It is commonplace for agricultural data to contain spatial information typically the longitude and latitude at which each data point was collected. However, one of the challenges this introduces is that different datasets often do not contain data that were collected at the same locations. A specific example involves planting and harvest data from precision farming equipment. As a planter moves through a field, it records thousands of measurements of its location along with various data about the planting operation (such as the variety of seed planted, the seeding rate, and the spacing between seeds). Similarly, as a harvester moves through a field, it records thousands of measurements of its location and the crop’s yield at that point. A question of agricultural interest is to estimate the relationship between yield and the variables measured during planting. However, the harvest data points do not necessarily exactly overlap the planting data points, so before any analysis can be conducted to probe these relationships, it’s necessary to determine which planting points are associated with which harvest points.

The planting and harvest files for one corn field are attached (both files are from the same year). Both tables contain a row for each data point collected by the machine. The columns in each file are as follows:

1. planting\_sample\_data.csv

1. long: longitude where data point was collected.
2. lat: latitude where data point was collected.
3. variety: the seed variety planted at that location.
4. seeding\_rate: continuous variable specifying the number of seeds planted per acre (in thousands).
5. seed\_spacing: continuous variable specifying the distance between seeds (inches).

2. harvest\_sample\_data.csv

1. long: longitude where data point was collected
2. lat: latitude where data point was collected
3. yield: continuous variable specifying the yield of the crop (in bushels/acre).

Your task is twofold:

1. Write a function that takes two arguments: the planting filename and the harvest filename. The function should read in the files from your local filesystem, and then determine the values of the planting variables (variety, seeding rate, and seed spacing) that should be associated with each harvest point. Please design, describe, and implement an algorithm that performs this association. The function should return a data frame containing the same number of rows as the harvest file, but with three extra columns, containing the values of variety, seeding rate, and seed spacing associated with each harvest point.
2. Use the output of the function you wrote in step 1 to perform some exploratory data analysis on the data provided. The goal is to quantify how the three planting variables (variety, seeding rate, and seed spacing) are associated with yield. This exploratory analysis should involve some data visualization and some implementation of statistical models.

Please provide the output of the function in step 1 as a .csv file, any plots or relevant data summaries from step 2, a written summary of your findings (describe your plant harvest point association algorithm, its efficiency, and your data analysis/modeling approach from step 2), and all code necessary to recreate this analysis.

The code below has been scripted using ipython. Please ensure pointers point to correct file location.

#####################################PART1###############################

import pandas as pd

import numpy as np

import time

from math import radians, sin, cos, asin, sqrt, pi, atan2

earth\_radius\_km = 6371 # Radius of the Earth in kilometers

# Importing Files (Please ensure pointers point to correct file locations)

planting\_input = 'planting\_sample\_data.csv'

harvest\_input = 'harvest\_sample\_data.csv'

# Association Function

def association(planting,harvest):

planting = pd.read\_csv(planting)

harvest = pd.read\_csv(harvest)

# Haversine Formula

def haversine(point1, point2):

dlat = np.radians(point2[:,0]) - radians(point1[0])

dlon = np.radians(point2[:,1]) - radians(point1[1])

a = np.square(np.sin(dlat/2.0)) + cos(radians(point1[0])) \* np.cos(np.radians(point2[:,0])) \* np.square(np.sin(dlon/2.0))

great\_circle\_distance = 2 \* np.arcsin(np.minimum(np.sqrt(a), np.repeat(1, len(a))))

d = earth\_radius\_km \* great\_circle\_distance

return np.argmin(d)

plan = np.column\_stack((np.array(planting.lat),np.array(planting.long)))

harv = np.column\_stack((np.array(harvest.lat),np.array(harvest.long)))

# Finding Nearest Neigbour

output = pd.DataFrame([])

for i in range(len(harv)):

clo = haversine(harv[i],plan)

tmp = pd.DataFrame([[planting.variety[clo],planting.seeding\_rate[clo],planting.seed\_spacing[clo]]],columns=list(['variety','seeding\_rate','seed\_spacing']))

output = output.append(tmp,ignore\_index=True)

# Concatenate Result to Harvest Dataframe

final = pd.concat([harvest, output], axis=1)

# Export Dataframe to csv

final.to\_csv('Harvest\_Cleaned.csv', sep=',')

# Call Function & Time

start = time.clock()

association(planting\_input,harvest\_input)

elapsed = time.clock()

elapsed = elapsed - start

print 'Total time elapsed =', elapsed, 'seconds'

**Methodology:**

To determine the values of the planting data variables that should be associated with each harvest data point I have utilized the ‘Haversine Formula’ which determines the great-circle distance (the shortest distance between two points on the surface of a sphere, measured along the surface of the sphere, as opposed to a straight line through the sphere's interior) between two points on a sphere given their longitudes and latitudes.

I have calculated the distance between every pair of points (harvest & planting co-ordinates) and found the closest point (planting data) lying next to the harvest coordinate data. I have then associated that planting point’s data with the harvest point data. Repeating this process I have associated every harvest data point with the nearest planting data point and concatenated their data. This method provides for a sufficiently accurate means of measuring distances while taking the curvature of the earth into account.

