Final Project

Amin Baabol

08/02/2020

Synopsis

This project carries on from the analysis performed in the midterm project. Essensially, we are combining five years worth of data gathered on a single field through yield monitoring system. Our task is to divide the field into optimal grid cells(which we have done in the midterm project) and then compute yield estimate for each optimized cell unit for the purpose of merging our data by cell unit across all five years. Finally, given that various crops are grown in parallel sequence we will have to normalize our merged data to calculate normalized yield estiamtes and standard deviation for each individual cell unit across all five year to account for the discrepencies in yield estimates amongs the various crop.

Assumption(s)

Assumption(s) made in this project: 1. 120 grid cells are assummed to be the optimal grid cell number per Dr. Claussen's recommendations

Steps:

1. Upload the csv files for all five year
2. Plot the raw data to get a visual representation before undergoing analysis
3. Sample uniformity:Check to esnure the harvest interval is less than 1 week
4. Cell Division,aggregation and normalization of yield estimates
5. Ranking the merged data
6. Classifications and plotting grid cells according to the normalized mean and standard deviation criteria:
    a. if the normalized mean/standardard deviation is in the top 25th percentile then classify it as high/unstable yield
    b. if the normalized mean/standard deviation is in the bottom 25th percentile then classify it as low/stable yield
    c. if the normalized mean/standard deviation is in between a and b then classify it as average yield

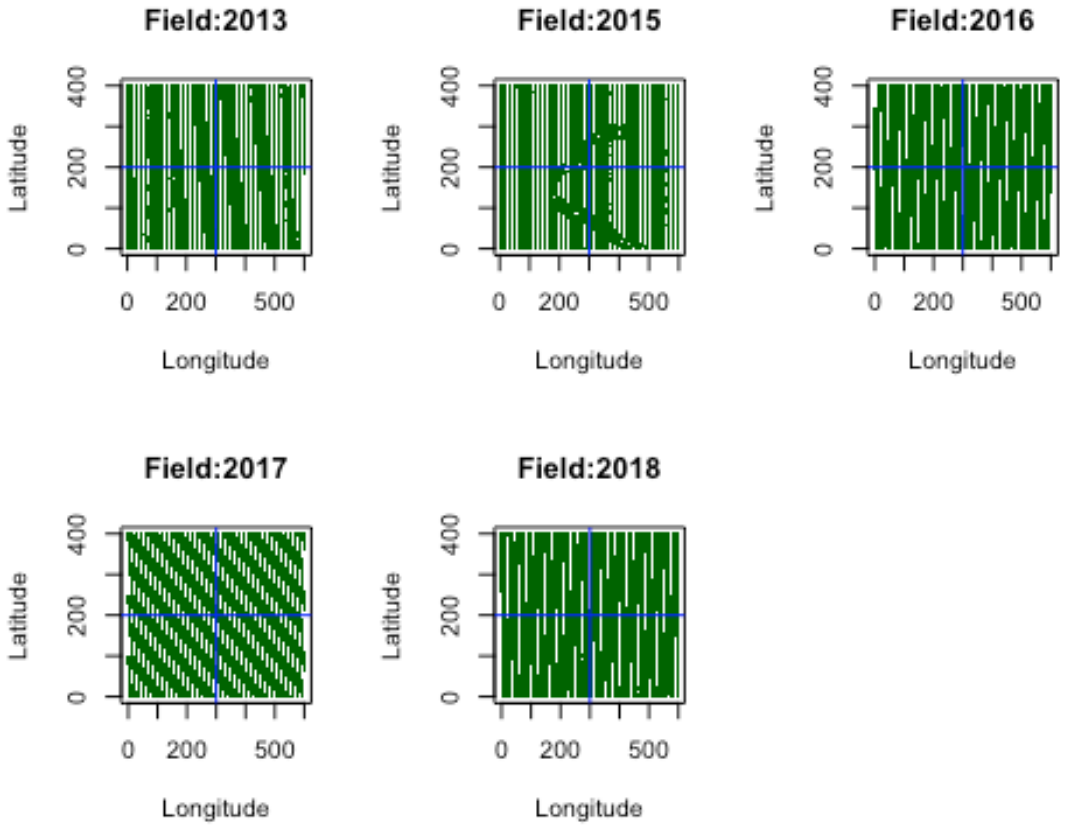
##Step 1:Upload the csv files for all five year

```
data.2013 <- data.frame(read.csv("~/Desktop/work/GradSchool/Summer2020/STATS600/FinalProject/home.2013.csv", header=T, sep = ","))
data.2015 <- data.frame(read.csv("~/Desktop/work/GradSchool/Summer2020/STATS600/FinalProject/home.2015.csv", header=T, sep = ","))
```

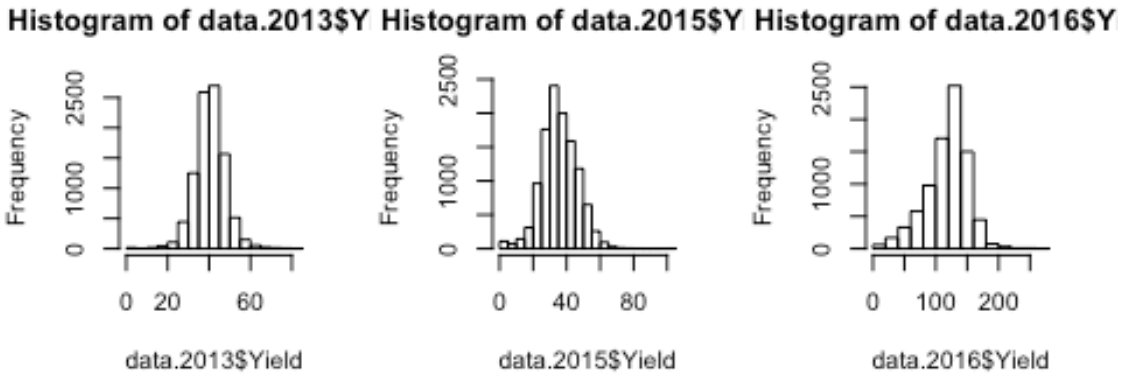
```
data.2016 <- data.frame(read.csv("~/Desktop/work/GradSchool/Summer2020/STATS6
00/FinalProject/home.2016.csv", header=T, sep = ","))
data.2017 <- data.frame(read.csv("~/Desktop/work/GradSchool/Summer2020/STATS6
00/FinalProject/home.2017.csv", header=T, sep = ","))
data.2018 <- data.frame(read.csv("~/Desktop/work/GradSchool/Summer2020/STATS6
00/FinalProject/home.2018.csv", header=T, sep = ","))
```

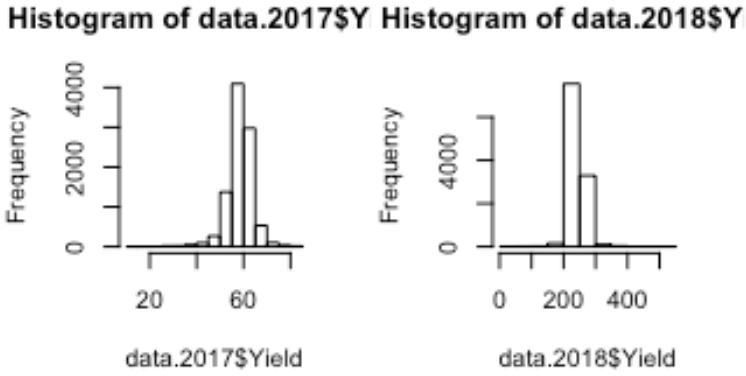
##Step 2:Plot the data to get a visual representation

```
par(mfrow=c(2,3))
plot(Latitude ~ Longitude, data=data.2013,main="Field:2013", col="dark green"
,pch='.')
abline(v=300, col='blue',pch=".")
abline(h=200,col='blue',pch=".")
plot(Latitude ~ Longitude, data=data.2015,main="Field:2015", col="dark green"
,pch='.')
abline(v=300, col='blue',pch=".")
abline(h=200,col='blue',pch=".")
plot(Latitude ~ Longitude, data=data.2016,main="Field:2016", col="dark green"
,pch='.')
abline(v=300, col='blue',pch=".")
abline(h=200,col='blue',pch=".")
plot(Latitude ~ Longitude, data=data.2017,main="Field:2017", col="dark green"
,pch='.')
abline(v=300, col='blue',pch=".")
abline(h=200,col='blue',pch=".")
plot(Latitude ~ Longitude, data=data.2018,main="Field:2018", col="dark green"
,pch='.')
abline(v=300, col='blue',pch=".")
abline(h=200,col='blue',pch=".")

par(mfrow=c(2,3))
```

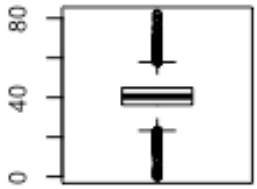


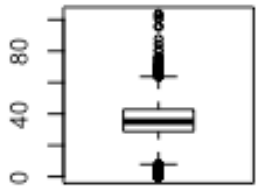

```
hist(data.2013$Yield)
hist(data.2015$Yield)
hist(data.2016$Yield)
hist(data.2017$Yield)
hist(data.2018$Yield)

par(mfrow=c(2,3))
```

