Final Project

Rylie Fleckenstein

08/07/2020

Introduction

The objective of this final project is to take our field that is broken down into optimal grid cells, and then determine the level of productivity that can be expected out of each individual cell and how reliable that level of productivity is from year to year within that cell. By understanding the nature of the field and the cells within it, we are able to make sure, while conducting some future experiment between two different agricultural practices, that the only independent variable of the experiment is the agricultural practice. We will have insight into what type of variance is normal within the field when only one agricultural practice is being used, and therefore will be able to formulate a plan of implementation for our experiment that spreads the practices evenly across all types of cells. The midterm project supplemented this project because it allowed us to determine the optimum number of grid cells we should use for an experiment. For this project we are taking that optimum number of grid cells and then expanding upon that concept and diving into the nature of each grid cell in order to understand the nature of the field as a whole.

Data

```
##      Yield Latitude Longitude           TimeStamp
## 1 43.57736 399.4129  3.274650 2013-09-30 18:47:00
## 2 47.40136 397.4667  3.257958 2013-09-30 18:47:01
## 3 46.38477 395.5015  3.216063 2013-09-30 18:47:02
## 4 49.19995 393.5342  3.257626 2013-09-30 18:47:03
## 5 42.86166 391.6324  3.258953 2013-09-30 18:47:04
## 6 47.11753 389.7365  3.253979 2013-09-30 18:47:05
```

Functions

Explanation

Below is the function I wrote, "timeSpan". The purpose of the function is to convert the date strings found in the data into Date Time objects so that I can calculate the span of time over which the field was harvested. One of the constraints for this analysis was that only sets of data which were gathered over a period of time less than 7 days were to be used. The reason for the 7 day time constraint was to make sure the variance within the data was not due to different growing periods. In this function I used the lubridate library and its function 'parse_date_time()' to convert the date strings in the data files into date time objects. I then calculated the time span by subtrating the min(Time) found from the max(Time).

```r
timeSpan <- function(df){
  #creates a date time object from the date string found in the data
  df$Time <- lubridate::parse_date_time(df$TimeStamp, orders = "ymdHMS", tz="America/Chicago")
  #calculating the span of time over the entire data set
  time_span <- max(df$Time) - min(df$Time)
return(time_span)
}
```

Data Screening

Explanation

Again, part of the data screening process is to make sure that only data sets with time spans less than 7 days are being used. In order to screen each data file I called my function "timeSpan" on each data set. Below we can see that harvest2013 took 6.9 hours from start to finish, harvest2015 took 2.9 days, harvest2016 took 4.7 days, harvest2017 took 5.4 hours, and harvest2018 took 5.7 hours. Therefore, we can use all sets of data because they all meet the time constraint.

```
## Time span of 2013 data:

## Time difference of 6.999167 hours

## Time span of 2015 data:

## Time difference of 2.946551 days

## Time span of 2016 data:

## Time difference of 4.717743 days

## Time span of 2017 data:

## Time difference of 5.479722 hours

## Time span of 2018 data:

## Time difference of 5.774722 hours
```

Grid (20 rows, 6 columns, 120 cells)

Explanation

The original data (harvest*Year*.dat) consists of 4 attributes, Yield, Latitude, Longitude, and time stamp. There are around 10,000 observations for each year. The goal of the midterm project was to determine the optimimal number of grid cells the field (Yield data) should be broken into to be able to see percent difference in two different agricultural practices. It was concluded by Proffessor Clausen that, for the final project, we were to break the field down into 6 columns and 20 rows which equates to 120 grid cells. The

code above is written to create a column index, a row index, and then a cell index. The above code was written by proffessor Clausen and was shown to the students during office hours. I have used his code here for this part of my project. The code does two simple things. First, the data is broken down by longitude and latitude in such a way that there are now 120 grid cells with each observation having a col column and row column showing us which grid cell they belong to. The code then combines the row index and column index in order to make a complete Cell index. This is important for later parts of this analysis where I merge other data frames with this one or one like it and want to make sure that all attributes of a cell are properly indexed and lined up with each other.

```
##      Yield Latitude Longitude           TimeStamp col row Cell
## 1 43.57736 399.4129  3.274650 2013-09-30 18:47:00   1  20 20:1
## 2 47.40136 397.4667  3.257958 2013-09-30 18:47:01   1  20 20:1
## 3 46.38477 395.5015  3.216063 2013-09-30 18:47:02   1  20 20:1
## 4 49.19995 393.5342  3.257626 2013-09-30 18:47:03   1  20 20:1
## 5 42.86166 391.6324  3.258953 2013-09-30 18:47:04   1  20 20:1
## 6 47.11753 389.7365  3.253979 2013-09-30 18:47:05   1  20 20:1
```

