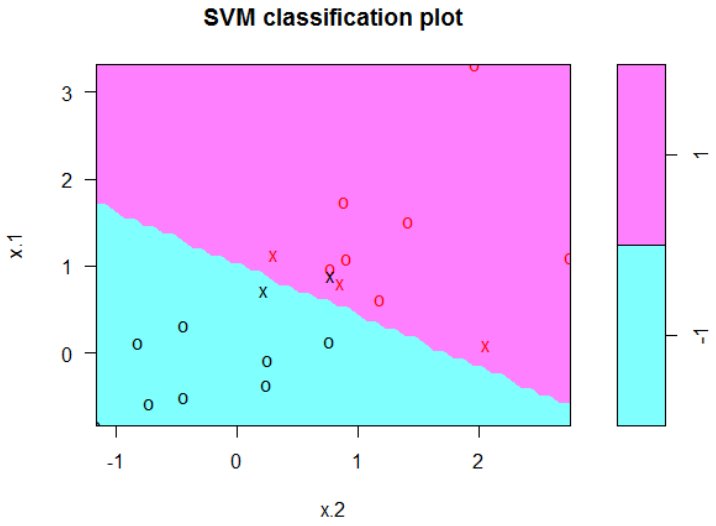
```
accuracy = (tp + tn )/sum_all
accuracy

error_rate = (fp + fn)/sum_all
error_rate

sensitivity = tp/(tp + fn)
sensitivity

specificity = tn/(fp + tn)
specificity

precision = tp/(tp + fp)
precision

recall = tp/(tp + fn)
recall

f_measure = (2*precision*recall)/(precision+recall)
f_measure
```

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Hasilnya:

```
> accuracy = (tp + tn )/sum_all
> accuracy
[1] 0.95
>
> error_rate = (fp + fn)/sum_all
> error_rate
[1] 0.05
>
> sensitivity = tp/(tp + fn)
> sensitivity
[1] 0.8888889
>
> specificity = tn/(fp + tn)
> specificity
[1] 1
>
> precision = tp/(tp + fp)
> precision
[1] 1
>
> recall = tp/(tp + fn)
> recall
[1] 0.8888889
>
> f_measure = (2*precision*recall)/(precision+recall)
> f_measure
[1] 0.9411765
```

Hasil visualisasi SVM



Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Tugas 2

Screen shot code program dengan menggunakan 800 dataset:

```
# Construct a linearly seperable dataset on 2-D plane
set.seed(100)
x=matrix(rnorm(800*2), ncol=2)
y=c(rep(-1,400),rep(1,400))
x[y==1,]=x[y==1,]+1

plot(x, col=(3-y))

dat=data.frame(x=x,y=as.factor(y))

# Load the libsvm R interface
# Use Liblinear for very large problem
library('e1071')

svmfit=svm(y ~ ., data=dat, kernel='linear', cost=10, scale=FALSE)
plot(svmfit,dat)
svmfit$index
summary(svmfit)

# Find optimal tuning parameter
set.seed (1)
tune.out=tune(svm,y ~ .,data=dat, kernel="linear", ranges = list(cost=c(0.001, 0.01, 0.1, 1,5, 10, 100)))
bestmod=tune.out$best.model
summary(bestmod)

# Construct the best data
xtest=matrix(rnorm(200*2), ncol=2)
ytest=sample(c(-1,1), 20, rep=TRUE)
xtest[ytest==1,]=xtest[ytest==1,]+1
testdat=data.frame(x=xtest, y=as.factor(ytest))

ypred=predict(bestmod, testdat)
```

```
#confusion matrix
tbl = table(predict=ypred, truth=testdat$y)

tn = as.vector(tbl[1])
tn
fn = as.vector(tbl[3])
fn
fp = as.vector(tbl[2])
fp
tp = as.vector(tbl[4])
tp

sum_all = tn + fn + fp + tp
sum_all
accuracy = (tp + tn )/sum_all
accuracy
error_rate = (fp + fn)/sum_all
error_rate
sensitivity = tp/(tp + fn)
sensitivity
specificity = tn/(fp + tn)
specificity
precision = tp/(tp + fp)
precision
recall = tp/(tp + fn)
recall
f_measure = (2*precision*recall)/(precision+recall)
f_measure
```

Berikut ini adalah penjelasan dari kode program.