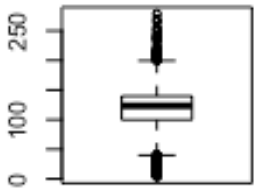


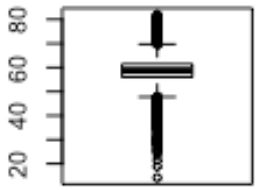


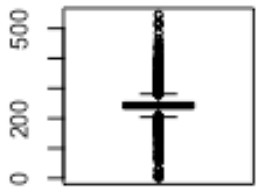

```r
boxplot(data.2013$Yield)
boxplot(data.2015$Yield)
boxplot(data.2016$Yield)
boxplot(data.2017$Yield)
boxplot(data.2018$Yield)

par(mfrow=c(2,3))
```



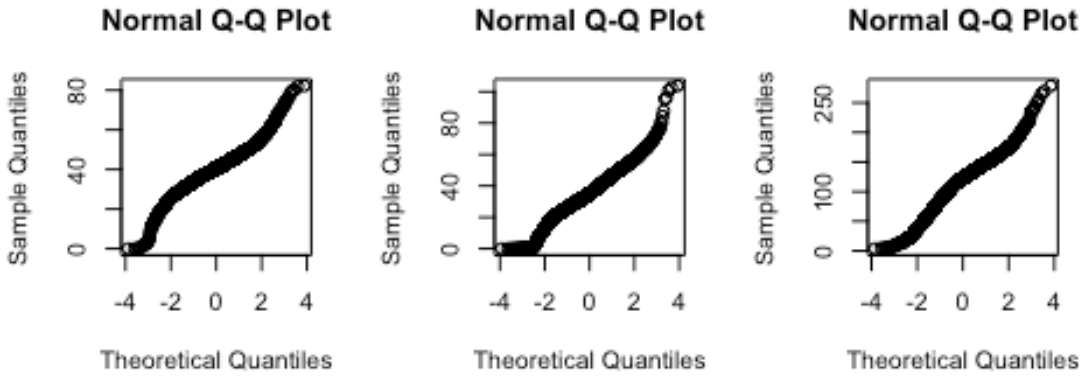


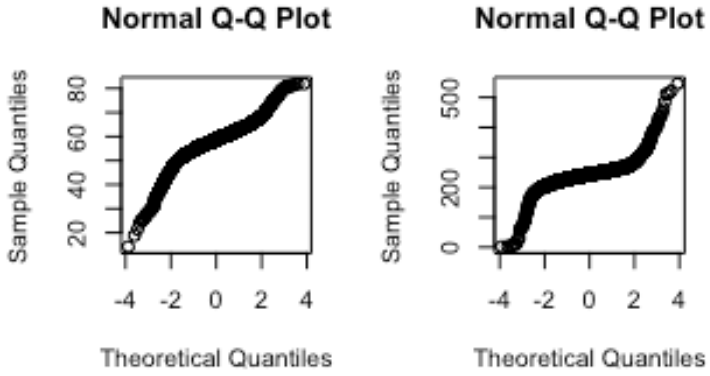






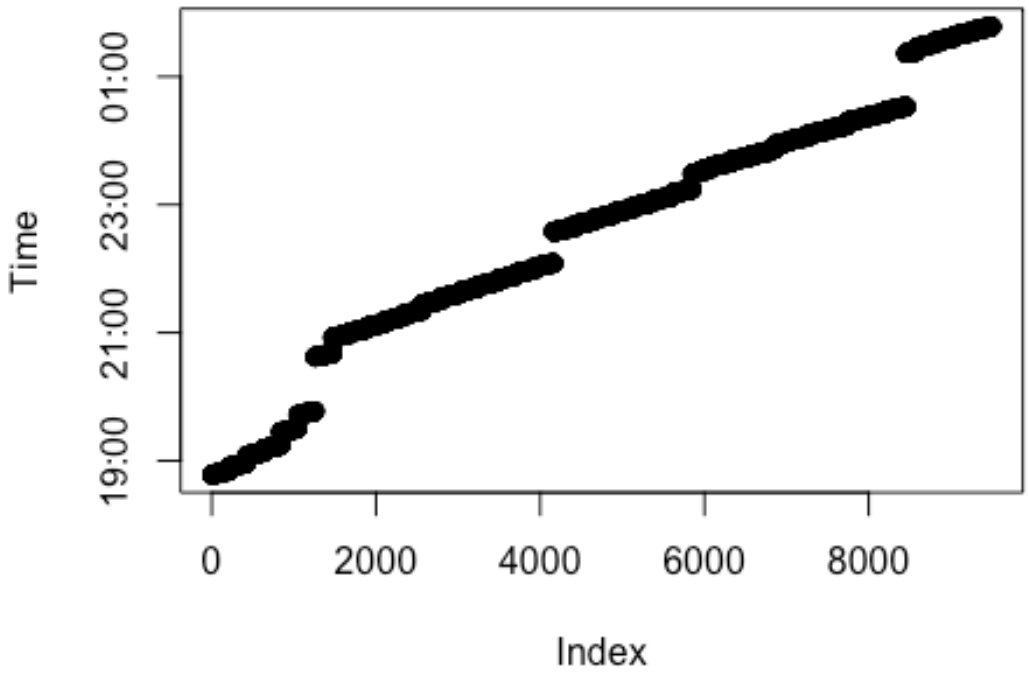

```r
qqnorm(data.2013$Yield)
qqnorm(data.2015$Yield)
qqnorm(data.2016$Yield)
qqnorm(data.2017$Yield)
qqnorm(data.2018$Yield)
```



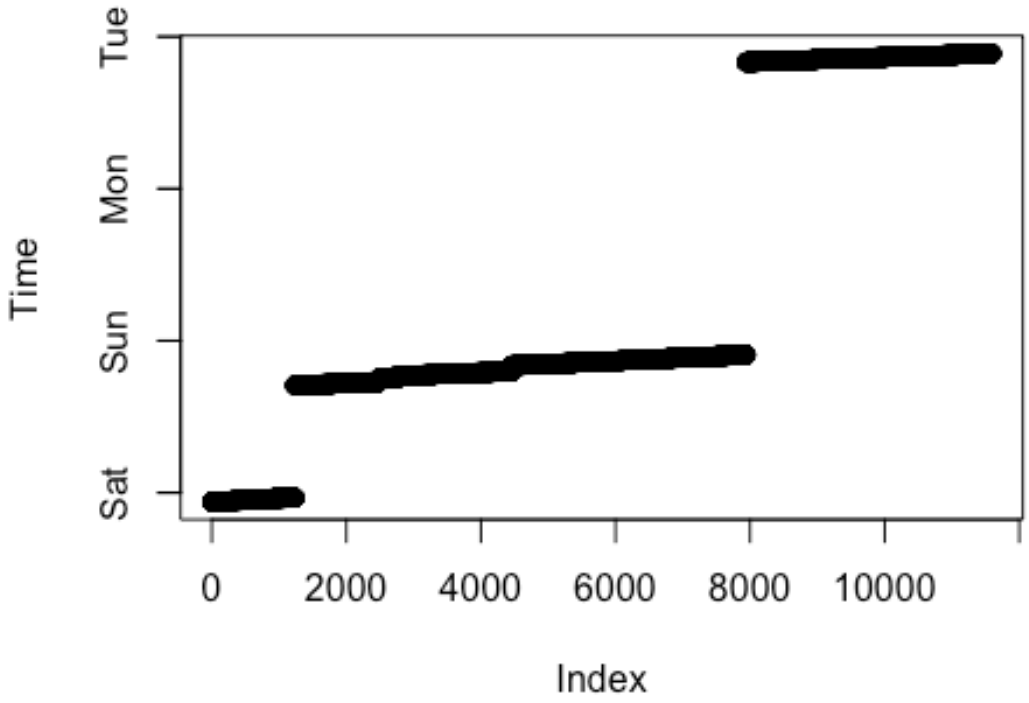


Step 3:Check to ensure the harvest interval is less than 1 week

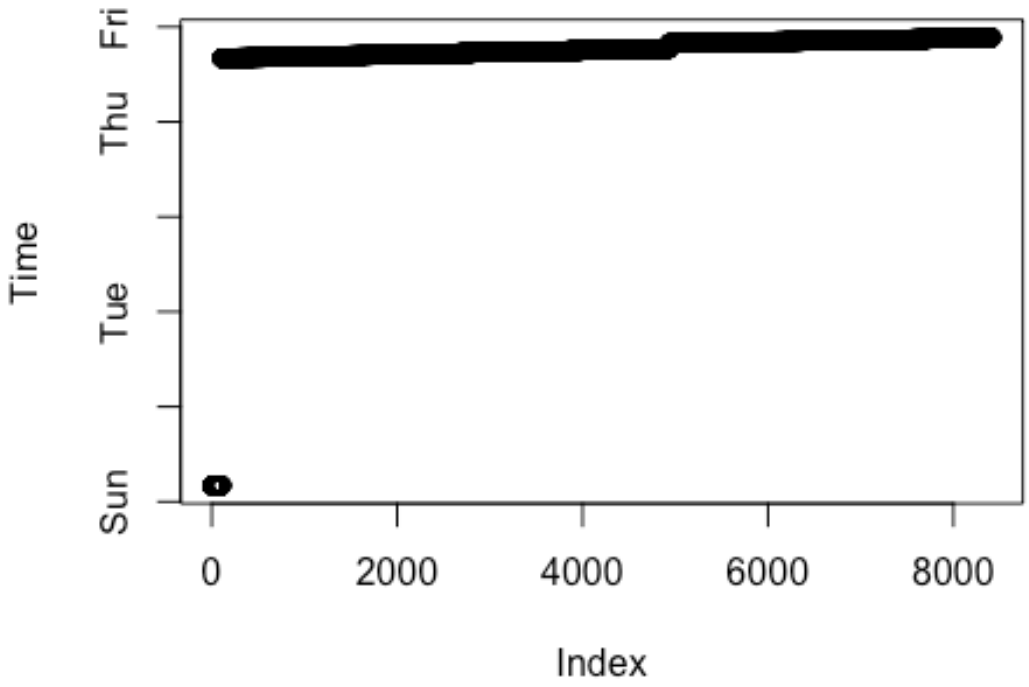
```
plot(as.POSIXct(data.2013$TimeStamp), ylab = "Time")
```



