SVM Classification

Sai Gonugunla

10/15/2022

source: https://www.kaggle.com/datasets/uciml/adult-census-income (https://www.kaggle.com/datasets/uciml/adult-census-income)

```
df <- read.csv("income_evaluation.csv", header=TRUE)
str(df)
```

```
## 'data.frame':    32561 obs. of  15 variables:
##  $ age           : int  39 50 38 53 28 37 49 52 31 42 ...
##  $ workclass     : chr  " State-gov" " Self-emp-not-inc" " Private" " Private" ...
##  $ fnlwgt        : int  77516 83311 215646 234721 338409 284582 160187 209642 4578
1 159449 ...
##  $ education     : chr  " Bachelors" " Bachelors" " HS-grad" " 11th" ...
##  $ education.num : int  13 13 9 7 13 14 5 9 14 13 ...
##  $ marital.status: chr  " Never-married" " Married-civ-spouse" " Divorced" " Marri
ed-civ-spouse" ...
##  $ occupation    : chr  " Adm-clerical" " Exec-managerial" " Handlers-cleaners" "
Handlers-cleaners" ...
##  $ relationship  : chr  " Not-in-family" " Husband" " Not-in-family" " Husband" ..
.
##  $ race          : chr  " White" " White" " White" " Black" ...
##  $ sex           : chr  " Male" " Male" " Male" " Male" ...
##  $ capital.gain  : int  2174 0 0 0 0 0 0 0 14084 5178 ...
##  $ capital.loss  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hours.per.week: int  40 13 40 40 40 40 16 45 50 40 ...
##  $ native.country: chr  " United-States" " United-States" " United-States" " Unite
d-States" ...
##  $ income        : chr  " <=50K" " <=50K" " <=50K" " <=50K" ...
```

Divide into train, test, validate

```
incomeEval <- df[,c(1,5, 13,15)]
attach(incomeEval)

set.seed(3)
spec <- c(train=.6, test=.2, validate=.2)
i <- sample(cut(1:nrow(incomeEval),
                nrow(incomeEval)*cumsum(c(0,spec)), labels=names(spec)))
train <- incomeEval[i=="train",]
test <- incomeEval[i=="test",]
vald <- incomeEval[i=="validate",]
```

```
attach(incomeEval)
```

```
## The following objects are masked from incomeEval (pos = 3):
##
##     age, education.num, hours.per.week, income
```

```
head(incomeEval,n=3, data = train)
```

```
##   age education.num hours.per.week income
## 1  39            13             40  <=50K
## 2  50            13             13  <=50K
## 3  38             9             40  <=50K
```

```
tail(incomeEval,n=5, data = train)
```

```
##       age education.num hours.per.week income
## 32557  27            12             38  <=50K
## 32558  40             9             40   >50K
## 32559  58             9             40  <=50K
## 32560  22             9             20  <=50K
## 32561  52             9             40   >50K
```

```
mean(age, data = train)
```

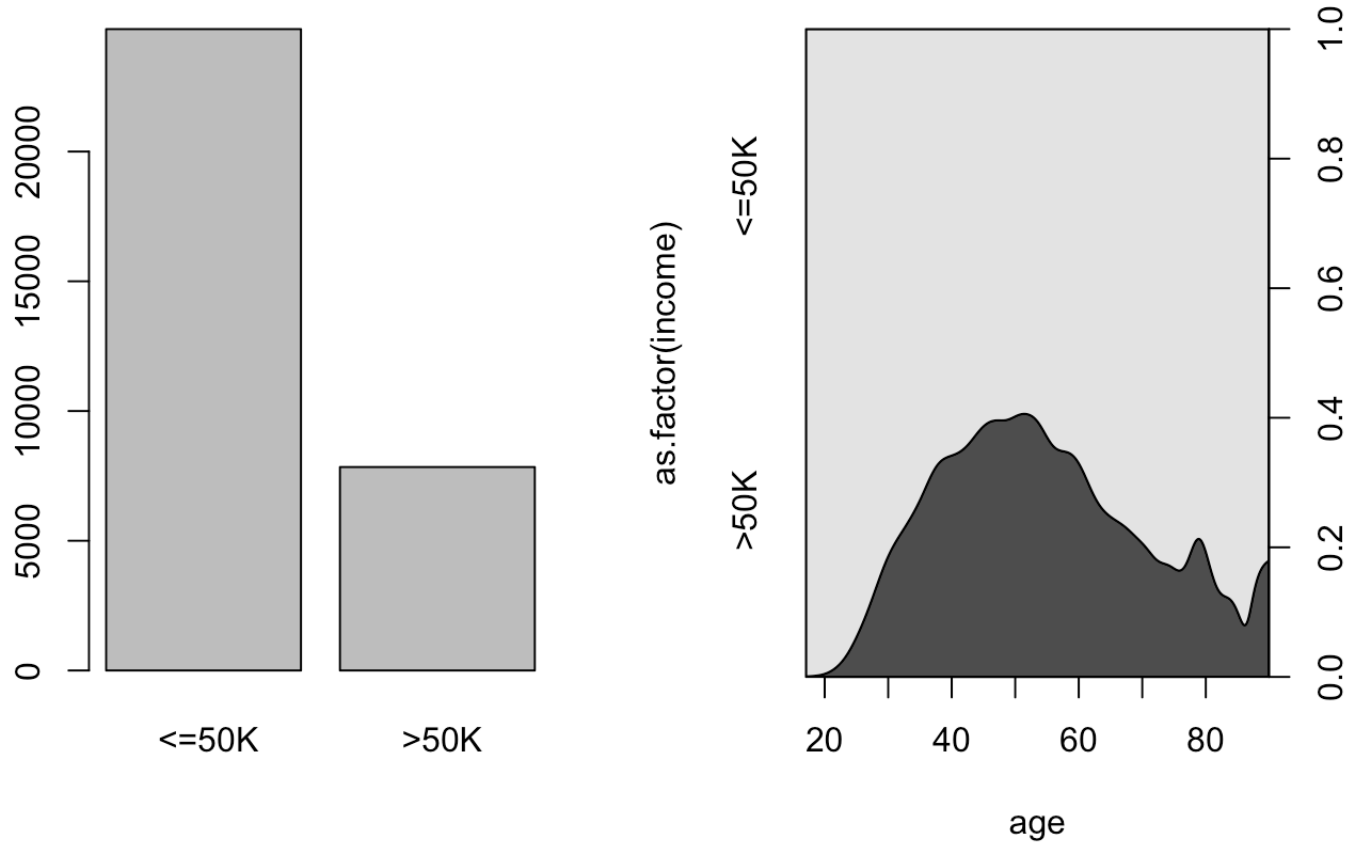
```
## [1] 38.58165
```

```
median(education.num, data = train)
```

