Assignment 5 (Machine Learning)

Krishu Thapa

2022-11-01

Question 1

```r
# Loading the wine data.
wineQuality = read.csv('winequality-red.csv', sep=',', header = TRUE)
```

1a.

The multiple linear regression on the pH response using all the predictors except fixed_acidity is shown below:

```r
lm.fit=lm(pH~.-fixed_acidity,data=wineQuality)

summary(lm.fit)
```

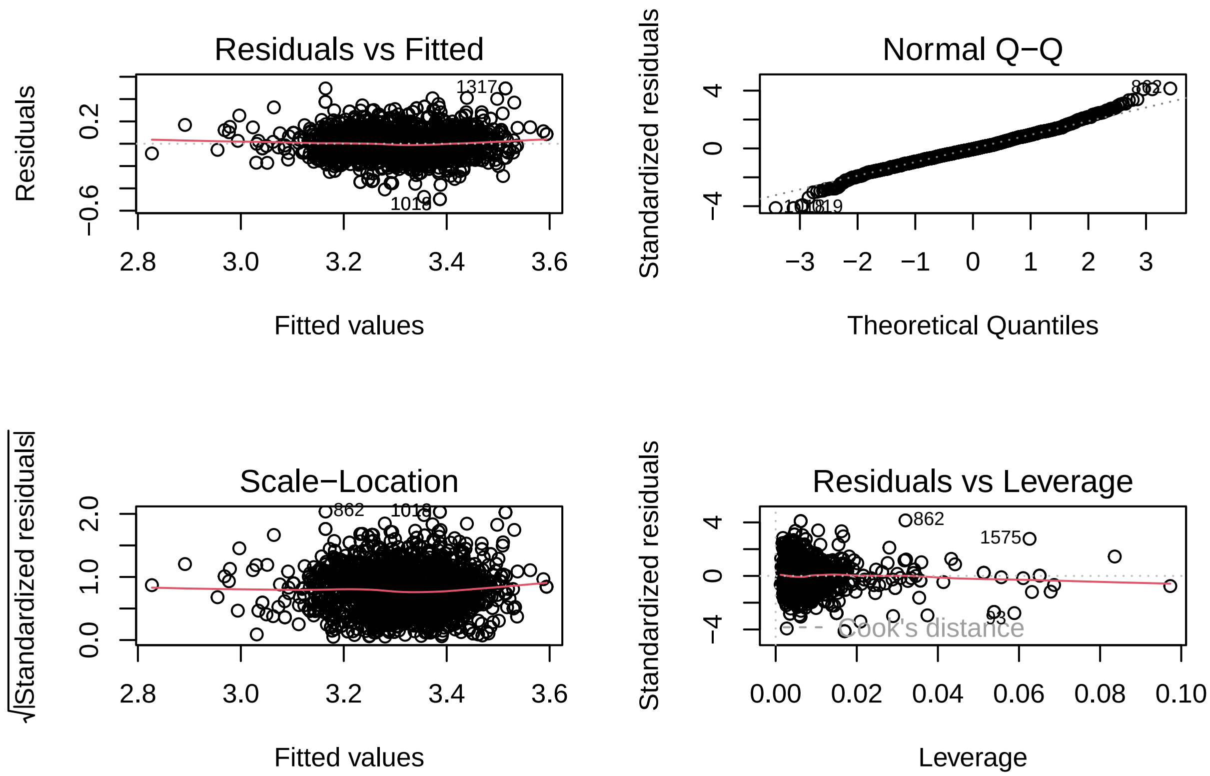
```
##
## Call:
## lm(formula = pH ~ . - fixed_acidity, data = wineQuality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.49686 -0.07741 -0.00613  0.07701  0.49572
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.3570729  2.5169763   0.539  0.58985
## volatile_acidity     -0.0324090  0.0232021  -1.397  0.16266
## citric_acid          -0.4326611  0.0236253 -18.313  < 2e-16 ***
## residual_sugar       -0.0029144  0.0025352  -1.150  0.25050
## chlorides            -0.3214631  0.0765066  -4.202 2.80e-05 ***
## free_sulfur_dioxide   0.0014447  0.0004032   3.583  0.00035 ***
## total_sulfur_dioxide -0.0002790  0.0001336  -2.088  0.03693 *
## density               1.7920994  2.5077019   0.715  0.47494
## sulphates            -0.0063721  0.0215864  -0.295  0.76789
## alcohol               0.0422034  0.0042522   9.925  < 2e-16 ***
## quality              -0.0191678  0.0046761  -4.099 4.36e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1214 on 1588 degrees of freedom
## Multiple R-squared:  0.3857, Adjusted R-squared:  0.3818
## F-statistic:  99.7 on 10 and 1588 DF,  p-value: < 2.2e-16
```

