Analysis: Classification Trees & Random Forest

Hannah Norman

```
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(mlbench)
library(rattle)
```

```
## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':
##
##     importance

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(ROCR)
```

Original NASA Asteroids Data Set

```
nasa_orig <- read.csv("nasa_original.csv")
# nasa_orig
```

Cleaned NASA Asteroids Data Set

```
nasa <- read.csv("nasa.csv")
# nasa
```

Numerical and Graphical Summaries of Response Variable

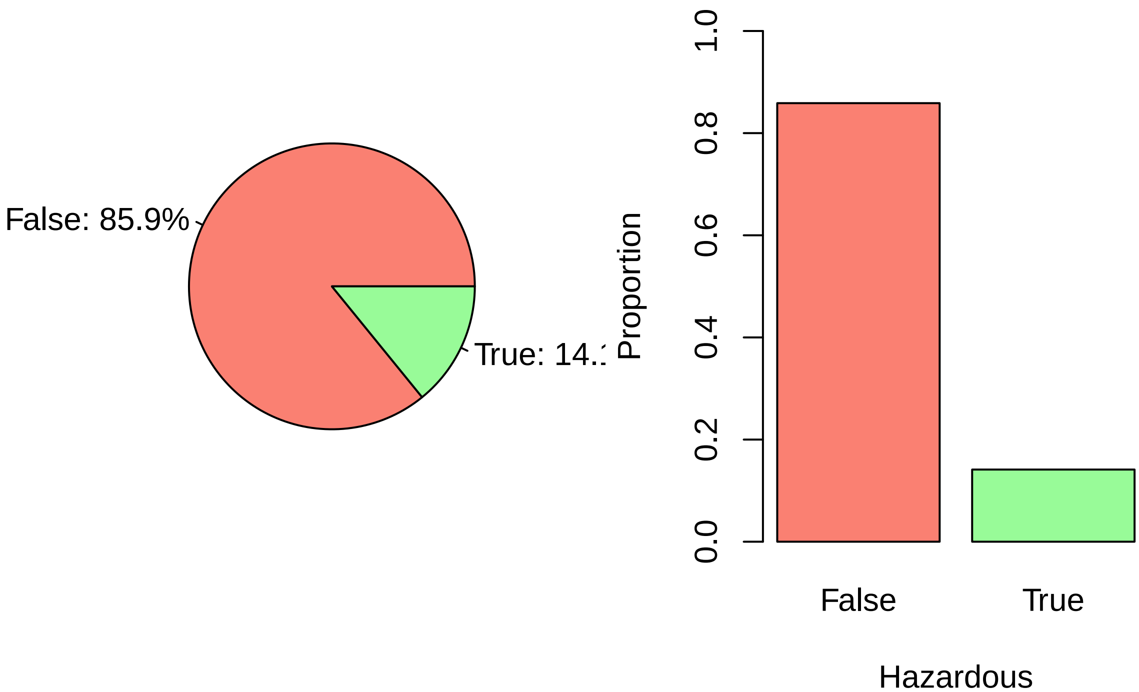
```
prop.hazardous <- prop.table(table(nasa$Hazardous))
prop.hazardous
```

```
##
##     False      True
## 0.8587204 0.1412796
```

```
par(mfrow=c(1,2))

# pie chart
count.hazardous <- table(nasa$Hazardous)
lbls <- paste(levels(as.factor(nasa$Hazardous)), ": ",
              round(prop.hazardous,3)*100, "%", sep="")
pie(count.hazardous, labels=lbls, col=c("salmon", "palegreen"))

# bar plot
barplot(prop.hazardous, xlab="Hazardous", ylab="Proportion", ylim=c(0, 1.0),
        col=c("salmon", "palegreen"))
```



Analyses of Predictors

(in search of outliers that might skew analysis)

```
# numerical summary + box plots
summary(nasa)
```

```
##  Absolute.Magnitude Est.Dia.in.KM.min. Est.Dia.in.KM.max. Est.Dia.in.KM.range
##  Min.   :14.40      Min.   :0.001011   Min.   :0.00226    Min.   :0.001249
##  1st Qu.:20.40      1st Qu.:0.030518   1st Qu.:0.06824    1st Qu.:0.037722
##  Median :22.30      Median :0.092163   Median :0.20608    Median :0.113919
##  Mean   :22.53      Mean   :0.172936   Mean   :0.38670    Mean   :0.213761
```

