Practice 8_DA5030

Shivam Verma

Problem 1.

Step 2 – exploring and preparing the data

```
teens <- read.csv("snsdata.csv")
str(teens)
```

```
## 'data.frame':    30000 obs. of  40 variables:
##  $ gradyear    : int  2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
##  $ gender      : Factor w/ 2 levels "F","M": 2 1 2 1 NA 1 1 2 1 1 ...
##  $ age         : num  19 18.8 18.3 18.9 19 ...
##  $ friends     : int  7 0 69 0 10 142 72 17 52 39 ...
##  $ basketball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ football    : int  0 1 1 0 0 0 0 0 0 0 ...
##  $ soccer      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ softball    : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ volleyball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ swimming    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cheerleading: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ baseball    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ tennis      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sports      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cute        : int  0 1 0 1 0 0 0 0 0 1 ...
##  $ sex         : int  0 0 0 0 1 1 0 2 0 0 ...
##  $ sexy        : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ hot         : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ kissed      : int  0 0 0 0 5 0 0 0 0 0 ...
##  $ dance       : int  1 0 0 0 1 0 0 0 0 0 ...
##  $ band        : int  0 0 2 0 1 0 1 0 0 0 ...
##  $ marching    : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ music       : int  0 2 1 0 3 2 0 1 0 1 ...
##  $ rock        : int  0 2 0 1 0 0 0 1 0 1 ...
##  $ god         : int  0 1 0 0 1 0 0 0 0 6 ...
##  $ church      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ jesus       : int  0 0 0 0 0 0 0 0 0 2 ...
##  $ bible       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hair        : int  0 6 0 0 1 0 0 0 0 1 ...
##  $ dress       : int  0 4 0 0 0 1 0 0 0 0 ...
##  $ blonde      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mall        : int  0 1 0 0 0 0 2 0 0 0 ...
##  $ shopping    : int  0 0 0 0 2 1 0 0 0 1 ...
##  $ clothes     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hollister   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ abercrombie : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ die         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ death       : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ drunk       : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ drugs       : int  0 0 0 0 1 0 0 0 0 0 ...
```

```r
table(teens$gender, useNA = "ifany")
```

```
##
##     F     M  <NA>
## 22054  5222  2724
```

```r
teens$age <- ifelse(teens$age >= 13 & teens$age < 20, teens$age, NA)
summary(teens$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   13.03   16.30   17.27   17.25   18.22   20.00    5523
```

Data preparation – dummy coding missing values

```r
teens$female <- ifelse(teens$gender == "F" & !is.na(teens$gender), 1, 0)
teens$no_gender <- ifelse(is.na(teens$gender), 1, 0)
table(teens$gender, useNA = "ifany")
```

```
##
##     F     M  <NA>
## 22054  5222  2724
```

```r
table(teens$female, useNA = "ifany")
```

```
##
##     0     1
##  7946 22054
```

```r
table(teens$no_gender, useNA = "ifany")
```

```
##
##     0     1
## 27276  2724
```

Data preparation – imputing the missing values

```r
mean(teens$age, na.rm = TRUE)
```

```
## [1] 17.25243
```

```r
aggregate(data = teens, age ~ gradyear, mean, na.rm = TRUE)
```

```
##   gradyear      age
## 1     2006 18.65586
## 2     2007 17.70617
## 3     2008 16.76770
## 4     2009 15.81957
```

```
ave_age <- ave(teens$age, teens$gradyear, FUN = function(x) mean(x, na.rm = TRUE))
teens$age <- ifelse(is.na(teens$age), ave_age, teens$age)
summary(teens$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.03   16.28   17.24   17.24   18.21   20.00
```

Standardizing the data

```
interests <- teens[5:40]
interests_z <- as.data.frame(lapply(interests, scale))
```

Step 3 – training a model on the data

```
set.seed(2345)
teen_clusters <- kmeans(interests_z, 5)
```

Step 4 – evaluating model performance

```
teen_clusters$size
```

```
## [1]  1038   601  4066  2696 21599
```