i. From the summary above we can see that the statistically significant predictors in our model are citric_acid, chlorides , free_sulfur_dioxide, alcohol and quantity because they have the p-value less than 0.01 , which shows that they are significant to reject their null hypotheis.

ii. The coefficient of free_sulfur_dioxide is that if there is change in the value of sulfur_dioxide by +1 or -1 , the value of pH changes by +0.001447 or -0.001447 respectively which is the coefficient value.

1b. The diagnostic plot for the above regression model is shown below:

```
par(mfrow=c(2,2))
plot(lm.fit)
```



No, the leverage plot doesnot show any unusually high leverage as no points lie outside the cook's distance as seen in the plot above. From the residual plot we can see that there are few outliers like the points 1317, 1018 etc.

1c. For the interaction with the alcohol we obatin the following models:

```
# For density and alcohol
lm.fit1 = lm(pH~.-fixed_acidity+ density*alcohol, data = wineQuality)
summary(lm.fit1)$coefficients[,4]
```

```
##          (Intercept)     volatile_acidity          citric_acid
##         2.039398e-03         2.497219e-01         3.344873e-67
##       residual_sugar            chlorides  free_sulfur_dioxide
##         1.190213e-01         2.182108e-05         2.350150e-04
## total_sulfur_dioxide              density            sulphates
##         1.605190e-02         9.505836e-04         7.186692e-01
##              alcohol              quality      density:alcohol
##         1.105169e-03         3.200207e-05         1.239411e-03
```

```
# For interaction of residual_sugar and alcohol
lm.fit2 = lm(pH~.-fixed_acidity+ residual_sugar*alcohol, data = wineQuality)
summary(lm.fit2)$coefficients[,4]
```

```
##             (Intercept)       volatile_acidity            citric_acid
##            6.615849e-01           1.717713e-01           5.556184e-67
##          residual_sugar              chlorides    free_sulfur_dioxide
##            1.228919e-01           2.535154e-05           1.383752e-03
##    total_sulfur_dioxide                density              sulphates
##            6.380471e-02           4.347886e-01           7.376950e-01
##                 alcohol                quality residual_sugar:alcohol
##            3.538745e-14           3.142359e-05           8.677542e-02
```

```
# For interaction quality and alcohol
lm.fit3 = lm(pH~.-fixed_acidity+ quality*alcohol, data = wineQuality)
summary(lm.fit3)$coefficients[,4]
```

```
##          (Intercept)     volatile_acidity          citric_acid
##         5.694933e-01         1.799224e-01         3.833611e-67
##       residual_sugar            chlorides  free_sulfur_dioxide
##         2.536785e-01         3.439700e-05         4.060102e-04
## total_sulfur_dioxide              density            sulphates
##         5.028663e-02         5.633128e-01         7.663612e-01
##              alcohol              quality      alcohol:quality
##         1.735121e-03         5.129480e-01         2.450643e-01
```

The p values for lm.fit1 ,lm.fit2 and lm.fit3 are 0.00123 , 0.086 and 0.245 respectively. Hence we can say that the interaction in model lm.fit1 (i.e density and alcohol) is the most significant interaction as its p value is lowest and <0.01.

Question 2.