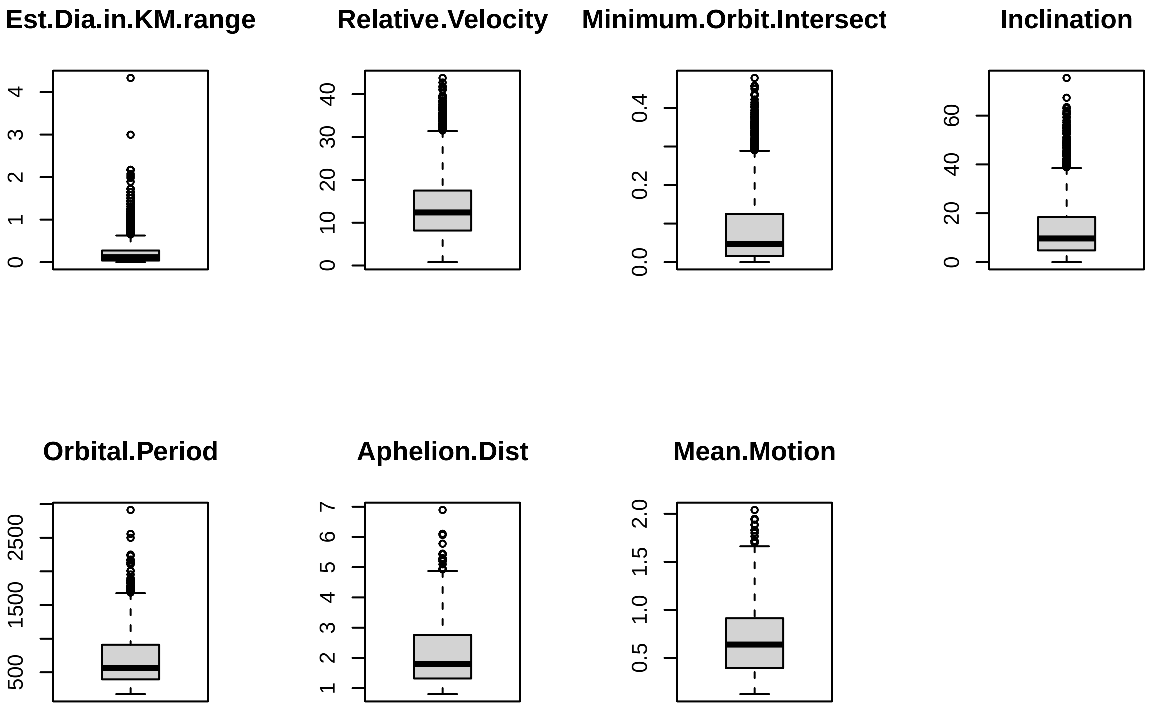
```
##  3rd Qu.:24.70      3rd Qu.:0.221083   3rd Qu.:0.49436    3rd Qu.:0.273273
##  Max.   :32.10      Max.   :3.503926   Max.   :7.83502    Max.   :4.331091
##  Close.Approach.Date Relative.Velocity.in.KM.per.sec Miss.Dist.in.KM
##  Min.   :19950101    Min.   : 0.8002                 Min.   :   26610
##  1st Qu.:20010765    1st Qu.: 8.1683                 1st Qu.:16224266
##  Median :20070908    Median :12.3900                 Median :37032388
##  Mean   :20066238    Mean   :13.6152                 Mean   :36468199
##  3rd Qu.:20120922    3rd Qu.:17.5084                 3rd Qu.:56281652
##  Max.   :20160908    Max.   :43.7899                 Max.   :74781600
##  Orbit.Uncertainity Minimum.Orbit.Intersection Jupiter.Tisserand.Invariant
##  Min.   :0.000      Min.   :0.0000021          Min.   :2.196
##  1st Qu.:1.000      1st Qu.:0.0151384          1st Qu.:3.804
##  Median :5.000      Median :0.0473248          Median :4.798
##  Mean   :4.099      Mean   :0.0823078          Mean   :4.852
##  3rd Qu.:7.000      3rd Qu.:0.1248985          3rd Qu.:5.774
##  Max.   :9.000      Max.   :0.4778910          Max.   :9.025
##   Eccentricity     Semi.Major.Axis   Inclination       Asc.Node.Longitude
##  Min.   :0.01296   Min.   :0.6159   Min.   : 0.01451   Min.   :  0.0019
##  1st Qu.:0.24918   1st Qu.:1.0530   1st Qu.: 4.78919   1st Qu.: 83.6762
##  Median :0.38253   Median :1.3349   Median : 9.68518   Median :173.6808
##  Mean   :0.39329   Mean   :1.4850   Mean   :12.83906   Mean   :173.5626
##  3rd Qu.:0.52595   3rd Qu.:1.8388   3rd Qu.:18.38036   3rd Qu.:258.6316
##  Max.   :0.96026   Max.   :3.9908   Max.   :75.40667   Max.   :359.9059
##  Orbital.Period   Perihelion.Distance Perihelion.Arg     Aphelion.Dist
##  Min.   : 176.6   Min.   :0.08074     Min.   :  0.0069   Min.   :0.8038
##  1st Qu.: 394.7   1st Qu.:0.67808     1st Qu.: 95.6430   1st Qu.:1.3191
##  Median : 563.3   Median :0.87390     Median :188.5239   Median :1.7918
##  Mean   : 692.9   Mean   :0.84320     Mean   :184.1443   Mean   :2.1268
##  3rd Qu.: 910.7   3rd Qu.:1.01935     3rd Qu.:272.5059   3rd Qu.:2.7545
##  Max.   :2912.0   Max.   :1.29983     Max.   :359.9931   Max.   :6.8918
##   Mean.Anomaly       Mean.Motion       Hazardous
##  Min.   :  0.0032   Min.   :0.1236   Length:3079
##  1st Qu.: 83.5492   1st Qu.:0.3953   Class :character
##  Median :183.9847   Median :0.6391   Mode  :character
##  Mean   :180.1871   Mean   :0.6810
##  3rd Qu.:276.3719   3rd Qu.:0.9121
##  Max.   :359.9180   Max.   :2.0390
```

```r
par(mfrow=c(2,4))
boxplot(nasa$Est.Dia.in.KM.range, main="Est.Dia.in.KM.range")
boxplot(nasa$Relative.Velocity, main="Relative.Velocity")
boxplot(nasa$Minimum.Orbit.Intersection, main="Minimum.Orbit.Intersection")
boxplot(nasa$Inclination, main="Inclination")
boxplot(nasa$Orbital.Period, main="Orbital.Period")
boxplot(nasa$Aphelion.Dist, main="Aphelion.Dist")
boxplot(nasa$Mean.Motion, main="Mean.Motion")

# to find row number of outlier observations
nasa[which.max(nasa$Est.Dia.in.KM.range),]
```

```
##     Absolute.Magnitude Est.Dia.in.KM.min. Est.Dia.in.KM.max.
## 695               14.4           3.503926           7.835018
##     Est.Dia.in.KM.range Close.Approach.Date Relative.Velocity.in.KM.per.sec
## 695            4.331091            20001222                        21.19854
##     Miss.Dist.in.KM Orbit.Uncertainity Minimum.Orbit.Intersection
```

```
## 695        21340634                  0                  0.0282524
##     Jupiter.Tisserand.Invariant Eccentricity Semi.Major.Axis Inclination
## 695                       3.573    0.6343498        1.982214    6.705068
##     Asc.Node.Longitude Orbital.Period Perihelion.Distance Perihelion.Arg
## 695           294.8956       1019.352           0.7247968       236.3403
##     Aphelion.Dist Mean.Anomaly Mean.Motion Hazardous
## 695      3.239631       338.28   0.3531656      True
```