The formulas are given below:

For any two points on a sphere, the haversine of the central angle between them is given by



* hav is the haversine function:

{\displaystyle \operatorname {hav} (\theta )=\sin ^{2}\left({\frac {\theta }{2}}\right)={\frac {1-\cos(\theta )}{2}}}

* *d* is the distance between the two points (along a [great](https://en.wikipedia.org/wiki/Great_circle) circle of the sphere),
* *r* is the radius of the sphere,
* *φ*1, *φ*2: latitude of point 1 and latitude of point 2, in radians
* *λ*1, *λ*2: longitude of point 1 and longitude of point 2, in radians

{\displaystyle d=r\operatorname {hav} ^{-1}(h)=2r\arcsin \left({\sqrt {h}}\right)}

More information can be found [here](https://en.wikipedia.org/wiki/Haversine_formula)

I have used numpy and pandas packages along with several others. I have also referred to stackoverflow as and when required.

**Process:**

Step 1: Import data from local system

Step 2: Write function to calculate haversine formula and calculate planting data point closest to harvesting data point

Step 3: Using index of planting data point, create a dataframe with corresponding data from planting dataframe, namely, variety, seeding rate, and seed spacing

Step 4: Once the dataframe is created concatenate it with the harvest dataframe by index

Step 5: We now have the required dataframe which is then exported to the local machines in csv format.

##############################PART2#########################

import matplotlib.pyplot as plt

from scipy import stats

from scipy.stats import linregress

final = pd.read\_csv('Harvest\_Cleaned.csv')

#Sort by variety

var1 = final[final['variety'] == 'DKC63-33RIB']

var2 = final[final['variety'] == 'P1498']

# Variety count

% matplotlib

var\_count = final['variety'].value\_counts()

var\_count.plot(kind='barh', stacked=True)

plt.title('Variety Used Per Harvest Point')

plt.ylabel('Type of Variety')

plt.xlabel('Variety Used Count')

plt.grid(True)

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

byvar = final.groupby('variety')

desc = byvar['yield'].describe()

plt.text(11500, 0.40, desc, fontsize=12)

byvar = final.groupby('variety')

desc1 = byvar['seed\_spacing'].describe()

plt.text(7000, 0.40, desc1, fontsize=12)

byvar = final.groupby('variety')

desc2 = byvar['seeding\_rate'].describe()

plt.text(2500, 0.40, desc2, fontsize=12)

# Box Plot Yield By Type

fig1 = plt.figure()

ax1 = fig1.add\_subplot(121)

plt.title('DKC63-33RIB')

var1.boxplot('yield')

plt.xlabel('Type of Variety')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

ax2 = fig1.add\_subplot(122)

plt.title('P1498')

var2.boxplot('yield')

plt.xlabel('Type of Variety')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

# Historgrams

fig1 = plt.figure()

ax1 = fig1.add\_subplot(121)

plt.title('DKC63-33RIB')

var1['yield'].hist(bins=50)

plt.ylabel('Frequency/Number of Harvest Points')

plt.xlabel('Yield (bushels/acre)')

plt.grid(True)

ax2 = fig1.add\_subplot(122)

plt.title('P1498')

var2['yield'].hist(bins=50)

plt.ylabel('Frequency/Number of Harvest Points')

plt.xlabel('Yield (bushels/acre)')

plt.grid(True)

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

# Yield vs Seeding Rate

fig2 = plt.figure()

ax1 = fig2.add\_subplot(121)

x1 = var1.ix[:,5]

y1 = var1.ix[:,3]

fit = np.polyfit(x1,y1,1)

fit\_fn = np.poly1d(fit)

plt.plot(x1,y1, 'ro', x1, fit\_fn(x1), '--k')

plt.xlim(0, 50)

plt.title('DKC63-33RIB')

plt.xlabel('Seeds Planted per Acre (in thousands)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x1,y1)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 380, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

ax1 = fig2.add\_subplot(122)

x2 = var2.ix[:,5]

y2 = var2.ix[:,3]

fit = np.polyfit(x2,y2,1)

fit\_fn = np.poly1d(fit)

plt.plot(x2,y2, 'go', x2, fit\_fn(x2), '--k')

plt.xlim(0, 50)

plt.title('P1498')

plt.xlabel('Seeds Planted per Acre (in thousands)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x2,y2)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 380, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