Examining the coordinates of the cluster centroids

```
teen_clusters$centers
```

```
##     basketball    football      soccer   softball  volleyball    swimming
## 1  0.362160730  0.37985213  0.13734997  0.1272107  0.09247518  0.26180286
## 2 -0.094426312  0.06691768 -0.09956009 -0.0379725 -0.07286202  0.04578401
## 3  0.003980104  0.09524062  0.05342109 -0.0496864 -0.01459648  0.32944934
## 4  1.372334818  1.19570343  0.55621097  1.1304527  1.07177211  0.08513210
## 5 -0.186822093 -0.18729427 -0.08331351 -0.1368072 -0.13344819 -0.08650052
##   cheerleading    baseball      tennis      sports         cute          sex
## 1    0.2159945  0.25312305  0.11991682  0.77040675  0.475265034  2.043945661
## 2   -0.1070370 -0.11182941  0.04027335 -0.10638613 -0.027044898 -0.042725567
## 3    0.5142451 -0.04933628  0.06703386 -0.05435093  0.796948359 -0.003156716
## 4    0.0400367  1.09279737  0.13887184  1.08316097 -0.005291962 -0.033193640
## 5   -0.1092056 -0.13616893 -0.03683671 -0.15903307 -0.171452198 -0.092301138
##           sexy          hot      kissed        dance        band   marching
## 1  0.547956598  0.314845390  3.02610259  0.455501275  0.39009330 -0.0105463
## 2 -0.027913348 -0.035027022 -0.04581067  0.050772118  4.09723438  5.2196105
## 3  0.266741598  0.623263396 -0.01284964  0.650572336 -0.03301257 -0.1131486
## 4  0.003036966  0.009046774 -0.08755418 -0.001993853 -0.07317758 -0.1039509
## 5 -0.076149916 -0.132614350 -0.13080557 -0.145524147 -0.11740538 -0.1104553
```

```
##          music       rock         god     church       jesus       bible
## 1  1.21014015  1.2014998  0.41743650  0.1621804  0.12698409  0.07464400
## 2  0.51624366  0.1865286  0.09706027  0.0675347  0.05333966  0.05836708
## 3  0.24527495  0.1166274  0.32867738  0.5195729  0.26142784  0.23946855
## 4  0.07102323  0.1565155  0.04902918  0.1320602  0.01776986  0.01719220
## 5 -0.12755935 -0.1044230 -0.09075500 -0.1239664 -0.05901846 -0.05243708
##           hair      dress      blonde         mall    shopping      clothes
## 1  2.59053048  0.5312082  0.36322464  0.622896285  0.27607550  1.245121599
## 2 -0.05146837  0.0492724 -0.01238629 -0.087713363 -0.03710273 -0.004395251
## 3  0.35590025  0.5837827  0.03301526  0.808620531  1.07073115  0.616207360
## 4  0.01714820 -0.0653358  0.03690938 -0.004723697  0.03497875  0.016201064
## 5 -0.19220150 -0.1286412 -0.02793327 -0.179127117 -0.21816580 -0.177738408
##      hollister abercrombie          die       death       drunk       drugs
## 1  0.31525537   0.4131560  1.712160983  0.94713629  1.83371069  2.73878856
## 2 -0.16788599  -0.1413652  0.008941101  0.05464759 -0.08699556 -0.06414588
## 3  0.85951603   0.7935060  0.062399295  0.12642222  0.03594162 -0.05888141
## 4 -0.08381546  -0.0861708 -0.067312427 -0.01611162 -0.06891763 -0.08795059
## 5 -0.16182051  -0.1545430 -0.085876102 -0.06882571 -0.08386703 -0.10777278
```

Step 5 – improving model performance

```
teens$cluster <- teen_clusters$cluster
teens[1:5, c("cluster", "gender", "age", "friends")]
```

```
##   cluster gender    age friends
## 1       5      M 18.982       7
## 2       3      F 18.801       0
## 3       5      M 18.335      69
## 4       5      F 18.875       0
## 5       1   <NA> 18.995      10
```

```
aggregate(data = teens , age ~ cluster, mean)
```

```
##   cluster      age
## 1       1 17.09319
## 2       2 17.38488
## 3       3 17.03773
## 4       4 17.03759
## 5       5 17.30265
```

```
aggregate(data = teens, female ~ cluster, mean)
```

```
##   cluster    female
## 1       1 0.8025048
## 2       2 0.7237937
## 3       3 0.8866208
## 4       4 0.6984421
## 5       5 0.7082735
```

```
aggregate(data = teens, friends ~ cluster, mean)
```

```
##   cluster  friends
## 1       1 30.66570
## 2       2 32.79368
## 3       3 38.54575
## 4       4 35.91728
## 5       5 27.79221
```

Problem 2.

Ques 1. What are some of the key differences between SVM and Random Forest for classification? When is each algorithm appropriate and preferable? Provide examples:

SVMs : Classification is done using hyperplanes, it is mostly used for classification problems with two classes. SVM can also be used for classification or numeric prediction problems. SVMs are a black box algorithm which means it is less interpretable. SVM is more effort demanding than Random Forest since finding the best model requires the testing of various combinations of kernels and model parameters.SVM is less subject to changes of data than Random Forest. SVM can be used in text categorization such as identification of the language used in a document or classification of documents by subject matter also for Optical Chracter Recognition.

Random Forests : It is used for classification problems with more than two classes. Random Forest is prone to overfitting as compared to SVM. The result of the random forest classification is the probability of belonging to a class. Small changes in data can result in large changes in decisions thus varies final predictions. Random forests could be used for Credit scoring models, Diagnosis of medical conditions.

Ques 2. Why might it be preferable to include fewer predictors over many?

It is to avoid redundancy and irrelevance. It is unnecessary to contain those variables in the model since it will not only lower the training speed but also not elevate the model performance.If we include too many features it will be hard for intrepretation and can be time consuming for models that work on black box algorithms. Another problem is overfitting of model, it can work well on trining data but tends to perform poor on unseen data.

Ques 3. You are asked to provide R-Squared for a kNN regression model. How would you respond to that request?

R- square is generally used in estimation of accuracy of linear regression models. As KNN algorithm works on finding distance betwwen data then select n nearest observations to make predictions. As there is no regression model, thus we cannot find fitness of the model which is measured by R-squared.

Ques 4. How can you determine which features to include when building a multiple regression model?

We can you determine which features to include when building a multiple regression model using backward and forward elimination methods. Forward selection begins with an empty equation. Predictors are added one at a time beginning with the predictor with the highest correlation with the dependent variable. Whereas, backward elimination works in the reverse process that is, all the independent variables are entered into the equation first and each one is deleted one at a time if they do not contribute to the regression equation. In terms of elimination or feature addition, the selection can be done on multiple metrics - p value, AIC, R-squared values. step() function is helpful in performing these methods.