Membangun dataset yang tergolong ke dalam dua kelas. Didefinisikan 800 data dengan 400 data berkelas positif (1) dan 400 data berkelas negatif (-1).

```
# Tugas 2
# Construct a linearly seperable dataset on 2-D plane
set.seed(100)
x=matrix(rnorm(800*2), ncol=2)
y=c(rep(-1,400),rep(1,400))
x[y==1,]=x[y==1,]+1
```

Plot menunjukkan bahwa data yang kita *generate* di *step* sebelumnya tidak terpisah secara linear.

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

```
plot(x, col=(3-y))
```

Selanjutnya, kita simpan data tersebut dalam *data frame* yang secara sederhana kita sebut *data matrix.*

```
dat=data.frame(x=x,y=as.factor(y))
```

Memanggil *library* e1071 yang merupakan *library* libsvm di R.

```
library('e1071')
```

Memanggil fungsi **svm()** dan melakukan pemilihan *kernel function* dan *error parameter* C (*cost function*). Argumen **scale=FALSE** berfungsi untuk memberitahu fungsi **svm()** untuk tidak menskalakan masing-masing fitur supaya memiliki mean nol atau standar deviasi satu

```
svmfit=svm(y ~ ., data=dat, kernel='linear', cost=10, scale=FALSE)
plot(svmfit,dat)
```

Mengidentifikasi identitas *support vector*

```
svmfit$index
```

Hasilnya:

```
  [1]   2   3   4   5   6   8  12  14  15  18  20  22  23  24  27  28  30  31  32  33  34  36  37  38  39  40  41  42  44  45  46  47  48  49  52  53  54  56
 [39]  57  58  59  61  62  63  64  66  67  68  70  71  74  76  79  86  87  88  89  90  91  92  94  95  96  98  99 100 102 108 109 115 117 120 124 125 128 130
 [77] 133 137 141 144 147 148 149 151 152 154 160 161 162 163 164 165 167 168 169 170 171 172 174 175 177 179 180 182 183 185 188 190 193 194 195 197 198 201
[115] 202 203 204 205 208 210 211 216 219 222 223 224 225 227 229 231 232 233 237 238 239 240 241 242 245 246 249 250 253 254 255 258 261 262 265 266 267 269
[153] 270 271 272 273 274 275 277 278 279 281 282 283 284 287 288 289 291 294 296 298 299 304 307 308 313 314 316 317 318 320 323 324 325 326 327 328 330 332
[191] 334 335 341 344 346 348 349 352 355 356 357 358 359 362 363 364 365 366 369 370 371 372 374 377 378 380 381 386 387 389 392 393 395 396 402 404 405 407
[229] 408 413 415 416 418 419 420 423 426 427 428 431 432 434 435 436 440 441 444 445 447 448 449 450 451 452 454 455 459 460 462 463 464 465 466 468 469 472
[267] 474 475 477 480 483 484 485 486 487 488 491 494 495 496 497 498 499 500 501 502 503 504 505 506 508 509 511 514 518 519 521 522 523 526 529 533 534 536
[305] 537 538 541 542 544 545 546 548 550 551 553 554 555 556 557 558 562 563 564 566 567 568 570 571 574 575 577 578 581 583 587 588 589 590 591 594 595 596
[343] 597 598 603 606 607 608 610 611 613 614 616 617 618 619 622 623 625 626 632 633 636 637 640 641 642 643 645 647 648 649 650 652 653 654 655 660 661 662
[381] 663 666 670 672 674 675 676 680 682 683 686 689 690 692 693 696 698 699 700 701 705 710 711 712 713 715 718 723 724 725 727 728 730 733 734 737 738 741
[419] 742 743 744 745 749 750 752 755 756 757 760 764 765 766 768 771 772 773 774 778 779 780 784 785 787 789 790 791 792 796 798
```

Menampilkan informasi ringkas mengenai model yang telah dibuat dengan menggunakan summary().

```
summary(svmfit)
```