# Yield vs Seeding Rate Without Outliers Within 3 Standard Deviations

mean = var1['seeding\_rate'].mean()

std = var1['seeding\_rate'].std()

var3 = var1[var1['seeding\_rate'] < mean + 3 \* std]

var3 = var3[var3['seeding\_rate'] > mean - 3 \* std]

mean = var1['yield'].mean()

std = var1['yield'].std()

var3 = var1[var1['yield'] < mean + 3 \* std]

var3 = var3[var3['yield'] > mean - 3 \* std]

mean = var2['seeding\_rate'].mean()

std = var2['seeding\_rate'].std()

var4 = var2[var2['seeding\_rate'] < mean + 3 \* std]

var4 = var4[var4['seeding\_rate'] > mean - 3 \* std]

mean = var2['yield'].mean()

std = var2['yield'].std()

var4 = var2[var2['yield'] < mean + 3 \* std]

var4 = var4[var4['yield'] > mean - 3 \* std]

fig2 = plt.figure()

ax1 = fig2.add\_subplot(121)

x1 = var3.ix[:,5]

y1 = var3.ix[:,3]

fit = np.polyfit(x1,y1,1)

fit\_fn = np.poly1d(fit)

plt.plot(x1,y1, 'ro', x1, fit\_fn(x1), '--k')

plt.xlim(20, 50)

plt.ylim(100, 350)

plt.title('DKC63-33RIB (Without Outliers)')

plt.xlabel('Seeds Planted per Acre (in thousands)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x1,y1)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 345, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

ax1 = fig2.add\_subplot(122)

x2 = var4.ix[:,5]

y2 = var4.ix[:,3]

fit = np.polyfit(x2,y2,1)

fit\_fn = np.poly1d(fit)

plt.plot(x2,y2, 'go', x2, fit\_fn(x2), '--k')

plt.xlim(20, 50)

plt.ylim(100, 350)

plt.title('P1498 (Without Outliers)')

plt.xlabel('Seeds Planted per Acre (in thousands)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x2,y2)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 345, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

# Yield vs Seed Spacing

fig3 = plt.figure()

ax1 = fig3.add\_subplot(121)

x1 = var1.ix[:,6]

y1 = var1.ix[:,3]

fit = np.polyfit(x1,y1,1)

fit\_fn = np.poly1d(fit)

plt.plot(x1,y1, 'ro', x1, fit\_fn(x1), '--k')

plt.xlim(0, 50)

plt.title('DKC63-33RIB')

plt.xlabel('distance between seeds (inches)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x1,y1)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 380, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

ax1 = fig3.add\_subplot(122)

x2 = var2.ix[:,6]

y2 = var2.ix[:,3]

fit = np.polyfit(x2,y2,1)

fit\_fn = np.poly1d(fit)

plt.plot(x2,y2, 'go', x2, fit\_fn(x2), '--k')

plt.xlim(0, 50)

plt.title('P1498')

plt.xlabel(' distance between seeds (inches)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x2,y2)

tmp = 'correlation coefficient:', r\_value

plt.text(50, 380, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

figManager = plt.get\_current\_fig\_manager()

figManager.window.showMaximized()

# Yield vs Seed Spacing Without Outliers Within 3 Standard Deviations

mean = var1['seed\_spacing'].mean()

std = var1['seed\_spacing'].std()

var5 = var1[var1['seed\_spacing'] < mean + 3 \* std]

var5 = var5[var5['seed\_spacing'] > mean - 3 \* std]

mean = var1['yield'].mean()

std = var1['yield'].std()

var5 = var1[var1['yield'] < mean + 3 \* std]

var5 = var5[var5['yield'] > mean - 3 \* std]

mean = var2['seed\_spacing'].mean()

std = var2['seed\_spacing'].std()

var6 = var2[var2['seed\_spacing'] < mean + 3 \* std]

var6 = var6[var6['seed\_spacing'] > mean - 3 \* std]

mean = var2['yield'].mean()

std = var2['yield'].std()

var6 = var2[var2['yield'] < mean + 3 \* std]

var6 = var6[var6['yield'] > mean - 3 \* std]

fig3 = plt.figure()

ax1 = fig3.add\_subplot(121)

x1 = var5.ix[:,6]

y1 = var5.ix[:,3]

fit = np.polyfit(x1,y1,1)

fit\_fn = np.poly1d(fit)

plt.plot(x1,y1, 'ro', x1, fit\_fn(x1), '--k')

plt.xlim(4,8)

plt.ylim(100,350)

plt.title('DKC63-33RIB (Without Outliers)')

plt.xlabel('distance between seeds (inches)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x1,y1)

tmp = 'correlation coefficient:', r\_value

plt.text(8, 345, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

ax1 = fig3.add\_subplot(122)

x2 = var6.ix[:,6]

y2 = var6.ix[:,3]

fit = np.polyfit(x2,y2,1)

fit\_fn = np.poly1d(fit)

plt.plot(x2,y2, 'go', x2, fit\_fn(x2), '--k')

plt.xlim(4,8)

plt.ylim(100,350)

plt.title('P1498 (Without Outliers)')

plt.xlabel(' distance between seeds (inches)')

plt.ylabel('Yield (bushels/acre)')

plt.grid(True)

