IE5561 Miniproject 1 - Code

Jingzhe Wang

March 3, 2018

Data preparation

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(rpart)
library(partykit)
```

```
## Loading required package: grid
```

```
## Loading required package: libcoin
```

```
## Loading required package: mvtnorm
```

```
library(randomForest)
```

```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ncf)
```

```
# Load in data
# 3367 observations collected from farm area from Area of Interest (AOI)
farm.soil = read.table('farm_soil_pts.txt', header = TRUE, sep = ',')
farm.slope = read.table('farm_slope_pts.txt', header = TRUE, sep = ',')
farm.ldcover = read.table('farm_ldcover_pts.txt', header = TRUE, sep = ',')
farm.canopy = read.table('farm_canopy_pts.txt', header = TRUE, sep = ',')
farm.pt = read.table('farm_pts.txt', header = TRUE, sep = ',')

colnames(farm.slope)[4] = 'slope'
```

```
colnames(farm.ldcover)[4] = 'landcover'
colnames(farm.canopy)[4] = 'canopy'

farm.soil.subset = select(farm.soil, c('stype','organic_percent', 'avail_water'))
farm.slope.subset = select(farm.slope, 'slope')
farm.ldcover.subset = select(farm.ldcover, 'landcover')
farm.canopy.subset = select(farm.canopy, 'canopy')
farm.pt.subset = select(farm.pt, c('POINT_X', 'POINT_Y'))

farm.data = bind_cols(farm.soil.subset, farm.slope.subset,
          farm.ldcover.subset, farm.canopy.subset, farm.pt.subset)

rm(farm.soil,farm.slope,farm.ldcover,farm.canopy,farm.pt,
  farm.soil.subset, farm.slope.subset,farm.ldcover.subset,
  farm.canopy.subset, farm.pt.subset)

# assign the value for response
farm.data$farmland = 'Y'


# 2746 observations collected from nonfarm area from AOI
nonfarm.soil = read.table('nonfarm_soil_pts.txt', header = TRUE, sep = ',')
nonfarm.slope = read.table('nonfarm_slope_pts.txt', header = TRUE, sep = ',')
nonfarm.ldcover = read.table('nonfarm_ldcover_pts.txt', header = TRUE, sep = ',')
nonfarm.canopy = read.table('nonfarm_canopy_pts.txt', header = TRUE, sep = ',')
nonfarm.pt = read.table('nonfarm_pts.txt', header = TRUE, sep = ',')

colnames(nonfarm.slope)[4] = 'slope'
colnames(nonfarm.ldcover)[4] = 'landcover'
colnames(nonfarm.canopy)[4] = 'canopy'

nonfarm.soil.subset = select(nonfarm.soil, c('stype','organic_percent', 'avail_water'))
nonfarm.slope.subset = select(nonfarm.slope, 'slope')
nonfarm.ldcover.subset = select(nonfarm.ldcover, 'landcover')
nonfarm.canopy.subset = select(nonfarm.canopy, 'canopy')
nonfarm.pt.subset = select(nonfarm.pt, c('POINT_X', 'POINT_Y'))

nonfarm.data = bind_cols(nonfarm.soil.subset, nonfarm.slope.subset,
               nonfarm.ldcover.subset, nonfarm.canopy.subset,
               nonfarm.pt.subset)

rm(nonfarm.soil,nonfarm.slope,nonfarm.ldcover,nonfarm.canopy,
               nonfarm.soil.subset, nonfarm.slope.subset,
               nonfarm.ldcover.subset, nonfarm.canopy.subset,
               nonfarm.pt.subset, nonfarm.pt)

# assign the value for response
nonfarm.data$farmland = 'N'

# row bind all the farm.data and nonfarm.data
proj.data = rbind(farm.data, nonfarm.data)

# check and specify the factor variables
```

```r
sapply(proj.data, class)
```

```
##           stype organic_percent     avail_water           slope
##        "factor"       "numeric"       "numeric"       "numeric"
##       landcover          canopy         POINT_X         POINT_Y
##       "integer"       "integer"       "numeric"       "numeric"
##        farmland
##     "character"
```

```r
# conver the corresponding nonfactor columns to factors
proj.data[,c(5,9)] = lapply(proj.data[,c(5,9)], factor)
```

Data description for 'proj.data'

```r
summary(proj.data)
```

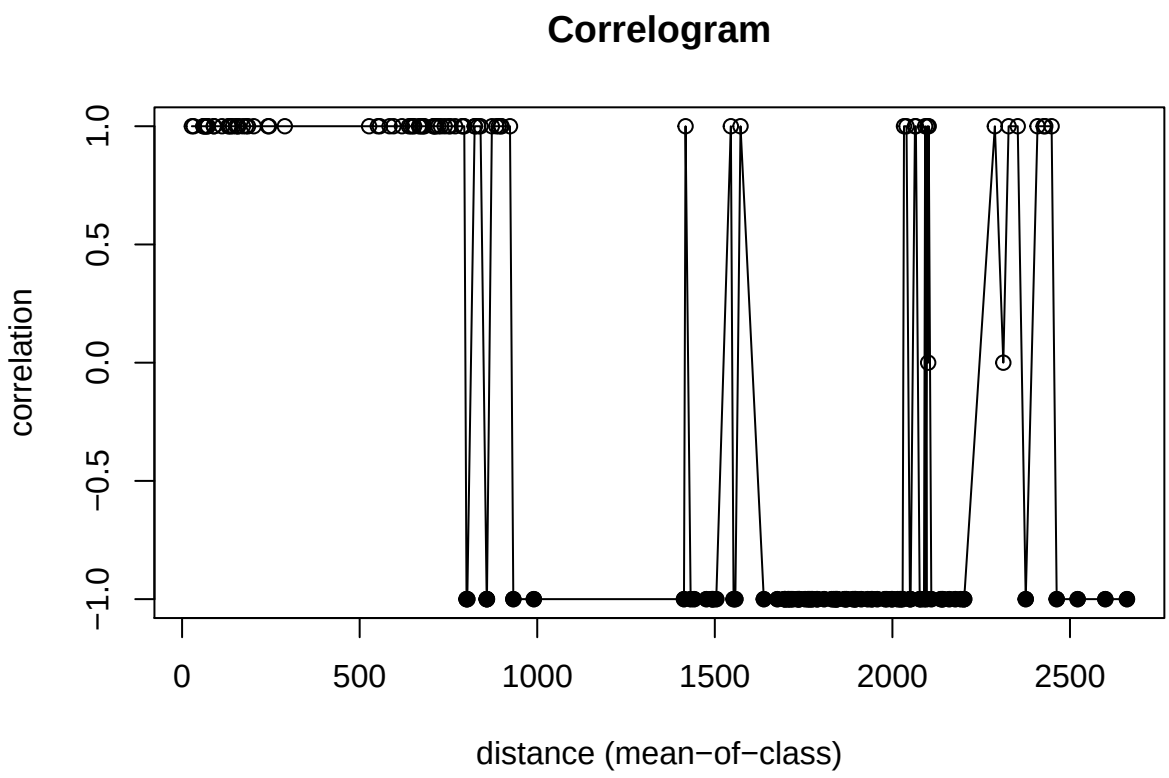
```
##   stype      organic_percent  avail_water        slope         landcover
##  CoF2: 749   Min.   :2.500   Min.   : 7.82   Min.   : 0.3383   21: 312
##  HmD :  73   1st Qu.:2.500   1st Qu.:23.67   1st Qu.:11.7326   22:1068
##  HmE2:1138   Median :4.600   Median :23.74   Median :17.7378   42:3388
##  HmF2:1040   Mean   :4.845   Mean   :20.44   Mean   :18.6737   71:1345
##  MxF :1309   3rd Qu.:7.500   3rd Qu.:23.97   3rd Qu.:24.6096
##  CoE :1573   Max.   :7.500   Max.   :24.30   Max.   :41.8858
##  DaE2: 231
##      canopy          POINT_X          POINT_Y       farmland
##  Min.   :  1.00   Min.   :129507   Min.   :240148   N:2746
##  1st Qu.:  8.00   1st Qu.:130454   1st Qu.:240689   Y:3367
##  Median : 63.00   Median :130984   Median :241495
##  Mean   : 54.79   Mean   :130968   Mean   :241416
##  3rd Qu.: 81.00   3rd Qu.:131474   3rd Qu.:242282
##  Max.   :104.00   Max.   :131914   Max.   :242961
##
```

- Factor variables: farmland(response), stype(soil type), landcover type
- Numeric variables: soil organic matter percentage, soil available water, slope tree canopy coverage rate, and XY-Coordinates(in meters).

Check spatial Autocorrelation pattern