Estimates and Ranking

Where the $i^{th} Yield$ observation for Year j is denoted as y_{ij}, we replaced y_{ij} with $r_{ij} = rank(y_{ij})$ and we determine ranks independently for $j = 1, 2, \ldots, J$ for years $\{2013, 2015, \ldots, 2018\}$

Explanation

The original data sets are all around 10,000 observations which I broke down into 120 cells. Therefore, I needed to create new data frames, of 120 observations, for each year's data set. The above 'cells_*Year*.dat' data frames all have 3 columns which are the following: the cell indentifier 'Cell', the cell Yield estimate 'Estimates' and the cell's rank based upon the yield estimate 'Rank'. The yield estimate is calculated by taking the mean of all yield observations within a particular cell. This value is then normalized through rank. The farmer planted several different crops in the field throughout the different years so in order to properly compare the yield of a single cell against itself over the years, the data needs to be normalized or viewed on the same scale.

```
##   Cell Estimates Rank
## 1  1:1  40.78269   55
## 2  1:2  44.30403  109
## 3  1:3  49.37674  120
## 4  1:4  43.24956  100
## 5  1:5  41.30299   64
## 6  1:6  37.92880   27
```

Merging Data

We merged the data sets by grid cell number. For each cell i, we calculated a normalized mean and a standard deviation over the J estimates. That is, if n_{ij} is the normalized yield value for grid cell i in year j, then calculate the following (mean, variance, standard deviation)

$$\bar{n_{i\cdot}} = \frac{\sum_{j=1}^{J} n_{ij}}{J}$$

$$s_{i.}^2 = \frac{\sum_{j=1}^{J}(n_{ij} - \bar{n_{.j}})^2}{J-1}$$

$$\sigma_{i.} = \sqrt{s_{i.}^2}$$

Explanation

Here I conducted a large merge with all 5 cell data sets so that I could have a dataframe where each observation is a unique cell and all 5 ranks for that cell correlated to their respective year.

```
##   Cell Rank.2013 Rank.2015 Rank.2016 Rank.2017 Rank.2018
## 1  1:1        55         8        60        39       118
## 2  1:2       109       102        48        48        56
## 3  1:3       120       119        79        64        70
## 4  1:4       100       113       103        99        25
## 5  1:5        64        97       118        36        32
## 6  1:6        27       115        97       102        13
```

Calculating mean

```
for (i in 1:length(Cells_combined.dat$Cell)){
  Cells_combined.dat$Rank.Mean[i]=((sum(
    Cells_combined.dat[i,2],Cells_combined.dat[i,3],
    Cells_combined.dat[i,4], Cells_combined.dat[i,5],
    Cells_combined.dat[i,6]))/5)

}
```

Explanation

Since the yearly ranks for each cell were all in a row I needed to write a for loop to calculate the rank mean for each cell using the above formula. The loop iterates over each cell observation and for each observation takes the sum of columns 2-6 (which are all the ranks) and divides this number by 5 which is how many years there are. This returns the mean rank for each cell.

The reason the Rank.Mean is important and something that needed to be calculated is because we want to know how well each cell did overall. We needed to know each cell's average rank over the 5 observable years.

```
##   Cell Rank.Mean
## 1  1:1      56.0
## 2  1:2      72.6
## 3  1:3      90.4
## 4  1:4      88.0
## 5  1:5      69.4
## 6  1:6      70.8
```

Calculating Variance and Standard Deviation

```
  for (i in 1:length(Cells_combined.dat$Cell)){
  Cells_combined.dat$Variance[i]=(sum(
    ((Cells_combined.dat[i,2]-Cells_combined.dat$Rank.Mean[i])**2),
    ((Cells_combined.dat[i,3]-Cells_combined.dat$Rank.Mean[i])**2),
    ((Cells_combined.dat[i,4]-Cells_combined.dat$Rank.Mean[i])**2),
    ((Cells_combined.dat[i,5]-Cells_combined.dat$Rank.Mean[i])**2),
    ((Cells_combined.dat[i,6]-Cells_combined.dat$Rank.Mean[i])**2)
    )/4)

  }
Cells_combined.dat$SD <- sqrt(Cells_combined.dat$Variance)
```

