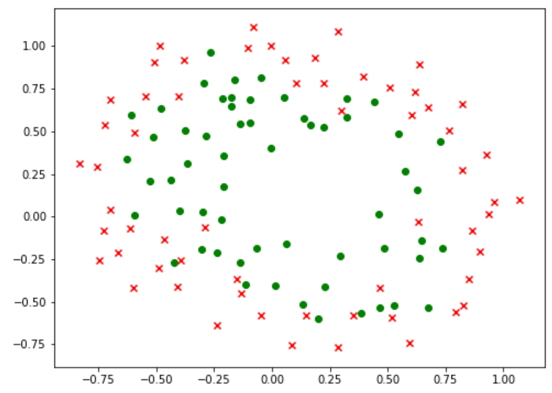
Project 2: Classification with Logistic Regression (10% of final grade)

Due: Before midnight on October 5, 2020 ($-3^{(n-1)}$ for late programs)

For this project we will apply Logistic Regression to predict whether capacitors from a fabrication plant pass quality control based (QC) on two different tests. To train your system and determine its reliability you have a set of 118 examples. The plot of these examples is show below where a red x is capacitor that failed QC and the green circles represent capacitors that passed QC.



I have already randomized the data into two data sets: a training set of 83 examples and a test set of 35 examples. Both are formatted as

- First line: m and n, tab separated
- Each line after that has two real numbers representing the results of the two tests, followed by a 1.0 if the capacitor passed QC and a 0.0 if it failed QC—tab separated.

Your assignment is to use what you have learned from the class videos and homework to create (from scratch in Python, not by using a Logistic Regression library function!) a binary classifier to predict whether each capacitor in the test set will pass QC. You are free to use any model variation and any testing or training approach we have discussed for logistic regression. In particular, since this data is not linear, I assume you will want to add new features based on powers of the original two features to create a good decision boundary. $w_0 + w_1x_1 + w_2x_2$ is not going to work! One choice might be

$w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6 + w_7x_7 + w_8x_8$ where the new features are created as follows:

New Features	From Original Features
x_1	x^1
x_2	x_1^2
x_3	x_2
x_4	x_1x_2
x_5	$x_1x_2^2$
x_6	x_2^2
x_7	$x_1^2x_2$
x_8	$x_1^2x_2^2$

Note that it is easy to create a small Python program that reads in your original features, uses a nested loop to create the new features and then writes them to a file.

```
thePower = 2
for j in range(thePower+1):
    for i in range(thePower+1):
        temp = (x1**i)*(x2**j)
        if (temp != 1):
            fout1.write(str(temp)+"\t")
fout1.write(str(y)+"\n")
```

With a few additions to the code you can make a program to create combinations of any powers of x_1 and x_2!

What to Turn In Via Canvas

1. ***A single py file (lastname_firstname_P2.py)*** that prompts for a training file name, computes weights using gradient descent, prints out a plot of iterations vs. J, and then prompts for a test file

name, and using the computed weights prints out final J, FP, FN, TP, TN, accuracy, precision, recall and F1 for the test set. All values should be clearly labelled.

2. ***Your training set file (lastname_firstname_P2Train.txt).*** First line should contain integers m and n, tab separated. Each line after that should have n real numbers representing the new feature data, followed by a 1.0 if the capacitor passed QC and a 0.0 if it failed QC—tab separated.
3. ***Your test set file (lastname_firstname_P2Train.txt).*** First line should contain integers m and n, tab separated. Each line after that should have n real numbers representing the new feature data, followed by a 1.0 if the capacitor passed QC and a 0.0 if it failed QC—tab separated.
4. ***A pdf file (lastname_firstname_P2.pdf)*** that includes
   - A description of your model and testing procedure, including
     - Description of your model
     - Initial values that you chose for your weights, alpha, and the initial value for ***J***.
     - Final values for alpha, your weights, how many iterations your learning algorithm went through and your final value of ***J*** on your training set.
     - Include a plot of ***J*** (vertical axis) vs. number of iterations (horizontal axis).
     - Value of ***J*** on your test set.
   - A confusion matrix showing your results on your test set.
   - A description of your final results that includes accuracy, precision, recall and F1 values.

Do not assume that any files are available to you besides files you turn in!

Zip your files into one zip file named ***lastname_firstname*** and upload them to Canvas.