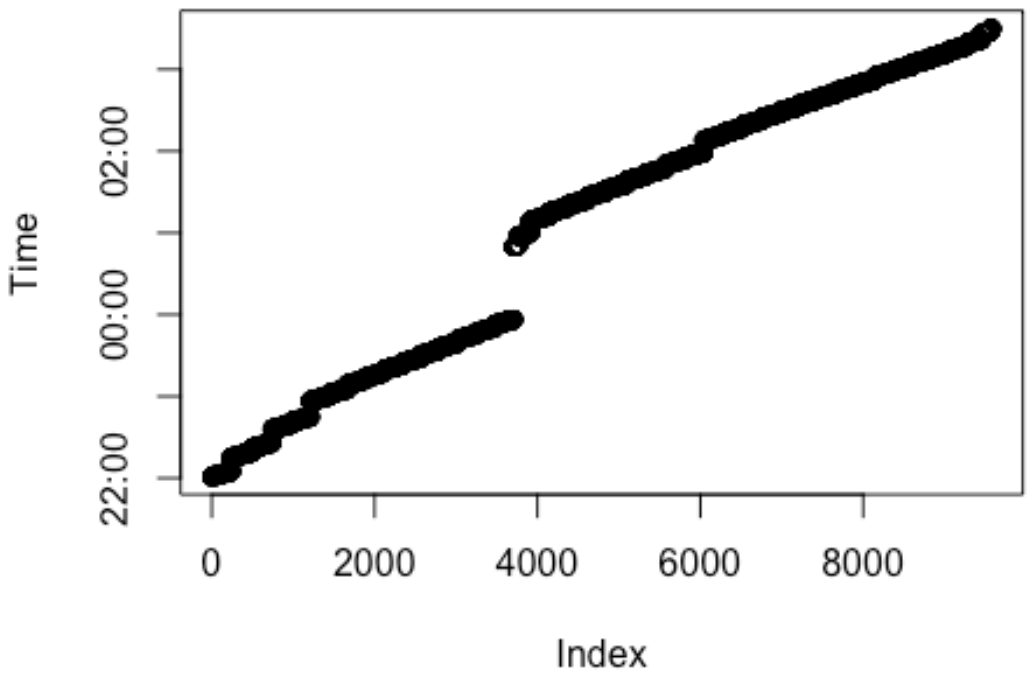

```r
plot(as.POSIXct(data.2015$TimeStamp), ylab = "Time")
```



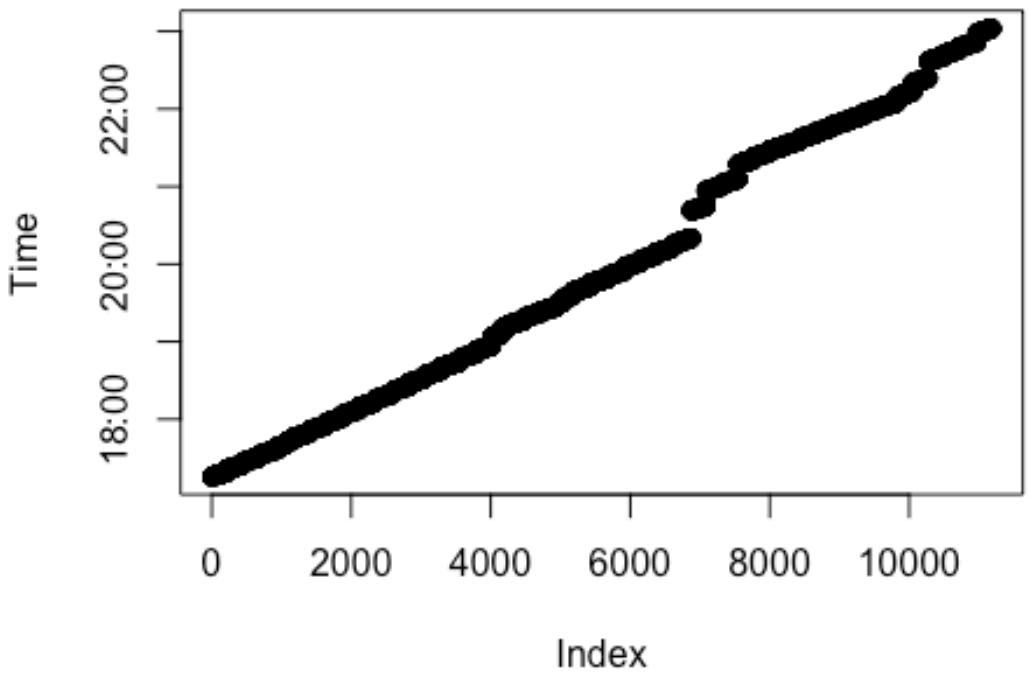

```r
plot(as.POSIXct(data.2016$TimeStamp), ylab = "Time")
```




```r
plot(as.POSIXct(data.2017$TimeStamp), ylab = "Time")
```




```r
plot(as.POSIXct(data.2018$TimeStamp), ylab = "Time")
```