```r
# explore the Moran's I pattern with respect to
# XY-coordinate values and the farmland value

# As the total number observation in the proj.data is large (6000+)
# I randomly choose 20 observations to explore spatial autocorrelation
set.seed(5561)
# randomly choose 100 obs
choose.index = sample(1:nrow(proj.data), 20)
x = proj.data[choose.index,]$POINT_X
y = proj.data[choose.index,]$POINT_Y
z = proj.data[choose.index,]$farmland

# the correlog() function does not take factor value for z
```

```
# I transformed it to numeric (I assumed this could still help me
# obtain the Moran's I values)
ncf.cor <- correlog(x, y, as.numeric(z), increment = 2, resamp=500,
                    quiet = TRUE)
plot(ncf.cor)
```

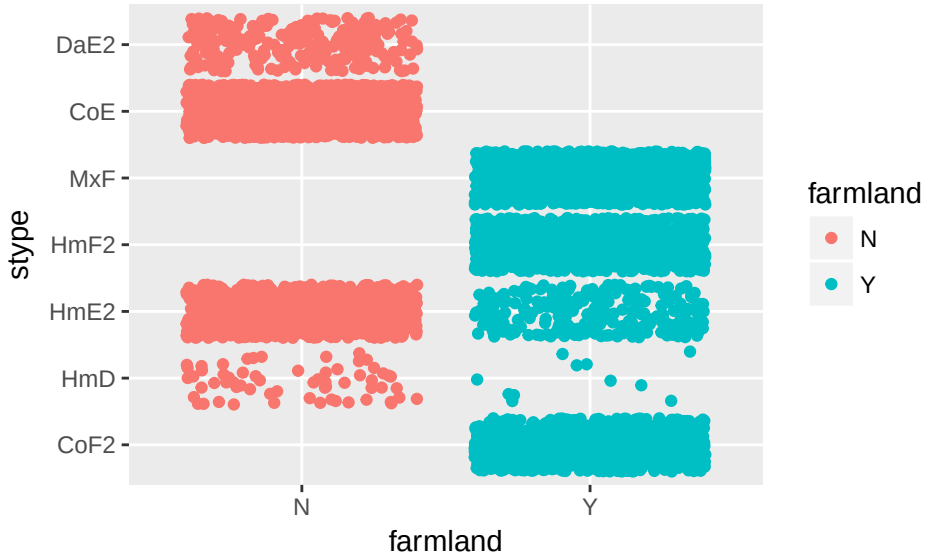


Correlation coeffecient here refers to Moran's I correlation coefficient, positive Moran's I indicates the location points with similar 'farmland' values (same in our case here); negative Moran's I indicates the location points with different 'farmland' values. This result was consistent with what I expected.

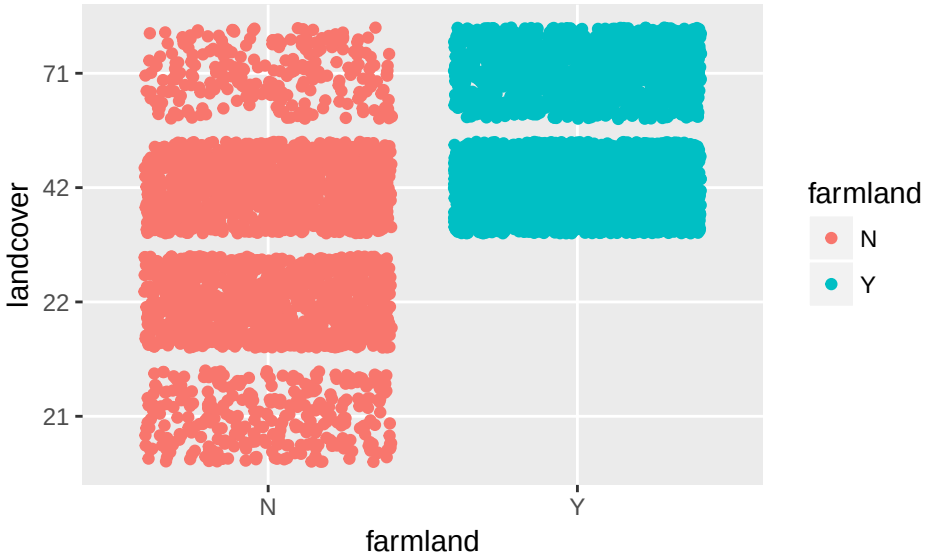
Obviously, the spatial autocorrelation warned us to use the common generalized linear regression model. Try the nonparametric decision tree and bagging methods.

Scatter plots

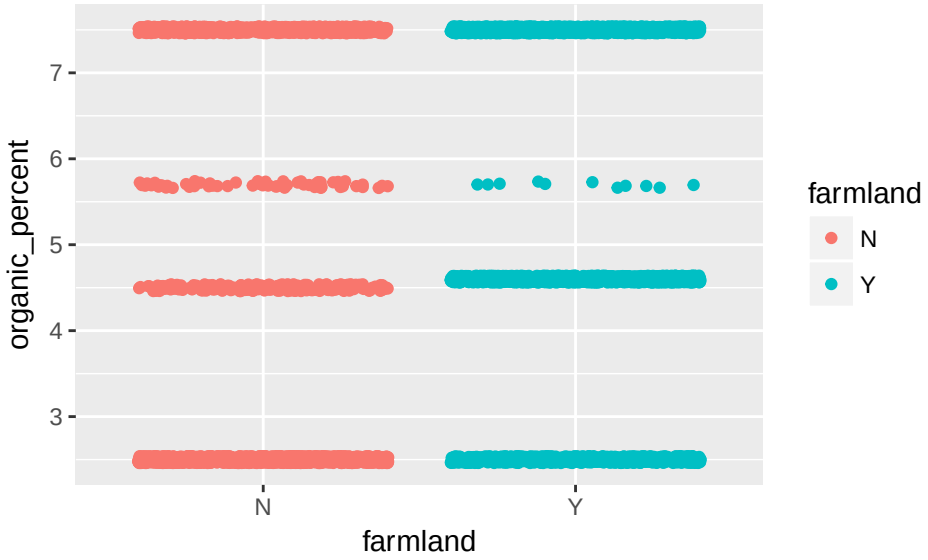
```
# farmland vs soil type
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = stype,
group = farmland, col = farmland))
```



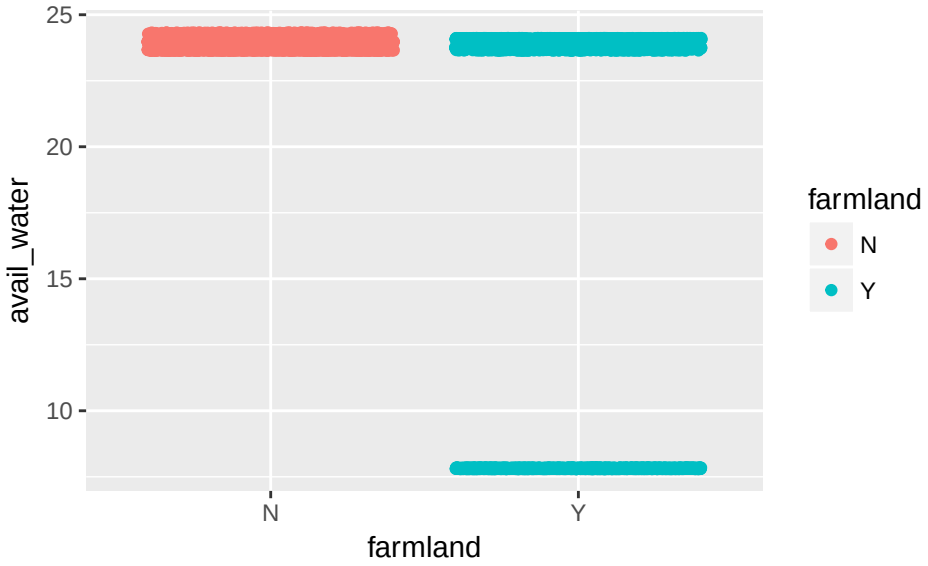