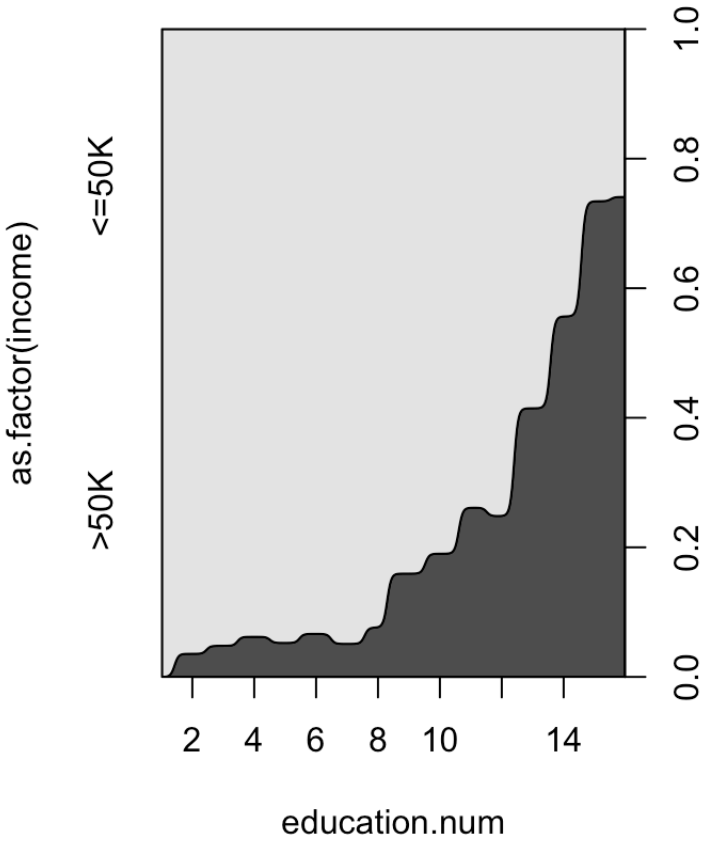
```
## [1] 10
```

there are more observations that are <50k than >=50k. The second set of pots are conditional density plots. The total probability space is the rectangle, with the darker grey indicating income >50k. We can see that as the years of education increases salary increases but this is harder to see with the age cd plot

```
par(mfrow=c(1,2))
plot(as.factor(income))
cdplot(as.factor(income)~age)
```




```
cdplot(as.factor(income)~education.num)
```



###Linear SVM Support Vectors: 9544

```
library(e1071)
svm1 <- svm(as.factor(income)~., data=train, kernel="linear", cost=10, scale=TRUE, ty
pe = "C-classification")
summary(svm1)
```

```
##
## Call:
## svm(formula = as.factor(income) ~ ., data = train, kernel = "linear",
##     cost = 10, type = "C-classification", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  9544
##
##  ( 4774 4770 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

evaluate mean accuracy is 0.783

```
pred1 <- predict(svm1, newdata=test)
table(pred1, test$income)
```

```
##
## pred1     <=50K  >50K
##    <=50K   4871  1311
##    >50K     100   230
```

```
mean(pred1==test$income)
```

```
## [1] 0.7833231
```

#linear kernel tune performed the best at c of .001 and gave .232