Step 4: Cell Division,aggregation and normalization of yield estimates

```
#a function to append yield samples
function1 <- function(mat, Yield, Latitude, Longitude){

  min.latitude  <- 0
  max.latitude  <- max(Latitude)
  lat.range     <- max.latitude-min.latitude
  min.longitude <- 0
  max.longitude <- max(Longitude)
  lon.range     <- max.longitude - min.longitude


  mat$Row  <- ceiling(20*mat$Latitude/lat.range)
  mat$Col  <- ceiling(6*mat$Longitude/lon.range)
  mat$Cell <- (mat$Row*1000 + mat$Col)
  mat$rank <- rank(mat$Yield)
  return(mat)}

data.2013 <- function1(mat=data.2013, Yield = data.2013$Yield, Latitude = dat
a.2013$Latitude, Longitude = data.2013$Longitude)
data.2015 <- function1(mat=data.2015, Yield = data.2015$Yield, Latitude = dat
a.2015$Latitude, Longitude = data.2015$Longitude)
```

```r
data.2016 <- function1(mat=data.2016, Yield = data.2016$Yield, Latitude = dat
a.2016$Latitude, Longitude = data.2016$Longitude)
data.2017 <- function1(mat=data.2017, Yield = data.2017$Yield, Latitude = dat
a.2017$Latitude, Longitude = data.2017$Longitude)
data.2018 <- function1(mat=data.2018, Yield = data.2018$Yield, Latitude = dat
a.2018$Latitude, Longitude = data.2018$Longitude)


#a loop function for grid divisions for all five years
cell.matrix <- function(mat, Yield, Latitude, Longitude){
  # range of latitude
  min.latitude <- 0
  max.latitude <- max(Latitude)
  lat.range <- max.latitude-min.latitude
  min.longitude <- 0
  max.longitude <- max(Longitude)
  lon.range <- max.longitude - min.longitude

  Grid <- data.frame(Divisions=1)
  Grid$MinYield=NA
  Grid$MaxYield=NA
  Grid$Cells=NA
  Grid$mean=NA
  Grid$sd=NA

  for (i in 1:length(Grid$Divisions)){
    required.replicates <- function(cv,percent_diff,alpha=0.05,beta=0.2){
    n <- ceiling(2*(((cv/percent_diff)^2)*(qnorm((1-alpha/2)) + qnorm((1-beta
)))^2))
    y <- (n )
    return(y)}

    j <- i
    mat$Row <- ceiling(20*j*Latitude/lat.range)
    mat$Col <- ceiling(6*j*Longitude/lon.range)
    mat$Cell <- mat$Row*1000 + mat$Col
    yield <- tapply(mat$Cell,mat$Cell,length)
    means <- tapply(mat$Yield,mat$Cell,mean)

    Grid$Cells[i] <- length(means)
    Grid$MinYield[i] <- min(yield)
    Grid$MaxYield[i] <- max(yield)
    Grid$mean[i] <- mean(means)
    Grid$sd[i] <- sd(means)
    Grid$cv[i] <- (100*Grid$sd[i]/Grid$mean[i])
    Grid$RR10 <- (required.replicates(cv=Grid$cv, percent_diff = 10))
    }
```

```
  return(Grid)}
cell.matrix(data.2013,ata.2013$Yield,data.2013$Latitude,data.2013$Longitude)

##   Divisions MinYield MaxYield Cells     mean       sd       cv RR10
## 1         1       67       93   120 40.46977 3.340354 8.253949   11

cell.matrix(data.2015,data.2015$Yield,data.2015$Latitude,data.2015$Longitude)

##   Divisions MinYield MaxYield Cells     mean       sd       cv RR10
## 1         1       87      131   120 35.78163 6.393858 17.86911   51

cell.matrix(data.2016,data.2016$Yield,data.2016$Latitude,data.2016$Longitude)

##   Divisions MinYield MaxYield Cells     mean       sd       cv RR10
## 1         1       58       91   120 117.6105 17.55513 14.92649   35

cell.matrix(data.2017,data.2017$Yield,data.2017$Latitude,data.2017$Longitude)

##   Divisions MinYield MaxYield Cells    mean       sd       cv RR10
## 1         1       68       96   120 58.4982 1.619086 2.767754    2

cell.matrix(data.2018,data.2018$Yield,data.2018$Latitude,data.2018$Longitude)

##   Divisions MinYield MaxYield Cells     mean       sd       cv RR10
## 1         1       86      134   120 242.6754 6.214401 2.560787    2

#aggregation and normalizatin

aggregation <- data.frame(grids=1:120,
                          yield.2013 = tapply(data.2013$Yield,data.2013$Cell,
mean),
                          yield.2015 = tapply(data.2015$Yield,data.2015$Cell,
mean),
                          yield.2016 = tapply(data.2016$Yield,data.2016$Cell,
mean),
                          yield.2017 = tapply(data.2017$Yield,data.2017$Cell,
mean),
                          yield.2018 = tapply(data.2018$Yield,data.2018$Cell,
mean))
head(aggregation)

##      grids yield.2013 yield.2015 yield.2016 yield.2017 yield.2018
## 1001     1   40.78269   26.21633   117.0053   57.68473   254.9570
## 1002     2   44.30403   42.19282   112.6399   58.18460   242.5142
## 1003     3   49.37674   51.13233   127.2392   58.73003   244.7709
## 1004     4   43.24956   46.71617   135.2822   60.05934   237.5016
## 1005     5   41.30299   41.58601   152.2427   57.64959   238.7490
## 1006     6   37.92880   47.29986   134.5051   60.16846   234.8132

RowSD = function(x){sqrt(rowSums((x-rowMeans(x))^2)/(dim(x)[2]-1))}

normalized1 <- data.frame(grids = 1:120,
```

```
                        norm.lat.2013 = tapply(data.2013$Latitude, data.2013
$Cell, mean),
                        norm.long.2013 = tapply(data.2013$Longitude, data.20
13$Cell, mean),
                        norm.yield.2013 = tapply(data.2013$Yield,data.2013$C
ell,mean),
                        norm.sd.2013 = tapply(data.2013$Yield,data.2013$Cell
,sd),

                        norm.lat.2015 = tapply(data.2015$Latitude, data.2015
$Cell, mean),
                        norm.long.2015 = tapply(data.2015$Longitude, data.20
15$Cell, mean),
                        norm.yield.2015 = tapply(data.2015$Yield,data.2015$C
ell,mean),
                        norm.sd.2015 = tapply(data.2015$Yield,data.2015$Cell
,sd),

                        norm.lat.2016 = tapply(data.2016$Latitude, data.2016
$Cell, mean),
                        norm.long.2016 = tapply(data.2016$Longitude, data.20
16$Cell, mean),
                        norm.yield.2016 = tapply(data.2016$Yield,data.2016$C
ell,mean),
                        norm.sd.2016 = tapply(data.2016$Yield,data.2016$Cell
,sd),

                        norm.lat.2017 = tapply(data.2017$Latitude, data.2017
$Cell, mean),
                        norm.long.2017 = tapply(data.2017$Longitude, data.20
17$Cell, mean),
                        norm.yield.2017 = tapply(data.2017$Yield,data.2017$C
ell,mean),
                        norm.sd.2017 = tapply(data.2017$Yield,data.2017$Cell
,sd),

                        norm.lat.2018 = tapply(data.2018$Latitude, data.2018
$Cell, mean),
                        norm.long.2018 = tapply(data.2018$Longitude, data.20
18$Cell, mean),
                        norm.yield.2018 = tapply(data.2018$Yield,data.2018$C
ell,mean),
                        norm.sd.2018 = tapply(data.2018$Yield,data.2018$Cell
,sd),

                        grand.mean = rowMeans(aggregation[,-1]),
                        SD = RowSD(aggregation[2:6]))
head(normalized1)
```

```
##      grids norm.lat.2013 norm.long.2013 norm.yield.2013 norm.sd.2013
## 1001     1      9.932726       47.01996        40.78269     3.493918
## 1002     2     10.286210      149.41001        44.30403     6.154753
## 1003     3      9.750030      251.94380        49.37674     8.346921
## 1004     4     10.026206      355.13635        43.24956     7.103325
## 1005     5      9.966944      448.33362        41.30299    11.859931
## 1006     6     10.111855      541.58970        37.92880    11.839294
##      norm.lat.2015 norm.long.2015 norm.yield.2015 norm.sd.2015
## 1001      9.951777       48.90171        26.21633     4.567857
## 1002      9.668017      147.84048        42.19282     8.804372
## 1003     10.032981      251.07624        51.13233    14.848197
## 1004     10.283762      350.63241        46.71617    11.972626
## 1005     10.088728      449.58727        41.58601    13.797646
## 1006      9.960026      551.83197        47.29986    10.233290
##      norm.lat.2016 norm.long.2016 norm.yield.2016 norm.sd.2016
## 1001      9.999348       46.67186        117.0053     13.16346
## 1002     10.026689      147.97435        112.6399     31.44776
## 1003      9.747628      242.70399        127.2392     37.81029
## 1004      9.936912      352.16938        135.2822     19.29458
## 1005     10.310580      449.67783        152.2427     34.80214
## 1006     10.349515      549.22244        134.5051     32.70149
##      norm.lat.2017 norm.long.2017 norm.yield.2017 norm.sd.2017
## 1001      9.942865       52.65224        57.68473     3.150681
## 1002     10.035862      153.36579        58.18460     2.867638
## 1003     10.020572      253.72792        58.73003     5.537253
## 1004      9.825151      347.79622        60.05934     3.489122
## 1005      9.707292      446.19098        57.64959    12.162859
## 1006     10.028260      545.69270        60.16846     6.900371
##      norm.lat.2018 norm.long.2018 norm.yield.2018 norm.sd.2018 grand.mean
## 1001      9.970083       48.56268        254.9570     15.84413   99.32921
## 1002      9.976924      149.67395        242.5142     20.63998   99.96710
## 1003      9.933193      249.54600        244.7709     14.60349  106.24983
## 1004     10.043223      351.67328        237.5016     17.12370  104.56177
## 1005     10.135355      451.94759        238.7490     38.05492  106.30607
## 1006     10.121530      553.58452        234.8132     15.14344  102.94307
##            SD
## 1001 93.59474
## 1002 84.64970
## 1003 83.90358
## 1004 83.22036
## 1005 87.22506
## 1006 82.97375
```