```
# farmland vs landcover type
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = landcover,
group = farmland, col = farmland))
```



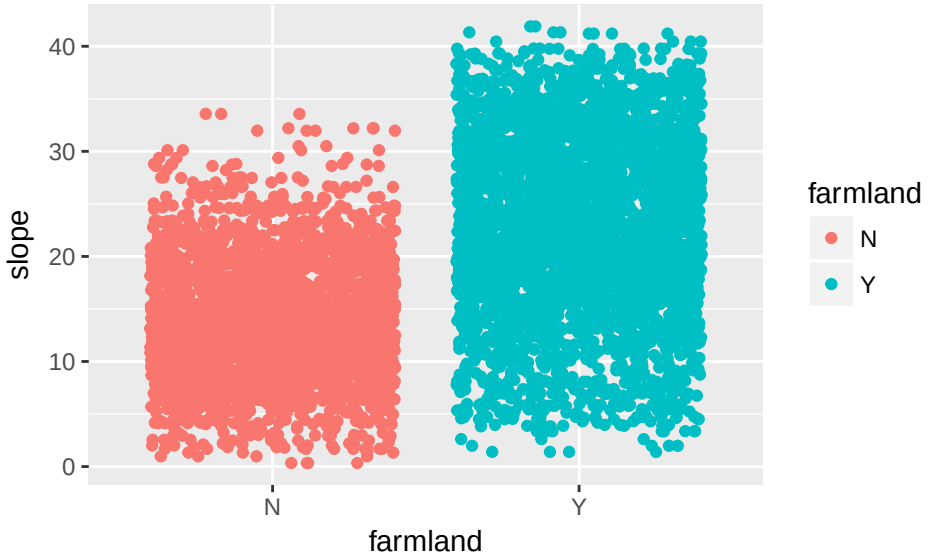

```
# farmland vs organic matter percentage in soil
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = organic_percent,
group = farmland, col = farmland))
```



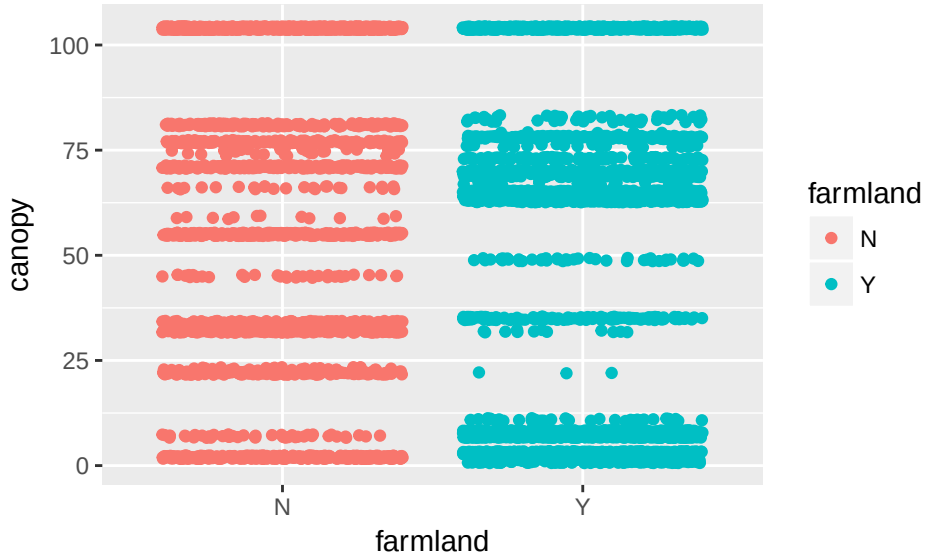

```
# farmland vs available water in soil
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = avail_water,
group = farmland, col = farmland))
```




```
# farmland vs slope
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = slope,
group = farmland, col = farmland))
```




```
# farmland vs tree canopy coverage rate
ggplot(data = proj.data) + geom_jitter(aes(x = farmland, y = canopy,
group = farmland, col = farmland))
```



From the plots, soil type, landcover type, slope seems to be very important predictors to determine if the points on a parcel of land are from farmland or nonfarmland.

Decision tree classification

Single decision tree

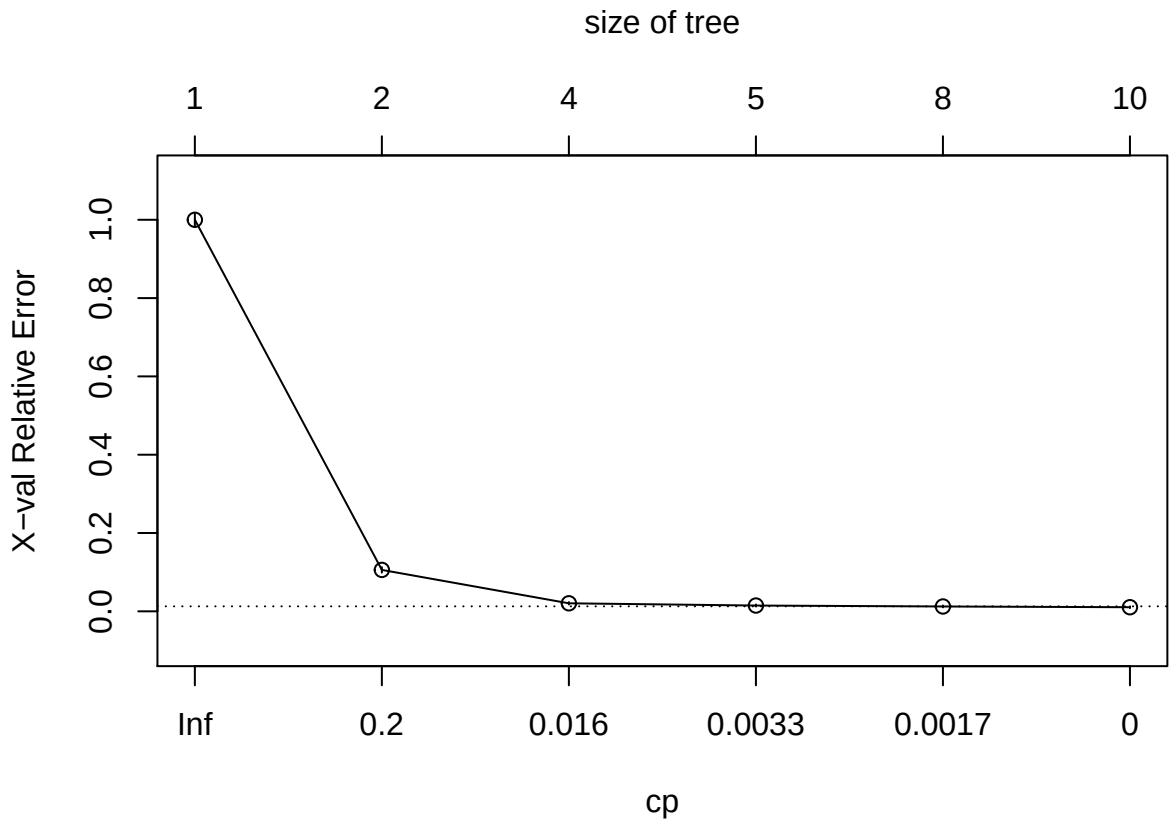
```
# use 75% of the observations for training
n.total = nrow(proj.data)
set.seed(300)
index.tr = sample(1:n.total, 0.75*n.total)
```