2a. Loading the boston data set from the mass library in R and seting up the model for each predictors.

```
library(MASS)

suppressMessages(attach(Boston))

lm.fit1=lm(crim~zn)
lm.fit2=lm(crim~indus)
lm.fit3=lm(crim~chas)
lm.fit4=lm(crim~nox)
lm.fit5=lm(crim~rm)
lm.fit6=lm(crim~age)
lm.fit7=lm(crim~dis)
lm.fit8=lm(crim~rad)
lm.fit9=lm(crim~tax)
lm.fit10=lm(crim~ptratio)
lm.fit11=lm(crim~black)
lm.fit12=lm(crim~lstat)
lm.fit13=lm(crim~medv)
```

2b. Running the above codes we found out that the statisitically significant predictors are zn, indus , nox, rm , age, dis ,rad ,tax, ptratio, black , lstat and medv since all of their

```r
# For nox
summary(lm.fit4)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -13.71988   1.699479 -8.072992 5.076814e-15
## nox          31.24853   2.999190 10.418989 3.751739e-23
```

```r
# For chas
summary(lm.fit3)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  3.744447  0.3961111  9.453021 1.239505e-19
## chas        -1.892777  1.5061155 -1.256727 2.094345e-01
```

```r
# For rm
summary(lm.fit5)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 20.481804  3.3644742  6.087669 2.272000e-09
## rm          -2.684051  0.5320411 -5.044819 6.346703e-07
```

```r
# For dis
summary(lm.fit7)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  9.499262  0.7303972 13.005611 1.502748e-33
## dis         -1.550902  0.1683300 -9.213458 8.519949e-19
```

```r
# For medv
summary(lm.fit13)$coefficients
```

```
##               Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 11.7965358 0.93418916 12.62757 5.934119e-32
## medv        -0.3631599 0.03839017 -9.45971 1.173987e-19
```

For nox,

There is a statistically significant association between this predictor and crim as $p < 0.01$. Also, for each unit change in the nox value there is the change in value of crim bt 31.2483 in the same direction which is the coefficient here. This is the most significant predictor as seen from the p-value in comparision to other predictors.

For chas,

Here, the association of the predictor is not statistically significant with the response crim as the p-value for chas is > 0.01.

For rm,

The association is statistically significant since $p < 0.01$ and for the unit increase in the value of predictor the value of crim decreases by 2.68 and vice versa.

For dis,

The association is statistically significant since p < 0.01 and for the unit increase in the value of predictor the value of crim decreases by 1.55 and vice versa.

For medv,

The association is statistically significant since p < 0.01 and for the unit increase in the value of predictor the value of crim decreases by 0.36 and vice versa.

2c. The multiple linear regression model is shown below:

```
lm.fitall = lm(crim~.,data= Boston)

summary(lm.fitall)
```

```
##
## Call:
## lm(formula = crim ~ ., data = Boston)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.924 -2.120 -0.353  1.019 75.051
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.033228   7.234903   2.354 0.018949 *
## zn            0.044855   0.018734   2.394 0.017025 *
## indus        -0.063855   0.083407  -0.766 0.444294
## chas         -0.749134   1.180147  -0.635 0.525867
## nox         -10.313535   5.275536  -1.955 0.051152 .
## rm            0.430131   0.612830   0.702 0.483089
## age           0.001452   0.017925   0.081 0.935488
## dis          -0.987176   0.281817  -3.503 0.000502 ***
## rad           0.588209   0.088049   6.680 6.46e-11 ***
## tax          -0.003780   0.005156  -0.733 0.463793
## ptratio      -0.271081   0.186450  -1.454 0.146611
## black        -0.007538   0.003673  -2.052 0.040702 *
## lstat         0.126211   0.075725   1.667 0.096208 .
## medv         -0.198887   0.060516  -3.287 0.001087 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.439 on 492 degrees of freedom
## Multiple R-squared:  0.454,  Adjusted R-squared:  0.4396
## F-statistic: 31.47 on 13 and 492 DF,  p-value: < 2.2e-16
```