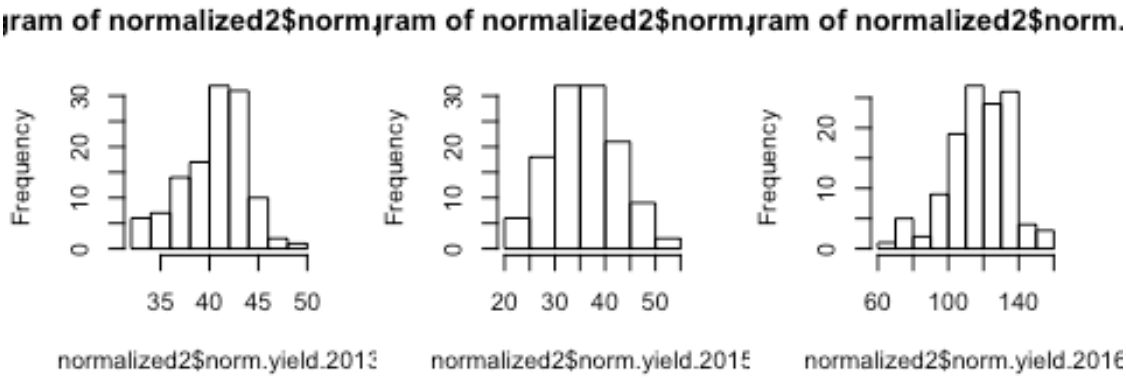
Step 5: Ranking the merged data

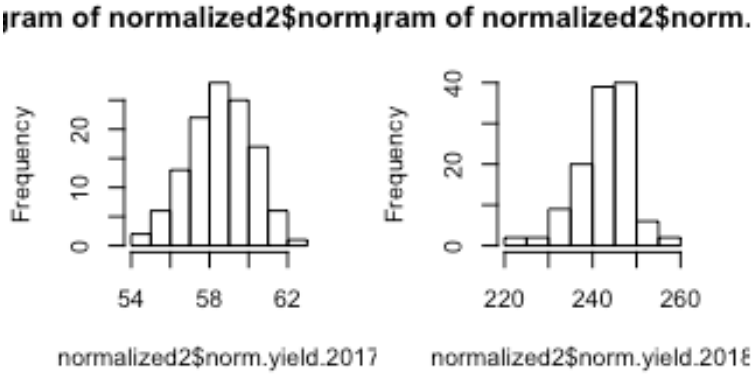
```
normalized2 <- data.frame(grids = 1:120,
                          norm.lat.2013 = tapply(data.2013$Latitude, data.2013
$Cell, mean),
                          norm.long.2013 = tapply(data.2013$Longitude, data.20
13$Cell, mean),
                          norm.yield.2013 = tapply(data.2013$Yield,data.2013$C
```

```
ell,mean),
                        norm.sd.2013 = tapply(data.2013$Yield,data.2013$Cell
,sd),
                        rank.2013 = rank(aggregation$yield.2013),

                        norm.lat.2015 = tapply(data.2015$Latitude, data.2015
$Cell, mean),
                        norm.long.2015 = tapply(data.2015$Longitude, data.20
15$Cell, mean),
                        norm.yield.2015 = tapply(data.2015$Yield,data.2015$C
ell,mean),
                        norm.sd.2015 = tapply(data.2015$Yield,data.2015$Cell
,sd),
                        rank.2015 = rank(aggregation$yield.2015),

                        norm.lat.2016 = tapply(data.2016$Latitude, data.2016
$Cell, mean),
                        norm.long.2016 = tapply(data.2016$Longitude, data.20
16$Cell, mean),
                        norm.yield.2016 = tapply(data.2016$Yield,data.2016$C
ell,mean),
                        norm.sd.2016 = tapply(data.2016$Yield,data.2016$Cell
,sd),
                        rank.2016 = rank(aggregation$yield.2016),

                        norm.lat.2017 = tapply(data.2017$Latitude, data.2017
$Cell, mean),
                        norm.long.2017 = tapply(data.2017$Longitude, data.20
17$Cell, mean),
                        norm.yield.2017 = tapply(data.2017$Yield,data.2017$C
ell,mean),
                        norm.sd.2017 = tapply(data.2017$Yield,data.2017$Cell
,sd),
                        rank.2017 = rank(aggregation$yield.2017),

                        norm.lat.2018 = tapply(data.2018$Latitude, data.2018
$Cell, mean),
                        norm.long.2018 = tapply(data.2018$Longitude, data.20
18$Cell, mean),
                        norm.yield.2018 = tapply(data.2018$Yield,data.2018$C
ell,mean),
                        norm.sd.2018 = tapply(data.2018$Yield,data.2018$Cell
,sd),
                        rank.2018 = rank(aggregation$yield.2018),

                        grand.mean = rowMeans(aggregation[,-1]),
                        SD = RowSD(aggregation[2:6]),
                        Ranking = rank(normalized1$grand.mean))
head(normalized2)
```

```
##      grids norm.lat.2013 norm.long.2013 norm.yield.2013 norm.sd.2013
## 1001     1      9.932726       47.01996        40.78269     3.493918
## 1002     2     10.286210      149.41001        44.30403     6.154753
## 1003     3      9.750030      251.94380        49.37674     8.346921
## 1004     4     10.026206      355.13635        43.24956     7.103325
## 1005     5      9.966944      448.33362        41.30299    11.859931
## 1006     6     10.111855      541.58970        37.92880    11.839294
##      rank.2013 norm.lat.2015 norm.long.2015 norm.yield.2015 norm.sd.2015
## 1001        55      9.951777       48.90171        26.21633     4.567857
## 1002       109      9.668017      147.84048        42.19282     8.804372
## 1003       120     10.032981      251.07624        51.13233    14.848197
## 1004       100     10.283762      350.63241        46.71617    11.972626
## 1005        64     10.088728      449.58727        41.58601    13.797646
## 1006        27      9.960026      551.83197        47.29986    10.233290
##      rank.2015 norm.lat.2016 norm.long.2016 norm.yield.2016 norm.sd.2016
## 1001         8      9.999348       46.67186        117.0053     13.16346
## 1002       102     10.026689      147.97435        112.6399     31.44776
## 1003       119      9.747628      242.70399        127.2392     37.81029
## 1004       113      9.936912      352.16938        135.2822     19.29458
## 1005        96     10.310580      449.67783        152.2427     34.80214
## 1006       115     10.349515      549.22244        134.5051     32.70149
##      rank.2016 norm.lat.2017 norm.long.2017 norm.yield.2017 norm.sd.2017
## 1001        60      9.942865       52.65224        57.68473     3.150681
## 1002        48     10.035862      153.36579        58.18460     2.867638
## 1003        79     10.020572      253.72792        58.73003     5.537253
## 1004       103      9.825151      347.79622        60.05934     3.489122
## 1005       118      9.707292      446.19098        57.64959    12.162859
## 1006        97     10.028260      545.69270        60.16846     6.900371
##      rank.2017 norm.lat.2018 norm.long.2018 norm.yield.2018 norm.sd.2018
## 1001        39      9.970083       48.56268        254.9570     15.84413
## 1002        48      9.976924      149.67395        242.5142     20.63998
## 1003        64      9.933193      249.54600        244.7709     14.60349
## 1004        98     10.043223      351.67328        237.5016     17.12370
## 1005        36     10.135355      451.94759        238.7490     38.05492
## 1006       102     10.121530      553.58452        234.8132     15.14344
##      rank.2018 grand.mean       SD Ranking
## 1001       118   99.32921 93.59474      57
## 1002        57   99.96710 84.64970      60
## 1003        70  106.24983 83.90358     114
## 1004        25  104.56177 83.22036     104
## 1005        32  106.30607 87.22506     116
## 1006        13  102.94307 82.97375      87
```

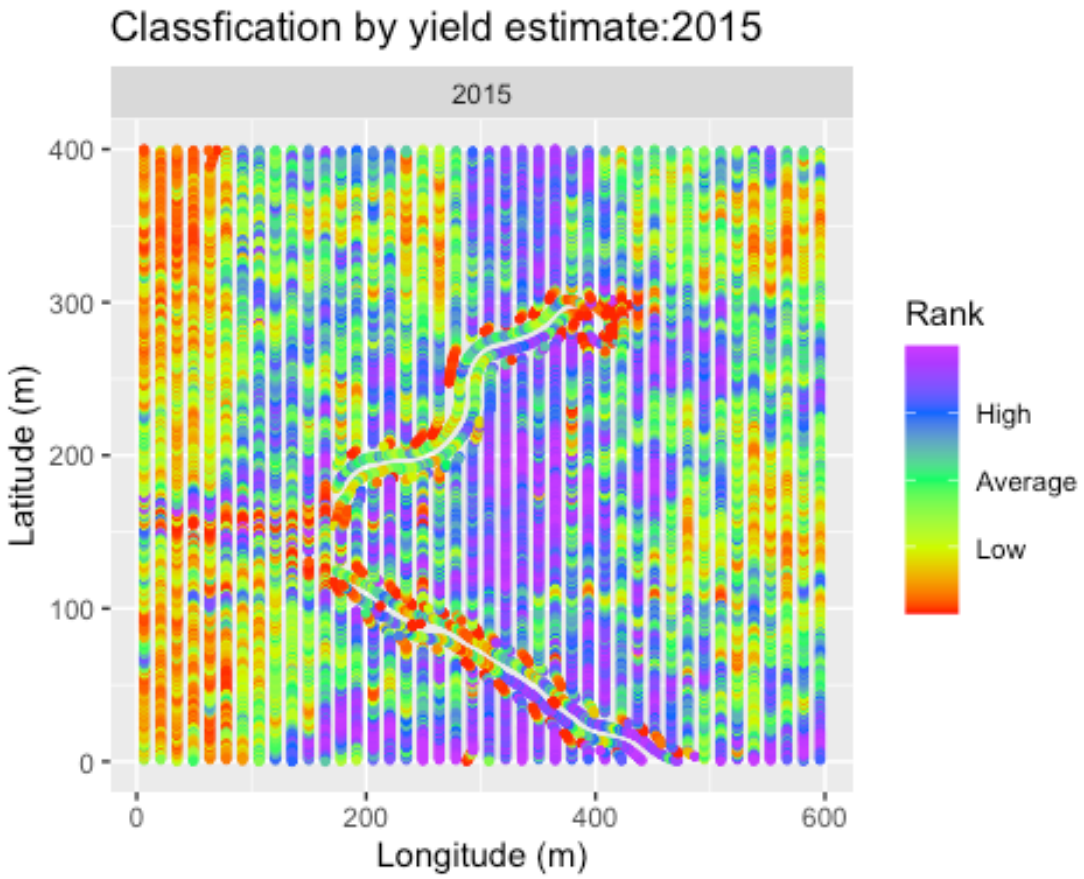
```r
par(mfrow=c(2,3))
hist(normalized2$norm.yield.2013)
hist(normalized2$norm.yield.2015)
hist(normalized2$norm.yield.2016)
hist(normalized2$norm.yield.2017)
hist(normalized2$norm.yield.2018)
```