```r
index.te = setdiff(1:n.total, index.tr)
# split the data
train = proj.data[index.tr,]
test = proj.data[index.te,]
```

```r
# full model
ctrl = rpart.control(cp=0)
farm.fit = rpart(farmland ~ stype + landcover + organic_percent +
           avail_water + slope + canopy, data = train, control = ctrl)
printcp(farm.fit)
```

```
##
## Classification tree:
## rpart(formula = farmland ~ stype + landcover + organic_percent +
##     avail_water + slope + canopy, data = train, control = ctrl)
##
## Variables actually used in tree construction:
## [1] canopy    landcover slope     stype
##
## Root node error: 2099/4584 = 0.4579
##
## n= 4584
##
##          CP nsplit rel error   xerror      xstd
## 1 0.8942354      0 1.0000000 1.000000 0.0160707
## 2 0.0426394      1 0.1057646 0.105765 0.0069244
## 3 0.0057170      3 0.0204859 0.020486 0.0031094
## 4 0.0019057      4 0.0147689 0.014769 0.0026436
## 5 0.0014293      7 0.0090519 0.012387 0.0024224
## 6 0.0000000      9 0.0061934 0.010481 0.0022292
```

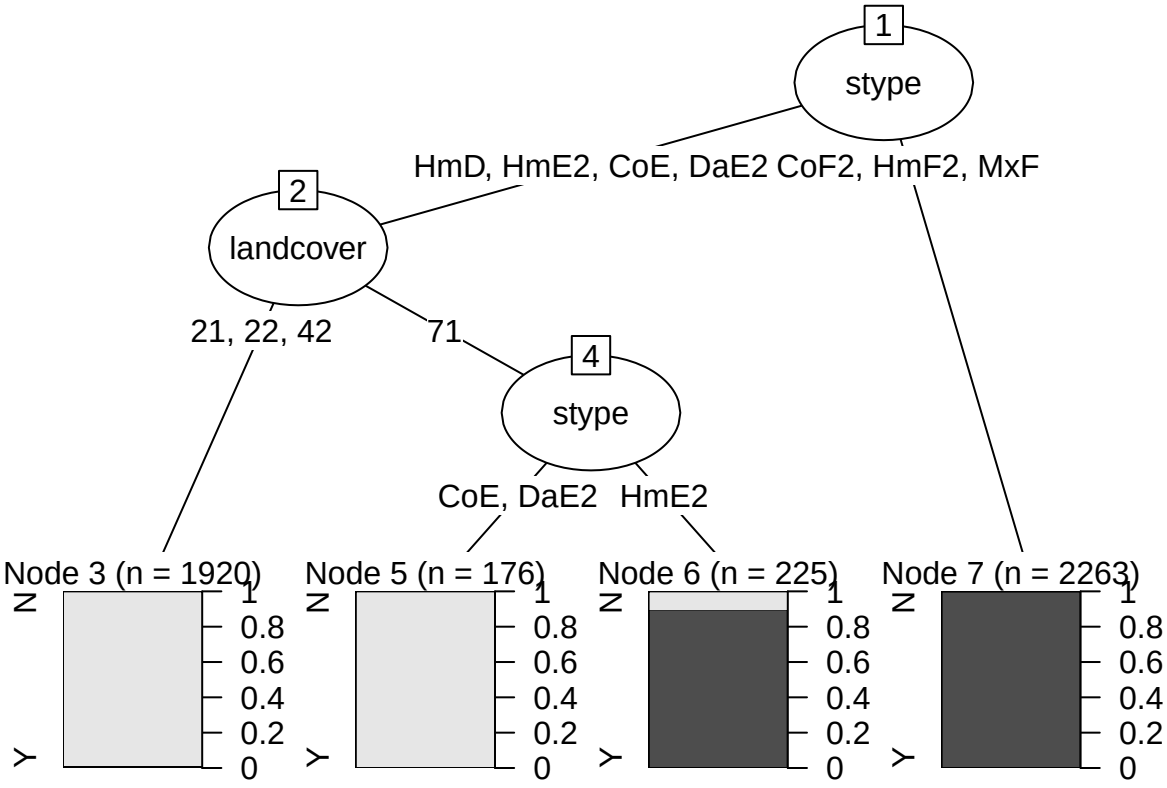
```r
plotcp(farm.fit)
```




```
# prune it back using cp = 0.01
farm.fit2 = prune(farm.fit, cp = 0.01)
printcp(farm.fit2)
```

```
##
## Classification tree:
## rpart(formula = farmland ~ stype + landcover + organic_percent +
##     avail_water + slope + canopy, data = train, control = ctrl)
##
## Variables actually used in tree construction:
## [1] landcover stype
##
## Root node error: 2099/4584 = 0.4579
##
## n= 4584
##
##         CP nsplit rel error   xerror      xstd
## 1 0.894235      0  1.000000 1.000000 0.0160707
## 2 0.042639      1  0.105765 0.105765 0.0069244
## 3 0.010000      3  0.020486 0.020486 0.0031094
```

```
plot(as.party(farm.fit2))
```




```
as.party(farm.fit2)
```

```
##
## Model formula:
## farmland ~ stype + landcover + organic_percent + avail_water +
##     slope + canopy
##
## Fitted party:
## [1] root
## |   [2] stype in HmD, HmE2, CoE, DaE2
## |   |   [3] landcover in 21, 22, 42: N (n = 1920, err = 1.0%)
## |   |   [4] landcover in 71
## |   |   |   [5] stype in CoE, DaE2: N (n = 176, err = 0.0%)
## |   |   |   [6] stype in HmE2: Y (n = 225, err = 10.2%)
## |   [7] stype in CoF2, HmF2, MxF: Y (n = 2263, err = 0.0%)
##
## Number of inner nodes:    3
## Number of terminal nodes: 4
```

```
# check the performance
farm.pred.cl = predict(farm.fit2, test, type = 'class')
table(farm.pred.cl, test$farmland)
```

```
##
## farm.pred.cl   N   Y
##            N 638   5
##            Y   9 877
```

