

Classification

Created by John Lawler - JML190001

CS4375.004 with Karen Mazidi

Created on 2/13/2023, last worked on 2/20/2023

Introduction

Linear models for classification are a method that uses a linear function to separate data into different classes. This is done by assigning weight to predictor variables and determining a model that can predict whether a value falls within a threshold or not. The strengths of these linear models are their simplicity and interpretability as they are easy to understand, and use in real world applications. The limitations of this mainly lie in the fact that complex data sets or data sets that use more than two variables will find the data learned from these methods unhelpful.

Getting the File

To run any of the chunks of R script in this file, please insert *weatherAUS.csv* (<https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package>) into the same folder as this file. You can also find this csv on my Github (<https://github.com/Billy-Budd/mensch-maschine/blob/main/blue-monday/weatherAUS.csv>) repository. Any other information that pertains to other documents can be found in the read me (<https://github.com/Billy-Budd/mensch-maschine#readme>). Unfortunately, the prediction function would not work with NA data, so I had to remove it which probably skews the overall results of this. Major sections have headers that are bolded and there are some intermediary sections that just have headers and are not bolded.

```
df <- read.csv("weatherAUS.csv", header = TRUE)
df <- na.omit(df) # remove missing data
```

Creating an 80/20 Split

a. Divide into 80/20 train/test

We set the seed so that our split does not change between runs of the `sample()` function. We then separate it into a list called `train` (the 80 split) and `test` (the 20 split). Output lengths of each split. Also outputs the date of data collection start and end to get a sense of the time period of this data.

```
set.seed(1234)
split <- sample(1:nrow(df), nrow(df)*.8, replace=FALSE) # split the data into 80/20 samples
train <- df[ split, ] # set train as the 80 split
test <- df[ -split, ] # set test as the 20 split

# I used MinTemp here to show observations, but any column header would be acceptable
tmp <- c("Number of observations in train:", length(train$MinTemp),
"Number of observations in test:", length(test$MinTemp),
"Date of data collection start: ", min(df$Date),
"Date of data collection end: ", max(df$Date))
cat(tmp, sep = '\n')
```

```
## Number of observations in train:
## 45136
## Number of observations in test:
## 11284
## Date of data collection start:
## 2007-11-01
## Date of data collection end:
## 2017-06-25
```

Summary Data

b. Use at least 5 R functions for data exploration, using the training data

This is some data from the training data, and below it are some tables that show the frequency of occurrence for some of the variables we will be looking at in our classification data.

```
head(train)
```

	Date <chr>	Location <chr>	MinTemp <dbl>	MaxTemp <dbl>	Rainfall <dbl>	Evaporation <dbl>	Sunshine <dbl> ▶
104049	2013-05-04	Nuriootpa	10.9	16.2	0.0	2.6	5.2
104142	2013-08-05	Nuriootpa	9.6	18.6	0.8	0.8	9.9
106046	2010-04-29	Woomera	10.4	21.3	0.0	4.6	9.7
46691	2010-11-08	Canberra	9.4	24.2	4.4	5.4	7.5
88792	2013-07-08	Cairns	22.3	27.6	0.2	7.0	8.8
94459	2012-03-23	Townsville	24.0	31.1	0.4	7.0	5.9

6 rows | 1-8 of 24 columns

```
tail(train)
```

	Date <chr>	Location <chr>	MinTemp <dbl>	MaxTemp <dbl>	Rainfall <dbl>	Evaporation <dbl>	Sunshine <dbl> ▶
65178	2011-10-14	MelbourneAirport	6.8	25.0	0.0	4.0	11.1
139250	2008-11-16	Darwin	25.6	35.0	0.0	7.4	8.7
120115	2016-01-19	PerthAirport	12.8	25.9	0.0	5.2	5.5
106938	2012-11-06	Woomera	14.5	27.1	9.4	11.0	5.4
78695	2010-12-07	Watsonia	19.9	28.4	0.0	6.0	1.0
75329	2009-12-15	Portland	6.8	31.7	0.0	5.4	13.2