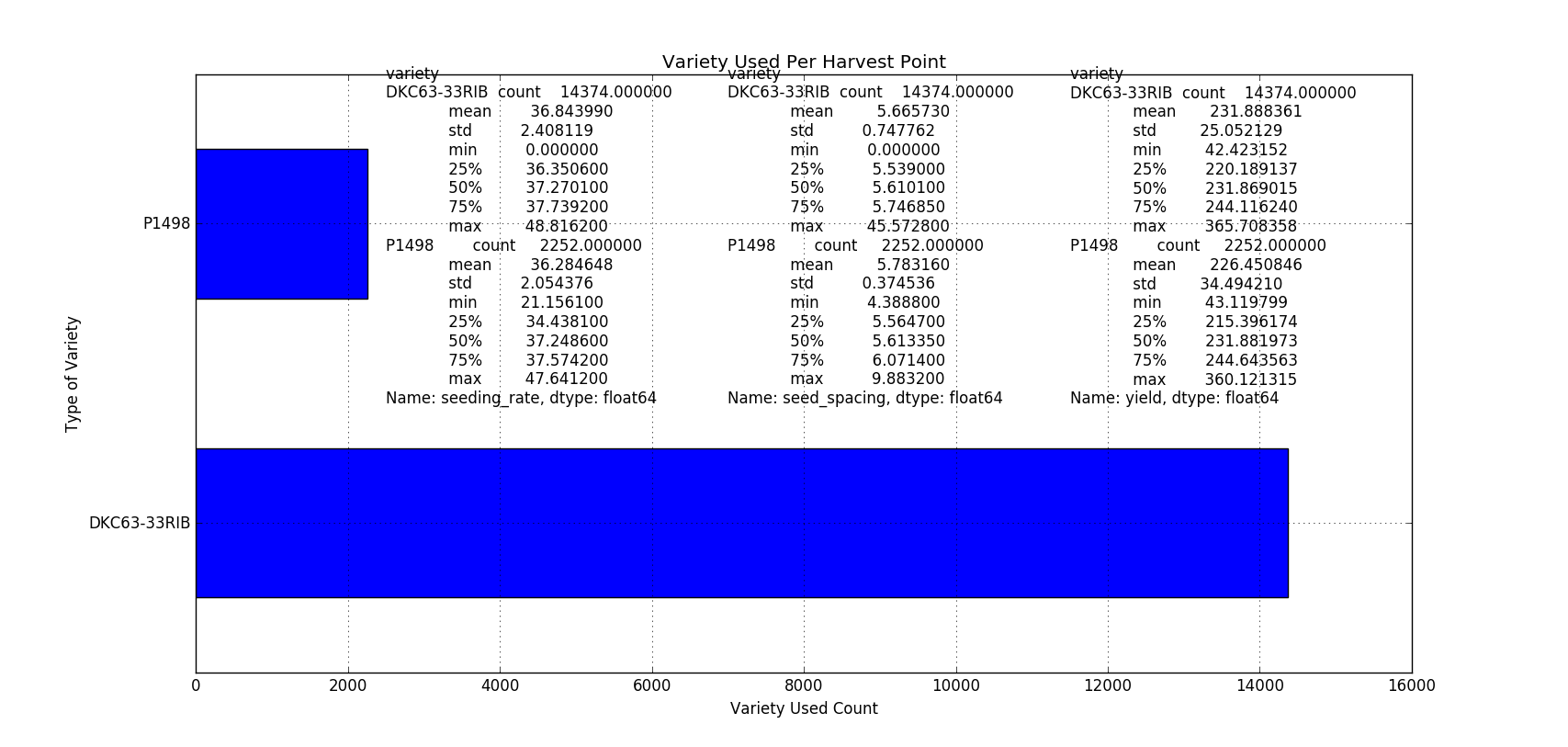
slope, intercept, r\_value, p\_value, std\_err = stats.linregress(x2,y2)