```
# overall error rate
1 - (638 + 877) / nrow(test)
```

```
## [1] 0.009156311
```

The classification results for training set data:

- Stype(Soil type) and landcover type turned out to be the most important predictors.
- For the 2263 location points with soil type CoF2, HmF2 or MxF, they are classified as farmland without any misclassification.
- For the 225 location points with soil type HmE2 and landcover type 71 (grassland/herbaceous), they are classified as farmland with 10.2% of error rate.
- For the 176 location points with soil type CoE or DaE2 and landcover type 71 (grassland/herbaceous), they are classified as non-farmland with 0% error rate.
- For the rest of 1920 location points with soil type HmD, HmE2, CoE or DaE2, landcover type 21(developed, open space), 22(developed, low intensenty) or 42(evergreen forest), they are classified as non-farmland with 1% error rate.

Note: landcover type interpretaton comes from U.S. Department of the Interior, U.S. Geological Survey, 2013

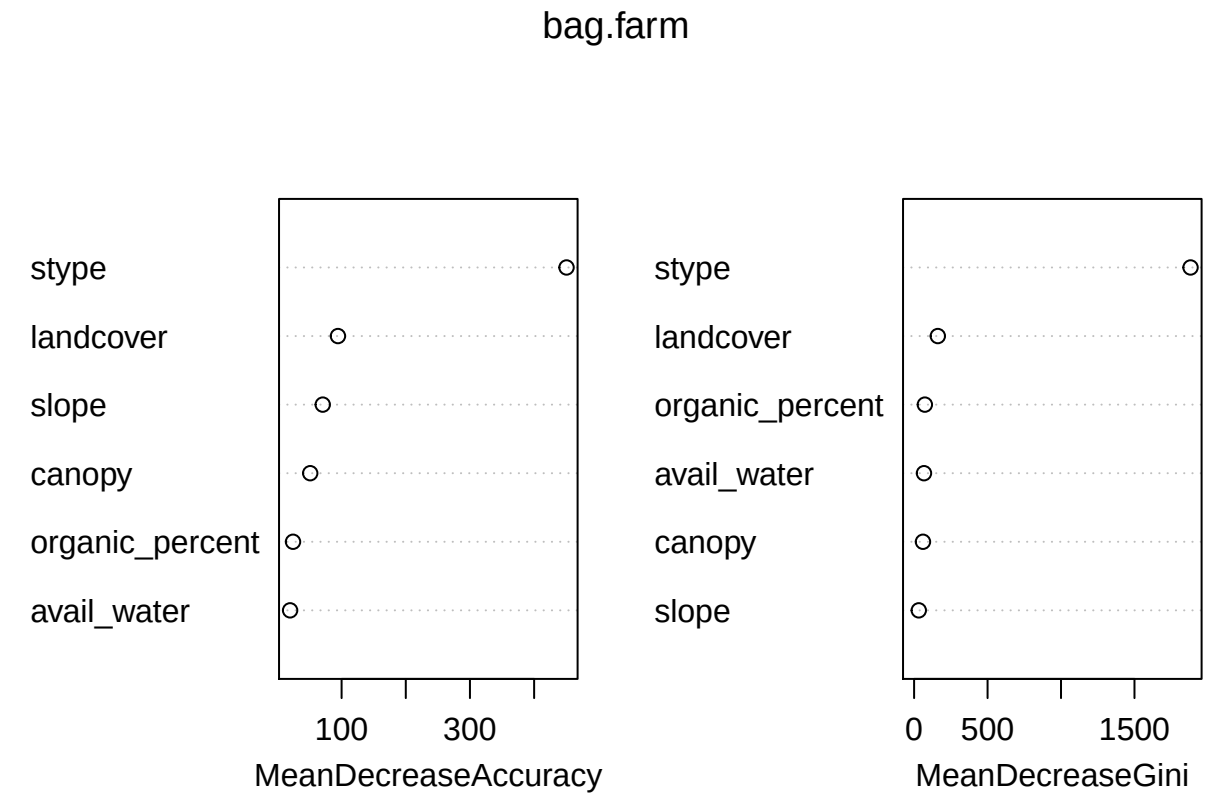
- Misclassification rate for test set data was around 0.92%.

Bagging

```
# bagging
# specify the p
p = 6
# bagging
bag.farm = randomForest(farmland ~ stype + landcover + organic_percent +
 avail_water + slope + canopy, data = train, mtry = p, importance = TRUE)

# which predictors are important?
importance(bag.farm)
```

```
##                          N         Y MeanDecreaseAccuracy MeanDecreaseGini
## stype           674.138590 219.98335             450.50581       1881.49080
## landcover        55.560657 116.56536              94.31051        161.46812
## organic_percent  10.426758  22.24320              24.30702         73.28344
## avail_water       9.919307  19.72325              19.74785         67.28420
## slope            68.222963  28.19136              70.61539         31.84374
## canopy           22.565267  81.85722              51.07500         60.23032
```

```r
varImpPlot(bag.farm)
```




```r
# prediction using bagging
farm.pred.bag = predict(bag.farm, test, type = 'class')
table(farm.pred.bag, test$farmland)
```

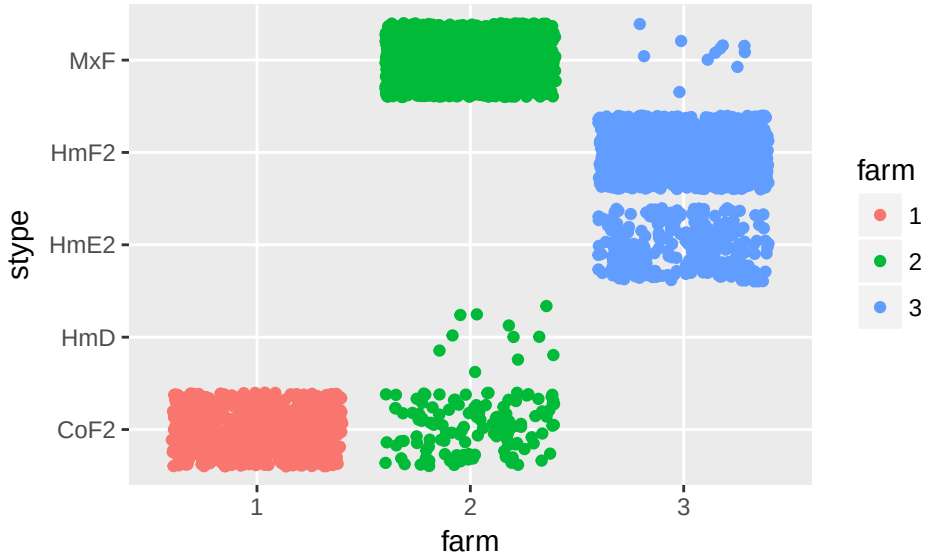
```
##
## farm.pred.bag   N   Y
##             N 647   0
##             Y   0 882
```

Soil type turned out to be the most important variable, according to 'mean decrease accuracy'. No misclassified location points in the test set data.

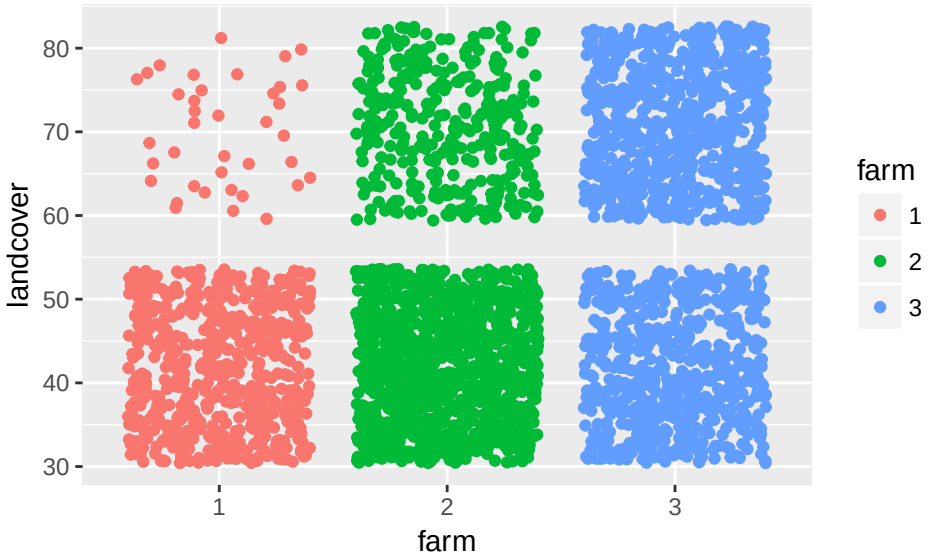
Compare three farms