Explanation

Again, since the yearly ranks for each cell were all in a row I needed to write a for loop to calculate the rank variance for each cell using the above formula. I was able to the sum of the the result squared for each calculated Rank.Mean subtracted from each yearly Rank. I then divided this result by 4 because J is 5 for the number of data sets and we needed $J-1$. I then simply took the square root of the variance to find the standard deviation of each cell over the years.

The reason we want to know the variance and standard deviation of the cells is beacause we want to know how stable or reliable the grid cells yield estimate is. We want to see how its rank differed over the years and by how much. Therefore, we calculated the cell's rank standard deviation over the 5 years.

```
##   Cell Rank.Mean Variance       SD
## 1  1:1      56.0   1613.5 40.16840
## 2  1:2      72.6    918.8 30.31171
## 3  1:3      90.4    734.3 27.09797
## 4  1:4      88.0   1271.0 35.65109
## 5  1:5      69.4   1416.8 37.64040
## 6  1:6      70.8   2218.2 47.09777
```

Classification

If the mean normalized score for a grid cell is in the largest 25% percent of all cells, classify this as a High yielding cell. If the mean normalized score is in the smallest 25%, classify this cell as Low yielding. Otherwise, classify the cell as Average yield.

Similarly, if the standard deviation of the normalized scores for a grid cell is in the largest 25% percent of all cells, classify this as an Unstable yielding cell. If the standard deviation of the normalized scores is in the smallest 25%, classify this cell as Stable yielding. Otherwise, classify the cell as Average yield.

Catergorizing means

```
 for (i in 1:length(Cells_combined.dat$Rank.Mean)){
   #top 25% is from 91:120
   if (Cells_combined.dat$Rank.Mean[i] %in% sort(Cells_combined.dat$Rank.Mean)[91:120]){
Cells_combined.dat$MeanCategory[i] <- 'High'
   }
   #bottom 25% is from 1:30
   else if (Cells_combined.dat$Rank.Mean[i] %in% sort(Cells_combined.dat$Rank.Mean)[1:30]){
Cells_combined.dat$MeanCategory[i] <- 'Low'
   }
   #middle 60% is the rest
   else {
     Cells_combined.dat$MeanCategory[i] <- 'Average'
   }
 }
```

Explanation

The above for loop was written to iterate through all instances of Rank.Mean and assign either 'High', 'Average', or 'Low' based on the above criteria. I deciced to sort the column Rank.Mean and index the top 25% and the bottom 25%.

```
##   Cell MeanCategory
## 1  1:1      Average
## 2  1:2      Average
## 3  1:3         High
## 4  1:4         High
## 5  1:5      Average
## 6  1:6      Average
```

Categorizing SD's

```
 for (i in 1:length(Cells_combined.dat$SD)){
   #top 25% is from 91:120
   if (Cells_combined.dat$SD[i] %in% sort(Cells_combined.dat$SD)[91:120]){
Cells_combined.dat$SDCategory[i] <- 'Unstable'
   }
   #bottom 25% is from 1:30
   else if (Cells_combined.dat$SD[i] %in% sort(Cells_combined.dat$SD)[1:30]){
Cells_combined.dat$SDCategory[i] <- 'Stable'
   }
    #middle 60% is the rest
   else {
     Cells_combined.dat$SDCategory[i] <- 'Average'
   }
 }
```

Explanation

The above for loop was written to iterate through all instances of SD and assign either 'Unstable', 'Average', or 'Stable' based on the above criteria. I deciced to sort the column SD and index the top 25% and the

bottom 25%.

```
##   Cell MeanCategory SDCategory
## 1  1:1      Average   Unstable
## 2  1:2      Average    Average
## 3  1:3         High    Average
## 4  1:4         High   Unstable
## 5  1:5      Average   Unstable
## 6  1:6      Average   Unstable
```

