Anly501 NB and SVM

Yangyi Li

```
# we set align to be the label
# we want to see if any other elements affect the align
marvel <- read.csv("marvel.csv")
names(marvel) <- tolower(names(marvel))

head(marvel)
```

```
##   page_id                                    name
## 1    1678             spider-man (peter parker)
## 2    7139       captain america (steven rogers)
## 3   64786 wolverine (james \\"logan\\" howlett)
## 4    1868   iron man (anthony \\"tony\\" stark)
## 5    2460                    thor (thor odinson)
## 6    2458            benjamin grimm (earth-616)
##                                   urlslug               id              align
## 1             \\/spider-man_(peter_parker)  secret identity    good characters
## 2       \\/captain_america_(steven_rogers)  public identity    good characters
## 3 \\/wolverine_(james_%22logan%22_howlett)  public identity neutral characters
## 4   \\/iron_man_(anthony_%22tony%22_stark)  public identity    good characters
## 5                   \\/thor_(thor_odinson) no dual identity    good characters
## 6            \\/benjamin_grimm_(earth-616)  public identity    good characters
##          eye       hair             sex gsm             alive appearances
## 1 hazel eyes brown hair male characters     living characters        4043
## 2  blue eyes white hair male characters     living characters        3360
## 3  blue eyes black hair male characters     living characters        3061
## 4  blue eyes black hair male characters     living characters        2961
## 5  blue eyes blond hair male characters     living characters        2258
## 6  blue eyes    no hair male characters     living characters        2255
##   first.appearance year
## 1           aug-62 1962
## 2           mar-41 1941
## 3           oct-74 1974
## 4           mar-63 1963
## 5           nov-50 1950
## 6           nov-61 1961
```

```
marvel <- marvel %>% select(id, align, sex, alive, appearances, year)
marvel <- marvel %>% select(align, everything())
marvel <- subset(marvel, align!="")