```r
# specify the three farms
# indices for three farms were found through GIS software tools
# e.g. attribute table and on-screen location selection query

# I simplied the names of the farms as 1, 2, 3
# create a vector to store the farm names
farm = numeric(nrow(farm.data))
farm[1:622] = 1
farm[623:2058] = 2
```

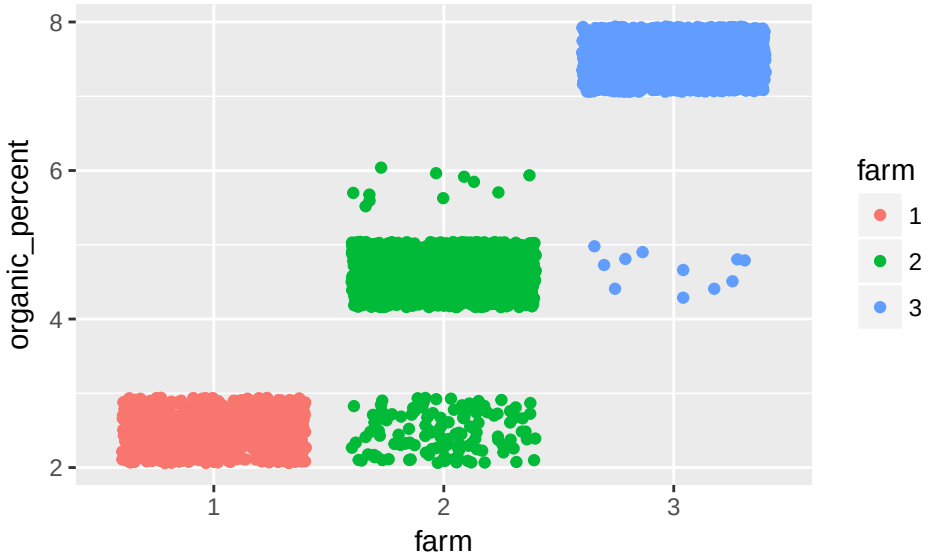
```r
farm[2059:3367] = 3

# attach the vector to the dataframe as a new column
# new dataframe 'farm.data2' will be used through out
# the rest of the analysis
farm.data2 = cbind(farm.data, farm)
# farm names were transformed to three levels of a factor variable
farm.data2$farm = as.factor(farm.data2$farm)

# Some scatter plots
# farm vs soil type
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = stype,
group = farm, col = farm))
```



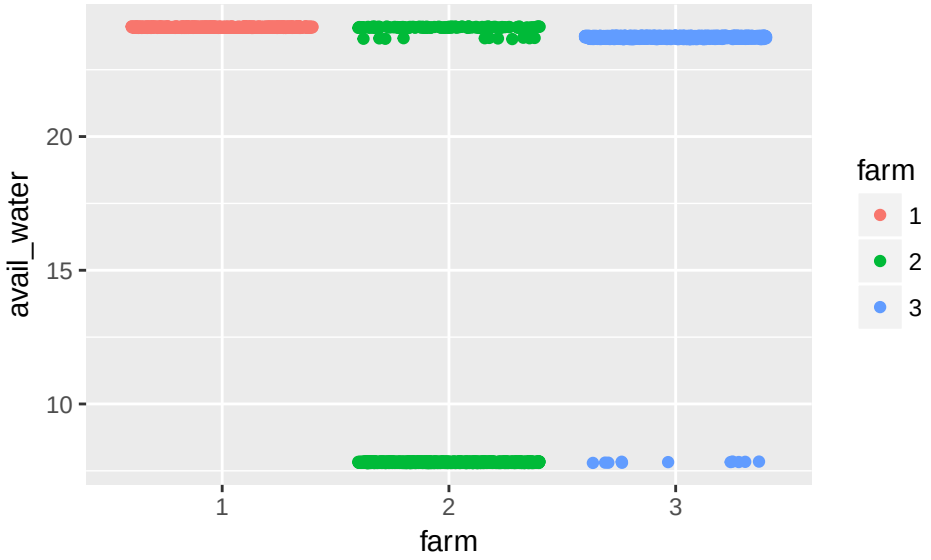

```r
# farm vs landcover type
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = landcover,
group = farm, col = farm))
```



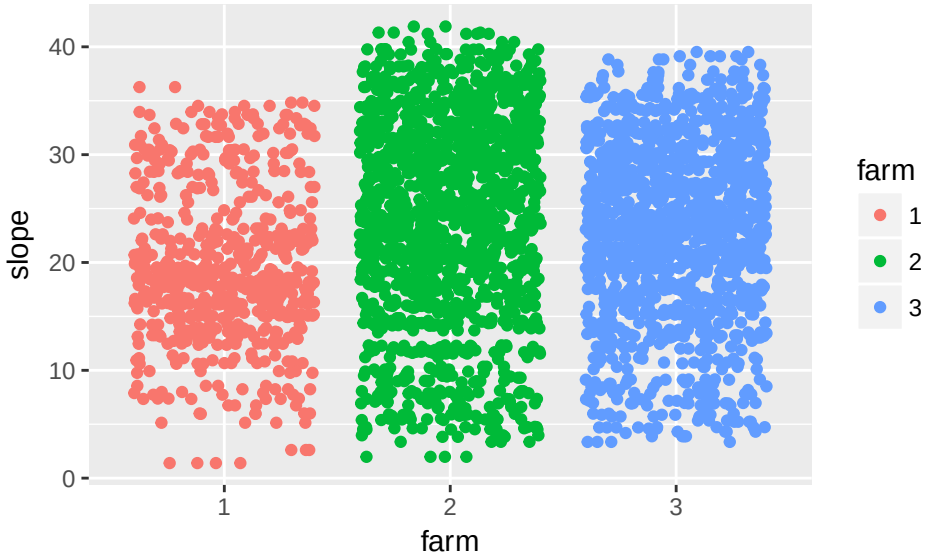

```
# farm vs organic matter percentage in soil
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = organic_percent,
group = farm, col = farm))
```



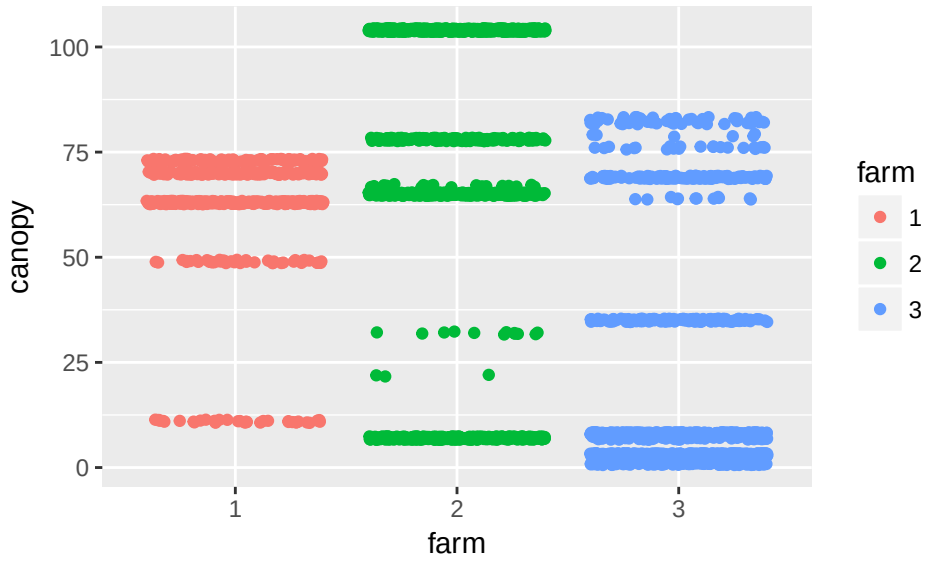

