Tree-based Alternatives to Generalized Linear Models

Ryan Miller

Introduction

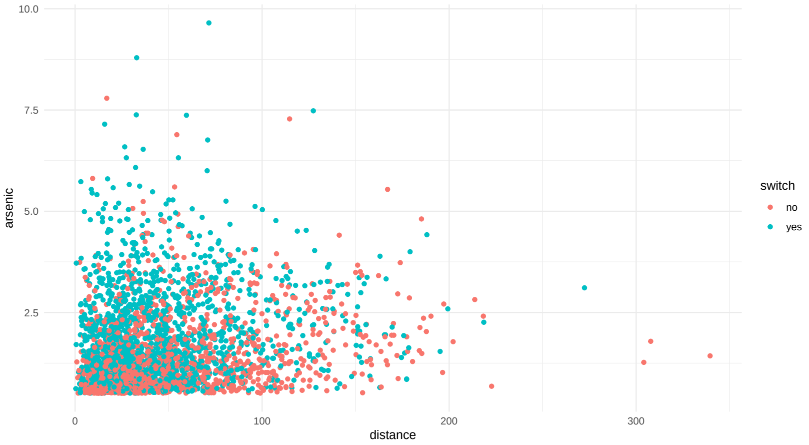
- We've spent the majority of the semester studying *generalized linear models* (linear and logistic regression in particular)
  - The roles of individual predictors are clearly understood
  - The models can be used for statistical inference

Introduction

- We've spent the majority of the semester studying *generalized linear models* (linear and logistic regression in particular)
  - The roles of individual predictors are clearly understood
  - The models can be used for statistical inference
- However, these models can be poorly suited for applications with a high degrees of interaction between predictors, or complex non-linear relationships
  - This week, we'll explore some *non-parametric* alternatives to GLMs

Well Switching

At the end of Lab #9 was an application involving households in Bangladesh switching from high arsenic wells to safer ones:



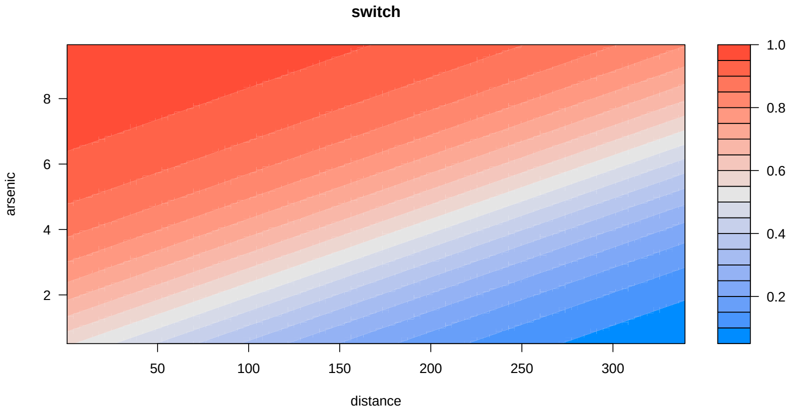
Well Switching - Logistic Regression

Looking at a summary of the logistic regression model: `switch ~ distance + arsenic`, how do these predictors influence the likelihood of switching?

```
##
## Call:
## glm(formula = switch ~ distance + arsenic, family = "binomial",
##     data = Wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6351  -1.2139   0.7786   1.0702   1.7085
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.002749   0.079448   0.035    0.972
## distance    -0.008966   0.001043  -8.593   <2e-16 ***
## arsenic      0.460775   0.041385  11.134   <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: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3930.7  on 3017  degrees of freedom
## AIC: 3936.7
##
## Number of Fisher Scoring iterations: 4
```

Well Switching - Logistic Regression (Visualization)