Additional Merging

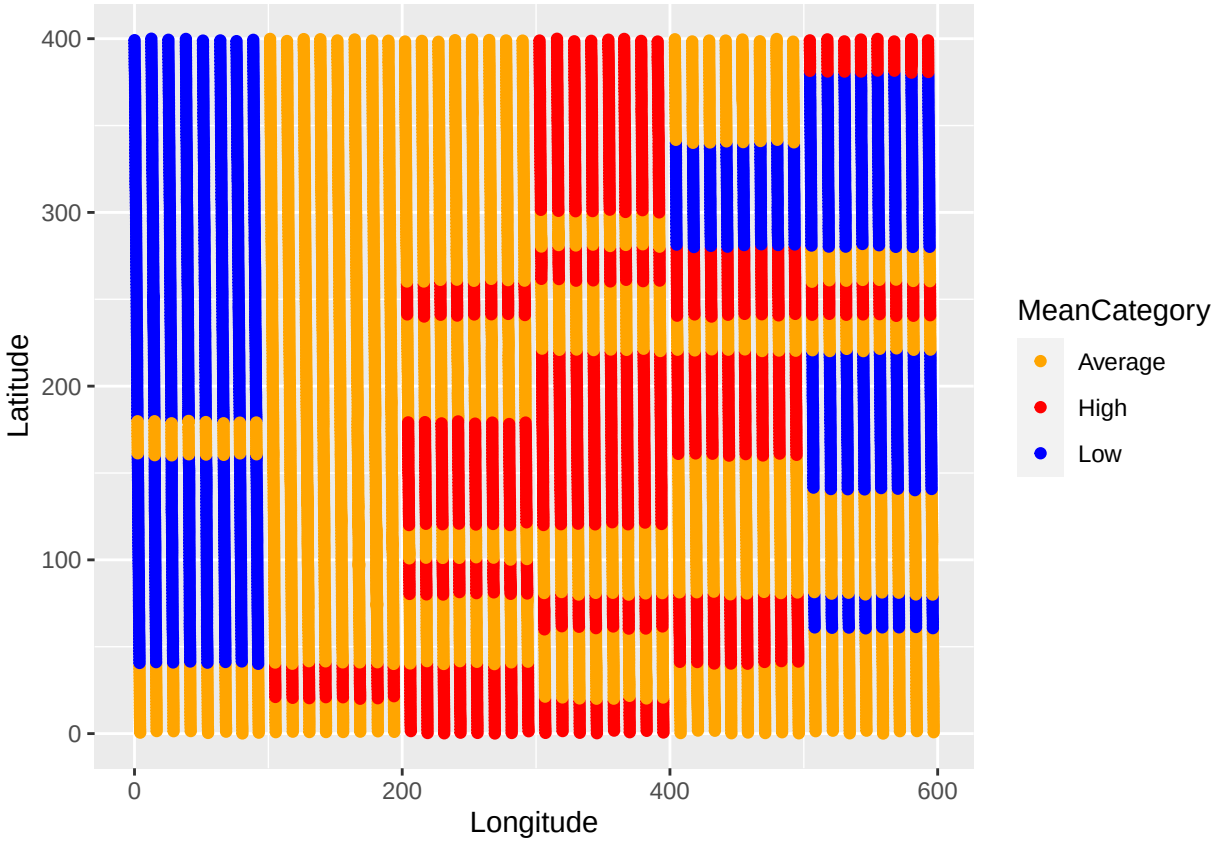
Now that I have each cell categorized I needed to plot field. In order to plot the field I need coordinates in the format of longitude and latitude. To accomplish this created the harvest.dat dataframe which is the longitude latitude for the field already indexed by cell number. I then merged this dataframe with the categories, by cell number, in order to create the categorized data frame. This ensured proper cell number indexing.

```
##   Latitude Longitude Cell
## 1 399.2460  264.9795 20:3
## 2 397.4478  265.0273 20:3
## 3 395.7073  265.0585 20:3
## 4 393.9365  265.0588 20:3
## 5 392.1895  265.0903 20:3
## 6 390.4362  265.1195 20:3
```