```
# farm vs available water rate in soil
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = avail_water,
group = farm, col = farm))
```




```
# farm vs slope
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = slope,
group = farm, col = farm))
```




```
# farm vs tree canopy coverage rate
ggplot(data = farm.data2) + geom_jitter(aes(x = farm, y = canopy,
group = farm, col = farm))
```



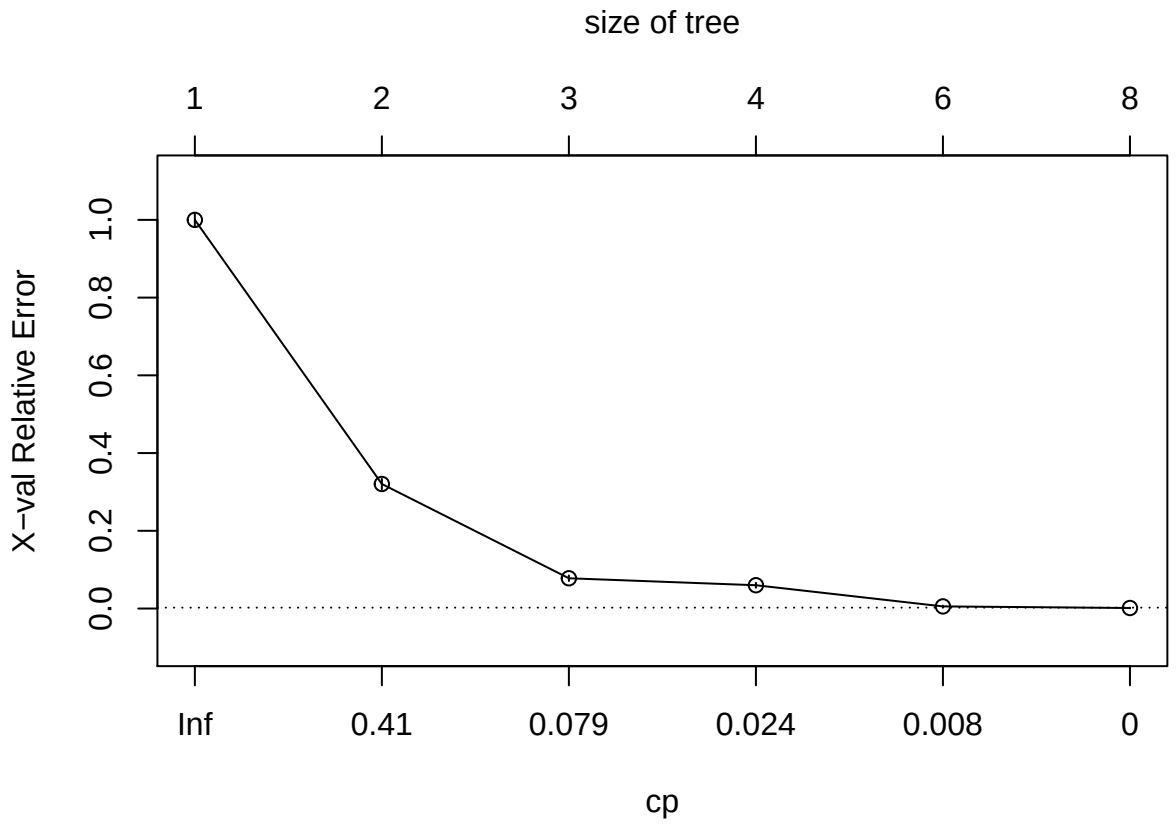
Decision tree for distiguishing three farms

```
# split the 'farm.data2'
# use 75% of the observations in 'farm.data2' for training set
n.total2 = nrow(farm.data2)
set.seed(300)
index.tr2 = sample(1:n.total2, 0.75*n.total2)
index.te2 = setdiff(1:n.total2, index.tr2)
# split the farm.data2
train2 = farm.data2[index.tr2,]
test2 = farm.data2[index.te2,]

# full model
ctrl = rpart.control(cp=0)
farm.fit3 = rpart(farm ~ stype + landcover + organic_percent +
           avail_water + slope + canopy, data = train2, control = ctrl)
printcp(farm.fit3)
```

```
##
## Classification tree:
## rpart(formula = farm ~ stype + landcover + organic_percent +
##     avail_water + slope + canopy, data = train2, control = ctrl)
##
## Variables actually used in tree construction:
## [1] canopy    landcover stype
##
## Root node error: 1451/2525 = 0.57465
##
## n= 2525
##
##          CP nsplit rel error    xerror       xstd
## 1 0.6795314      0 1.0000000 1.0000000 0.01712134
## 2 0.2425913      1 0.3204686 0.3204686 0.01342338
## 3 0.0254997      2 0.0778773 0.0778773 0.00716028
```

```
## 4 0.0234321      3 0.0523777 0.0599586 0.00631653
## 5 0.0027567      5 0.0055134 0.0055134 0.00194620
## 6 0.0000000      7 0.0000000 0.0013784 0.00097426
```

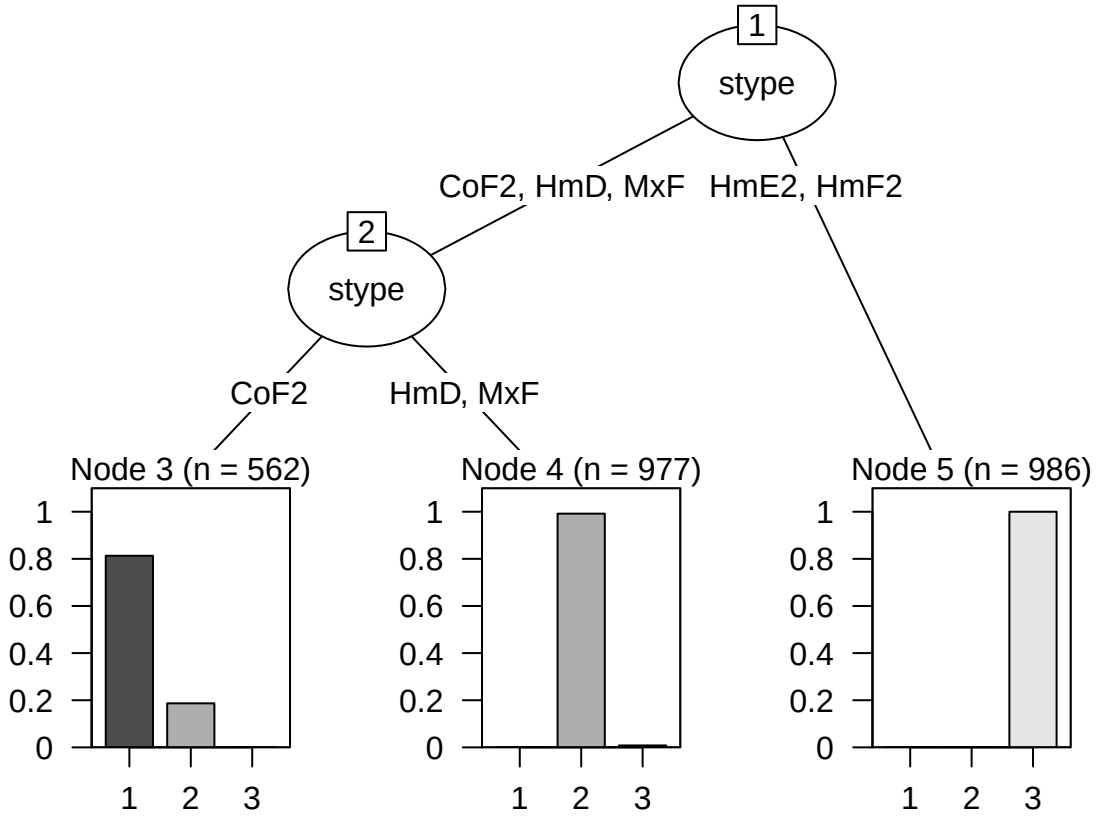
```
plotcp(farm.fit3)
```




