SVM Classification

Ethan Huynh and Ryan Gagliardi

Load packages and data. Factor the necessary columns.

```
library(e1071)
library(MASS)
df <- read.csv('adult.csv')
colnames(df) <- c("age","workclass","fnlwgt","education","education-num","marital-status","occup
ation","relationship","race","sex","capital-gain","capital-loss","hours-per-week","native-countr
y","income")
df$income <- factor(df$income)
df$education <- factor(df$education)
df$sex <- factor(df$sex)
df$race <- factor(df$race)
```

Divide into train, test, validate

Cut the data set from 32000 rows to 13000 rows randomly Then divide into train, test, and validate sets

```
set.seed(6164)
i <- sample(1:nrow(df), 0.4*nrow(df), replace=FALSE)
df2 <- df[i,]

spec <- c(train=.6, test=.2, validate=.2)
i2 <- sample(cut(1:nrow(df2),nrow(df2)*cumsum(c(0,spec)), labels=names(spec)))
train <- df2[i2=="train",]
test <- df2[i2=="test",]
vald <- df2[i2=="validate",]
```

Data exploration

Let’s explore the data with R functions and plots.

```
str(df)
```

```
## 'data.frame':    32560 obs. of  15 variables:
##  $ age           : int  50 38 53 28 37 49 52 31 42 37 ...
##  $ workclass     : chr  " Self-emp-not-inc" " Private" " Private" " Private" ...
##  $ fnlwgt        : int  83311 215646 234721 338409 284582 160187 209642 45781 159449 280464
...
##  $ education     : Factor w/ 16 levels " 10th"," 11th",..: 10 12 2 10 13 7 12 13 10 16 ...
##  $ education-num : int  13 9 7 13 14 5 9 14 13 10 ...
##  $ marital-status: chr  " Married-civ-spouse" " Divorced" " Married-civ-spouse" " Married-civ
-spouse" ...
##  $ occupation    : chr  " Exec-managerial" " Handlers-cleaners" " Handlers-cleaners" " Prof-s
pecialty" ...
##  $ relationship  : chr  " Husband" " Not-in-family" " Husband" " Wife" ...
##  $ race          : Factor w/ 5 levels " Amer-Indian-Eskimo",..: 5 5 3 3 5 3 5 5 5 3 ...
##  $ sex           : Factor w/ 2 levels " Female"," Male": 2 2 2 1 1 1 2 1 2 2 ...
##  $ capital-gain  : int  0 0 0 0 0 0 0 14084 5178 0 ...
##  $ capital-loss  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hours-per-week: int  13 40 40 40 40 16 45 50 40 80 ...
##  $ native-country: chr  " United-States" " United-States" " United-States" " Cuba" ...
##  $ income        : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 2 2 2 2 ...
```

```
dim(df)
```

```
## [1] 32560    15
```

```
head(df)
```

	... <int>	workclass <chr>	fnlwgt <int>	education <fct>	education-num <int>	marital-status <chr>	occupation <chr>
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-manag
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cle
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cle
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-special
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-manag
6	49	Private	160187	9th	5	Married-spouse-absent	Other-servic

6 rows | 1-8 of 16 columns

Build a logistic regression model for baseline

```
glm1 <- glm(income~education+sex+race, data=train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = income ~ education + sex + race, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9016  -0.6681  -0.4476  -0.1665   2.7506
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -4.55549    0.50079  -9.097  < 2e-16 ***
## education 11th           -0.20293    0.38554  -0.526 0.598642
## education 12th            0.64539    0.43939   1.469 0.141880
## education 1st-4th       -13.02941  232.39690  -0.056 0.955290
## education 5th-6th         0.05238    0.50970   0.103 0.918145
## education 7th-8th         0.13687    0.41136   0.333 0.739347
## education 9th             0.20315    0.42192   0.481 0.630173
## education Assoc-acdm      1.64047    0.31551   5.199 2.00e-07 ***
## education Assoc-voc       1.70004    0.30730   5.532 3.16e-08 ***
## education Bachelors       2.33443    0.28482   8.196 2.48e-16 ***
## education Doctorate       3.70400    0.35908  10.315  < 2e-16 ***
## education HS-grad         0.94143    0.28412   3.314 0.000921 ***
## education Masters         2.92146    0.29570   9.880  < 2e-16 ***
## education Preschool     -12.85725  392.74428  -0.033 0.973884
## education Prof-school     3.95641    0.34510  11.464  < 2e-16 ***
## education Some-college    1.33306    0.28510   4.676 2.93e-06 ***
## sex Male                  1.25581    0.07359  17.065  < 2e-16 ***
## race Asian-Pac-Islander   0.74331    0.44325   1.677 0.093552 .
## race Black                0.48689    0.42867   1.136 0.256033
## race Other                0.15871    0.57527   0.276 0.782636
## race White                0.97225    0.41419   2.347 0.018908 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8618.4  on 7813  degrees of freedom
## Residual deviance: 7186.6  on 7793  degrees of freedom
## AIC: 7228.6
##
## Number of Fisher Scoring iterations: 14
```