Predictive Performance Stats

```
get_stats <- function(CM) {
  TP <- CM[2,2]
  FP <- CM[1,2]
  TN <- CM[1,1]
  FN <- CM[2,1]

  acc <- (TP+TN) / (TP+TN+FN+FP)
  err <- (FP+FN) / (TP+TN+FN+FP)
  pre <- (TP) / (TP+FP)
  sen <- (TP) / (TP+FN)
  spe <- (TN) / (TN+FP)
  fme <- (2*pre*sen) / (pre+sen)
  mcc_denom <- sqrt(TP+FP)*sqrt(TP+FN)*sqrt(TN+FP)*sqrt(TN+FN)
  mcc <- (TP*TN - FP*FN) / mcc_denom

  name <- c("accuracy", "error rate", "precision", "sensitivity", "specificity", "F-measure", "Matthew'
  value <- c(acc, err, pre, sen, spe, fme, mcc)
  stats <- data.frame(name, value)

  return (stats)
}
```

Classification Tree Analysis

1. Make sure that the model assumptions, if any, are satisfied.

No model assumptions of decision trees to be satisfied? We are fitting a classification tree as opposed to a regression tree because the response variable is categorical and binary.

2. Assess the model fit and perform diagnostics, if appropriate.

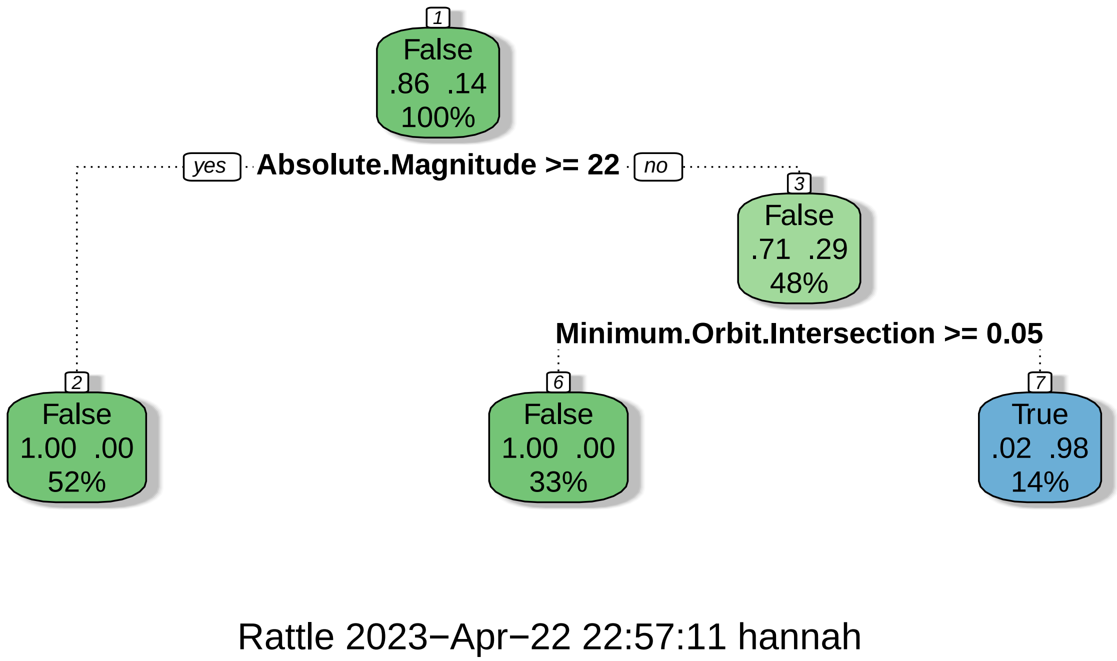
Both methods appear to give identical results. Note the identical accuracy and kappa ratings across all cp tuning parameters in the plain rpart method...

```
set.seed(1)

# rpart
nasa.CVrpart <- train(Hazardous ~ ., data=nasa,
                      method="rpart",
                      tuneGrid = expand.grid(cp=seq(0.005, 0.05, length=10)),
                      trControl=trainControl(method="cv", number=10,
                                             savePredictions=TRUE,
                                             classProbs=TRUE,
                                             selectionFunction = "oneSE"))
nasa.CVrpart
```

```
## CART
##
## 3079 samples
##   20 predictor
##    2 classes: 'False', 'True'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2771, 2771, 2772, 2770, 2771, 2772, ...
## Resampling results across tuning parameters:
##
##   cp     Accuracy  Kappa
##   0.005  0.993829  0.9743262
##   0.010  0.993829  0.9743262
##   0.015  0.993829  0.9743262
##   0.020  0.993829  0.9743262
##   0.025  0.993829  0.9743262
##   0.030  0.993829  0.9743262
##   0.035  0.993829  0.9743262
##   0.040  0.993829  0.9743262
##   0.045  0.993829  0.9743262
##   0.050  0.993829  0.9743262
##
## Accuracy was used to select the optimal model using  the one SE rule.
## The final value used for the model was cp = 0.05.
```

```
# print tree
fancyRpartPlot(nasa.CVrpart$finalModel)
```