marvel <- subset(marvel, id!="")
marvel <- subset(marvel, sex!="")
marvel <- marvel %>% filter(sex == "male characters"|
           sex == "female characters")
marvel <- marvel %>% drop_na()
colnames(marvel)[1] <- "label"
marvel <- marvel[1:1000,]
head(marvel)
```

```
##                label               id             sex              alive
## 1    good characters  secret identity male characters living characters
## 2    good characters  public identity male characters living characters
## 3 neutral characters  public identity male characters living characters
## 4    good characters  public identity male characters living characters
## 5    good characters no dual identity male characters living characters
## 6    good characters  public identity male characters living characters
##   appearances year
## 1        4043 1962
## 2        3360 1941
## 3        3061 1974
## 4        2961 1963
## 5        2258 1950
## 6        2255 1961
```

```
str(marvel)
```

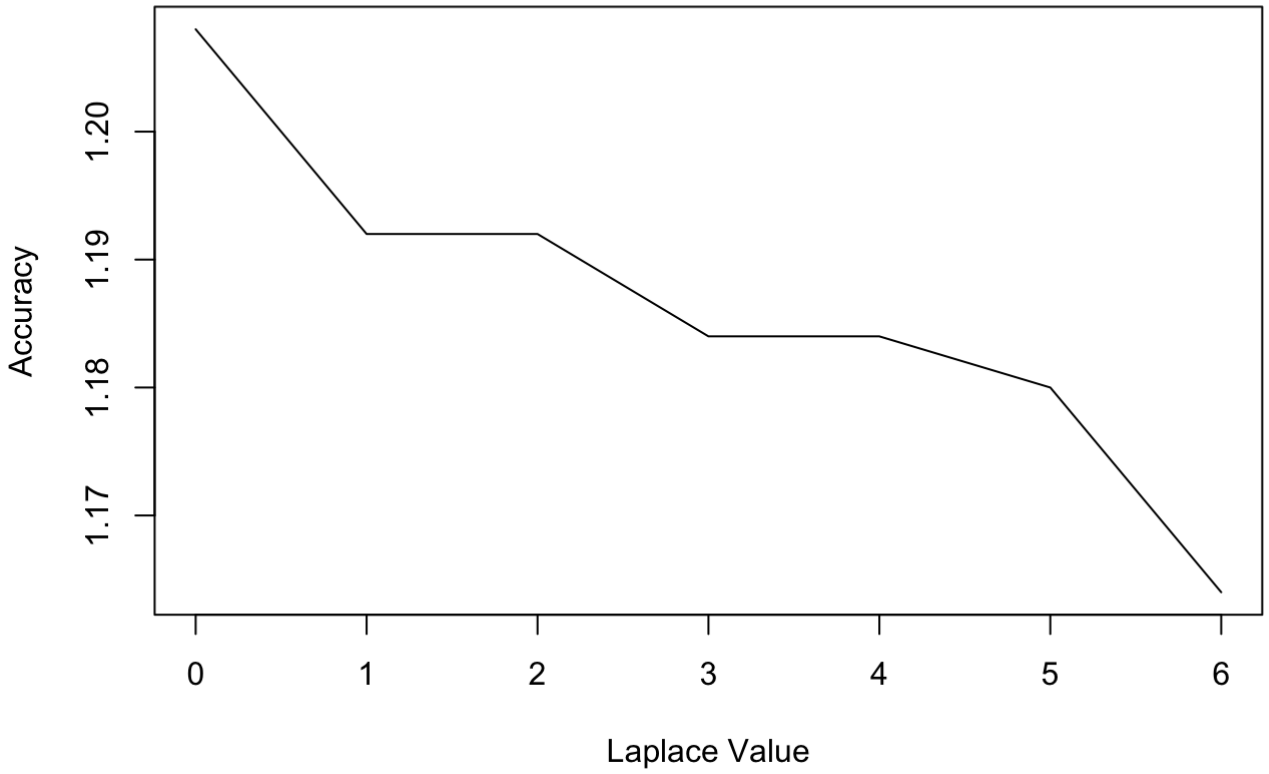
```
## 'data.frame':    1000 obs. of  6 variables:
##  $ label      : chr  "good characters" "good characters" "neutral characters" "good c
haracters" ...
##  $ id         : chr  "secret identity" "public identity" "public identity" "public id
entity" ...
##  $ sex        : chr  "male characters" "male characters" "male characters" "male char
acters" ...
##  $ alive      : chr  "living characters" "living characters" "living characters" "liv
ing characters" ...
##  $ appearances: int  4043 3360 3061 2961 2258 2255 2072 2017 1955 1934 ...
##  $ year       : int  1962 1941 1974 1963 1950 1961 1961 1962 1963 1961 ...
```

```
## If necessary - correct data types
marvel$label <- as.factor(marvel$label)
marvel$id <- as.factor(marvel$id)
marvel$sex <- as.factor(marvel$sex)
marvel$alive <- as.factor(marvel$alive)
```

```
DataSize=nrow(marvel)
TrainingSet_Size<-floor(DataSize*(3/4))
TestSet_Size <- DataSize - TrainingSet_Size

MyTrainSample <- sample(nrow(marvel),TrainingSet_Size,replace=FALSE)
