Classification

Matthew Naughton

CS 4375.003

Portfolio: Linear Models

Linear models for classification are very similar to regression, as they are ways to make predictions. However, with classification, we only look at the sign of whatever result we get, negative or positive. Positive means we will choose to predict one class, and negative means the other class. One strength of using it for classification is the accuracy, as linear regression is highly impacted by outliers, this is not as noticable when it is used for classification.

```
SmokeDetection <- read.csv(file = 'smoke_detection_iot.csv')

# a. Divide the Data into train and test data 80/20
i <- sample(1:nrow(SmokeDetection), nrow(SmokeDetection)*0.80, replace=FALSE)
train <- SmokeDetection[i,]
test <- SmokeDetection[-i,]

# b. Use 5 R functions for data exploration using the training data
head(train)

##           X        UTC Temperature.C. Humidity... TVOC.ppb. eCO2.ppm.
Raw.H2
## 58092 58091 1655125513         -9.041       46.30        89       400
12777
## 3232   3231 1654736562          8.718       65.40        39       400
13257
## 56664 56663 1654717708         51.280       13.71      2151       400
13597
## 33510 33509 1654769858         19.440       50.64       350       400
13083
## 22456 22455 1654755786        -18.479       48.41      1366       400
12971
## 36457 36456 1654772805         24.130       53.71      1099       640
12799
##       Raw.Ethanol Pressure.hPa. PM1.0 PM2.5 NC0.5 NC1.0 NC2.5   CNT
Fire.Alarm
## 58092       20636       937.492  2.16  2.24 14.86 2.317 0.052  1205
0
## 3232        20177       939.696  2.56  2.66 17.62 2.747 0.062  3231
1
## 56664       20115       936.682  0.98  1.02  6.74 1.051 0.024  5521
0
## 33510       19910       939.343  0.30  0.31  2.04 0.318 0.007  8515
```

```
1
## 22456       19404       938.721  1.86  1.93 12.78 1.993 0.045 22455
1
## 36457       19455       939.074  2.37  2.46 16.29 2.541 0.057 11462
1
```

```
mean(train$Temperature.C., na.rm=TRUE)
```

```
## [1] 15.98954
```

```
mean(train$Humidity..., na.rm=TRUE)
```

```
## [1] 48.54485
```

```
range(train$Temperature.C.)
```

```
## [1] -22.01  59.93
```

```
range(train$Humidity...)
```

```
## [1] 10.74 75.20
```

```
names(train)
```

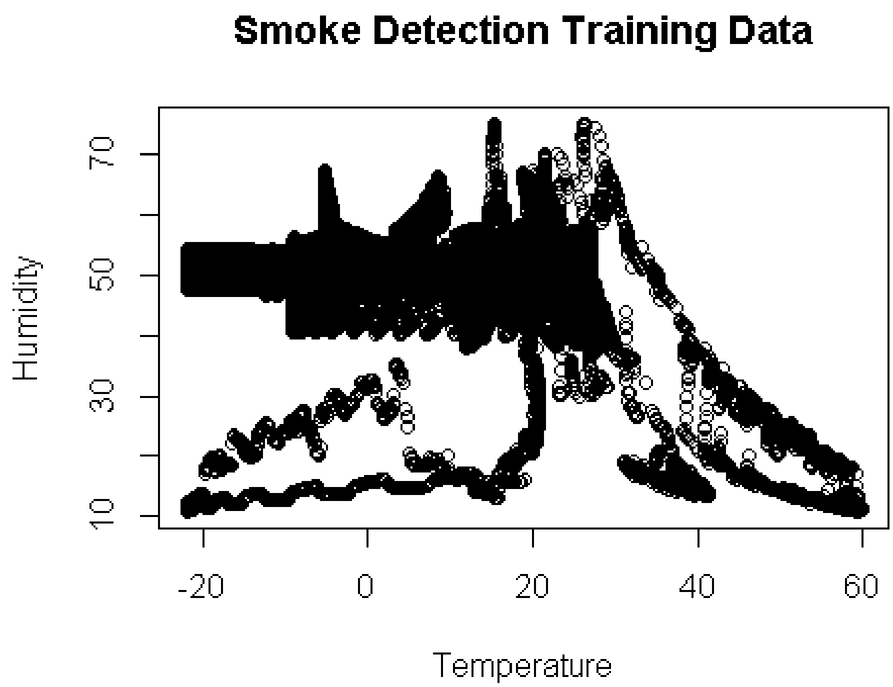
```
##  [1] "X"              "UTC"              "Temperature.C." "Humidity..."
##  [5] "TVOC.ppb."      "eCO2.ppm."        "Raw.H2"         "Raw.Ethanol"
##  [9] "Pressure.hPa."  "PM1.0"            "PM2.5"          "NC0.5"
## [13] "NC1.0"          "NC2.5"            "CNT"            "Fire.Alarm"
```

```
str(train)
```