```
# prune it back using cp = 0.03
farm.fit4 = prune(farm.fit3, cp = 0.03)
printcp(farm.fit4)
```

```
##
## Classification tree:
## rpart(formula = farm ~ stype + landcover + organic_percent +
##     avail_water + slope + canopy, data = train2, control = ctrl)
##
## Variables actually used in tree construction:
## [1] stype
##
## Root node error: 1451/2525 = 0.57465
##
## n= 2525
##
##         CP nsplit rel error   xerror      xstd
## 1 0.67953      0  1.000000 1.000000 0.0171213
## 2 0.24259      1  0.320469 0.320469 0.0134234
## 3 0.03000      2  0.077877 0.077877 0.0071603
```

```
plot(as.party(farm.fit4))
```




```
as.party(farm.fit4)
```

```
##
## Model formula:
## farm ~ stype + landcover + organic_percent + avail_water + slope +
##     canopy
##
## Fitted party:
## [1] root
## |   [2] stype in CoF2, HmD, MxF
## |   |   [3] stype in CoF2: 1 (n = 562, err = 18.7%)
## |   |   [4] stype in HmD, MxF: 2 (n = 977, err = 0.8%)
## |   [5] stype in HmE2, HmF2: 3 (n = 986, err = 0.0%)
##
## Number of inner nodes:    2
## Number of terminal nodes: 3
```

```
# check the performance
farm.pred.cl2 = predict(farm.fit4, test2, type = 'class')
table(farm.pred.cl2, test2$farm)
```

```
##
## farm.pred.cl2   1   2   3
##             1 165  22   0
##             2   0 340   3
```

```
##             3   0   0 312
```

```
# overall error rate
1 - (165+340+312) / nrow(test2)
```

```
## [1] 0.02969121
```

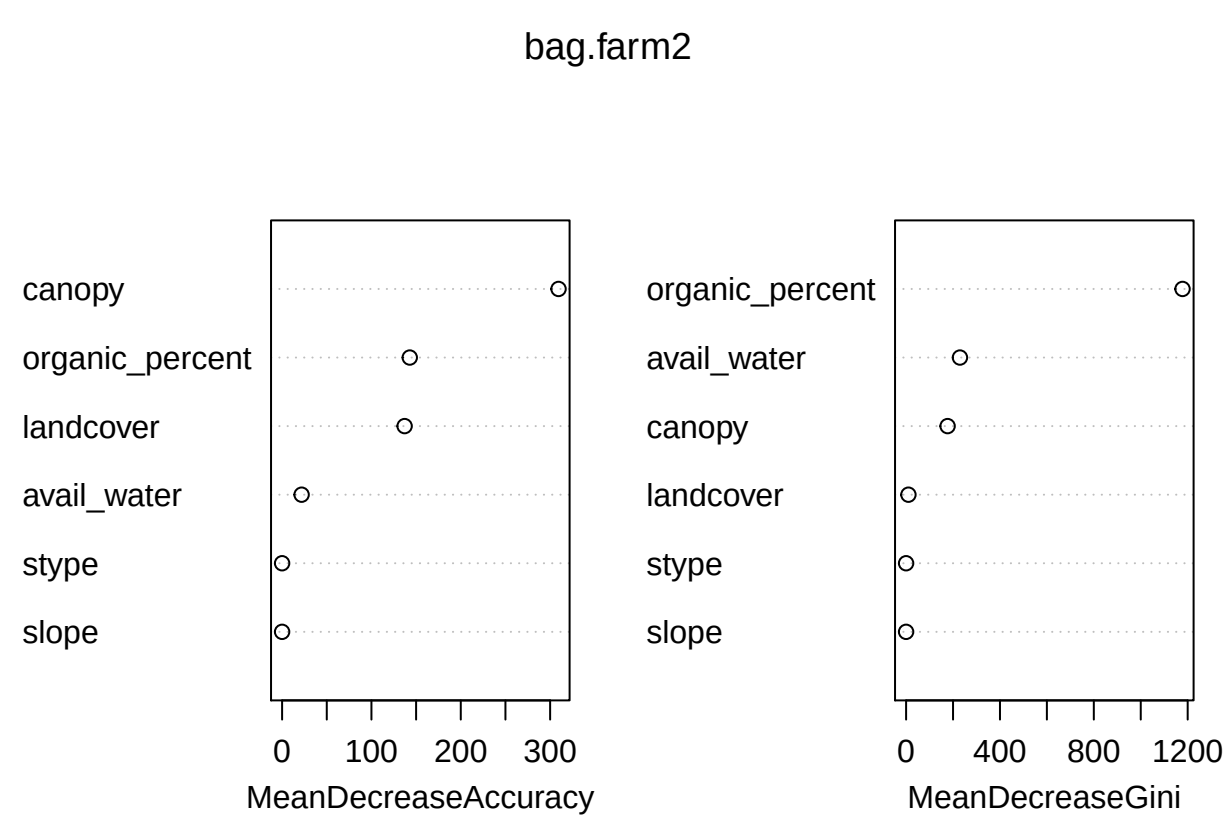
- Soil type is the most important predictor to distinguish three farms. For the 986 location points with soil type HmE2 and HmF2 in our training set, they are classified as farm3 without any misclassification.
- For the 977 location points with soil type HmD and MxF in our training set, they are classified as farm2 with error rate 0.8%.
- For the 562 location points with soil type CoF1, they are classified as farm1 with error rate 18.7%.
- The classification error rate for the prediction on our test data is around 2.97%.

Bagging

```
# bagging
# specify the p
p = 6
# bagging
bag.farm2 = randomForest(farm ~ stype + landcover + organic_percent +
 avail_water + slope + canopy, data = train2, mtry = p, importance = TRUE)

# which predictors are important?
importance(bag.farm2)
```

```
##                         1         2         3 MeanDecreaseAccuracy
## stype             0.00000   0.00000   0.00000              0.00000
## landcover         0.00000 133.96133  37.64683            137.13036
## organic_percent  67.00458 265.37861 154.09342            142.88294
## avail_water      21.88085  21.18083  11.25450             21.83289
## slope             0.00000   0.00000   0.00000              0.00000
## canopy          236.60182 217.70389  34.28624            309.42261
##                 MeanDecreaseGini
## stype                   0.000000
## landcover               9.978786
## organic_percent      1178.085692
## avail_water           229.453753
## slope                   0.000000
## canopy                176.816696
```

```
varImpPlot(bag.farm2)
```




```
# prediction using bagging
farm.pred.bag2 = predict(bag.farm2, test2, type = 'class')
table(farm.pred.bag2, test2$farm)
```

```
##
## farm.pred.bag2   1   2   3
##              1 165   0   0
##              2   0 362   0
##              3   0   0 315
```

- Based on 'Mean decrease accuracy', canopy is the most important predictor, organic_percent and landcover follows. This result is quite different from the one in single decision tree.
- No misclassified location points in the test set data.

```
# summarize the counts for location points in each farm in training data
train2 %>% group_by(farm) %>%
  summarize(count = n())
```

```
## # A tibble: 3 x 2
##   farm  count
##   <fct> <int>
## 1 1       457
## 2 2      1074
## 3 3       994
```

Maybe in the resampling process of bagging, it is more likely to pick location points from farm2 and farm3 rather than farm1. The resulting decision trees will have fairly different important variables be selected then as the variance for each single decision tree is high.