MyTrainingSET <- marvel[MyTrainSample,]
MyTestSET <- marvel[-MyTrainSample,]

train_label <- MyTrainingSET$label
test_label <- MyTestSET$label

old_set <- MyTrainingSET
MyTrainingSET<-MyTrainingSET[ , -which(names(MyTrainingSET) %in% c("label"))]
head(MyTrainingSET)
```

```
##                    id               sex               alive appearances year
## 515  public identity   male characters deceased characters          71 1968
## 620  secret identity   male characters   living characters          58 2005
## 939  secret identity female characters   living characters          36 1993
## 664  secret identity   male characters   living characters          52 1984
## 999  public identity female characters   living characters          34 1996
## 254 no dual identity female characters deceased characters         156 2005
```

```
Accuracy=c()
for(laplace in 0:6){
  model<-naiveBayes(label~.,data=old_set,laplace=laplace)
  prediction<-predict(model,MyTrainingSET)
  Accuracy<-c(Accuracy,sum(prediction==MyTestSET$label)/nrow(MyTestSET))
}
plot(0:6,Accuracy,'l',xlab='Laplace Value')
```



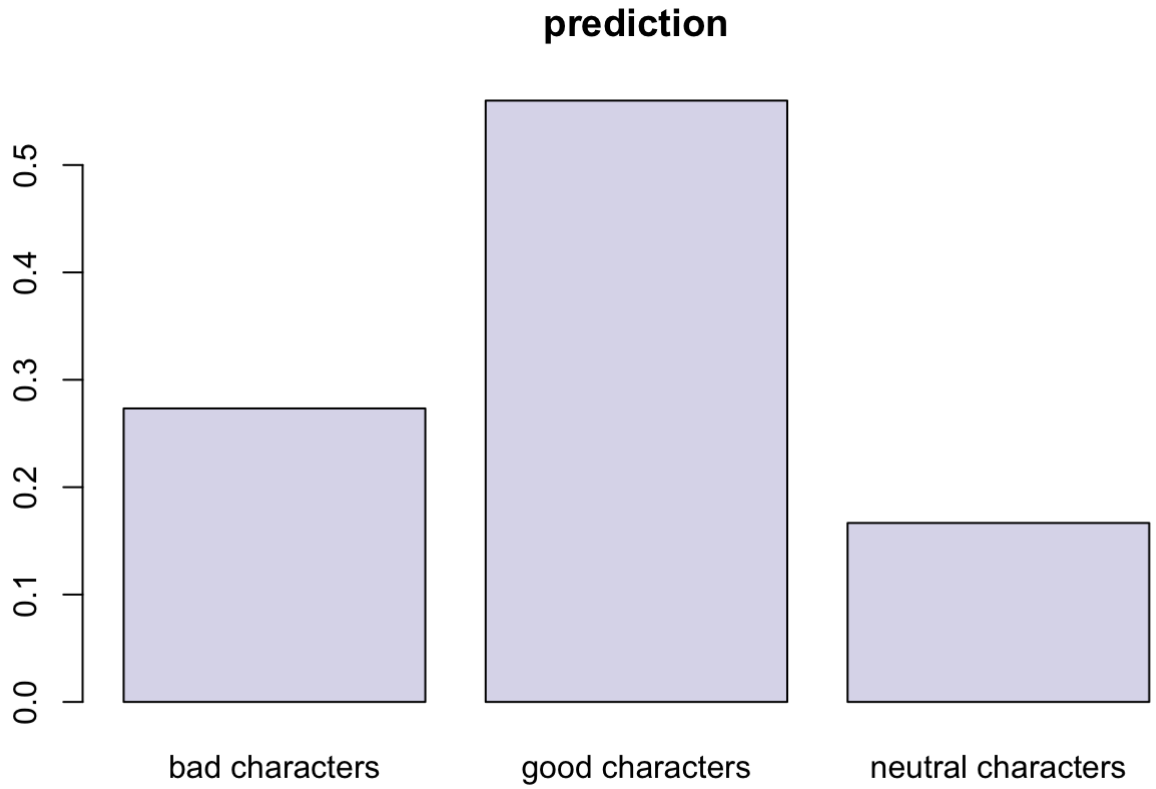

```
(NB_e1071_2<-naiveBayes(MyTrainingSET,
                        train_label,
                        laplace = 1))
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = MyTrainingSET, y = train_label, laplace = 1)
##
## A-priori probabilities:
## train_label
##     bad characters    good characters neutral characters
##          0.2733333          0.5600000          0.1666667
##
## Conditional probabilities:
##                     id
## train_label          known to authorities identity no dual identity
##   bad characters                      0.004784689      0.105263158
##   good characters                     0.009433962      0.202830189
##   neutral characters                  0.007751938      0.178294574
##                     id
## train_label          public identity secret identity
##   bad characters         0.325358852     0.564593301
##   good