tmp = 'correlation coefficient:', r\_value

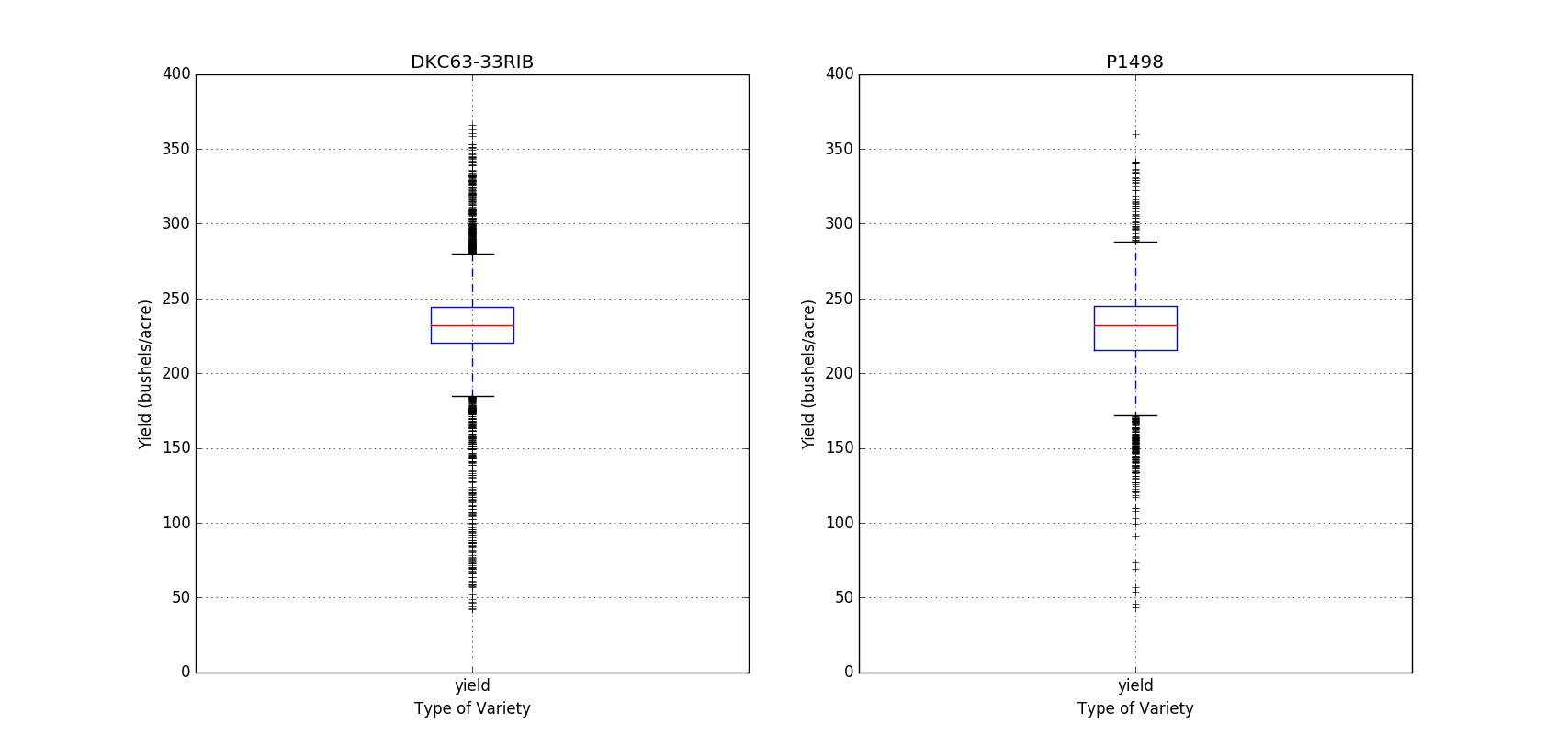
plt.text(8, 345, tmp, fontsize=12,horizontalalignment='right',verticalalignment='top')

figManager = plt.get\_current\_fig\_manager()

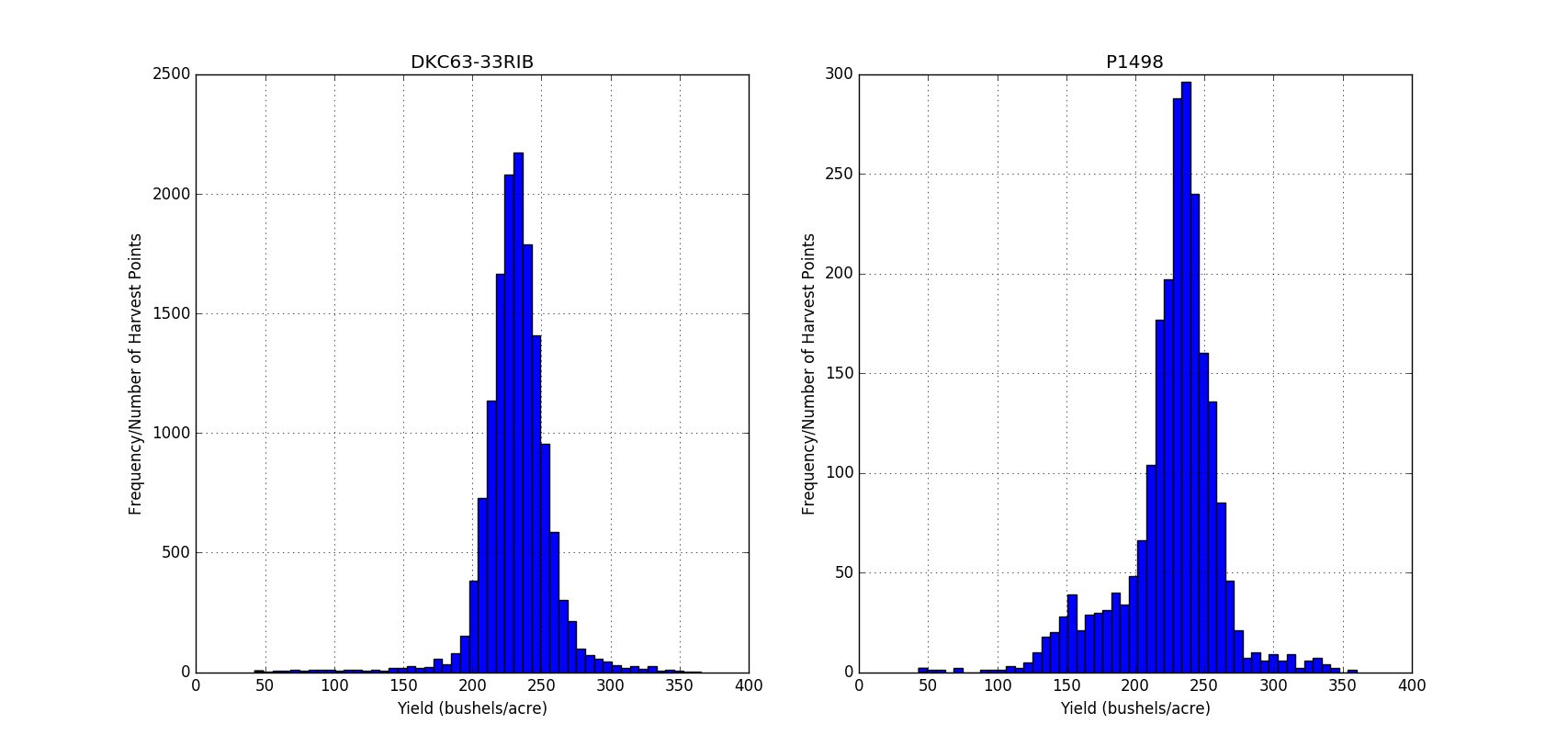
figManager.window.showMaximized()