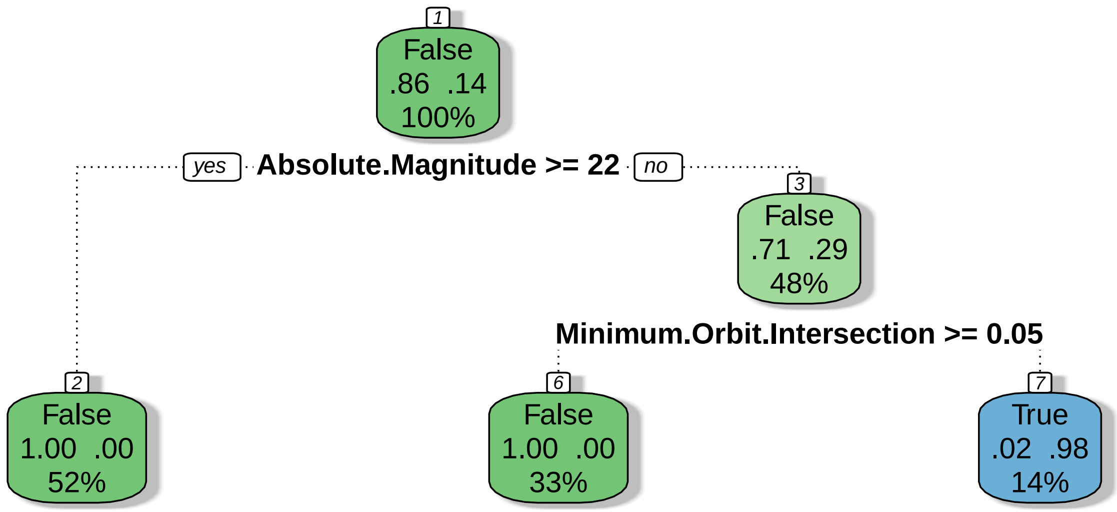



```r
set.seed(1)

# rpart1SE
nasa.CVrpart1SE <- train(Hazardous ~ ., data=nasa,
                         method="rpart1SE",
                         trControl=trainControl(method="cv", number=10,
                                                savePredictions=TRUE))
nasa.CVrpart1SE
```

```
## CART
##
## 3079 samples
##   20 predictor
##    2 classes: 'False', 'True'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2771, 2771, 2772, 2770, 2771, 2772, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.993829  0.9743262
```

```r
# print tree
fancyRpartPlot(nasa.CVrpart1SE$finalModel)
```