6 rows | 1-8 of 24 columns

```
summary(train)
```

```
##      Date      Location      MinTemp      MaxTemp
## Length:45136      Length:45136      Min.   :-6.70      Min.    : 6.30
## Class :character  Class :character  1st Qu.: 8.60      1st Qu.:18.70
## Mode  :character  Mode  :character  Median :13.20      Median :23.90
##                                     Mean   :13.46      Mean   :24.22
##                                     3rd Qu.:18.40      3rd Qu.:29.70
##                                     Max.    :31.40      Max.    :48.10
##      Rainfall      Evaporation      Sunshine      WindGustDir
## Min.   : 0.000      Min.   : 0.000      Min.   : 0.00      Length:45136
## 1st Qu.: 0.000      1st Qu.: 2.800      1st Qu.: 5.10      Class :character
## Median : 0.000      Median : 5.000      Median : 8.60      Mode  :character
## Mean   : 2.148      Mean   : 5.511      Mean   : 7.73
## 3rd Qu.: 0.600      3rd Qu.: 7.400      3rd Qu.:10.70
## Max.   :206.200      Max.   :72.200      Max.   :14.50
## WindGustSpeed      WindDir9am      WindDir3pm      WindSpeed9am
## Min.   : 9.00      Length:45136      Length:45136      Min.   : 2.00
## 1st Qu.:31.00      Class :character  Class :character  1st Qu.: 9.00
## Median :39.00      Mode  :character  Mode  :character  Median :15.00
## Mean   :40.87                                     Mean :15.65
## 3rd Qu.:48.00                                     3rd Qu.:20.00
## Max.   :124.00                                     Max.   :67.00
## WindSpeed3pm      Humidity9am      Humidity3pm      Pressure9am
## Min.   : 2.00      Min.   : 0.00      Min.   : 0.00      Min.   : 980.5
## 1st Qu.:13.00      1st Qu.:55.00      1st Qu.:35.00      1st Qu.:1012.7
## Median :19.00      Median :67.00      Median :51.00      Median :1017.2
## Mean   :19.77      Mean   :65.91      Mean   :49.61      Mean   :1017.2
## 3rd Qu.:26.00      3rd Qu.:79.00      3rd Qu.:63.00      3rd Qu.:1021.8
## Max.   :76.00      Max.   :100.00      Max.   :100.00      Max.   :1040.4
## Pressure3pm      Cloud9am      Cloud3pm      Temp9am
## Min.   : 977.1      Min.   :0.000      Min.   :0.000      Min.   : -0.70
## 1st Qu.:1010.1      1st Qu.:1.000      1st Qu.:2.000      1st Qu.:13.10
## Median :1014.7      Median :5.000      Median :5.000      Median :17.80
## Mean   :1014.8      Mean   :4.243      Mean   :4.326      Mean   :18.20
## 3rd Qu.:1019.4      3rd Qu.:7.000      3rd Qu.:7.000      3rd Qu.:23.23
## Max.   :1038.9      Max.   :8.000      Max.   :9.000      Max.   :39.40
## Temp3pm      RainToday      RainTomorrow
## Min.   : 4.60      Length:45136      Length:45136
## 1st Qu.:17.40      Class :character  Class :character
## Median :22.40      Mode  :character  Mode  :character
## Mean   :22.71
## 3rd Qu.:27.90
## Max.   :46.10
```

```
str(train)
```

```
## 'data.frame':   45136 obs. of  23 variables:
## $ Date       : chr  "2013-05-04" "2013-08-05" "2010-04-29" "2010-11-08" ...
## $ Location   : chr  "Nuriootpa" "Nuriootpa" "Woomera" "Canberra" ...
## $ MinTemp    : num  10.9 9.6 10.4 9.4 22.3 24 26.1 11.9 12.2 7.3 ...
## $ MaxTemp    : num  16.2 18.6 21.3 24.2 27.6 31.1 34 15.7 19.9 14.8 ...
## $ Rainfall   : num  0 0.8 0 4.4 0.2 0.4 0 0 38.6 0.6 ...
## $ Evaporation : num  2.6 0.8 4.6 5.4 7 7 7.2 8.2 4.6 4 ...
## $ Sunshine   : num  5.2 9.9 9.7 7.5 8.8 5.9 11.1 0.4 10.4 7.2 ...
## $ WindGustDir : chr  "SE" "NW" "S" "N" ...
## $ WindGustSpeed: int  43 59 31 50 63 31 37 37 48 67 ...
## $ WindDir9am  : chr  "E" "NW" "S" "WNW" ...
## $ WindDir3pm  : chr  "E" "NW" "S" "N" ...
## $ WindSpeed9am : int  24 20 11 4 39 15 13 13 22 31 ...
## $ WindSpeed3pm : int  17 33 13 24 48 15 28 17 26 39 ...
## $ Humidity9am  : int  85 79 65 71 68 72 64 75 72 63 ...
## $ Humidity3pm  : int  35 55 45 72 57 61 56 68 62 54 ...
## $ Pressure9am  : num  1026 1016 1027 1017 1020 ...
## $ Pressure3pm  : num  1024 1010 1024 1016 1018 ...
## $ Cloud9am     : int  5 4 6 3 5 7 6 7 2 1 ...
## $ Cloud3pm     : int  7 5 1 5 3 7 2 7 3 7 ...
## $ Temp9am      : num  13.1 13.3 16.5 18.1 25.3 27.2 31 13.2 15.6 10.4 ...
## $ Temp3pm      : num  15.8 17.9 20.8 18.9 25.7 30.3 33.2 14.3 19.6 12.5 ...
## $ RainToday    : chr  "No" "No" "No" "Yes" ...
## $ RainTomorrow : chr  "No" "No" "No" "Yes" ...
## - attr(*, "na.action")= 'omit' Named int [1:89040] 1 2 3 4 5 6 7 8 9 10 ...
## .. attr(*, "names")= chr [1:89040] "1" "2" "3" "4" ...
```

