ISS4003 – Data Mining

Praktikum 2

Support Vector Machines

No. Praktikum	:	1
Minggu / Sesi	:	10 / 3
Tujuan	:	• Memahami penggunaan SVM sebagai *classifier* • Memahami penerapan *classifier evaluation metric*
Setoran	:	**Softcopy**: laporan tugas dalam format pdf dan satu program R
Waktu Penyetoran	:	23 November 2017: Pukul 11:59 WIB

I. Referensi

1. R Interface, "An Example of Using the R interface," https://www.csie.ntu.edu.tw/~cjlin/libsvm/R_example.html (diakses 16 November 2017).
2. Gareth James, Daniela Witten, Trevor Hastie, dan Robert Tibshirani, "An Introduction to Statistical Learning with Applications in R," Springer, 2013.

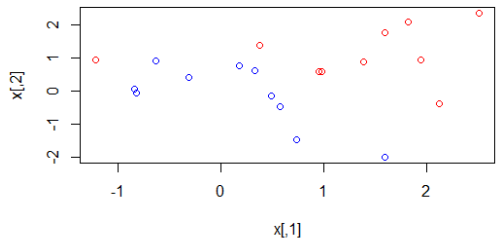
II. Support Vector Machine

Di praktikum kali ini kita akan mencoba menggunakan *library* e1071 dalam R.

1. Pasanglah *library* e1071. Silahkan lihat modul praktikum 1 jika Anda lupa cara memasang *library* di RStudio.
2. Ketiklah *source code* berikut:

```
# Construct a linearly seperable dataset on 2-D plane
set.seed (100)
x=matrix(rnorm(20*2), ncol=2)
y=c(rep(-1,10), rep(1,10))
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 test data
xtest=matrix(rnorm(20*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
table(predict=ypred, truth=testdat$y)
```

3. Kita mulai dengan membangun dataset yang tergolong ke dalam dua kelas. Didefinisikan 20 data dengan 10 data berkelas positif (1) dan 10 data berkelas negatif (-1) Ini ditunjukkan dengan baris 2 sampai 5.
4. Setelah itu, lakukan pengecekan pada dataset tersebut dengan memvisualisasikannya. Ini ditunjukkan pada baris ke 7.



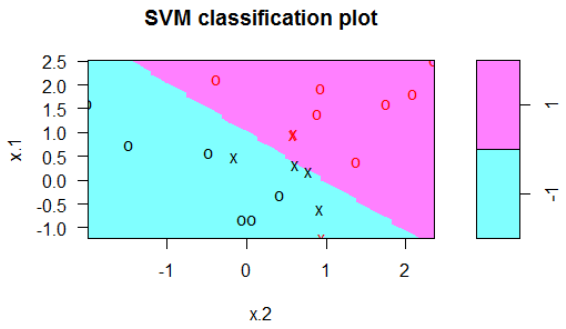
5. *Plot* menunjukkan bahwa data yang kita *generate* di *step* sebelumnya tidak terpisah secara linear. Selanjutnya, kita simpan data tersebut dalam *data frame* yang secara sederhana kita sebut *data matrix*. Secara opsional, Anda dapat menuliskan perintah berikut ini di baris ke 10 untuk melakukan pengecekan pada *data matrix*:

```
10  head(dat)
```

Akan didapatkan keluaran seperti berikut ini di dalam RConsole:

```
          x.1         x.2  y
1 -0.6264538  0.91897737 -1
2  0.1836433  0.78213630 -1
3 -0.8356286  0.07456498 -1
4  1.5952808 -1.98935170 -1
5  0.3295078  0.61982575 -1
6 -0.8204684 -0.05612874 -1
```

6. Data telah siap diolah. Sekarang, kita perlu memanggil *library* e1071 yang merupakan *library* libsvm di R. Ini ditunjukkan di baris ke 13.
7. Pada baris ke 15, kita panggil fungsi **svm()**. Di sini, kita 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; tergantung pada aplikasi, seseorang mungkin lebih suka menggunakan **scale=TRUE**.
8. Lakukan pengecekan dengan memvisualisasikan model svm yang didapat (baris 16):



