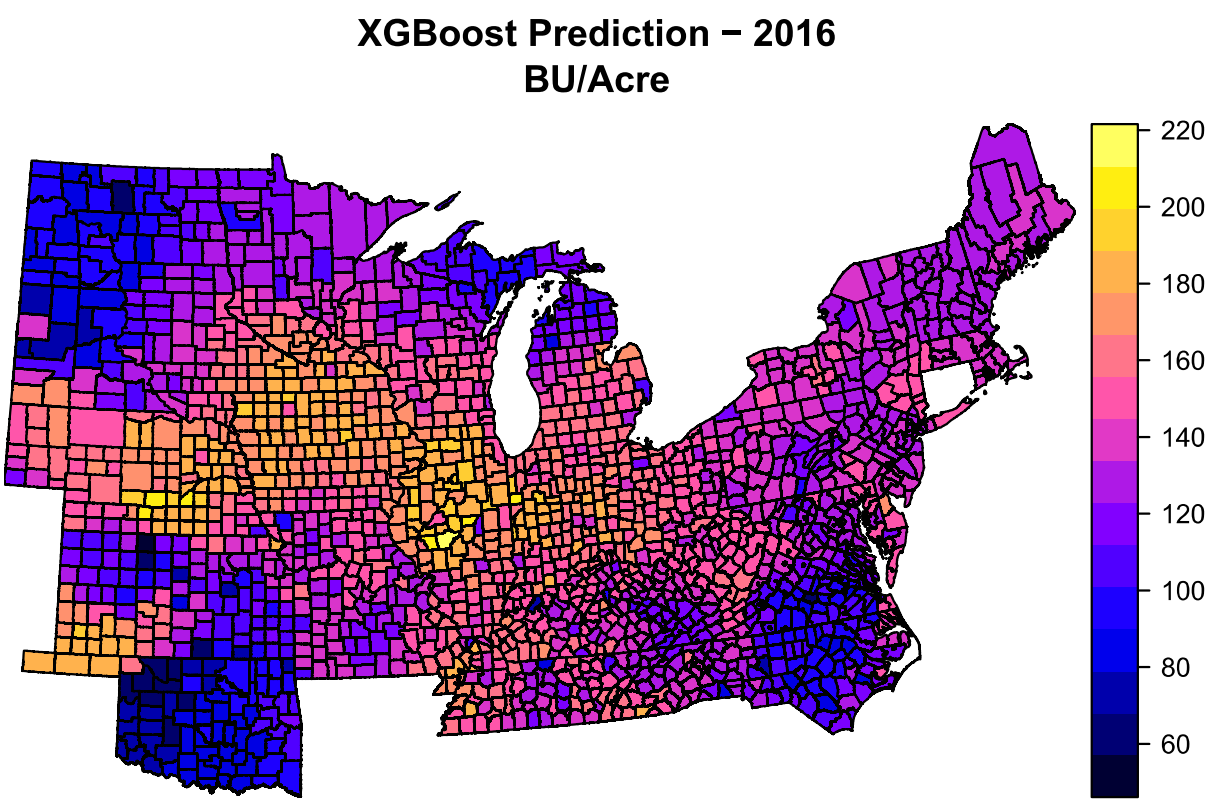
Modeling Corn Yield by County

Intro

There is not a ton of time in a few 45 minute interviews and I wanted to delve a bit in to the types of data that I'd likely be working with if hired for this position. Basically, I wanted to prepare for technical interviews by doing some modeling and playing with data. To that end, I put together this notebook of playing with predicting corn yield at the county level for 2016 by training a model on data prior to 2016. I grabbed data from NASS on corn yield to do this and a variety of remotely sensed imagery and other covariates.