Hasilnya:

```
Call:
svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10
      gamma:  0.5

Number of Support Vectors:  449

 ( 224 225 )


Number of Classes:  2

Levels:
 -1 1
```

Menemukan *optimal tuning parameter.*

```
set.seed (1)
```

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Menerapkan *cross-validation* dengan menggunakan fungsi *tune()* untuk melakukan 10-*fold cross-validation* pada sebuah set model dengan cara membandingkan beberapa model yang diperoleh dengan nilai *cost* yang berbedabeda

```
23 tune.out=tune(svm,y ~ .,data=dat, kernel="linear", ranges = list(cost=c(0.001, 0.01, 0.1, 1,5, 10, 100)))
```

Memilih dan menyimpan model yang terbaik:

```
24 bestmod=tune.out$best.model
```

Menampilkan informasi ringkas mengenai model yang telah dibuat dengan menggunakan summary().

```
25 summary(bestmod)
```

Hasilnya:

```
Call:
best.tune(method = svm, train.x = y ~ ., data = dat, ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1
      gamma:  0.5

Number of Support Vectors:  450

 ( 225 225 )


Number of Classes:  2

Levels:
 -1 1
```

Mendefininisikan *testing set* menggunakan 200 *testing set*

```
28 xtest=matrix(rnorm(200*2), ncol=2)
29 ytest=sample(c(-1,1), 200, rep=TRUE)
30 xtest[ytest==1,]=xtest[ytest==1,]+1
31 testdat=data.frame(x=xtest, y=as.factor(ytest))
```

Menggunakan fungsi *predict* untuk memprediksi label kelas masing masing testing set.

```
33 ypred=predict(bestmod, testdat)
```

Menampilkan hasil prediksi terhadap 200 testing data dalam confusion matrix:

```
35 table(predict=ypred, truth=testdat$y)
```

Hasilnya :

```
       truth
predict -1  1
     -1 83 29
     1  21 67
```

Mendefenisikan *variable* tbl sebagai *confusion matrix.*
Mendefenisikan *variable* tp untuk memperoleh nilai *true positive* diperoleh dari nilai *truth* 1, dan *predict* 1 yaitu nilai pada kotak ke-4.

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Mendefenisikan *variable* fn untuk memperoleh nilai *false negative* diperoleh dari nilai *truth* 1, dan *predict* -1 yaitu nilai pada kotak ke-3.
Mendefenisikan *variable* tp untuk memperoleh nilai *false positive* diperoleh dari nilai *truth* -1, dan *predict* 1 yaitu nilai pada kotak ke-2.
Mendefenisikan *variable* tp untuk memperoleh nilai *true neagative* diperoleh dari nilai *truth* 1, dan *predict* 1 yaitu nilai pada kotak ke-1.
Mendefenisikan *variable* sum_all untuk memperoleh nilai dari penjumlahan tp + tn + fp + fn.

```
tbl = table(predict=ypred, truth=testdat$y)

tp = as.vector(tbl[4])
tp

fn = as.vector(tbl[3])
fn

fp = as.vector(tbl[2])
fp

tn = as.vector(tbl[1])
tn

sum_all = tn + fn + fp + tp
sum_all
```

Menghitung *accuracy, error rate, sensitivity, specificity, precision, recall,* dan *f_measure:*

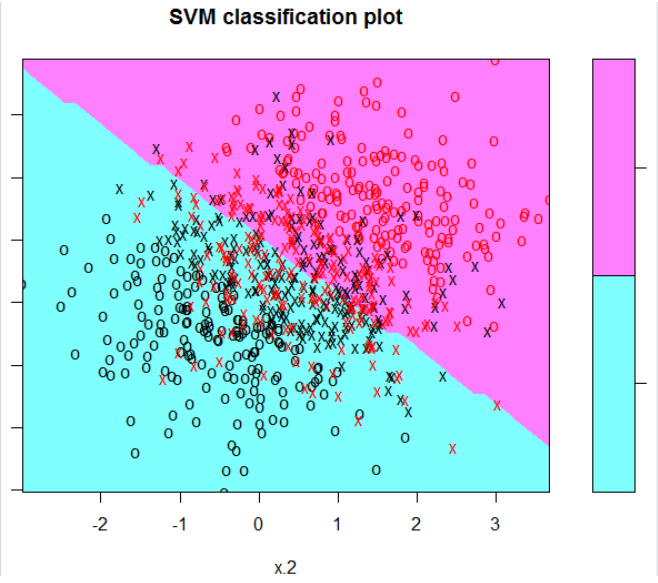
```
accuracy = (tp + tn )/sum_all
accuracy

error_rate = (fp + fn)/sum_all
error_rate

sensitivity = tp/(tp + fn)
sensitivity

specificity = tn/(fp + tn)
specificity

precision = tp/(tp + fp)
precision

recall = tp/(tp + fn)
recall

f_measure = (2*precision*recall)/(precision+recall)
f_measure
```

Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Hasilnya:

```
> tbl = table(predict=ypred, truth=testdat$y)
>
> tn = as.vector(tbl[1])
> tn
[1] 83
> fn = as.vector(tbl[3])
> fn
[1] 29
> fp = as.vector(tbl[2])
> fp
[1] 21
> tp = as.vector(tbl[4])
> tp
[1] 67
>
> sum_all = tn + fn + fp + tp
> sum_all
[1] 200
> accuracy = (tp + tn )/sum_all
> accuracy
[1] 0.75
> error_rate = (fp + fn)/sum_all
> error_rate
[1] 0.25
> sensitivity = tp/(tp + fn)
> sensitivity
[1] 0.6979167
> specificity = tn/(fp + tn)
> specificity
[1] 0.7980769
> precision = tp/(tp + fp)
> precision
[1] 0.7613636
> recall = tp/(tp + fn)
> recall
[1] 0.6979167
> f_measure = (2*precision*recall)/(precision+recall)
> f_measure
[1] 0.7282609
```

Hasil visualisasi SVM:



Praktikum 2
NIM : 12S14011
Nama : Agri Fina

Hasil Analisa Perbandingan Hasil dengan 20 Dataset dan 800 Dataset

Hasil visualisasi SVM juga menunjukkan bahwa support vector yang diperoleh dengan menggunakan 800 dataset lebih banyak dibandingkan dengan hanya menggunakan 20 dataset. Nilai *accuracy*, *precision*, *sensitivity*, *specificity*, *f-measure* yang diperoleh dengan menggunakan 20 dataset lebih tinggi dibandingkan dengan menggunakan 800 dataset, sementara *error rate* yang diperoleh dengan menggunakan 20 dataset lebih kecil daripada *error rate* yang diperoleh dengan menggunakan 800 dataset.

Hasil *run code program* 20 dataset:

x	num [1:20, 1:2] -0.5022 0.1315 -0.0789 0.8868 0.117 ...
xtest	num [1:20, 1:2] 2.512 0.39 -0.621 -1.215 2.125 ...
values	
accuracy	0.95
error_rate	0.05
f_measure	0.941176470588235
fn	1L
fp	0L
precision	1
recall	0.888888888888889
sensitivity	0.888888888888889
specificity	1
sum_all	20L
tbl	'table' int [1:2, 1:2] 11 0 1 8
tn	11L
tp	8L
y	num [1:20] -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
ypred	Factor w/ 2 levels "-1","1": 2 1 1 1 2 1 1 1 2 2 ...
ytest	num [1:20] 1 -1 -1 1 1 -1 -1 -1 1 1 ...

Hasil *run code program* 800 dataset:

x	num [1:800, 1:2] -0.5022 0.1315 -0.0789 0.8868 0.117 ...
xtest	num [1:200, 1:2] 1.074 1.896 -0.603 0.609 0.584 ...
values	
accuracy	0.75
error_rate	0.25
f_measure	0.728260869565217
fn	21L
fp	29L
precision	0.697916666666667
recall	0.761363636363636
sensitivity	0.761363636363636
specificity	0.741071428571429
sum_all	200L
tbl	'table' int [1:2, 1:2] 83 21 29 67
tn	83L
tp	67L
y	num [1:800] -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
ypred	Factor w/ 2 levels "-1","1": 1 2 1 2 2 1 1 1 2 1 ...
ytest	num [1:200] -1 -1 -1 1 1 -1 -1 -1 -1 -1 ...