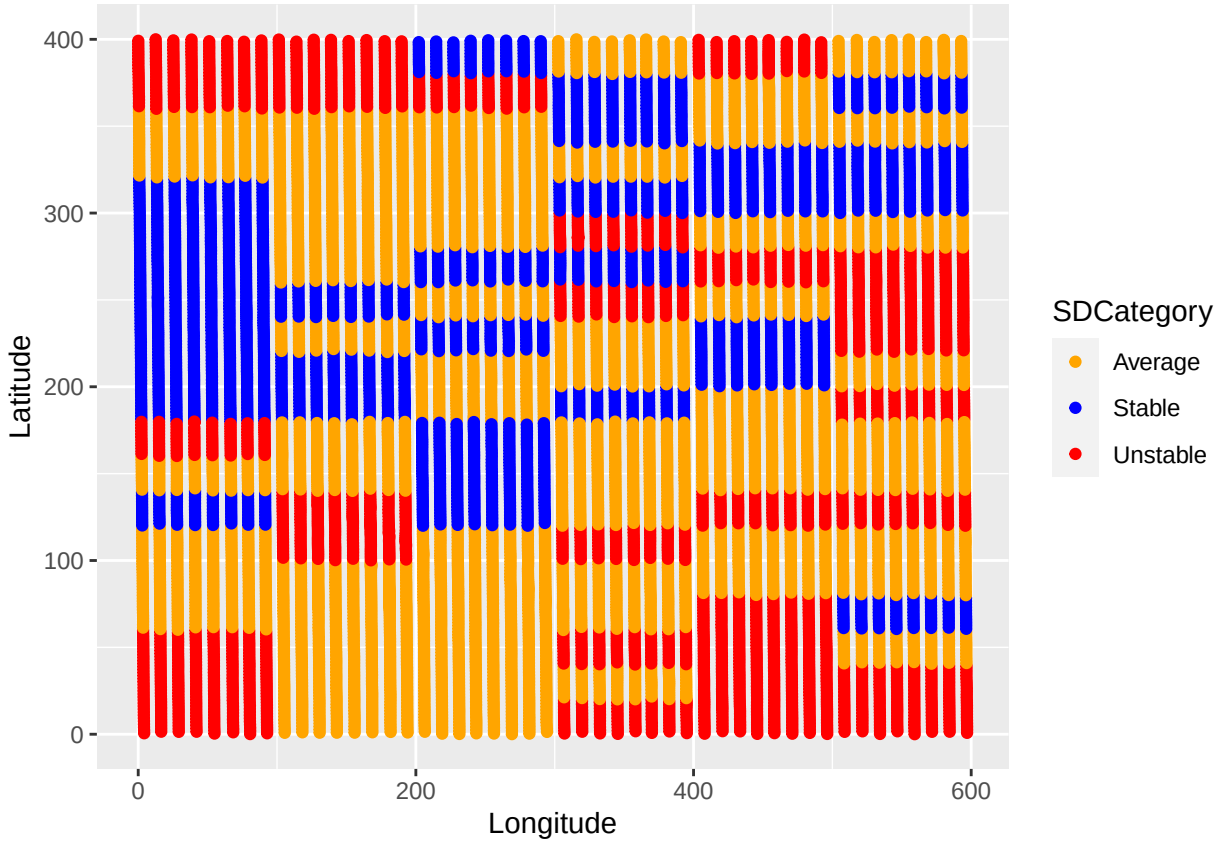
```
##   Cell   Latitude Longitude MeanCategory SDCategory
## 1  1:1 18.2787006 80.372048      Average   Unstable
## 2  1:1 16.4635406 80.442242      Average   Unstable
## 3  1:1  3.0934322 29.203547      Average   Unstable
## 4  1:1  1.3897430 29.214601      Average   Unstable
## 5  1:1  0.5576202  4.278226      Average   Unstable
## 6  1:1  4.8358266 29.174917      Average   Unstable
```

Plots

This plot shows the yield mean category. Whether or not the yield, over the 5 years, is high, average, or low.



This plot shows how stable each cell is over the years. If the cell is unstable then the standard deviation for that cell over the last 5 years is high. If the cell is stable then the standard deviation for that cell over the last 5



years is low.

Conclusion

We were able to successful plot which areas of the field are expected to have high, average, or low yields and which parts of the field are stable, unstable, or just average. This information can be put to use in some type of experiment where we test two different agricultural practices against each other. By understanding the nature of the field, we are able to implement an experiment plan that ensures the agricultural practice is the only independent variable and that there are not any other variables skewing the results of the test.