**Exploratory Data Analysis:** ****

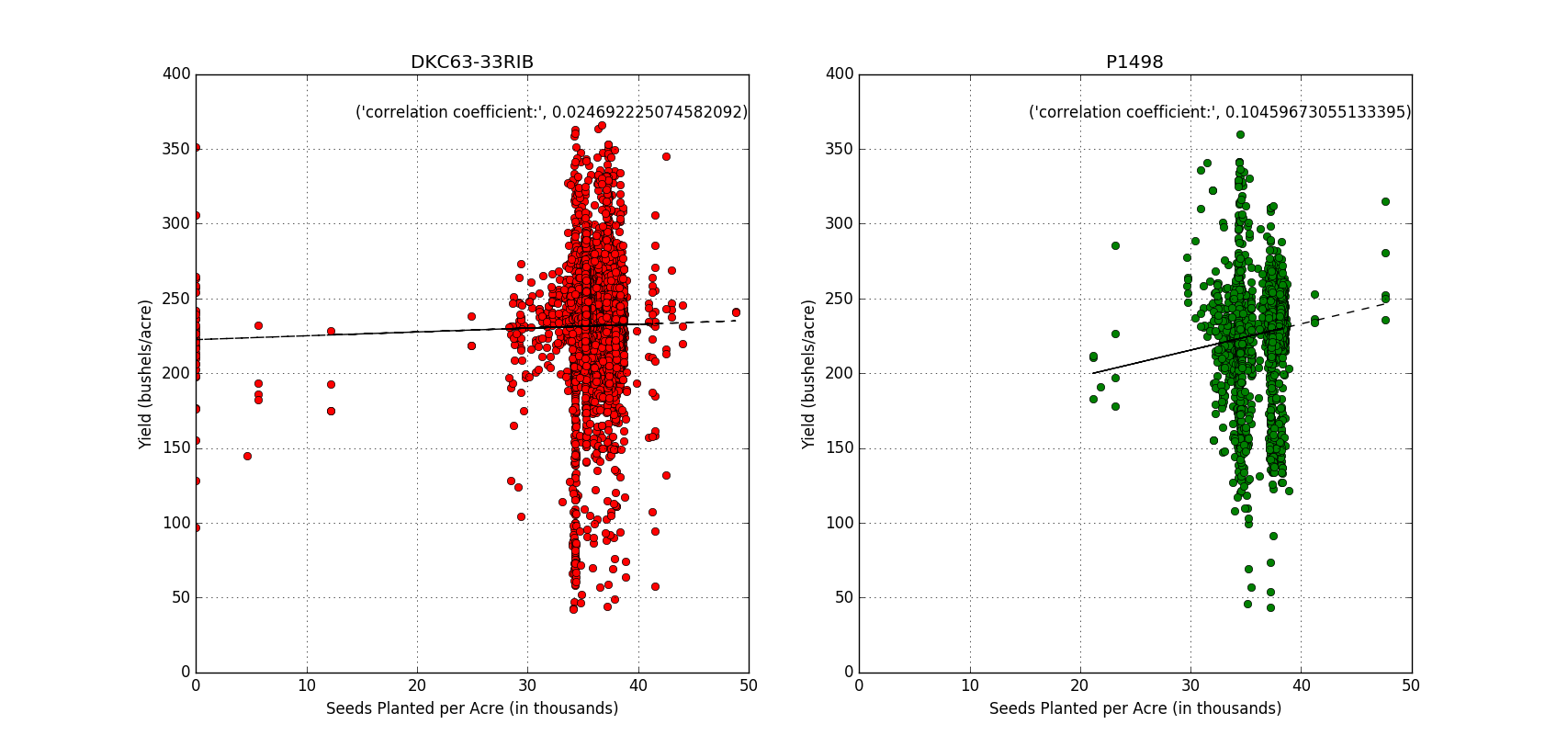
* The figure above gives the total count of each variety type that was harvested
* It can be observed that for every P1498 crop type that was harvested there were more than six DKC63-33RIB crop type that were harvested.
* The mean, standard deviation, minimum, maximum and percentiles (yield, seeding rate and seeding space) are given to the top right which is categorized by variety
* The DKC63-33RIB variety produces roughly 5 bushels/acre more than the P1498 crop variety
* Most descriptors are with ±5 units of each other with the exception of standard deviation