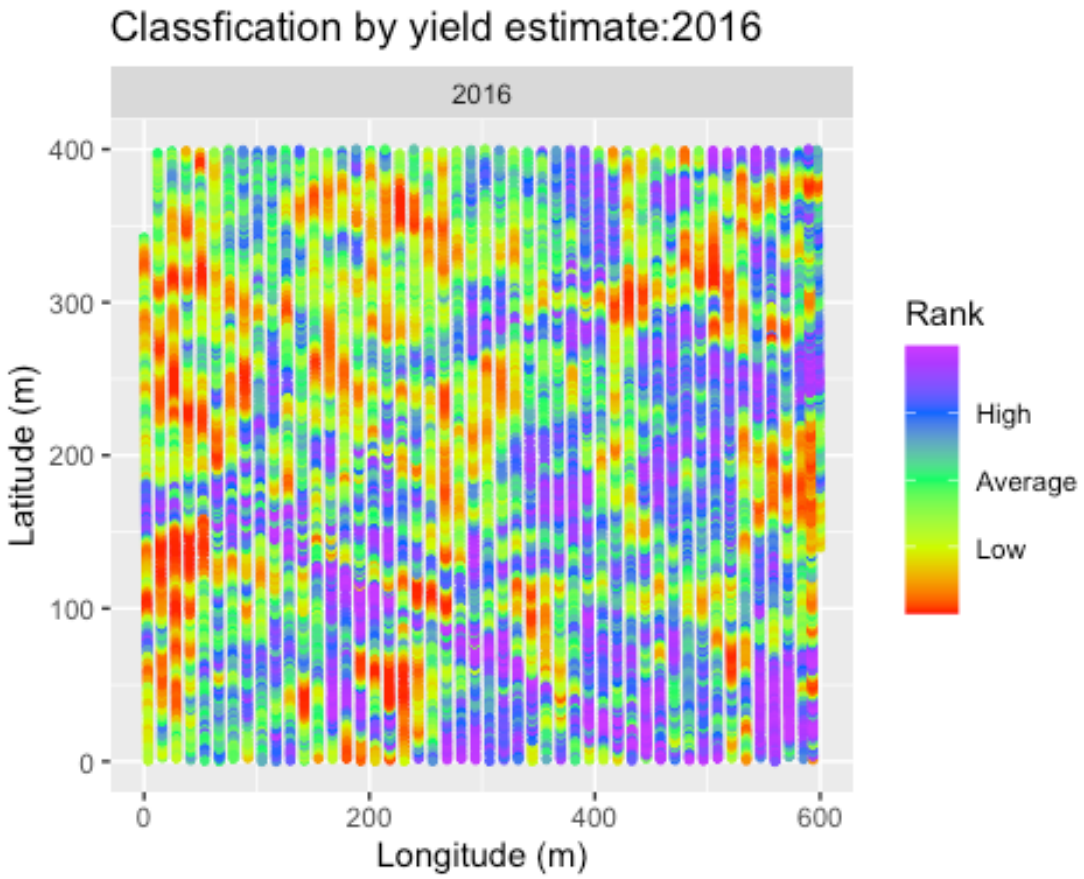


Step 6: Classifications and plotting grid cells according to the normalized mean and standard deviation criteria: a. if the normalized mean/standardard deviation is in the top 25th percentile then classify it as high/unstable yield b. if the normalized mean/standard deviation is in the bottom 25th percentile then classify it as low/stable yield c. if the normalized mean/standard deviation is in between a and b then classify it as average yield

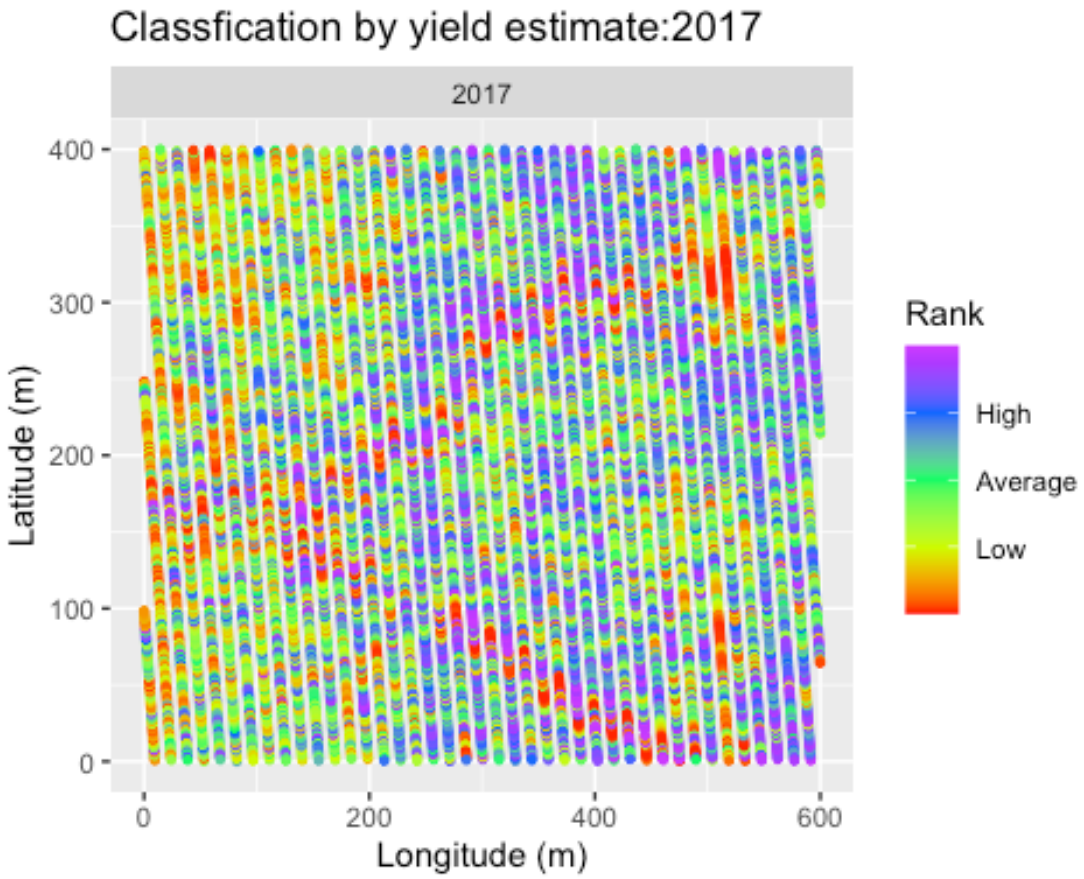
```
#classifications by normalized means
library(ggplot2)

#2015
ggplot(data = data.2015, mapping = aes(x = Longitude, y = Latitude))+
geom_point(aes(color = rank), size = 0.9)+
scale_colour_gradientn(colours = rainbow(5), breaks = c(2898,5796,8694), labe
ls = c("Low", "Average", "High"))+
labs(color = "Rank", x = "Longitude (m)", y = "Latitude (m)") + facet_wrap(~
2015) + ggtitle("Classfication by yield estimate:2015")
```



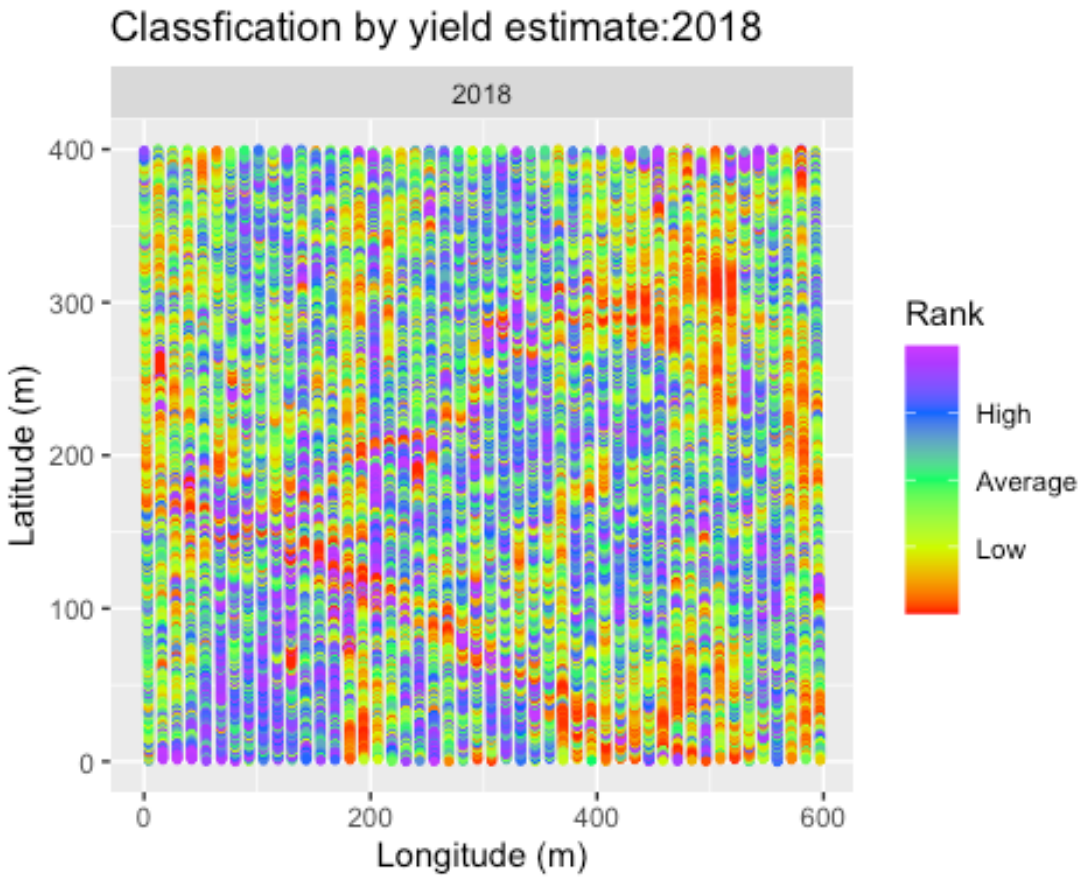

```
#2016
ggplot(data = data.2016, mapping = aes(x = Longitude, y = Latitude))+
geom_point(aes(color = rank), size = 0.9)+
scale_colour_gradientn(colours = rainbow(5), breaks = c(2104,4207,6310), labe
ls = c("Low", "Average", "High"))+
labs(color = "Rank", x = "Longitude (m)", y = "Latitude (m)") + facet_wrap(~
2016) + ggtitle("Classfication by yield estimate:2016")
```



