Classifying Weather Data

Mai H. Nguyen

2016-08-03

Preliminaries

Get data from rattle library

```
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 3.1.4 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
df <- weather
dim(df)
```

```
## [1] 366  24
```

Remove variable RISK_MM, which is the same as Rainfall for the next day

```
df$RISK_MM <- NULL
dim(df)
```

```
## [1] 366  23
```

Partition Data

Divide data into train and test sets

```
# Randomly select 70% of samples from dataset
set.seed(765)                       # Set seed for reproducibility
pct.trn <- 0.7                      # % of data used for training
nrows <- nrow(df)
index <- sample(1:nrows, size = pct.trn * nrows)

# Divide data into train and test sets
df.trn <- df[index,]
df.tst <- df[-index,]

# Statistics on train & test sets
dim(df.trn)
```

```
## [1] 256  23
```

```
table(df.trn$RainTomorrow)/nrow(df.trn)
```

```
##
##        No       Yes
## 0.8320312 0.1679688
```

```
# summary(df.trn)

dim(df.tst)
```

```
## [1] 110  23
```

```
table(df.tst$RainTomorrow)/nrow(df.tst)
```

```
##
##        No       Yes
## 0.7909091 0.2090909
```

```
# summary(df.tst)

# Save datasets.
# write.csv(df.trn, "data/weather-trn.csv",row.names=FALSE)  # Save as CSV f
iles
# write.csv(df.tst, "data/weather-tst.csv",row.names=FALSE)
saveRDS(df.trn, "data/weather-trn.rds")                      # Save as RDS f
iles
saveRDS(df.tst, "data/weather-tst.rds")
```

Analyses

Classification using Decision Tree - Predicting whether it will rain tomorrow

```
library(rpart)
library(rpart.plot)

# Remove variables not useful for classification (Date, Location, RainToday)
df.trn <- subset(df.trn,select=-c(Date,Location,RainToday))
df.tst <- subset(df.tst,select=-c(Date,Location,RainToday))
names(df.trn)               # Columns in datasets after removing some variab
les
```

```
##  [1] "MinTemp"       "MaxTemp"       "Rainfall"      "Evaporation"
##  [5] "Sunshine"      "WindGustDir"   "WindGustSpeed" "WindDir9am"
##  [9] "WindDir3pm"    "WindSpeed9am"  "WindSpeed3pm"  "Humidity9am"
## [13] "Humidity3pm"   "Pressure9am"   "Pressure3pm"   "Cloud9am"
## [17] "Cloud3pm"      "Temp9am"       "Temp3pm"       "RainTomorrow"
```

```
dim(df.trn)                 # Dimensions of datasets after removing some var
iables
```

```
## [1] 256  20
```