Resulting Model Predictions and Actual Yield

```
county <- readOGR("C:/Users/mhayes1/Desktop/CModel/FinalModelShape.shp", verbose = FALSE)
spplot(county, "Impute", main = "XGBoost Prediction - 2016\nBU/Acre", bty = "n",
    par.settings = list(axis.line = list(col = "transparent")), colorkey = list(axis.line = list(col =
```



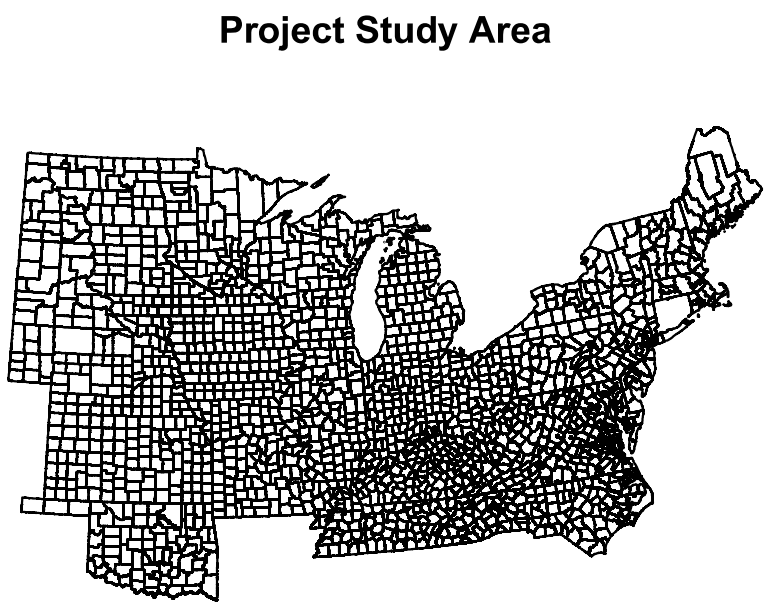
County Boundaries - Study Area

First off, we'll get the US county shapefile from the census bureau. We'll download it, unzip it and clip out iowa for processing the rest of our spatial data and to aggregate data to the county level. I'll also grab a state shapefile to clip out the weird artifacts of county boundaries here.

```
# Download the county boundary file from census bureau
download.file("ftp://ftp2.census.gov/geo/tiger/TIGER2017/COUNTY/tl_2017_us_county.zip",
    destfile = "C:/Users/mhayes1/Desktop/CModel/County/CountyBound.zip")

# Comes down as a zip file, unzip
unzip("C:/Users/mhayes1/Desktop/CModel/County/CountyBound.zip", exdir = "C:/Users/mhayes1/Desktop/CMode

# Download the state boundary file from census bureau
download.file("http://www2.census.gov/geo/tiger/GENZ2017/shp/cb_2017_us_state_500k.zip",
    destfile = "C:/Users/mhayes1/Desktop/CModel/StateBound/StateBound.zip")

# Comes down as a zip file, unzip
unzip("C:/Users/mhayes1/Desktop/CModel/StateBound/StateBound.zip", exdir = "C:/Users/mhayes1/Desktop/CM
```

```
# read into R as a shapefile
county <- readOGR("C:/Users/mhayes1/Desktop/CModel/County/tl_2017_us_county.shp",
    stringsAsFactors = F, verbose = F)

state <- readOGR("C:/Users/mhayes1/Desktop/CModel/StateBound/cb_2017_us_state_500k.shp",
    stringsAsFactors = F, verbose = F)

# Going to use albers equal area projection, good choice for this extent to
# minizmize spatial distortion
projectProjection <- "+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0 +datum=NAD83

# Reduce the counties to our study area. Manually selecting counties since i
# don't have a 'study area' file Basically, same counties as:
# https://granular.ag/accounting-by-soil-type-update/
SA <- county[county$STATEFP %in% c(38, 46, 31, 20, 40, 27, 19, 29, 55, 17, 18, 39,
    21, 47, 37, 51, 54, 26, 24, 10, 34, 42, 9, 36, 44, 25, 33, 50, 23), ]

# Transform the counties into study projection
SA <- spTransform(SA, projectProjection)

# Grab only the states that are in the study area
state <- state[state$STATEFP %in% SA$STATEFP, ]
# Transfrom the state into study projectin
state <- spTransform(state, projectProjection)

# Use raster package crop function to clean up the boundaries in the Study Area
# file
SA <- crop(SA, state)

# Save the SA county boundary
writeOGR(obj = SA, dsn = "C:/Users/mhayes1/Desktop/CModel/ReadySpatial", layer = "StudyArea",
    driver = "ESRI Shapefile", overwrite = T)
```

```r
# Save the State boundary
writeOGR(obj = state, dsn = "C:/Users/mhayes1/Desktop/CModel/ReadySpatial", layer = "StateStudyArea",
    driver = "ESRI Shapefile", overwrite = T)

# plot just Study Area
plot(SA, main = "Project Study Area")
```