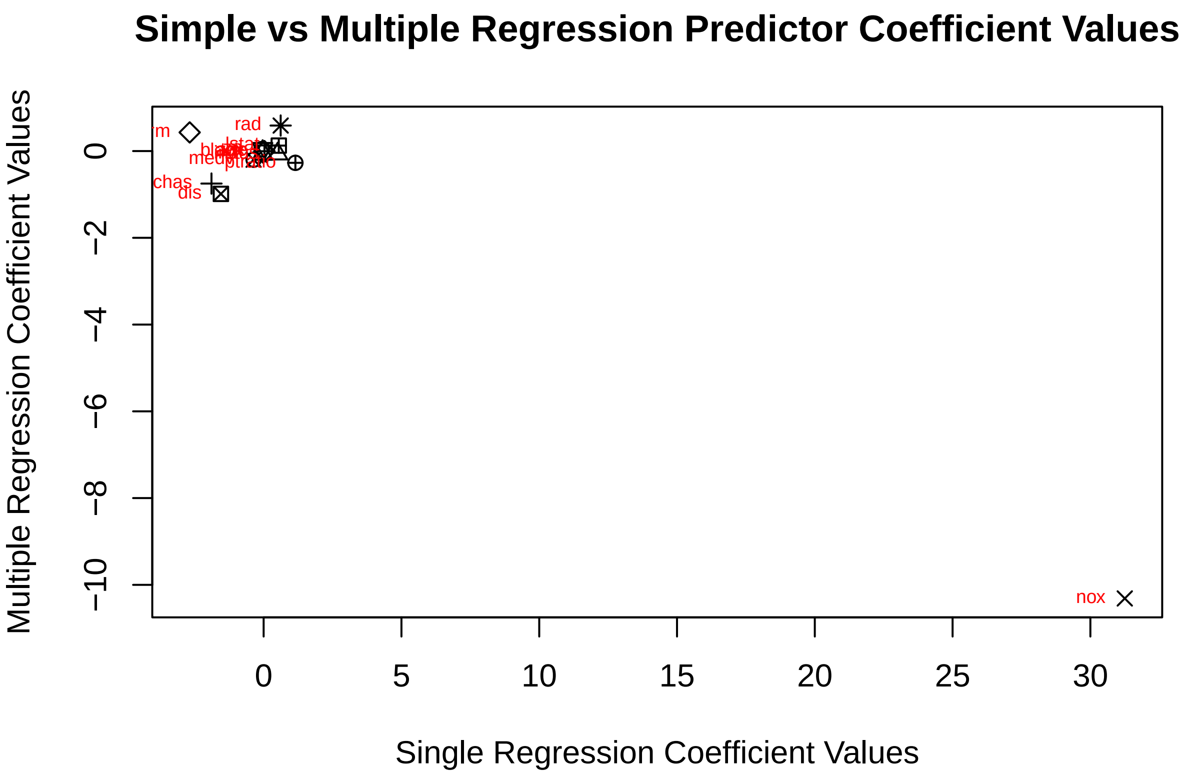
We can see that the pvalue for predictors **dis, rad** is <0.01. Hence we can reject the null hypothesis for these two predictors.

2d. Few of the predictors which had statistically significant association with the crim value when take individually in a. is no longer significant when we take all predictors at once for the model in c. For example: zn , indus , nox , rm ,age etc.

```r
univariateCoef <- c(coef(lm.fit1)[2], coef(lm.fit2)[2],coef(lm.fit3)[2],
                    coef(lm.fit4)[2],coef(lm.fit5)[2],coef(lm.fit6)[2],
                    coef(lm.fit7)[2],coef(lm.fit8)[2],coef(lm.fit9)[2],
                    coef(lm.fit10)[2],coef(lm.fit11)[2],coef(lm.fit12)[2],
                    coef(lm.fit13)[2])

multipleRegressionCoef <- coef(lm.fitall)[2:14]

plot(univariateCoef , multipleRegressionCoef, pch =1:13, xlab = "Single Regression Coefficient Values",
title(main = "Simple vs Multiple Regression Predictor Coefficient Values")
text(univariateCoef, multipleRegressionCoef, names(Boston)[2:14], cex=0.6,
     pos=2, col="red")
```



From the graph above we can notice that most of the coefficients have a low value expect for the one at the bottom right which is nox i.e Unit change in nox causes high value of change in the crim.