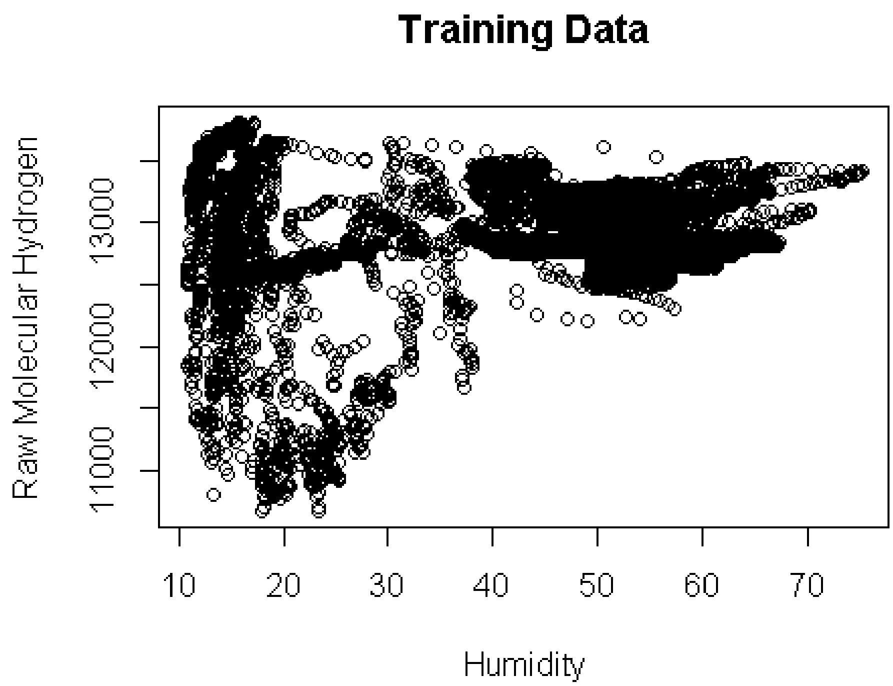
```
## 'data.frame':    50104 obs. of  16 variables:
##  $ X             : int  58091 3231 56663 33509 22455 36456 59266 10028
38986 35135 ...
##  $ UTC           : int  1655125513 1654736562 1654717708 1654769858
1654755786 1654772805 1655126688 1654743359 1654775335 1654771484 ...
##  $ Temperature.C.: num  -9.04 8.72 51.28 19.44 -18.48 ...
##  $ Humidity...   : num  46.3 65.4 13.7 50.6 48.4 ...
##  $ TVOC.ppb.     : int  89 39 2151 350 1366 1099 200 881 1061 944 ...
##  $ eCO2.ppm.     : int  400 400 400 400 400 640 449 720 490 729 ...
##  $ Raw.H2        : int  12777 13257 13597 13083 12971 12799 12762 12768
12854 12765 ...
##  $ Raw.Ethanol   : int  20636 20177 20115 19910 19404 19455 20515 19513
19464 19488 ...
##  $ Pressure.hPa. : num  937 940 937 939 939 ...
##  $ PM1.0         : num  2.16 2.56 0.98 0.3 1.86 2.37 1.87 2.07 1.58 2.34
...
##  $ PM2.5         : num  2.24 2.66 1.02 0.31 1.93 2.46 1.94 2.16 1.64 2.43
...
##  $ NC0.5         : num  14.86 17.62 6.74 2.04 12.78 ...
##  $ NC1.0         : num  2.317 2.747 1.051 0.318 1.993 ...
##  $ NC2.5         : num  0.052 0.062 0.024 0.007 0.045 0.057 0.045 0.05
```

```
0.038 0.057 ...
##  $ CNT           : int  1205 3231 5521 8515 22455 11462 2380 10028 13992
10141 ...
##  $ Fire.Alarm    : int  0 1 0 1 1 1 0 1 1 1 ...

# c. Create 2 informative graphs using the training data
plot(train$Temperature.C., train$Humidity..., xlab="Temperature",
ylab="Humidity", main="Smoke Detection Training Data")
```




```
plot(train$Humidity..., train$Raw.H2, xlab="Humidity", ylab="Raw Molecular
Hydrogen", main="Training Data")
```




```r
# d. Build a logistic regression model and output the summary. Write a
thorough explanation of the information in the summary.
glm1 <- glm(SmokeDetection$Fire.Alarm~SmokeDetection$Temperature.C.,
data=train, family=binomial)
summary(glm1)
```

```
##
## Call:
## glm(formula = SmokeDetection$Fire.Alarm ~ SmokeDetection$Temperature.C.,
##     family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0620  -1.4378   0.7726   0.8875   1.0731
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    1.3920222  0.0154637   90.02   <2e-16 ***
## SmokeDetection$Temperature.C. -0.0275713  0.0006854  -40.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74900  on 62629  degrees of freedom
## Residual deviance: 73130  on 62628  degrees of freedom
## AIC: 73134
```

```
##
## Number of Fisher Scoring iterations: 4
```