NASS Data to Model Yield

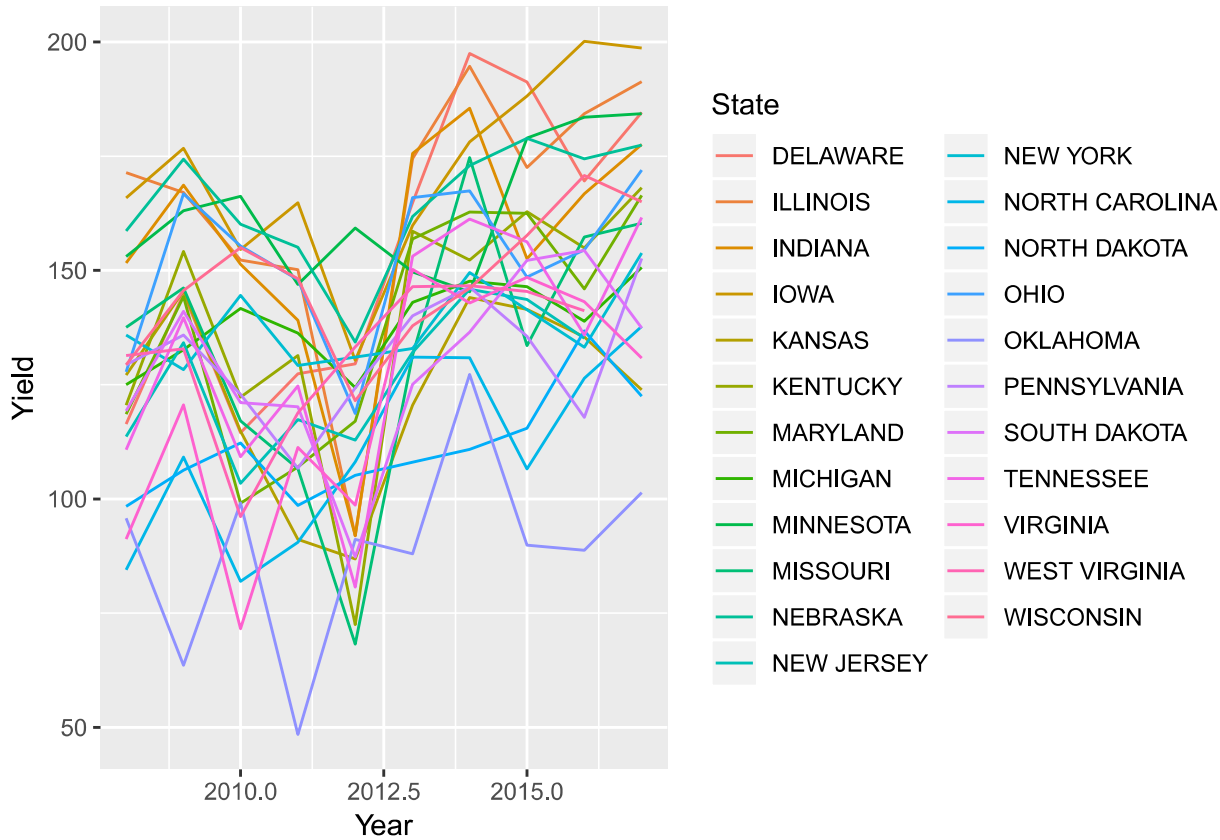
We have a study area file that looks good and now we need some data!