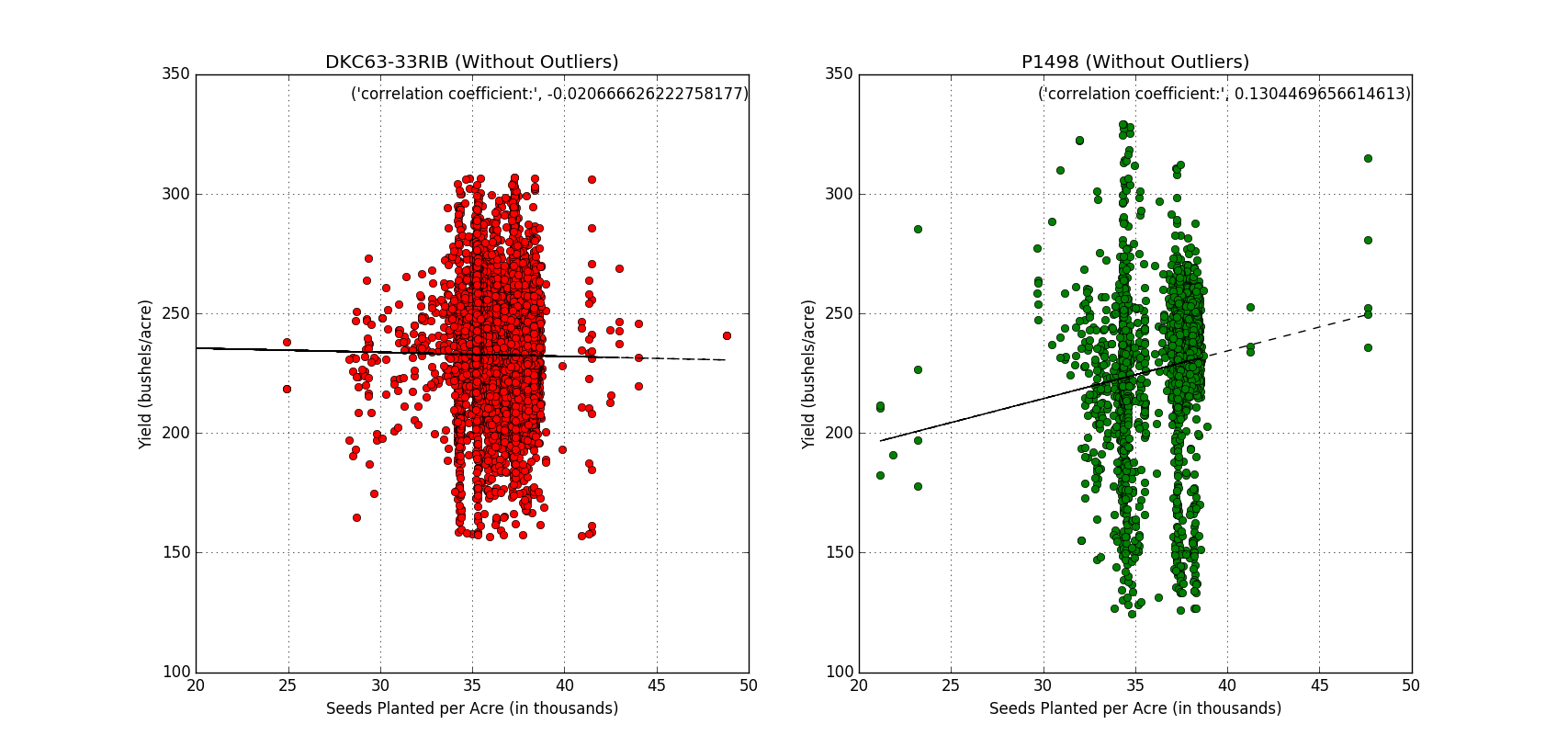
* The above is a box-plot to provide a graphical representation of the yield per variety. Both varieties are comparable