2e. Non-linear association of each predictors with the response variable crim is given by:

```r
lm.poly1=lm(crim~poly(zn,3,raw = TRUE))
lm.poly2=lm(crim~poly(indus, 3 , raw = TRUE))
lm.poly3=lm(crim~poly(chas,3,raw = TRUE))
lm.poly4=lm(crim~poly(nox,3, raw = TRUE))
lm.poly5=lm(crim~poly(rm,3, raw = TRUE))
lm.poly6=lm(crim~poly(age,3, raw = TRUE))
lm.poly7=lm(crim~poly(dis,3,raw = TRUE))
lm.poly8=lm(crim~poly(rad,3,raw = TRUE))
lm.poly9=lm(crim~poly(tax,3,raw = TRUE))
lm.poly10=lm(crim~poly(ptratio,3,raw = TRUE))
lm.poly11=lm(crim~poly(black,3,raw = TRUE))
lm.poly12=lm(crim~poly(lstat,3,raw = TRUE))
```

```
lm.poly13=lm(crim~poly(medv,3,raw = TRUE))
```

There seems to be some traces of slight deviation of the polynomial model from the linear fitting line however there are no evidence of complete nonlinearity in any of the predictors with the response variable crim.

Question 3.

3a. Given,

$X_1 = 32$
$X_2 = 3$
$X_3 = 11$

Now,

p(A) $= \frac{e^{\beta_0+\beta_1*X_1+\beta_2*X_2+\beta_3*X_3}}{1+e^{\beta_0+\beta_1*X_1+\beta_2*X_2+\beta_3*X_3}}$

p(A) $= \frac{e^{-8+0.1*32+1*3+-.04*11}}{1+e^{-8+0.1*32+1*3+-.04*11}}$

p(A) = 9.6%

Hence probability of getting A is 9.6%

3b. Given,

$p(A) = 0.65$
$X_2 = 3$
$X_3 = 11$

Now,

$log(\frac{p(x)}{1-p(x)}) = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \beta_3 * X_3$

$log(\frac{.65}{.35}) = -8 + 0.1 * X_1 + 1 * 3 + -.04 * 11$

Solving above equation we get,

$X_2 = 60.59hrs$

Hence to have a 65% chance of getting A , the person must study 60.59 hrs.

3c. Given,

$p(A) = 0.60$
$X_2 = 3$
$X_3 = 3$

Now,

$log(\frac{p(x)}{1-p(x)}) = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \beta_3 * X_3$

$log(\frac{.60}{.40}) = -8 + 0.1 * X_1 + 1 * 3 + -.04 * 3$

Solving above equation we get,

$X_2 = 55.25hrs$

Hence to have a 60% chance of getting A , the person must study 55.25 hrs.

Question 4.

Loading the bbc data:

```
# Loading the bbc data.
bbc = read.csv('bbc.csv', sep=',', header = TRUE)

bbc <- bbc[sample(1:nrow(bbc)), ]
```

4a. The preprocessing and tokenization step is shown below:

```
library(tidyr)
library(tm)
```

```
## Loading required package: NLP
```

```
suppressMessages(library(dplyr))

bbc_stemmed <- stemDocument(bbc$text, language="english")

bbc_corpus <- Corpus(VectorSource(bbc_stemmed))

bbc_corpus <- bbc_corpus %>% tm_map(content_transformer(tolower)) %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace) %>%
  tm_map(removeWords, c("the", "and", stopwords("english")))

bbc_dtm <- DocumentTermMatrix(bbc_corpus)

# Finding 10% least frequent terms.
freq <- colSums(as.matrix(bbc_dtm))
ord <- order(freq, decreasing = FALSE)
least_freq_words <- freq[head(ord, n = 0.1 * ncol(bbc_dtm))]

# Removing the 10% least used word and constructing final document term matrix.
final_corpus <- tm_map(bbc_corpus, removeWords , names(least_freq_words))
final_dtm <- DocumentTermMatrix(final_corpus)

# For the word count in the 2100 data.
bbc_data_frame_final <-  as.data.frame(as.matrix(final_dtm), stringsAsFactors=False)
```

```r
# Showing columns in row 2100 with value >4.
bbc_data_frame_final[2100,] %>% select(where(~ any(. > 4)))
```

```
##      veri game now season win chelsea arsenal mourinho
## 2100    6    6   6      6   5       7       6        8
```