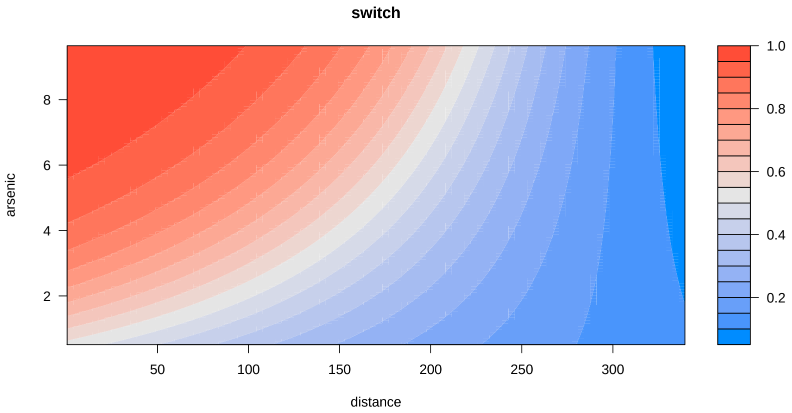
```r
library(visreg)
m <- glm(switch ~ distance + arsenic, data = Wells, family = "binomial")
visreg2d(m, xvar = "distance", yvar = "arsenic", scale = "response")
```



Well Switching - Interactions

However, perhaps there's an interaction between these two variables? How might we decide if this interaction is real?

```r
m <- glm(switch ~ distance*arsenic, data = Wells, family = "binomial")
visreg2d(m, xvar = "distance", yvar = "arsenic", scale = "response")
```



Well Switching - Interactions

A likelihood ratio test suggests borderline statistical evidence of an interaction. . .

```r
library(lmtest)
m1 <- glm(switch ~ distance + arsenic, data = Wells, family = "binomial")
m2 <- glm(switch ~ distance*arsenic, data = Wells, family = "binomial")
lrtest(m1, m2)
```

```
## Likelihood ratio test
##
## Model 1: switch ~ distance + arsenic
## Model 2: switch ~ distance * arsenic
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   3 -1965.3
## 2   4 -1963.8  1 3.0399    0.08124 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Alternatives to Logistic Regression

- In some circumstances, it might make sense to change the modeling approach rather than include numerous interactions in a logistic regression model
  - Logistic regression coefficients can already be difficult to interpret, and interactions will make interpreting the model even more complicated

Alternatives to Logistic Regression

- In some circumstances, it might make sense to change the modeling approach rather than include numerous interactions in a logistic regression model
  - Logistic regression coefficients can already be difficult to interpret, and interactions will make interpreting the model even more complicated
- Classification and Regression Trees (CART) are a type of *non-parametric* model that are well-suited for applications involving many interactive features
  - As you'll soon see, CART models are easily interpreted (even while including numerous interactions between features)

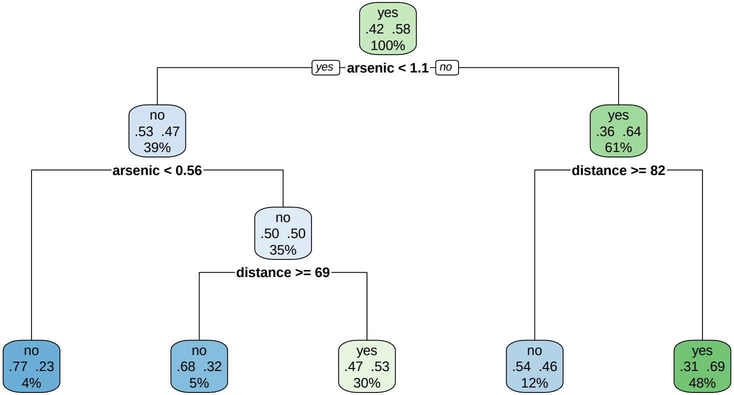
The CART Algorithm

The CART algorithm relies on a procedure known as *recursive binary splitting*:

1) Starting with a "parent" node, search for a splitting rule that maximizes the *homogeneity* or *purity* of the "child" nodes
2) Next, considering each node that hasn't yet been split, find another splitting rule that maximizes *purity*
3) Repeat until a stopping criteria has been reached

Example - CART