Rattle 2023−Apr−22 22:57:12 hannah

3. Identify tuning parameters to be used, if appropriate.

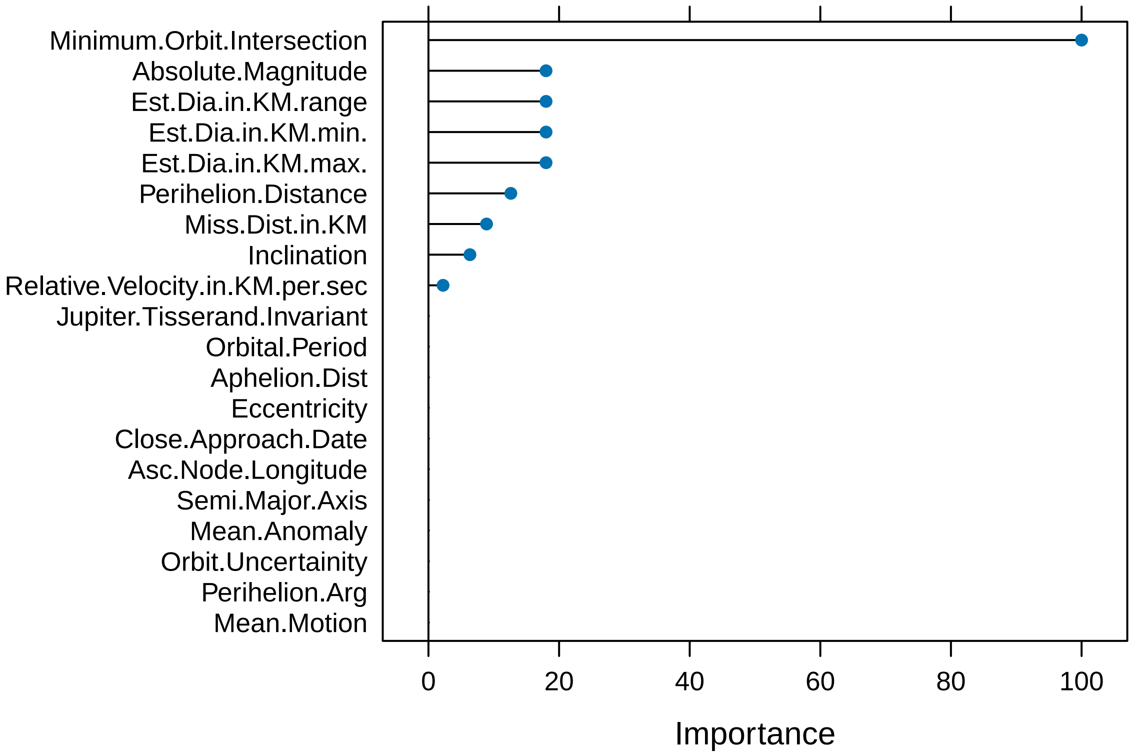
If the plain rpart method is utilized, it selects a cp (complexity parameter) value of 0.05 to maximize accuracy for the final model. Note that the accuracy and kappa values are identical for each of the cp values tried by the model, so any choice within that range should produce comparable results.

4. Identify and interpret the effect of selected variables.

```
# variable importance (rpart)
varImp(nasa.CVrpart)
```

```
## rpart variable importance
##
##                                 Overall
## Minimum.Orbit.Intersection      100.000
## Absolute.Magnitude               18.004
## Est.Dia.in.KM.range              18.004
## Est.Dia.in.KM.min.               18.004
## Est.Dia.in.KM.max.               18.004
## Perihelion.Distance              12.614
## Miss.Dist.in.KM                   8.894
## Inclination                       6.351
## Relative.Velocity.in.KM.per.sec   2.235
## Semi.Major.Axis                   0.000
## Jupiter.Tisserand.Invariant       0.000
## Perihelion.Arg                    0.000
## Close.Approach.Date               0.000
## Aphelion.Dist                     0.000
## Mean.Motion                       0.000
## Mean.Anomaly                      0.000
## Asc.Node.Longitude                0.000
## Orbit.Uncertainity                0.000
## Orbital.Period                    0.000
## Eccentricity                      0.000
```

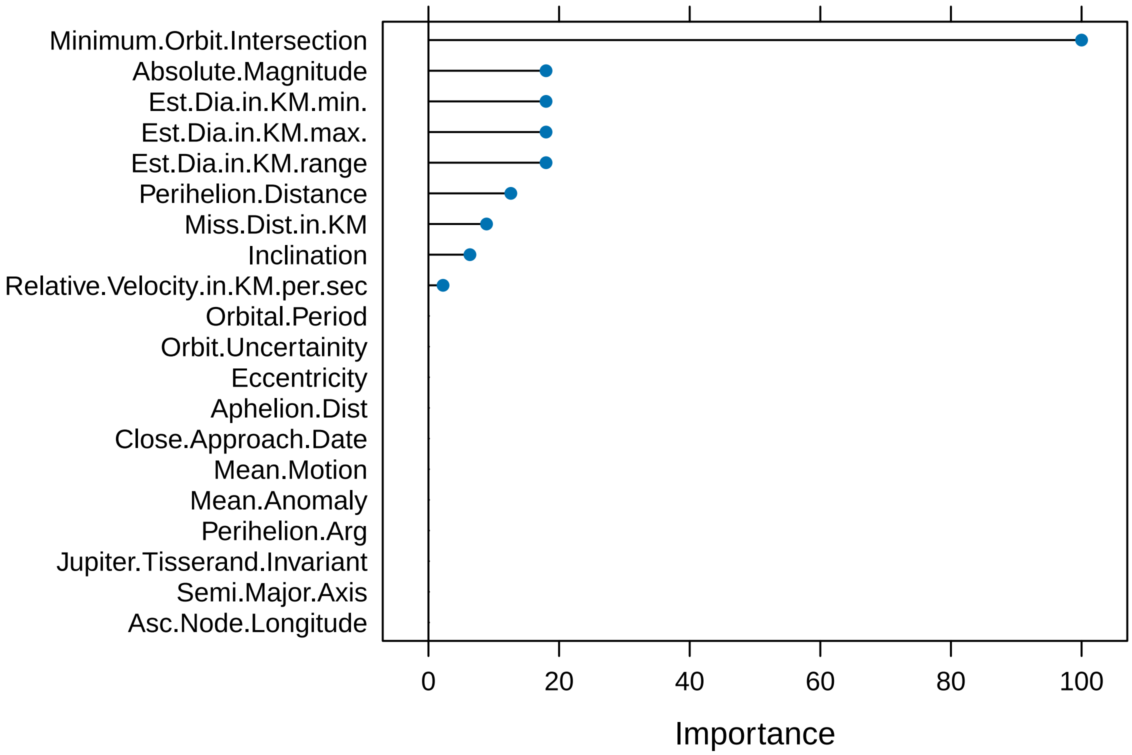
```r
plot(varImp(nasa.CVrpart))
```




```r
# variable importance (rpart1SE)
varImp(nasa.CVrpart1SE)
```

```
## rpart1SE variable importance
##
##                                  Overall
## Minimum.Orbit.Intersection       100.000
## Absolute.Magnitude                18.004
## Est.Dia.in.KM.range               18.004
## Est.Dia.in.KM.max.                18.004
## Est.Dia.in.KM.min.                18.004
## Perihelion.Distance               12.614
## Miss.Dist.in.KM                    8.894
## Inclination                        6.351
## Relative.Velocity.in.KM.per.sec    2.235
## Eccentricity                       0.000
## Orbital.Period                     0.000
## Mean.Motion                        0.000
## Semi.Major.Axis                    0.000
## Perihelion.Arg                     0.000
## Asc.Node.Longitude                 0.000
## Aphelion.Dist                      0.000
## Mean.Anomaly                       0.000
## Close.Approach.Date                0.000
## Orbit.Uncertainity                 0.000
## Jupiter.Tisserand.Invariant        0.000
```

```
plot(varImp(nasa.CVrpart1SE))
```