Note: Even after removing the 10% of less frequent terms there are large feature terms nearly 24k which makes the computation very slow.So, for this reference I have only kept the 20% higher frequency word for naive bayes as sparsity is extremely high.

4b. The naive bayes implementation is shown below:

```r
set.seed(7)

suppressMessages(library(caret))
suppressMessages(library(e1071))
suppressMessages(library(quanteda))
suppressMessages(library(randomForest))


# Getting the top 20% frequently repeated terms.

feature_matrix <- dfm_trim(as.dfm(final_dtm), min_docfreq = 100,
min_termfreq = 0.2, termfreq_type = "quantile")


document_matrix <- as.matrix(feature_matrix)

correlated_matrix <- cor(as.matrix(document_matrix))
correlated_terms <- findCorrelation(correlated_matrix , cutoff =.80)

# Removing the corelated terms
document_matrix <- document_matrix[,-c(correlated_terms)]


# Training and testing division

train_size <- floor(0.85 * nrow(document_matrix))

train_x <- document_matrix[1:train_size,]
train_y <- as.factor(bbc[1:train_size,]$category)

test_x <- document_matrix[(train_size+1) : nrow(document_matrix),]
test_y <- as.factor(bbc[(train_size+1): nrow(document_matrix),]$category)

naiveBayesModel <- naiveBayes(train_x,train_y)
prediction<- predict(naiveBayesModel,test_x)

# Confusion matrix

confusion_matrix <- confusionMatrix(prediction,test_y)

print("Confusion Matrix")
```

```
## [1] "Confusion Matrix"
```

```r
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      business entertainment politics sport tech
##   business            69             1        6     0    1
##   entertainment        0            50        1     0    1
##   politics             1             2       58     0    1
##   sport                0             9        0    78    0
##   tech                 2             5        0     0   49
##
## Overall Statistics
##
##                Accuracy : 0.9102
##                  95% CI : (0.8743, 0.9386)
##     No Information Rate : 0.2335
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8872
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: business Class: entertainment Class: politics
## Sensitivity                   0.9583               0.7463          0.8923
## Specificity                   0.9695               0.9925          0.9851
## Pos Pred Value                0.8961               0.9615          0.9355
## Neg Pred Value                0.9883               0.9397          0.9743
## Prevalence                    0.2156               0.2006          0.1946
## Detection Rate                0.2066               0.1497          0.1737
## Detection Prevalence          0.2305               0.1557          0.1856
## Balanced Accuracy             0.9639               0.8694          0.9387
##                      Class: sport Class: tech
## Sensitivity                1.0000      0.9423
## Specificity                0.9648      0.9752
## Pos Pred Value             0.8966      0.8750
## Neg Pred Value             1.0000      0.9892
## Prevalence                 0.2335      0.1557
## Detection Rate             0.2335      0.1467
## Detection Prevalence       0.2605      0.1677
## Balanced Accuracy          0.9824      0.9587
```

```r
# Precision

print("Precision:")
```

```
## [1] "Precision:"
```

```r
confusion_matrix$byClass[1:5,3]
```

```
##      Class: business Class: entertainment      Class: politics
##            0.8961039            0.9615385            0.9354839
##         Class: sport          Class: tech
```

```
##            0.8965517            0.8750000
```

```r
# Recall

print("Recall:")
```

```
## [1] "Recall:"
```

```r
confusion_matrix$byClass[1:5,1]
```

```
##      Class: business Class: entertainment      Class: politics
##            0.9583333            0.7462687            0.8923077
##         Class: sport          Class: tech
##            1.0000000            0.9423077
```