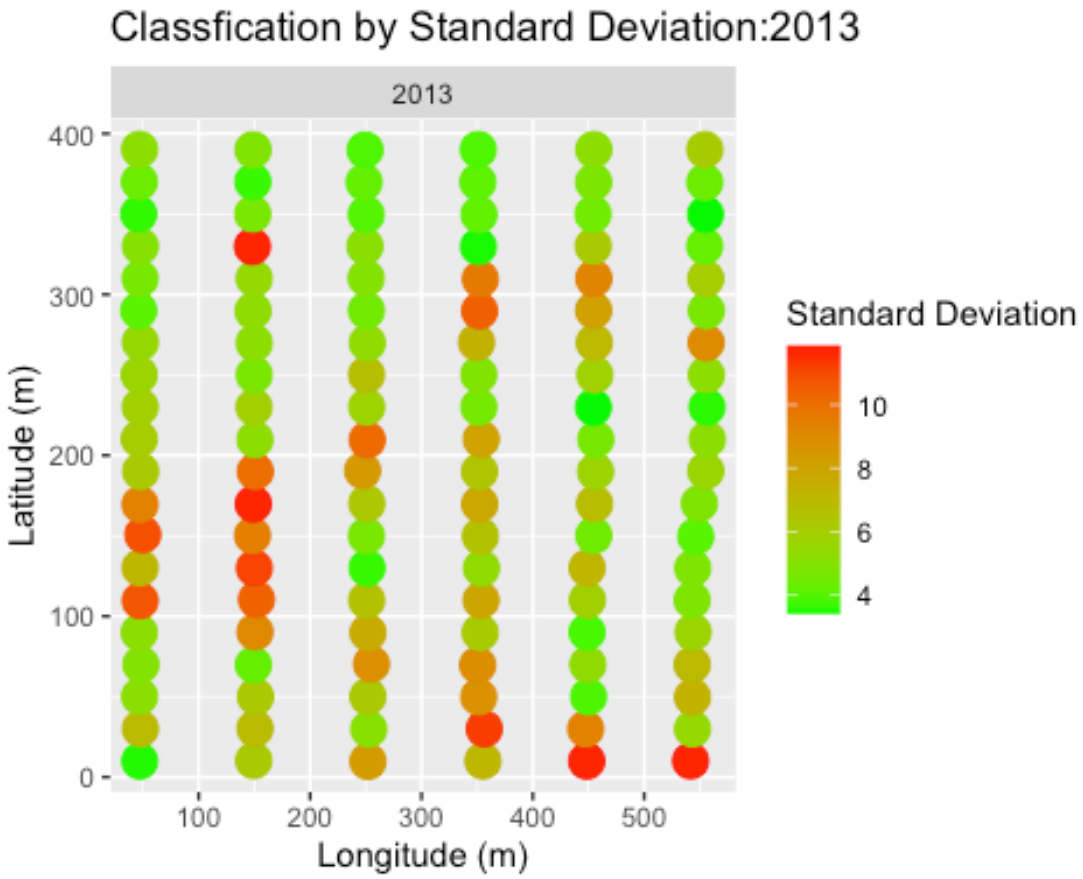

```
#2017
ggplot(data = data.2017, mapping = aes(x = Longitude, y = Latitude))+
geom_point(aes(color = rank), size = 0.9)+
scale_colour_gradientn(colours = rainbow(5), breaks = c(2396 ,4789,7184), lab
els = c("Low", "Average", "High"))+
labs(color = "Rank", x = "Longitude (m)", y = "Latitude (m)") + facet_wrap(~
2017) + ggtitle("Classfication by yield estimate:2017")
```



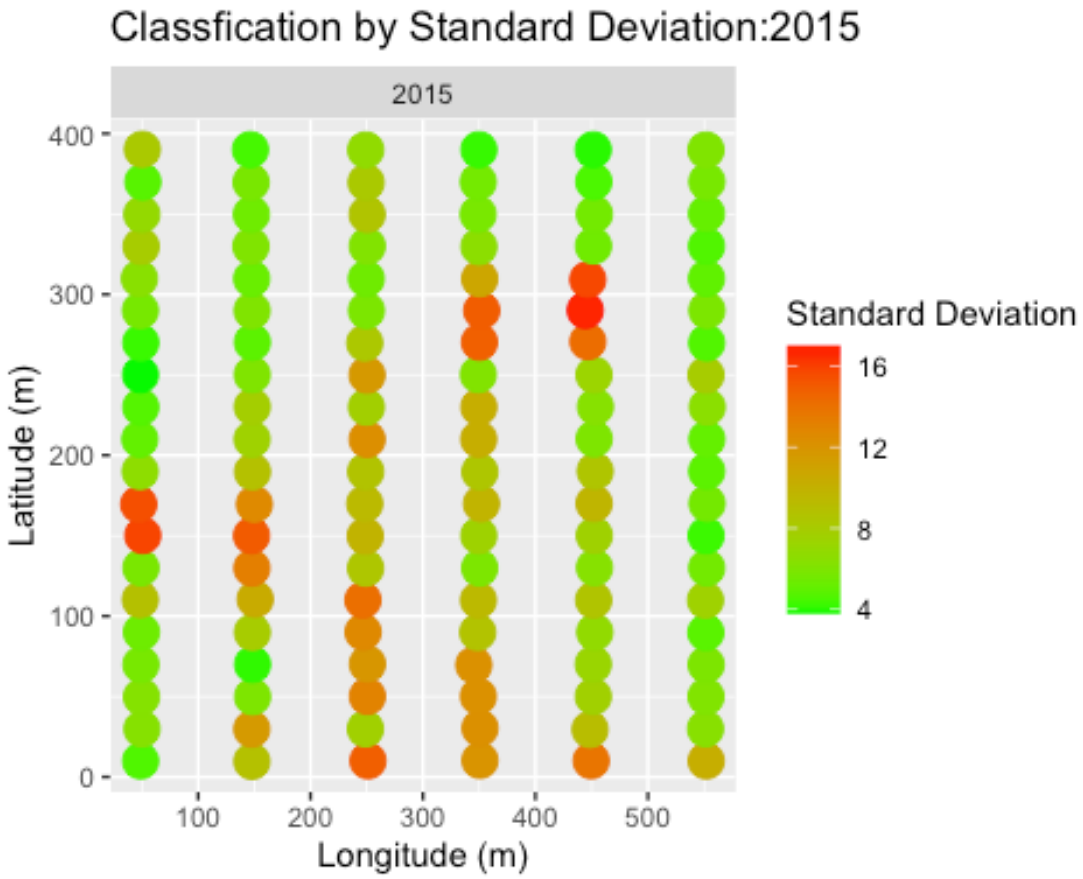

```
#2018
ggplot(data = data.2018, mapping = aes(x = Longitude, y = Latitude))+
geom_point(aes(color = rank), size = 0.9)+
scale_colour_gradientn(colours = rainbow(5), breaks = c(2796,5592,8388), labe
ls = c("Low", "Average", "High"))+
labs(color = "Rank", x = "Longitude (m)", y = "Latitude (m)") + facet_wrap(~
2018) + ggtitle("Classfication by yield estimate:2018")
```



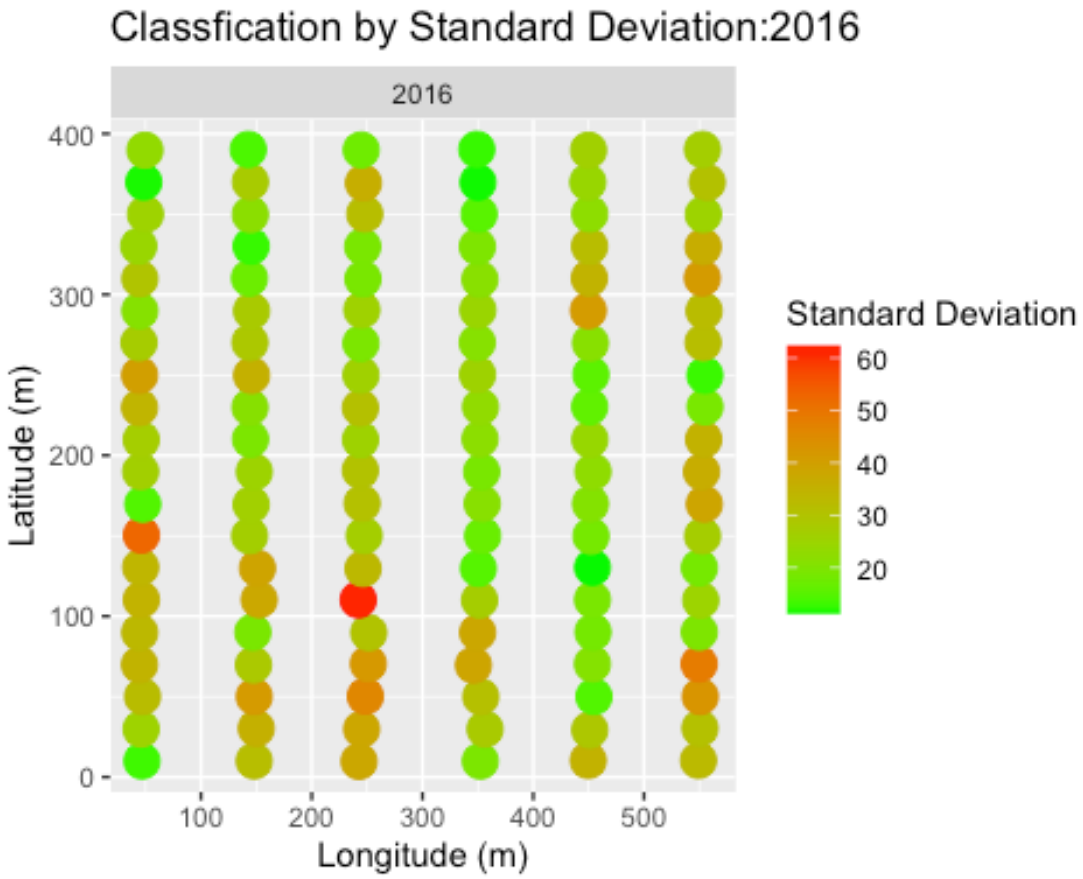

```
#Classifications by standard deviation of normalized means
#2013
ggplot(data = normalized2, mapping = aes(x = norm.long.2013, y = norm.lat.201
3))+
geom_point(aes(color = norm.sd.2013), size = 5)+
scale_colour_gradient(low = "green", high = "red") +
labs(color = "Standard Deviation", x = "Longitude (m)", y = "Latitude (m)") +
facet_wrap(~ 2013) + ggtitle("Classfication by Standard Deviation:2013")
```