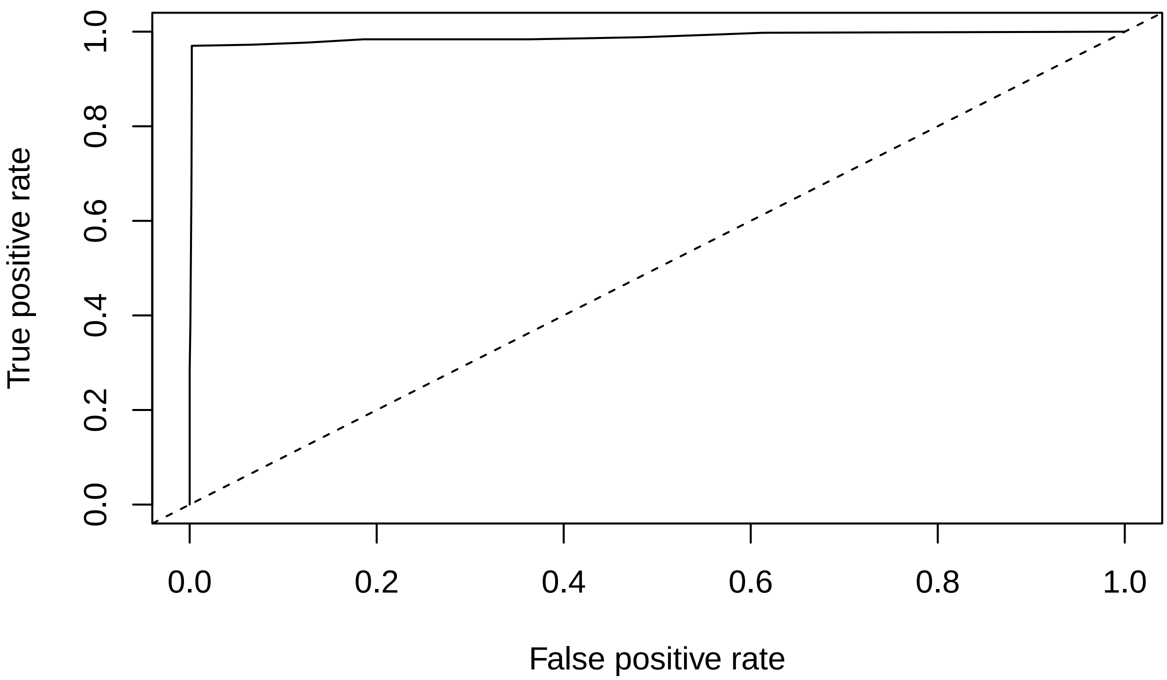


5. Evaluate the cross-validated (CV) predictive performance.

```
head(nasa.CVrpart$pred)
```

```
##    pred   obs rowIndex       False        True    cp Resample
## 1 False False       11 1.000000000 0.000000000 0.005   Fold01
## 2 False False       13 0.992572586 0.007427414 0.005   Fold01
## 3 False False       34 1.000000000 0.000000000 0.005   Fold01
## 4 False False       38 0.992572586 0.007427414 0.005   Fold01
## 5  True  True       66 0.002624672 0.997375328 0.005   Fold01
## 6  True  True       68 0.002624672 0.997375328 0.005   Fold01
```

```
pihatcv.rpart <- nasa.CVrpart$pred[nasa.CVrpart$pred$cp == 0.05,]
predcv.rpart <- prediction(pihatcv.rpart$True, pihatcv.rpart$obs)
perfcv.rpart <- performance(predcv.rpart, "tpr", "fpr")
plot(perfcv.rpart)
abline(a=0, b=1, lty=2)
```




```
aucCV.rpart <- performance(predcv.rpart, "auc")@y.values
aucCV.rpart
```

```
## [[1]]
## [1] 0.9884736
```

```
confMat <- table(pihatcv.rpart$obs, pihatcv.rpart$pred)
confMat
```

```
##
##          False True
##   False  2638    6
##   True     13  422
```

```
rpart.stats <- get_stats(confMat)
rpart.stats
```

```
##           name       value
## 1     accuracy 0.993829165
## 2   error rate 0.006170835
## 3    precision 0.985981308
## 4  sensitivity 0.970114943
## 5  specificity 0.997730711
## 6    F-measure 0.977983778
## 7 Matthew's CC 0.974439117
```