```
tune_svm1 <- tune(svm, as.factor(income)~., 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
##  0.001
##
## - best performance: 0.2321523
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.2321523 0.01916384
## 2 1e-02 0.2321523 0.01916384
## 3 1e-01 0.2321523 0.01916384
## 4 1e+00 0.2321523 0.01916384
## 5 5e+00 0.2321523 0.01916384
## 6 1e+01 0.2321523 0.01916384
## 7 1e+02 0.2321523 0.01916384
```

Evaluate on best linear svm

the best model has a mean accuracy of .763

```
pred1 <- predict(tune_svm1$best.model, newdata=test)
mean(pred1==test$income)
```

```
## [1] 0.76336
```

Try a polynomial kernel

mean accuracy is .788, number of Support Vectors: 8951

```
svm2 <- svm(as.factor(income)~., data=train, kernel="polynomial", cost=10, scale=TRUE
,type = "C-classification")
summary(svm2)
```

```
##
## Call:
## svm(formula = as.factor(income) ~ ., data = train, kernel = "polynomial",
##     cost = 10, type = "C-classification", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  8951
##
##  ( 4485 4466 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred2 <- predict(svm2, newdata=test)
mean(pred2==test$income)
```

```
## [1] 0.7880835
```

#polynomial kernel tune performed the best at c of 5 and gave .21004

```
tune_svm2 <- tune(svm, as.factor(income)~., data=vald, kernel="polynomial",
                  ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune_svm2)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     5
##
## - best performance: 0.2100395
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.2315360 0.01703540
## 2 1e-02 0.2149517 0.01692274
## 3 1e-01 0.2106540 0.01412549
## 4 1e+00 0.2101931 0.01527386
## 5 5e+00 0.2100395 0.01541543
## 6 1e+01 0.2100395 0.01541543
## 7 1e+02 0.2203234 0.02424896
```

Evaluate on best polynomial svm

the best model has a mean accuracy of .786

```
pred2 <- predict(tune_svm2$best.model, newdata=test)
mean(pred2==test$income)
```

```
## [1] 0.7857801
```

Try a radial kernel

Support Vectors: 8840. mean accuracy of 0.805

```
svm3 <- svm(as.factor(income)~., data=train, kernel="radial", cost=10, gamma=1, scale
=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = as.factor(income) ~ ., data = train, kernel = "radial",
##     cost = 10, gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  8840
##
##  ( 4808 4032 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

```
pred3 <- predict(svm3, newdata=test)
mean(pred3==test$income)
```

```
## [1] 0.8045147
```

Tune hyperperameters

best was cost 1, gamma .5. Performance of .201

```
set.seed(1234)
tune.out <- tune(svm, as.factor(income)~., 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
##     1   0.5
##
## - best performance: 0.2014445
##
## - Detailed performance results:
##     cost gamma     error dispersion
## 1  1e-01   0.5 0.2020591 0.01300229
## 2  1e+00   0.5 0.2014445 0.01423764
## 3  1e+01   0.5 0.2019039 0.01510260
## 4  1e+02   0.5 0.2058977 0.01572729
## 5  1e+03   0.5 0.2069728 0.01436059
## 6  1e-01   1.0 0.2035940 0.01258549
## 7  1e+00   1.0 0.2014449 0.01737836
## 8  1e+01   1.0 0.2032882 0.01599700
## 9  1e+02   1.0 0.2098911 0.01404901
## 10 1e+03   1.0 0.2149569 0.01458587
## 11 1e-01   2.0 0.2054383 0.01568417
## 12 1e+00   2.0 0.2014466 0.01720086
## 13 1e+01   2.0 0.2082033 0.01461715
## 14 1e+02   2.0 0.2152634 0.01278571
## 15 1e+03   2.0 0.2223283 0.01215440
## 16 1e-01   3.0 0.2065122 0.01652584
## 17 1e+00   3.0 0.2048239 0.01410378
## 18 1e+01   3.0 0.2086634 0.01240122
## 19 1e+02   3.0 0.2203328 0.01206690
## 20 1e+03   3.0 0.2287790 0.01534234
## 21 1e-01   4.0 0.2114253 0.01652127
## 22 1e+00   4.0 0.2037514 0.01515274
## 23 1e+01   4.0 0.2123486 0.01325584
## 24 1e+02   4.0 0.2244779 0.01369717
## 25 1e+03   4.0 0.2318451 0.01286463
```

###Evaluate on best radial svm mean accuracy of 0.803

```
pred4 <- predict(tune.out$best.model, newdata=test)
mean(pred4==test$income)
```

```
## [1] 0.8029791
```

Radial SVM performed the best with an accuracy of 0.803 followed by polynomial with an accuracy of 0.786. The SVM with the lowest accuracy is linear SVM with an accuracy of 0.763. Radial SVM performed the best likely because the dataset used may not be linearly separable. Linear SVM didnt perform as welllikely because it works the best when there are a lot of features but in this situation only 3 features are used.