Wind Gust Direction Frequency:

```
windGustDirFreq <- table(train$WindGustDir)
```

Location Frequency:

```
locationFreq <- table(train$Location)
```

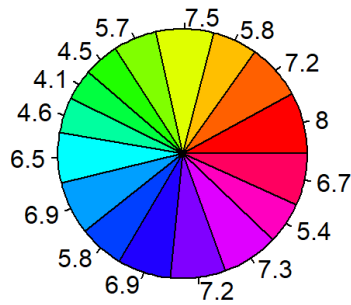
Simple Graphs

c. Create at least 2 informative graphs, using the training data

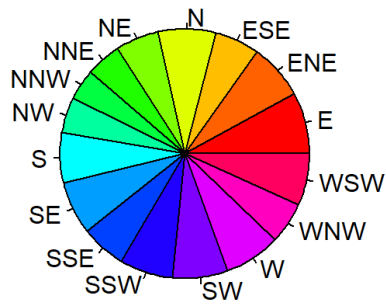
These are some simple graphs to just get an idea of what the data looks like. By the looks of these graphs, Pressure at 9AM seems to be a better predictor than maximum temperature at predicting the amount of rainfall in Australia. The graphs were a bit crowded, so I doubled them up to show percentages and what they are. The location frequency pie chart is still difficult to read, so I added a third graph to supplement. The third graph shows the probability of rain tomorrow given there was rain today. This shows that it rains nonsporadically. Given the green on the first day, it is more likely to rain on the second day.

```
par(mfrow=c(1,2)) # output side by side
piepercent1 <- round(100 * windGustDirFreq / sum(windGustDirFreq), 1) # get percents for graph 1
pie(windGustDirFreq, labels = piepercent1, main = "Wind Direction Frequency", col = rainbow(length(windGustDirFreq))) # percent graph 1
pie(windGustDirFreq, main = "Wind Direction Frequency", col = rainbow(length(windGustDirFreq))) # Labeled graph 1
```