Random Forest Analysis

1. Make sure that the model assumptions, if any, are satisfied.

No model assumptions of random forest to be satisfied?

2. Assess the model fit and perform diagnostics, if appropriate.

Both methods appear to give identical results. Note the identical accuracy and kappa ratings across all cp tuning parameters in the plain rpart method…

```r
set.seed(1)

nasa.rf <- randomForest(as.factor(Hazardous) ~ ., data=nasa)
nasa.rf
```

```
##
## Call:
##  randomForest(formula = as.factor(Hazardous) ~ ., data = nasa)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 0.55%
## Confusion matrix:
##       False True class.error
## False  2640    4 0.001512859
## True     13  422 0.029885057
```

```r
set.seed(1)

nasa.CVrf <- train(Hazardous ~ ., data=nasa,
                   method="rf",
                   trControl=trainControl(method="cv", number=10,
                                          savePredictions=TRUE,
                                          classProbs=TRUE))
nasa.CVrf
```

```
## Random Forest
##
## 3079 samples
##   20 predictor
##    2 classes: 'False', 'True'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2771, 2771, 2772, 2770, 2771, 2772, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9948020  0.9782101
##   11    0.9961018  0.9836952
##   20    0.9970769  0.9877668
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 20.
```

3. Identify tuning parameters to be used, if appropriate.

The final model selects an mtry tuning parameter value of 20 in order to maximize accuracy.

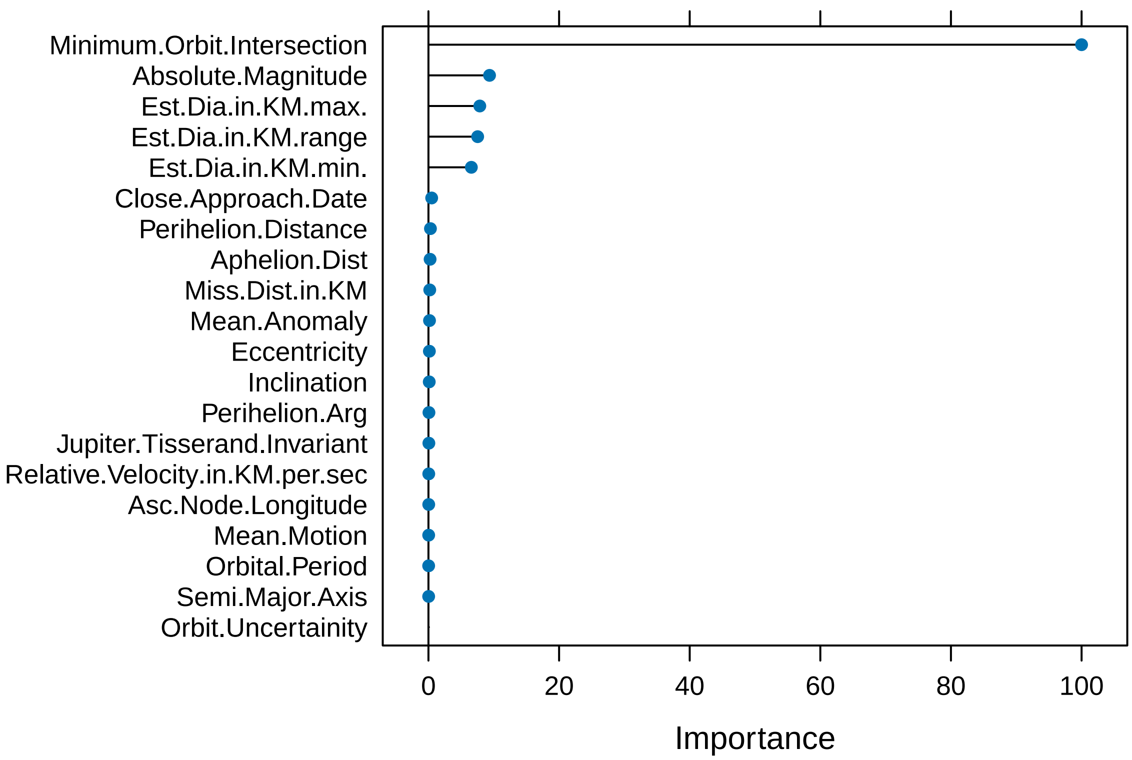
4. Identify and interpret the effect of selected variables.

```r
varImp(nasa.CVrf)
```

```
## rf variable importance
##
##                                      Overall
```

```
## Minimum.Orbit.Intersection      100.00000
## Absolute.Magnitude                9.35397
## Est.Dia.in.KM.max.                7.86373
## Est.Dia.in.KM.range               7.53785
## Est.Dia.in.KM.min.                6.56980
## Close.Approach.Date               0.49328
## Perihelion.Distance               0.30455
## Aphelion.Dist                     0.25229
## Miss.Dist.in.KM                   0.20076
## Mean.Anomaly                      0.16128
## Eccentricity                      0.14131
## Inclination                       0.12331
## Perihelion.Arg                    0.07658
## Jupiter.Tisserand.Invariant       0.06409
## Relative.Velocity.in.KM.per.sec   0.04976
## Asc.Node.Longitude                0.04545
## Mean.Motion                       0.03574
## Orbital.Period                    0.03012
## Semi.Major.Axis                   0.02982
## Orbit.Uncertainity                0.00000
```

```
plot(varImp(nasa.CVrf))
```