9. Pada visualisasi tersebut, kita dapat melihat bahwa *support vector* ditunjukkan dengan simbol "X". Kita dapat mengidentifikasi identitas support vector tersebut dengan kode program baris 17. Akan diperoleh hasil sebagai berikut dalam RConsole:

```
[1]  1  2  5  7 14 16 17
```

10. Kita dapat menampilkan informasi ringkas mengenai model yang telah dibuat dengan menggunakan summary() yang ditunjukkan pada baris 18:

```
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:  7

 ( 4 3 )


Number of Classes:  2

Levels:
 -1 1
```

Catatan: berikut merupakan opsi lain dari SVM-type dan SVM-Kernel (Anda dapat menggali informasi lebih lanjut di laman resmi libsvm - https://www.csie.ntu.edu.tw/~cjlin/libsvm/):

```
options:
-s svm_type : set type of SVM (default 0)
        0 -- C-SVC
        1 -- nu-SVC
        2 -- one-class SVM
        3 -- epsilon-SVR
        4 -- nu-SVR
-t kernel_type : set type of kernel function (default 2)
        0 -- linear: u'*v
        1 -- polynomial: (gamma*u'*v + coef0)^degree
        2 -- radial basis function: exp(-gamma*|u-v|^2)
        3 -- sigmoid: tanh(gamma*u'*v + coef0)
```

11. *Library* e1071 meliputi *built-in-function* **tune()**, untuk menerapkan *cross-validation*. Secara default, fungsi tersebut melakukan 10-fold cross-validation pada sebuah set model. Untuk dapat menggunakan fungsi ini, kita harus memasukkan informasi yang relevan mengenai set model yang kita pertimbangkan. Baris 21-22 menunjukkan bahwa kita membandingkan beberapa model yang diperoleh dengan nilai *cost* yang berbeda-beda.
12. Model yang terbaik dipilih dan disimpan (baris 23).
13. Lakukan pengecekan pada model tersebut dengan baris 24.

```
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:  0.1
      gamma:  0.5

Number of Support Vectors:  16

 ( 8 8 )


Number of Classes:  2

Levels:
 -1 1
```

14. Ternyata kita belum mendefininisikan *testing set*. Jangan khawatir! Buat aja data yang baru khusus untuk *testing set*, dengan kode program baris 27 sampai baris 30.
15. Gunakanlah fungsi **predict()** untuk memprediksi label kelas dari masing-masing testing set kita (baris 32). Tentu saja di tahap ini, kita menggunakan model yang terbaik tadi.
16. Baris 34 menunjukkan hasil prediksi terhadap 20 *testing* data.

III. Tugas

1. Tulis ulang program tersebut, dengan syarat:
    a. Manfaatkan nilai **ypred** pada baris 32 dan **testdat$y** untuk mendapatkan nilai *accuracy*, *error rate*, *sensitivity*, *specificity*, *precision*, *recall*, and *f measure* (*f1* atau *f-score*). Tambahkan *source code* evaluasi tersebut setelah baris 34. (Lihat *slide* kuliah "*Basic Concepts of Classification*" halaman 69 - 70)
2. Ubahlah jumlah *dataset* menjadi 800 *tuple* dengan distribusi 400 kelas negatif dan 400 kelas positif (baris 3), Gunakan 200 *testing set*. Bandingkan hasilnya dengan ketika jumlah *dataset* hanya 20 *tuple*. Berikan pandangan Anda!
3. Berikan pemahaman Anda sendiri untuk setiap baris kode dalam kedua program tersebut (bukan membuat baris komentar dalam program)!

IV. Setoran

1. Laporan berisi jawaban untuk soal tugas nomor 1 dan 2, dengan nama dokumen: **praktikum2_[nim].pdf**. Tulislah **Praktikum 2**, **NIM**, dan **Nama** di sudut kanan atas dokumen. Contoh:

Praktikum 2
NIM :
Nama :

1. Source code:
   ***
2. ***
3. ***

2. Program **support_vector_classifier.R** yang sudah dimodifikasi sesuai soal tugas nomor 1.

Keduanya disetor ke **ecourse.del.ac.id** pada halaman mata kuliah Data Mining.