characters        0.339622642     0.448113208
##   neutral characters     0.248062016     0.565891473
##
##                     sex
## train_label          female characters male characters
##   bad characters             0.1690821       0.8309179
##   good characters            0.4265403       0.5734597
##   neutral characters         0.2992126       0.7007874
##
##                     alive
## train_label          deceased characters living characters
##   bad characters               0.3140097         0.6859903
##   good characters              0.2180095         0.7819905
##   neutral characters           0.2204724         0.7795276
##
##                     appearances
## train_label               [,1]     [,2]
##   bad characters      93.70732  86.6154
##   good characters    213.65714 405.3904
##   neutral characters 161.81600 261.0325
##
##                     year
## train_label              [,1]     [,2]
##   bad characters     1975.776 13.48595
##   good characters    1979.888 18.23119
##   neutral characters 1978.936 14.03265
```

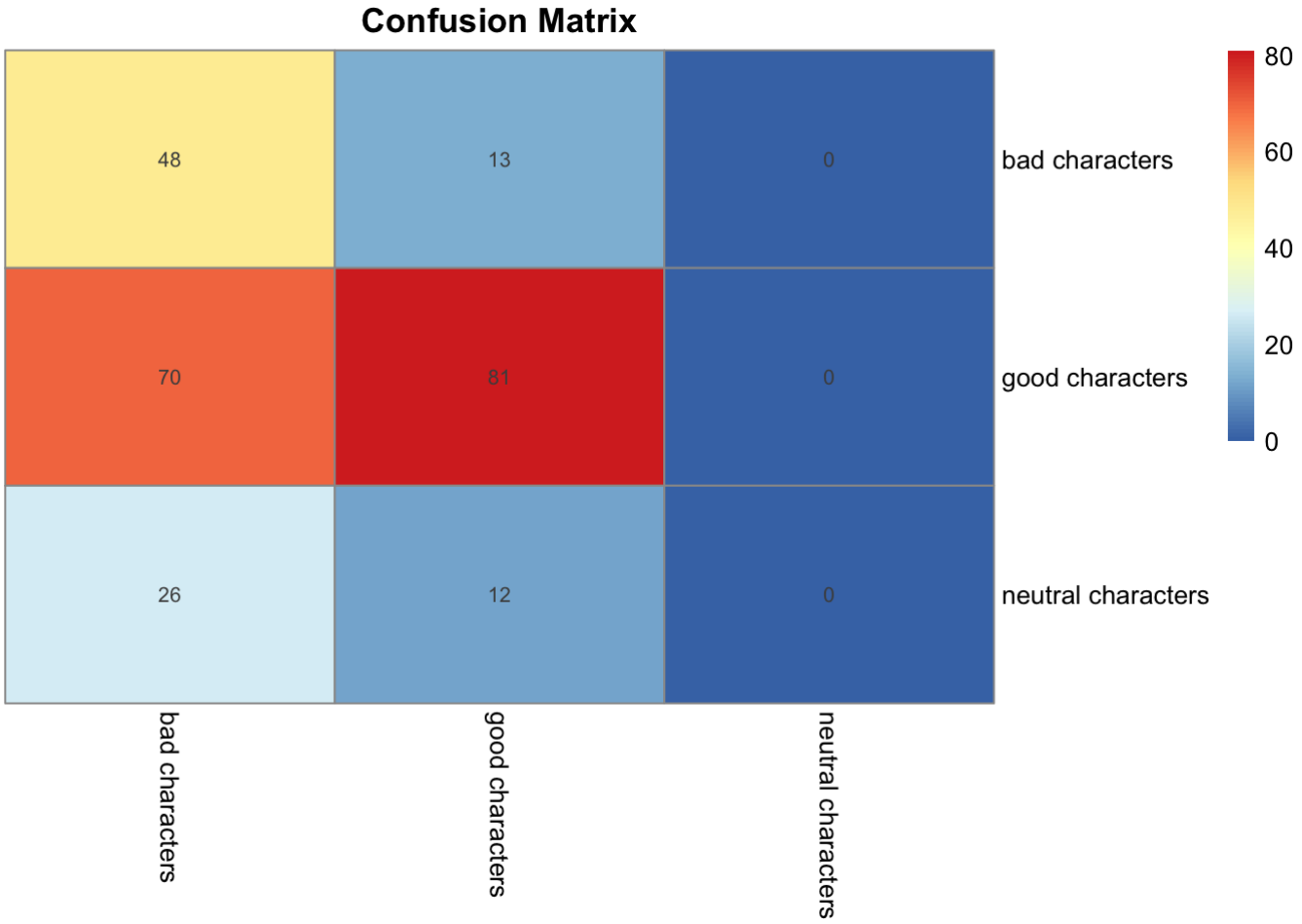
```
NB_e1071_Pred <- predict(NB_e1071_2, MyTestSET)

table(NB_e1071_Pred,test_label)
```

```
##                     test_label
## NB_e1071_Pred        bad characters good characters neutral characters
##   bad characters                 48              70                 26
##   good characters                13              81                 12
##   neutral characters              0               0                  0
```

```
model<-naiveBayes(label~.,data=old_set,laplace=1)
acu <- model$apriori/sum(model$apriori)
barplot(acu, col=rgb(0.2,0.2,0.6,0.2), main='prediction')
```




```
confusionMatrix<-table(MyTestSET$label,NB_e1071_Pred)
pheatmap(confusionMatrix,cluster_cols=F,cluster_rows=F,display_numbers=T,number_format =
"%.f",main='Confusion Matrix')
```


Confusion Matrix
48
13
0
70
81
0
26
12
0
bad characters
good characters
neutral characters
bad characters
good characters
neutral characters
80
60
40
20
0