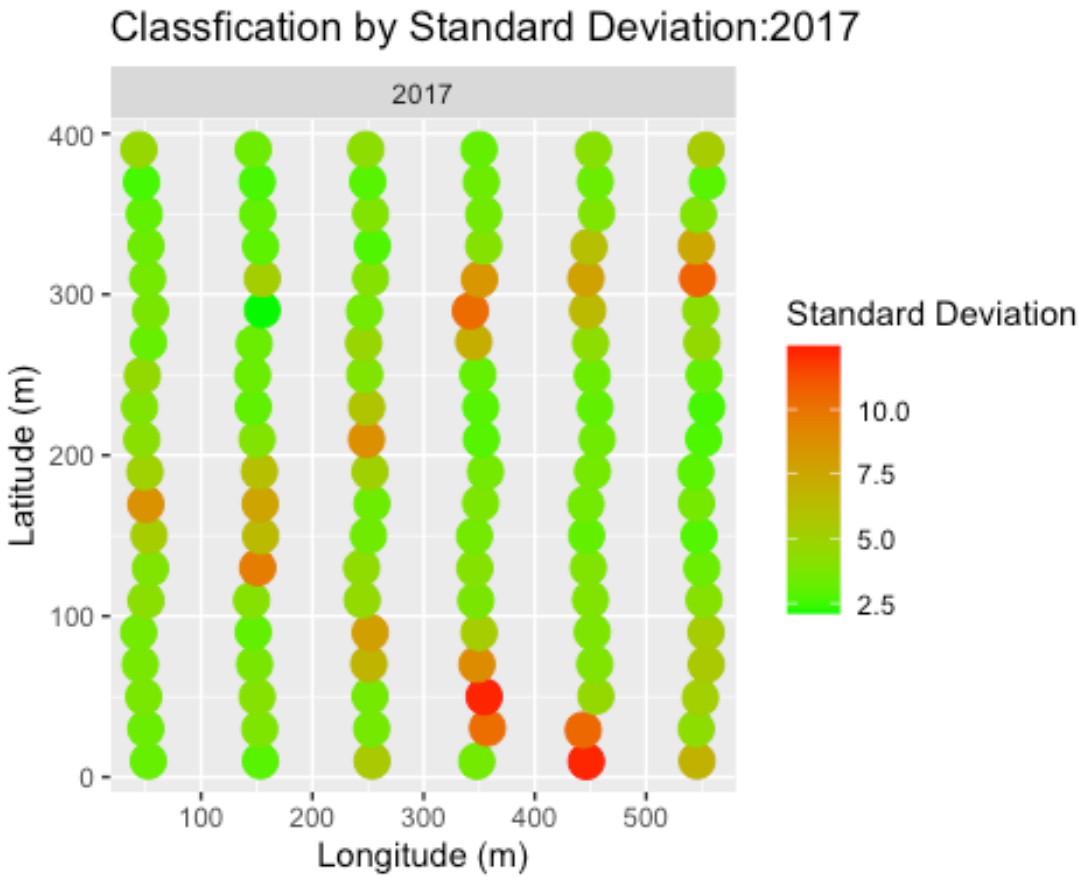

```
#2015
ggplot(data = normalized2, mapping = aes(x = norm.long.2015, y = norm.lat.201
5))+
geom_point(aes(color = norm.sd.2015), size = 5)+
scale_colour_gradient(low = "green", high = "red") +
labs(color = "Standard Deviation", x = "Longitude (m)", y = "Latitude (m)") +
facet_wrap(~ 2015) + ggtitle("Classfication by Standard Deviation:2015")
```



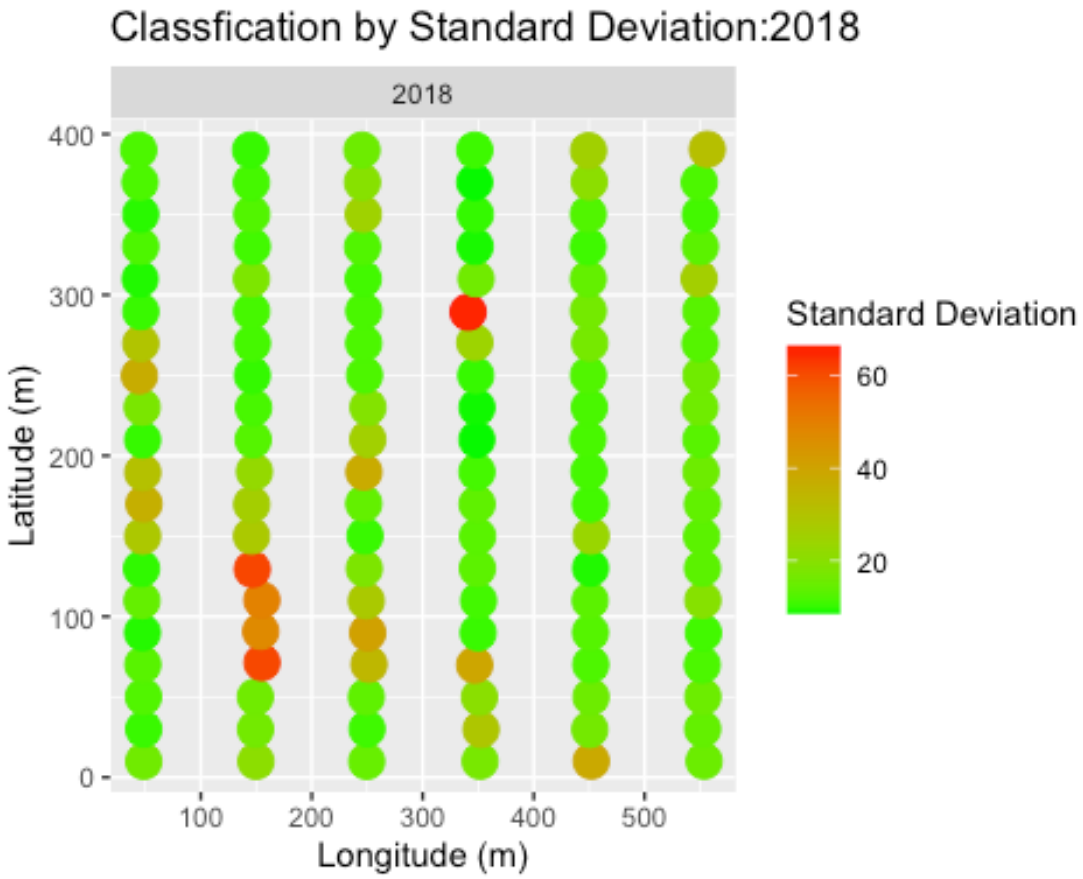

```
#2016
ggplot(data = normalized2, mapping = aes(x = norm.long.2016, y = norm.lat.201
6))+
geom_point(aes(color = norm.sd.2016), size = 5)+
scale_colour_gradient(low = "green", high = "red") +
labs(color = "Standard Deviation", x = "Longitude (m)", y = "Latitude (m)") +
facet_wrap(~ 2016) + ggtitle("Classfication by Standard Deviation:2016")
```




```r
#2017
ggplot(data = normalized2, mapping = aes(x = norm.long.2017, y = norm.lat.201
7))+
geom_point(aes(color = norm.sd.2017), size = 5)+
scale_colour_gradient(low = "green", high = "red") +
labs(color = "Standard Deviation", x = "Longitude (m)", y = "Latitude (m)") +
facet_wrap(~ 2017) + ggtitle("Classfication by Standard Deviation:2017")
```




```
#2018
ggplot(data = normalized2, mapping = aes(x = norm.long.2018, y = norm.lat.201
8))+
geom_point(aes(color = norm.sd.2018), size = 5)+
scale_colour_gradient(low = "green", high = "red") +
labs(color = "Standard Deviation", x = "Longitude (m)", y = "Latitude (m)") +
facet_wrap(~ 2018) + ggtitle("Classfication by Standard Deviation:2018")
```



Discussion:

Starting with the text processing, upon quickly converting the datetime instances and plotting it, it became evident that all five datasets were collected in less than seven days.We then proceeded by dividing the datasets into grid cells which then allowed us to compute yield estimates for each cell. Then, we aggreggated the yield samples for all five years. We creatted a series of histograms for all five datasets to get an understanding of how spread out the yield estimates were, which they were moderately spreadout. In order to overcome this spread, we normalized all five datasets and merged them together. After normalizing and merging them, we were able to classify each individual as high,average, or low yields and conversely, we were able to classify the standard deviation in each grid cell as stable, average, unstable.

Based on the rank analysis, the left part of the field, in particular, the cells between 0 to 100m longitude, seem to be under-performing in terms of harvest mean estimates. The standard deviation plot for those cells indicidate that there are quiet a few cells with significantly high standard deviation. Ignoring the random effects factor which might bloat our standard deviation values, I think it might be well advised to try different farming techniques to improve the harvest yields of that part of the field.