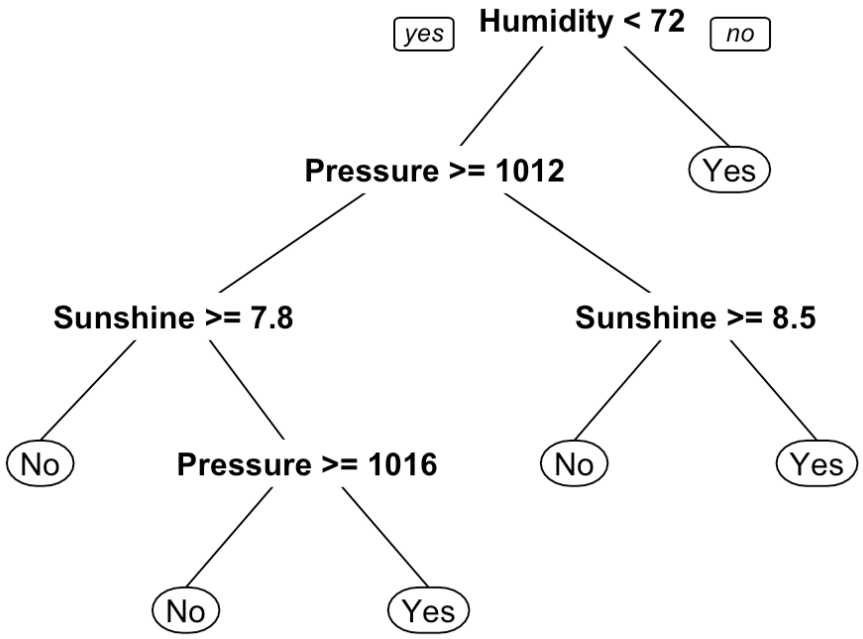
```
dim(df.tst)
```

```
## [1] 110  20
```

```
# Build tree with training data.
# Target variable is RainTomorrow, and input variables are the rest of the v
ariables.
set.seed(567)   # Set seed for x-val
tree.model <- rpart (RainTomorrow ~ .,data=df.trn, method="class")
printcp(tree.model)          # Print summary of trained model
```

```
##
## Classification tree:
## rpart(formula = RainTomorrow ~ ., data = df.trn, method = "class")
##
## Variables actually used in tree construction:
## [1] Humidity3pm Pressure3pm Sunshine
##
## Root node error: 43/256 = 0.16797
##
## n= 256
##
##         CP nsplit rel error  xerror    xstd
## 1 0.255814      0   1.00000 1.00000 0.13910
## 2 0.081395      1   0.74419 0.88372 0.13229
## 3 0.011628      3   0.58140 0.81395 0.12783
## 4 0.010000      5   0.55814 0.81395 0.12783
```

```
rpart.plot(tree.model)       # Plot resulting tree model
```




```
# Prediction error on TRAIN data (i.e., resubstition error)
pred.trn <- predict(tree.model,newdata=df.trn,type="class")
table(actual=df.trn$RainTomorrow,predicted=pred.trn)                # Confus
ion matrix
```

```
##       predicted
## actual  No Yes
##    No  197  16
##    Yes   8  35
```

```
err <- (1 - (sum(pred.trn==df.trn$RainTomorrow)) / nrow(df.trn))    # Miscla
ssification Error
paste("Error on TRAIN Data: ", err)
```

```
## [1] "Error on TRAIN Data:  0.09375"
```

```
# Prediction error on TEST data
pred.tst <- predict(tree.model,newdata=df.tst,type="class")
table(actual=df.tst$RainTomorrow,predicted=pred.tst)                # Confus
ion matrix
```

```
##       predicted
## actual No Yes
##    No  75  12
##    Yes 10  13
```

```
err <- (1 - (sum(pred.tst==df.tst$RainTomorrow)) / nrow(df.tst))    # Miscla
ssification Error
paste("Error on TEST Data: ", err)
```

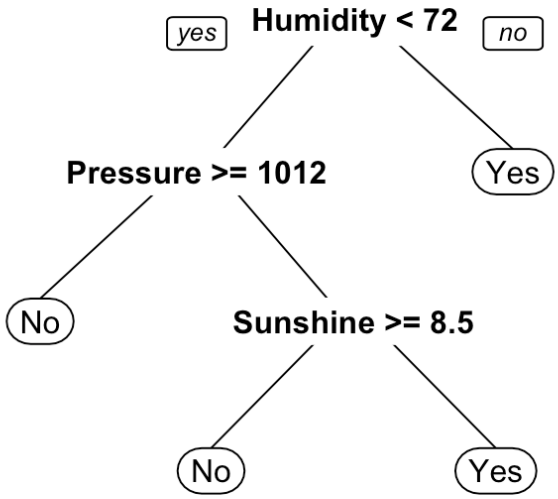
```
## [1] "Error on TEST Data:  0.2"
```