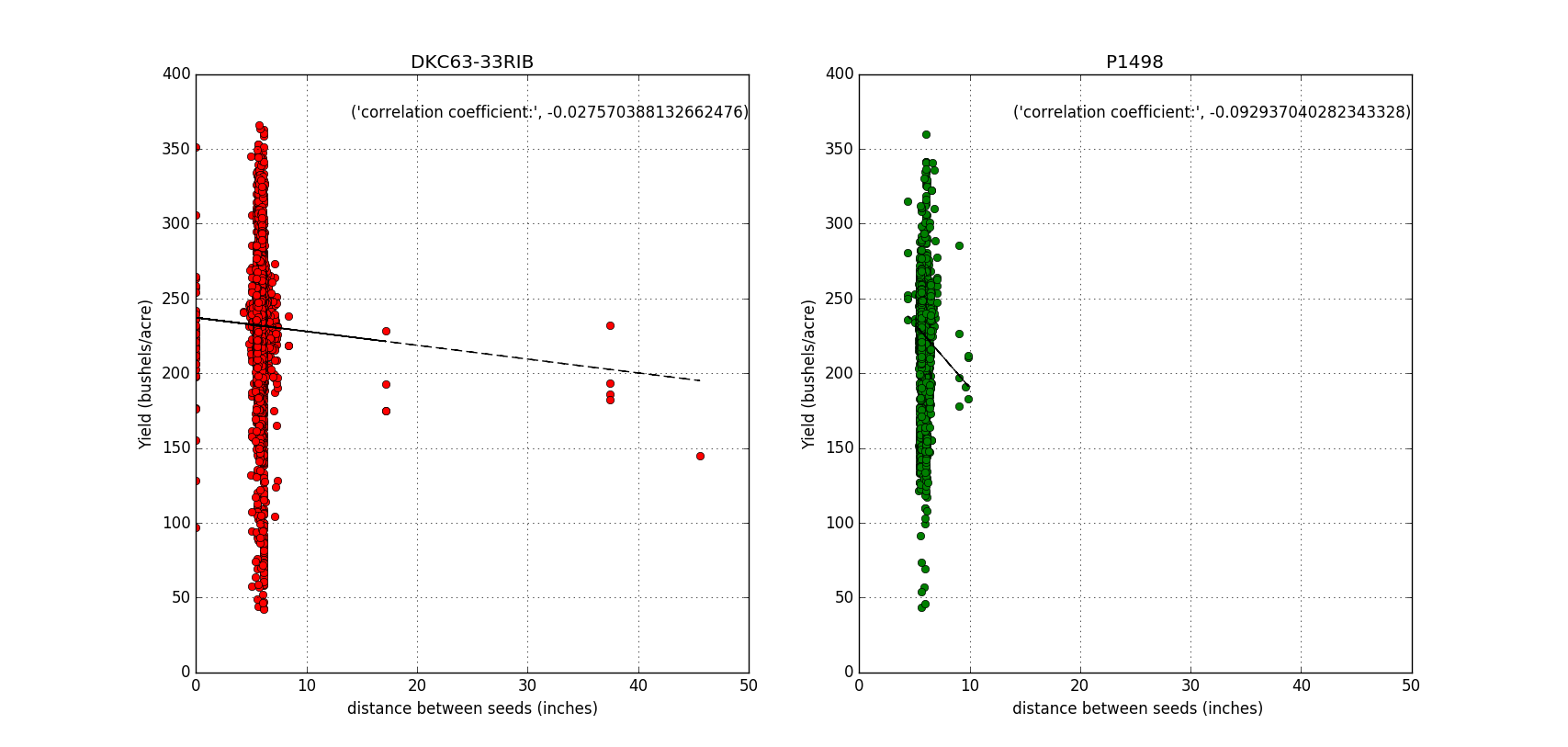


* Given above is a histogram of Yield as categorized by variety
* The DKC63-33RIB has the appearance of a normal distribution owing to the large number of records
* The P1498 appears to have a negative skew



* A linear regression model is given above (Seeding rate vs. yield) which is categorized by variety
* The correlation coefficient has been calculated for both models
* Both models show no linear correlation
* A best fit curve shows little improvement in yield (DKC63-33RIB) based on seeding rate
* However, P1498 shows a near 25 percent improvement in yield when seeding rate is increased from 20 to 50 thousand seeds planted per acre
* The model below is also a regression model between seeding rate and yield
* However only records falling with 3 standard deviations were preserved
* DKC63-33RIB variety shows no improvement (slight decline) in yield with an increase in seeding rate and the P1498 variety still shows a near 25 percent improvement in yield when seeding rate is increased from 20 to 50 thousand seeds planted per acre





* A linear regression model is given above (Seeding space vs. yield) which is categorized by variety
* The correlation coefficient has been calculated for both models
* Both models show no linear correlation and a decline in yield with an increase in seeding distance
* The model below is also a regression model between seeding distance and yield
* However only records falling with 3 standard deviations were preserved
* DKC63-33RIB variety shows no improvement (nearly stagnant) in yield with an increase in seeding distance and the P1498 variety still shows a near 25 percent decline in yield when seeding distance is increased from 4.5 to 8.0 inches between seeds
* It can be concluded that the DKC63-33RIB variety shows little improvement in yield with increase in seeding rate and seeding distance. However, the P1498 variety shows an increase in yield with an increase in seeding rate and decrease in yield with an increase in seeding distance