To start, the Call reminds us of what we defined when we made the model. Next, we have Deviance Residuals, these are a model of fit. Afterwards, we have the coefficients, along with their Standard Error, the Z-stat, and p-values. Below that are Null and Residual Deviances, and the AIC.

```
# e. Build a Naive Bayes model and output what the model learned. Write an
explanation of the data.
library(e1071)
nb1 <- naiveBayes(train$Fire.Alarm~., data=train)  #Dot "." does all the
columns
nb1

##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.2850271 0.7149729
##
## Conditional probabilities:
##    X
## Y       [,1]     [,2]
##   0 41667.10 21974.24
##   1 27229.93 14314.07
##
##    UTC
## Y         [,1]      [,2]
##   0 1654860678 185113.35
##   1 1654765069  27080.44
##
##    Temperature.C.
## Y       [,1]     [,2]
##   0 19.73904 14.97400
##   1 14.49479 13.84833
##
##    Humidity...
## Y       [,1]      [,2]
##   0 42.97281 11.913599
##   1 50.76617  5.949212
##
##    TVOC.ppb.
## Y       [,1]       [,2]
##   0 4565.059 14196.5040
```

```
##   1  883.065   549.1885
## 
##    eCO2.ppm.
## Y       [,1]     [,2]
##   0 943.2329 2814.378
##   1 551.9583 1229.620
## 
##    Raw.H2
## Y       [,1]     [,2]
##   0 12896.05 428.3781
##   1 12960.95 167.3410
## 
##    Raw.Ethanol
## Y       [,1]     [,2]
##   0 20085.42 950.3704
##   1 19622.73 307.4268
## 
##    Pressure.hPa.
## Y       [,1]     [,2]
##   0 938.1023 1.231646
##   1 938.8362 1.312855
## 
##    PM1.0
## Y        [,1]      [,2]
##   0 258.05347 1430.3282
##   1  36.58954  594.2152
## 
##    PM2.5
## Y        [,1]     [,2]
##   0 445.74016 2839.960
##   1  79.60856 1511.891
## 
##    NC0.5
## Y        [,1]     [,2]
##   0 1329.7735 7057.707
##   1  147.3262 2145.535
## 
##    NC1.0
## Y       [,1]     [,2]
##   0 489.4012 3165.495
##   1  89.0146 1710.576
## 
##    NC2.5
## Y       [,1]    [,2]
##   0 178.6811 1467.01
##   1  41.2613  909.74
## 
##    CNT
## Y        [,1]     [,2]
```

```
##   0  2404.044 1560.657
##   1 13756.070 6577.378
```

Ignoring some of the extreme values due to constants, we can see that some things have increased probability. For example, when the humidity was lower, the probability of a fire alarm increased, same as the temperature but not as sensitive.

```
# f. Predict and evaluate on the test data.
p1 <- predict(nb1, newdata=test, type="class")
table(p1, test$Fire.Alarm)

##
## p1     0    1
##   0 1878  217
##   1 1714 8717

mean(p1==test$Fire.Alarm)

## [1] 0.8458407

p1_raw <- predict(nb1, newdata = test, type="raw")
head(p1_raw)

##                 0         1
## [1,] 0.0094979472 0.9905021
## [2,] 0.0038270720 0.9961729
## [3,] 0.0014845913 0.9985154
## [4,] 0.0007788558 0.9992211
## [5,] 0.0006960389 0.9993040
## [6,] 0.0006336922 0.9993663
```

Looking at the data, we can see that the difference in mean is much higher even though it's only by one year. This information tells us that it has a high predictive value.

g. Write a paragraph listing the strengths and weaknesses of Naive Bayes and Logistic Regression.

Naive Bayes has a higher bias and a lower variance. The results are analyzed such that we can much more easily make predictions with fewer variables and overall less data. This algorithm gives us a faster solution for a few training sets while still considering independent features. On the other hand, Logistic Regression has a low bias and higher variance. We can use categorical and continuous variables to predict the probability. Whenever there are more classes, multi-class logistic regression should be used for data analysis.

h. Write a paragraph listing the benefits, drawbacks of each of the classification metrics used.

Each of the classification methods tell us something different, such as our accuracy. We can use this to determine how predictable certain values are. Some of the data in this set contained constant values which seemed to mess with the accuracy of what each metric was telling us. The sensitivity and specificity measure the true positive and true negative rate respectively.