I decided to explore the National Agriculture Statstics Service (NASS) Quick Stats site (https://quickstats.nass.usda.gov/). Here, I accessed Survey data for Corn at the county level. Turns out that NASS has an API with HTTP GET and JSON access. I created a function that builds the appropriate queries; the NASS site restricts downloads so you have to download lots of small pieces. So I make a list of every county, create the function and then use a parallel lapply call to speed up the work.

```r
# We are going to download data by state/county FIPS, so get all of them
downvec <- unique(SA$GEOID)
```

```r
# We'll do an lapply to download all of the data convert downvec to a list. 1,871
# counties worth of data.
downvec <- as.list(downvec)

# create a function to download the data
NASSDown <- function(x) {

    # Attempt to download the data, if download fails, return NULL
    intdat <- tryCatch({
        path <- paste0("https://quickstats.nass.usda.gov/api/api_GET/?key=8DC4E243-37AB-385D-9F74-0527D
            substr(x, 3, 5), "&state_fips_code=", substr(x, 1, 2), "&format=json")

        # Download Data via API from NASS
        intdat <- jsonlite::fromJSON(path)[[1]]
    }, error = function(cond) {
        return(NULL)
    }, warning = function(cond) {
        return(NULL)
    })
    # return whatever data was grabbed
    return(intdat)
}


# I am impatient and not using much bandwidth with these small API requests, so I
# cluster to speed up.

cl <- makeCluster(15, type = "SOCK")
# Parallelize NASSDown function Nice! Only takes 132 seconds! Probably saved
# about 18 minutes.
system.time(ddown <- parLapply(cl = cl, downvec, NASSDown))

# close the cluster
stopCluster(cl)

# Rbind everything into a data.frame
ddown <- do.call("rbind", ddown)

# subset data and fix the column names
ddown <- ddown[, c(2, 27, 7, 39, 18, 16, 33)]
names(ddown) <- c("State", "StateFips", "CountyFips", "Year", "Yield", "VarDesc",
    "Units")

# we get a few pieces of information but i just want to model yield in bu/acre
ddown <- ddown[ddown$VarDesc == "CORN, GRAIN - YIELD, MEASURED IN BU / ACRE", ]
```

```r
ddown <- read.csv("C:/Users/mhayes1/Desktop/CModel/YieldData.csv", stringsAsFactors = F)
# just take a quick look at the data
agg <- aggregate(as.numeric(ddown$Yield) ~ ddown$State + ddown$Year, FUN = mean)
names(agg) <- c("State", "Year", "Yield")


ggplot(data = agg, aes(x = Year, y = Yield, group = State, colour = State)) + geom_line()
```




```r
head(ddown)
```

```
##      State StateFips CountyFips Year Yield
## 1 NEBRASKA        31         39 2017 203.9
## 2 NEBRASKA        31         39 2016 197.8
## 3 NEBRASKA        31         39 2015 200.1
## 4 NEBRASKA        31         39 2014 181.1
## 5 NEBRASKA        31         39 2013 192.8
## 6 NEBRASKA        31         39 2012  95.7
##                                     VarDesc     Units
## 1 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
## 2 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
## 3 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
## 4 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
## 5 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
## 6 CORN, GRAIN - YIELD, MEASURED IN BU / ACRE BU / ACRE
```

```r
write.csv(ddown, "C:/Users/mhayes1/Desktop/CModel/YieldData.csv", row.names = F)
```