```r
library(rpart)
library(rpart.plot)
mytree <- rpart(switch ~ distance + arsenic, data = Wells)
rpart.plot(mytree, extra = 104)
```



How are Splits Determined?

- The CART algorithm works to split parent nodes into child nodes that are as homogeneous (or "pure") as possible
- There are dozens of ways to measure *purity*, but a couple popular ones are:
  - *Gini Index*: a criteria based upon the binomial variance, $p * (1 - p)$ - nodes that are more "pure" have less variance
  - *Information Gain*: A more sophisticated theoretical construct that compares the divergence of two probability distributions

Additional information:

- More on CART splits: http://pages.stat.wisc.edu/~loh/treeprogs/guide/wires11.pdf
- More on Information Gain: https://en.wikipedia.org/wiki/Information_gain_in_decision_trees

When does Splitting Stop?

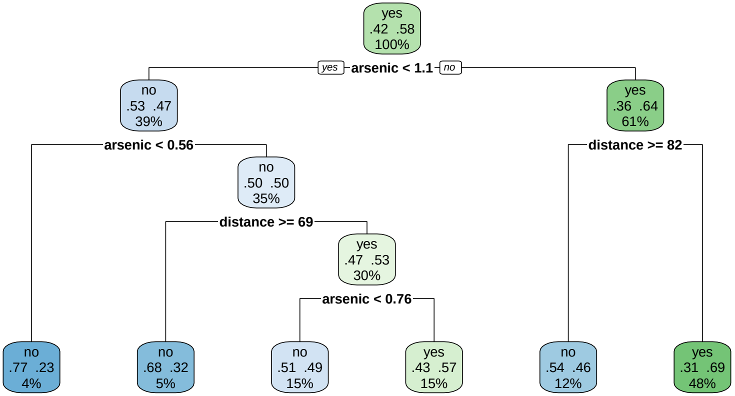
Two factors which determine when the CART algorithm terminates:

1) The *complexity parameter*, *cp*, which defines a minimum factor of improvement in purity that must be achieved in order for a split to be considered "worthwhile" (1% by default in `rpart()`)
2) The *minimum node size*, the minimum number of data-points that must belong to a node for it to be deemed eligible for splitting (20 by default in `rpart()`)

Tuning the CART Algorithm

Notice what happens when cp is set to 0.008 (rather than the default of 0.01). What do you think would happen if it were set to 0.001?

```
mytree <- rpart(switch ~ distance + arsenic, data = Wells,
                control = rpart.control(cp = 0.008, minsplit = 100))
rpart.plot(mytree, extra = 104)
```



Comments on Splitting/Tuning

- In the previous example, notice our new tree is merely our first tree with one additional split
    - This is not a coincidence, the CART algorithm is greedy

Comments on Splitting/Tuning

- In the previous example, notice our new tree is merely our first tree with one additional split
  - This is not a coincidence, the CART algorithm is greedy
- An implication is that we can always go from a larger CART model to a smaller one by ignoring splits that are beneath a certain depth
  - This idea is called "pruning", it is covered in greater detail in this week's lab
  - Pruning is unique to CART models, we couldn't do the same thing in logistic regression

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?
  - No, the models aren't nested.

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?
  - No, the models aren't nested.
- Can we use a model selection criterion like AIC or BIC?

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?
  - No, the models aren't nested.
- Can we use a model selection criterion like AIC or BIC?
  - No, the CART model doesn't involve a likelihood.

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?
  - No, the models aren't nested.
- Can we use a model selection criterion like AIC or BIC?
  - No, the CART model doesn't involve a likelihood.
- Can we compare performance summaries like classification accuracy, Cohen's kappa, or AUC?

Can CART Outperform Logistic Regression?

In our well switching application, how might we determine whether our CART model is better than a logistic regression model?

- Can we use a statistical test, such as the likelihood ratio test?
  - No, the models aren't nested.
- Can we use a model selection criterion like AIC or BIC?
  - No, the CART model doesn't involve a likelihood.
- Can we compare performance summaries like classification accuracy, Cohen's kappa, or AUC?
  - Yes, but we should be careful not to reward overfitting the sample data

Comparison Using Cross-Validation

```r
### Setup
set.seed(123)
fold_id <- sample(rep(1:5, length.out = nrow(Wells)), size = nrow(Wells))
preds1 <- preds2 <- preds3 <- preds4 <- numeric(nrow(Wells))