Prune tree if necessary

```
# Get min complexity parameter value from tree
cp.best <- tree.model$cptable[which.min(tree.model$cptable[,"xerror"]),"CP"]
tree.pruned <- prune(tree.model, cp=cp.best)                                #
Set cp criterion for pruning tree
printcp(tree.pruned)                                                        #
Print summary of pruned tree
```

```
##
## Classification tree:
## rpart(formula = RainTomorrow ~ ., data = df.trn, method = "class")
##
## Variables actually used in tree construction:
## [1] Humidity3pm Pressure3pm Sunshine
##
## Root node error: 43/256 = 0.16797
##
## n= 256
##
##         CP nsplit rel error  xerror    xstd
## 1 0.255814      0   1.00000 1.00000 0.13910
## 2 0.081395      1   0.74419 0.88372 0.13229
## 3 0.011628      3   0.58140 0.81395 0.12783
```

```
rpart.plot(tree.pruned)                                                    #
Plot pruned tree
```




```
pred.tst.pruned <- predict(tree.pruned,newdata=df.tst,type="class")        #
Apply tree to test data
table(actual=df.tst$RainTomorrow,predicted=pred.tst.pruned)                #
Confusion matrix
```

```
##       predicted
## actual No Yes
##    No  81   6
##    Yes 13  10
```

```
err <- (1 - (sum(pred.tst.pruned==df.tst$RainTomorrow)) / nrow(df.tst))    #
Misclassification Error
paste("Error on TEST Data (Pruned Tree): ", err)
```

```
## [1] "Error on TEST Data (Pruned Tree):  0.172727272727273"
```

Classification using Random Forest - Predicting whether it will rain tomorrow

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

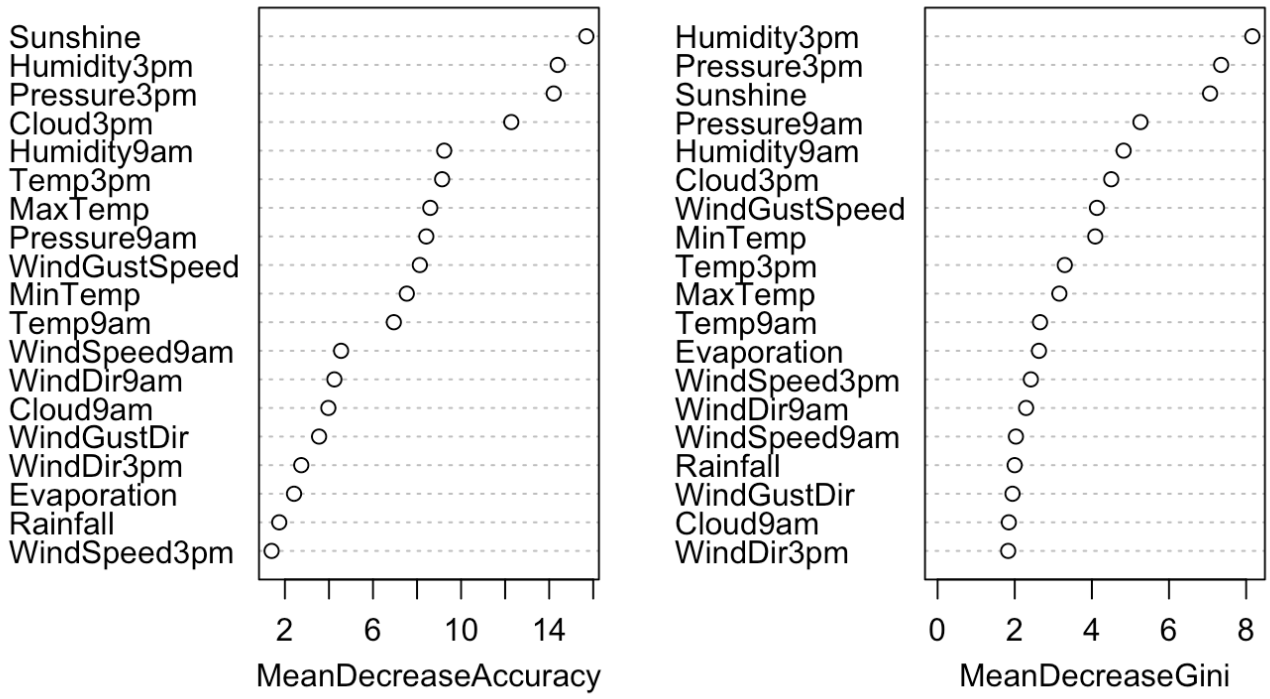
```
# Build random forest from training data.
# Target variable is RainTomorrow, and input variables are the rest of the v
ariables.
# Missing values are imputed.  Importance of variables are calculated.
set.seed(765)          # For reproducibility
rf.model <- randomForest (RainTomorrow ~ .,data=df.trn, na.action=na.roughfi
x, importance=TRUE)
print(rf.model)
```

```
##
## Call:
##  randomForest(formula = RainTomorrow ~ ., data = df.trn, importance = TRU
E,      na.action = na.roughfix)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 12.89%
## Confusion matrix:
##      No Yes class.error
## No  207   6  0.02816901
## Yes  27  16  0.62790698
```

```
# Variable importance
importance(rf.model)
```

```
##                     No        Yes MeanDecreaseAccuracy MeanDecreaseGini
## MinTemp        7.694933  0.1052833             7.534373         4.087539
## MaxTemp        8.690387 -0.7177258             8.598798         3.157335
## Rainfall       2.654562 -0.9834446             1.742222         1.997765
## Evaporation    3.112431 -0.7107073             2.414391         2.628264
## Sunshine      13.157019  8.1340611            15.688026         7.063623
## WindGustDir    4.709697 -1.0264793             3.549826         1.941946
## WindGustSpeed  8.584853  2.6556228             8.126043         4.131167
## WindDir9am     3.665868  2.3384873             4.245033         2.294442
## WindDir3pm     4.064071 -1.3133281             2.742124         1.829328
## WindSpeed9am   3.706726  2.0701231             4.549495         2.029935
## WindSpeed3pm   1.764287 -0.2583209             1.395117         2.417482
## Humidity9am    8.374986  5.3797973             9.231957         4.822744
## Humidity3pm   12.687647  7.9528922            14.386919         8.160509
## Pressure9am    8.507053  1.1854687             8.419845         5.259450
## Pressure3pm   11.170705 10.0607317            14.205019         7.350877
## Cloud9am       3.621276  1.3351240             3.970600         1.846066
## Cloud3pm       9.173409  7.3855950            12.279361         4.501596
## Temp9am        6.504138  1.0863092             6.946406         2.656024
## Temp3pm        9.563329 -0.3212291             9.137713         3.297733
```

```
varImpPlot(rf.model)
```