Wind Direction Frequency

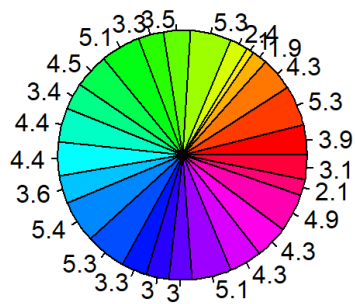


Wind Direction Frequency

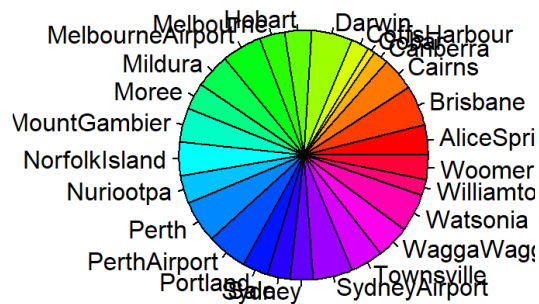


```
par(mfrow=c(1,2)) # output side by side
piepercent2 <- round(100 * locationFreq / sum(locationFreq), 1) # get percents for graph 2
pie(locationFreq, labels = piepercent2, main = "Location Frequency", col = rainbow(length(locationFreq))) # percent graph 2
pie(locationFreq, main = "Location Frequency", col = rainbow(length(locationFreq))) # percent graph 2
```

Location Frequency

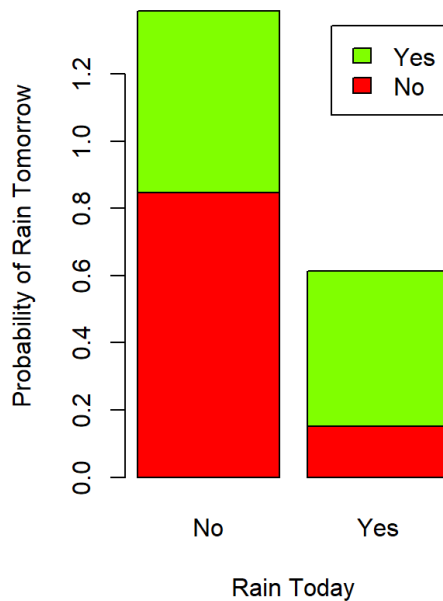


Location Frequency



```
tempTable <- prop.table(table(train$RainToday,train$RainTomorrow), margin = 1) # create a table for bar plot
barplot(tempTable, main = "Probability of Rain Tomorrow Given Rain Today", xlab = "Rain Today", ylab = "Probability of Rain Tomorrow", legend.text = rownames(tempTable), col = rainbow(length(tempTable))) # create bar plot
```

Probability of Rain Tomorrow Given Rain Today



Creating a Simple Linear Model

d. Build a logistic regression model and output the summary. Write a thorough explanation of the information in the model summary.

The deviance residuals show the difference between the observed and predicted models, and judging that the max difference between 0 and 1 is 1, the prediction is not always correct with a max difference of almost 2 and a minimum difference greater than -1. On the other hand, the median, 1Q, and 3Q are the same so the prediction is often off by the exact same amount which is a neat detail. The coefficients here show the change in the log odds of y for every 1 unit in the predictor change. The p-values are very low, which is a good thing. The null and residual deviance here tells us that the difference is statistically significant.

```
train$RainToday <- ifelse(train$RainToday=="Yes",1,0)
train$RainTomorrow <- ifelse(train$RainTomorrow=="Yes",1,0)
glm1 <- glm(RainTomorrow ~ RainToday, data = train, family = "binomial", maxit = 400) # create logistic regression model
summary(glm1)
```

```
##
## Call:
## glm(formula = RainTomorrow ~ RainToday, family = "binomial",
##      data = train, maxit = 400)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.1112  -0.5749  -0.5749  -0.5749   1.9400
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.71661    0.01485 -115.61  <2e-16 ***
## RainToday    1.55884    0.02495   62.48  <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: 47644  on 45135  degrees of freedom
## Residual deviance: 43791  on 45134  degrees of freedom
## AIC: 43795
##
## Number of Fisher Scoring iterations: 3
```

Naive Bayes

e. Build a naïve Bayes model and output what the model learned. Write a thorough explanation of the data.