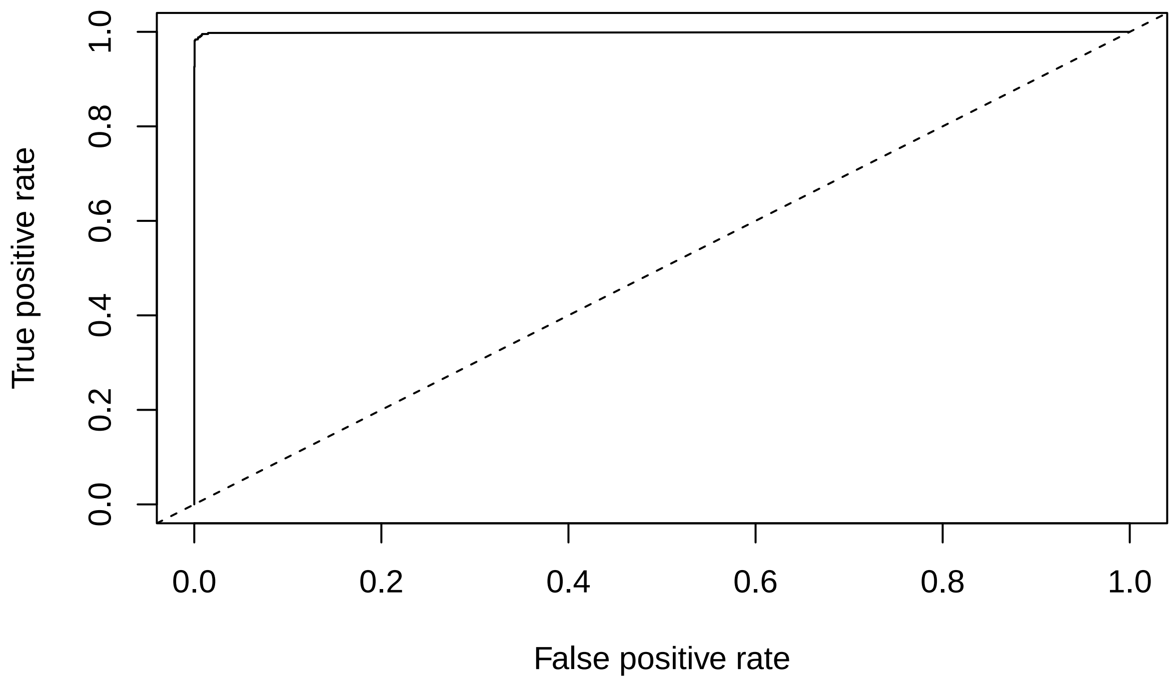
5. Evaluate the cross-validated (CV) predictive performance.

```
head(nasa.CVrf$pred)
```

```
##    pred   obs False  True rowIndex mtry Resample
## 1 False False 0.964 0.036       11    2   Fold01
## 2 False False 0.982 0.018       13    2   Fold01
## 3 False False 0.976 0.024       34    2   Fold01
```

```
## 4 False False 0.996 0.004       38    2   Fold01
## 5  True  True 0.164 0.836       66    2   Fold01
## 6  True  True 0.242 0.758       68    2   Fold01
```

```
pihatcv.rf <- nasa.CVrf$pred[nasa.CVrf$pred$mtry == 20,]
predcv.rf <- prediction(pihatcv.rf$True, pihatcv.rf$obs)
perfcv.rf <- performance(predcv.rf, "tpr", "fpr")
plot(perfcv.rf)
abline(a=0, b=1, lty=2)
```




```
aucCV.rf <- performance(predcv.rf, "auc")@y.values
aucCV.rf
```

```
## [[1]]
## [1] 0.9986302
```

```
confMat <- table(pihatcv.rf$obs, pihatcv.rf$pred)
confMat
```

```
##
##          False True
##   False   2643    1
##   True       8  427
```

```
rf.stats <- get_stats(confMat)
rf.stats
```

```
##           name       value
## 1     accuracy 0.997076973
## 2   error rate 0.002923027
## 3    precision 0.997663551
## 4  sensitivity 0.981609195
## 5  specificity 0.999621785
## 6    F-measure 0.989571263
## 7 Matthew's CC 0.987915632
```