Evaluate random forest

```
# Impute missing values
sum(is.na(df.tst))                                    # Number of NAs befor
e imputation
```

```
## [1] 15
```

```
df.tst.imp <- rfImpute(RainTomorrow ~ ., data=df.tst)  # Impute missing valu
es
```

```
## ntree      OOB      1      2
##   300:  27.27% 10.34% 91.30%
## ntree      OOB      1      2
##   300:  24.55%  8.05% 86.96%
## ntree      OOB      1      2
##   300:  23.64%  8.05% 82.61%
## ntree      OOB      1      2
##   300:  22.73%  6.90% 82.61%
## ntree      OOB      1      2
##   300:  21.82%  8.05% 73.91%
```

```
sum(is.na(df.tst.imp))                                    # Number of NAs after
 imputation
```

```
## [1] 0
```

```
# Evaluate RF model on test dataset
rf.pred.tst <- predict(rf.model,newdata=df.tst.imp)
table(actual=df.tst.imp$RainTomorrow,predicted=rf.pred.tst)
# Confusion matrix
```

```
##       predicted
## actual No Yes
##    No  87   0
##    Yes 17   6
```

```
err <- (1 - (sum(rf.pred.tst==df.tst.imp$RainTomorrow)) / nrow(df.tst.imp))
# Misclassification Error
paste("RF Error on TEST Data: ", err)
```

```
## [1] "RF Error on TEST Data:  0.154545454545455"
```