I did not include date or wind gust direction as they seem irrelevant to this particular calculation as they are not really dependent on the weather, and the date is unique since it includes the year. This function tells us that the prior probability of predicting rain tomorrow given the the independent variables. Here, we can see that we have a 22.069% of prior probabilities to predict rain tomorrow.

```
library(e1071)
nb1 <- naiveBayes(RainTomorrow~.-Date -WindGustDir -WindDir9am -WindDir3pm, data = train)
nb1
```

```

##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##      0      1
## 0.7793114 0.2206886
##
## Conditional probabilities:
##      Location
## Y  AliceSprings  Brisbane  Cairns  Canberra  Cobar  CoffsHarbour
## 0  0.045657427 0.052508884 0.038152097 0.019388770 0.010689410 0.020810235
## 1  0.014958338 0.052605160 0.060636482 0.018371649 0.005521534 0.034635077
##      Location
## Y      Darwin  Hobart  Melbourne MelbourneAirport  Mildura
## 0 0.050291400 0.034029851 0.031499645 0.050945274 0.051769723
## 1 0.062945487 0.038650738 0.037044473 0.051601245 0.022286919
##      Location
## Y      Moree MountGambier NorfolkIsland  Nuriootpa  Perth PerthAirport
## 0 0.037754087 0.040000000 0.038749112 0.037327647 0.056034115 0.054697939
## 1 0.021584178 0.058628652 0.060837265 0.031723723 0.048790282 0.045678145
##      Location
## Y      Portland  Sale  Sydney SydneyAirport  Townsville  WaggaWagga
## 0 0.025472637 0.030248756 0.029168443 0.049097370 0.046140725 0.045344705
## 1 0.060034133 0.028009236 0.033430378 0.057925911 0.030619416 0.033631162
##      Location
## Y      Watsonia Williamtown  Woomera
## 0 0.046794598 0.020184790 0.037242360
## 1 0.054914165 0.024997490 0.009938761
##
##      MinTemp
## Y      [,1]      [,2]
## 0 13.16067 6.366739
## 1 14.51064 6.449673
##
##      MaxTemp
## Y      [,1]      [,2]
## 0 24.76681 6.886300
## 1 22.27322 6.836883
##
##      Rainfall
## Y      [,1]      [,2]
## 0 1.182877 4.631336
## 1 5.555838 11.725871
##
##      Evaporation
## Y      [,1]      [,2]
## 0 5.771534 3.820559
## 1 4.590583 3.154288
##
##      Sunshine
## Y      [,1]      [,2]
## 0 8.629248 3.334472
## 1 4.556510 3.392962
##
##      WindGustSpeed
## Y      [,1]      [,2]
## 0 39.22724 12.18668
## 1 46.68286 15.47586
##
##      WindSpeed9am
## Y      [,1]      [,2]
## 0 15.27227 8.077631

```



```
## 1 16.99197 8.979033
##
## WindSpeed3pm
## Y      [,1]      [,2]
## 0 19.37845 8.269777
## 1 21.15400 9.254369
##
## Humidity9am
## Y      [,1]      [,2]
## 0 63.24569 18.38337
## 1 75.30931 15.69075
##
## Humidity3pm
## Y      [,1]      [,2]
## 0 44.71443 17.86888
## 1 66.90674 18.40364
##
## Pressure9am
## Y      [,1]      [,2]
## 0 1018.186 6.549436
## 1 1013.904 7.061550
##
## Pressure3pm
## Y      [,1]      [,2]
## 0 1015.648 6.542779
## 1 1011.787 7.102994
##
## Cloud9am
## Y      [,1]      [,2]
## 0 3.761251 2.770084
## 1 5.943178 2.159871
##
## Cloud3pm
## Y      [,1]      [,2]
## 0 3.782345 2.579391
## 1 6.243550 1.851493
##
## Temp9am
## Y      [,1]      [,2]
## 0 18.26153 6.539919
## 1 17.97265 6.568909
##
## Temp3pm
## Y      [,1]      [,2]
## 0 23.37767 6.706086
## 1 20.34447 6.692738
##
## RainToday
## Y      [,1]      [,2]
## 0 0.1534897 0.3604640
## 1 0.4629053 0.4986471
```

Classification Models

f. Using these two classification models, predict and evaluate on the test data using all of the classification metrics discussed in class. Compare the results and indicate why you think these results happened.

TODO

```
test$RainToday <- ifelse(test$RainToday=="Yes",1,0)
test$RainTomorrow <- ifelse(test$RainTomorrow=="Yes",1,0)
pred1 <- predict(glm1, newdata = test, type = "response")
acc1 <- mean(pred1)
```

Responses

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

TODO

h. Write a paragraph listing the benefits, drawbacks of each of the classification metrics used, and briefly describe what each metric tells you.

TODO