## Loop across CV folds
for(k in 1:5){

  ## Subset the data
  train <- Wells[fold_id != k, ]
  test <- Wells[fold_id == k, ]

  ## Fit models on the data
  m1 <- glm(switch ~ arsenic*distance, data = train, family = "binomial")
  m2 <- glm(switch ~ arsenic + distance, data = train, family = "binomial")
  m3 <- rpart(switch ~ distance + arsenic, data = train)
  m4 <- rpart(switch ~ distance + arsenic, data = train,
            control = rpart.control(cp = 0.008, minsplit = 100))

  ## Store predictions
  preds1[fold_id == k] <- predict(m1, newdata = test, type = "response")
  preds2[fold_id == k] <- predict(m2, newdata = test, type = "response")
  preds3[fold_id == k] <- predict(m3, newdata = test, type = "prob")[,2]
  preds4[fold_id == k] <- predict(m4, newdata = test, type = "prob")[,2]
}
```

Comparison Using Cross-Validation (continued)

Sadly, both of our CART models have lower *out-of-sample* accuracy than either logistic regression model

```
## Out-of-sample accuracy
pred_class1 <- ifelse(preds1 >= .5, "yes", "no")
out_acc1 <- sum(pred_class1 == Wells$switch)/nrow(Wells)
pred_class2 <- ifelse(preds2 >= .5, "yes", "no")
out_acc2 <- sum(pred_class2 == Wells$switch)/nrow(Wells)
pred_class3 <- ifelse(preds3 >= .5, "yes", "no")
out_acc3 <- sum(pred_class3 == Wells$switch)/nrow(Wells)
pred_class4 <- ifelse(preds4 >= .5, "yes", "no")
out_acc4 <- sum(pred_class4 == Wells$switch)/nrow(Wells)
c(out_acc1, out_acc2, out_acc3, out_acc4)
```

```
## [1] 0.6215232 0.6201987 0.6152318 0.6115894
```

In-sample Model Comparison

It's worthwhile noting our the CART model *does* have higher *in-sample accuracy* than the logistic regression model with an interaction. . .

```
## Larger tree
mytree <- rpart(switch ~ distance + arsenic, data = Wells,
                control = rpart.control(cp = 0.008, minsplit = 100))
preds_tree <- predict(mytree, newdata = Wells, type = "prob")[,2]
preds_tree_class <- ifelse(preds_tree >= .5, "yes", "no")

## Larger Logistic Regression
lreg <- glm(switch ~ arsenic*distance, data = Wells, family = "binomial")
preds_lreg <- predict(lreg, newdata = Wells, type = "response")
preds_lreg_class <- ifelse(preds_lreg >= .5, "yes", "no")

## In-sample accuracy
c(sum(preds_tree_class == Wells$switch)/nrow(Wells),
  sum(preds_lreg_class == Wells$switch)/nrow(Wells))
```

```
## [1] 0.6307947 0.6241722
```

Pros and Cons of Tree-based Models

Pros:

- Work great on highly interactive data
- Don't require the user to specify a parametric model (or perform model selection)
- Easy to visualize and understand
- Easily generalized to nominal or numeric outcomes

Pros and Cons of Tree-based Models

Pros:

- Work great on highly interactive data
- Don't require the user to specify a parametric model (or perform model selection)
- Easy to visualize and understand
- Easily generalized to nominal or numeric outcomes

Cons:

- Tend to overfit the sample data, leading a greater disparity between in-sample and out-of-sample performance (we'll talk more about this next time)
- Effects of individual predictors aren't distinct

Additional Comments

- It isn't necessary to use CART as a "final model", the method used to help discover interactions or non-linear relationships
  - So, even if your application suggests using regression, trees are a useful an exploratory method

Additional Comments

- It isn't necessary to use CART as a "final model", the method used to help discover interactions or non-linear relationships
    - So, even if your application suggests using regression, trees are a useful an exploratory method
- Alternatively, even if logistic regression offers superior predictive performance, a CART model is sometimes still preferable due to its simplicity
    - It's much easier to explain a set of splitting rules to most non-statisticians than it is to explain log-odds or odds ratios