Try a linear kernel

```
svm1 <- svm(income~education+sex+race, data=train, kernel="linear", cost=10, scale=TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = income ~ education + sex + race, data = train, kernel = "linear",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  3456
##
##  ( 1744 1712 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred <- predict(svm1, newdata=test)
table(pred, test$income)
```

```
##
## pred      <=50K  >50K
##    <=50K   1875   514
##    >50K      91   125
```

```
mean(pred==test$income)
```

```
## [1] 0.7677543
```

Tune

```
tune_svm1 <- tune(svm, income~education+sex+race, data=vald, kernel="linear"
                  ,ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune_svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   0.1
##
## - best performance: 0.2168833
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.2295697 0.02257700
## 2 1e-02 0.2295697 0.02257700
## 3 1e-01 0.2168833 0.01255157
## 4 1e+00 0.2172679 0.01146481
## 5 5e+00 0.2172679 0.01146481
## 6 1e+01 0.2172679 0.01146481
## 7 1e+02 0.2172679 0.01146481
```

Evaluate on best linear svm

```
pred <- predict(tune_svm1$best.model, newdata=test)
table(pred, test$income)
```

```
##
## pred      <=50K  >50K
##    <=50K   1947   584
##    >50K      19    55
```

```
mean(pred==test$income)
```

```
## [1] 0.7685221
```

Try a polynomial kernel

```
svm2 <- svm(income~education+sex+race, data=train, kernel="polynomial", cost=10, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = income ~ education + sex + race, data = train, kernel = "polynomial",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  3789
##
##  ( 1911 1878 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred <- predict(svm2, newdata=test)
table(pred, test$income)
```

```
##
## pred      <=50K  >50K
##    <=50K   1923   548
##    >50K      43    91
```

```
mean(pred==test$income)
```

```
## [1] 0.7731286
```

Try a radial kernel

```
svm3 <- svm(income~education+sex+race, data=train, kernel="radial", cost=10, gamma=1, scale=TRUE
)
summary(svm3)
```

```
##
## Call:
## svm(formula = income ~ education + sex + race, data = train, kernel = "radial",
##     cost = 10, gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  3416
##
##  ( 1748 1668 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred <- predict(svm3, newdata=test)
table(pred, test$income)
```

```
##
## pred      <=50K  >50K
##    <=50K   1912   537
##    >50K      54   102
```

```
mean(pred==test$income)
```

```
## [1] 0.7731286
```

Tune hyperperameters

```
set.seed(6164)
tune.out <- tune(svm, income~education+sex+race, data=vald, kernel="radial",
                 ranges=list(cost=c(0.1,1,10,100,1000),
                             gamma=c(0.5,1,2,3,4)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##   0.1     1
##
## - best performance: 0.2065399
##
## - Detailed performance results:
##     cost gamma     error dispersion
## 1  1e-01   0.5 0.2195859 0.02454082
## 2  1e+00   0.5 0.2157619 0.02140692
## 3  1e+01   0.5 0.2157604 0.02061177
## 4  1e+02   0.5 0.2157604 0.02061177
## 5  1e+03   0.5 0.2157604 0.02061177
## 6  1e-01   1.0 0.2065399 0.02349506
## 7  1e+00   1.0 0.2157619 0.02117711
## 8  1e+01   1.0 0.2165281 0.02032604
## 9  1e+02   1.0 0.2165281 0.02032604
## 10 1e+03   1.0 0.2165281 0.02032604
## 11 1e-01   2.0 0.2065399 0.02349506
## 12 1e+00   2.0 0.2165281 0.02032604
## 13 1e+01   2.0 0.2165281 0.02032604
## 14 1e+02   2.0 0.2165281 0.02032604
## 15 1e+03   2.0 0.2165281 0.02032604
## 16 1e-01   3.0 0.2065399 0.02349506
## 17 1e+00   3.0 0.2165281 0.02032604
## 18 1e+01   3.0 0.2165281 0.02032604
## 19 1e+02   3.0 0.2165281 0.02032604
## 20 1e+03   3.0 0.2165281 0.02032604
## 21 1e-01   4.0 0.2065399 0.02349506
## 22 1e+00   4.0 0.2165281 0.02032604
## 23 1e+01   4.0 0.2165281 0.02032604
## 24 1e+02   4.0 0.2165281 0.02032604
## 25 1e+03   4.0 0.2165281 0.02032604
```

```
svm4 <- svm(income~education+sex+race, data=train, kernel="radial", cost=100, gamma=0.5, scale=T
RUE)
summary(svm4)
```

```
##
## Call:
## svm(formula = income ~ education + sex + race, data = train, kernel = "radial",
##     cost = 100, gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  3393
##
##  ( 1726 1667 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred <- predict(svm4, newdata=test)
table(pred, test$income)
```

```
##
## pred      <=50K  >50K
##    <=50K   1910   537
##    >50K      56   102
```

```
mean(pred==test$income)
```

```
## [1] 0.7723608
```

Best method

In this case the data was correlated so well and in a way that all algorithms got almost the exact same accuracy. Both radial and polynomial got .77 accuracy while linear got .76. This is likely because both radial and polynomial are able to “bend” to get a more accurate fit for the data and be the most in line with the data points around it while linear has to be a straight line making it unable to conform to irregular data trends.