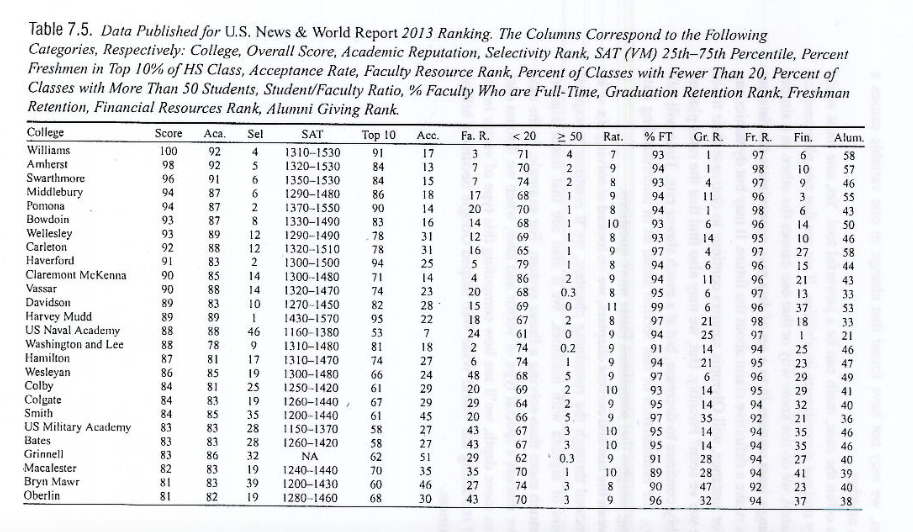
**College Rankings Project** Name:

You can open my college data file to find the following table from Life is Linear:



**Project Directions**

Create a Python program that does the following:

1. Uses *all* of the colleges in the above list except for Grinnell (who doesn’t publish SATs) and all of the variables above (Academic Reputation through Alumni Giving Rank) to create a matrix, A. In addition, create a vector, b, containing all of the colleges’ (except for Grinnell’s) US News World & Report Score (2nd column).

2. Normalizes by editing each entry to be (entry-minimum)/(maximum-minimum) for the min and max value in that column. (Why do we need to normalize?)

3. Solves for the weight of each ranking using the formula .

4. What variables affect a school’s ranking most in a positive way? In a negative way?

5. Use the weights to calculate the US News World & Report score for Colby. (Don’t enter Colby’s data manually but rather use the row in your matrix above).

**Python Refresher – Be sure to do this before getting started on your project!**

Open Anaconda-Jupyter Notebook and start a new project.

Put the csv collegedata file that I’ve sent you in the same directory as this project.

Always keep these two lines at the top of your file:

**import** numpy **as** np  
**from** scipy **import** linalg

In order to generate a matrix of the college data, type the following:

collegedata=np.genfromtxt('collegedata.csv',delimiter=',', encoding=None, dtype=None)

This command creates a matrix called collegedata that stores the information from the file collegedata.csv. It goes to another column in the matrix whenever it sees a comma between the numbers (that is what delimiter=’,’ does). It also doesn’t declare the table as numbers or letters, since the matrix contains both college names and numbers. (dtype=None). Be sure to use a capital “N.”

You can print the entire matrix by typing:

**print**(collegedata)

You can find how many colleges are in the list by typing:

**print**(len(collegedata))

You can find how many variables are in the list by typing:

**print**(len(collegedata[0]))

You can find out the data for just the Williams school by typing:

**print**(collegedata[0])

How would you find the data for Oberlin?

You’ll probably need to use a for loop in your program, so let’s review those. Remember that for loops require specifying a range. Create a list A=[2,3,5,7,10]. If you wanted to print out one entry at a time, you could type:

A=[2,3,5,7,10]  
**for** i **in** range(len(A)):  
 **print**(A[i])

Remember that range will always stop one short of the maximum value.

Suppose you wanted to find all of the SAT scores in the list, which is contained in the 5th column of the matrix. You could type:

**for** i **in** range(len(collegedata)):  
 **print**(collegedata[i][4])

How would you print out the alumni giving rank?

You might want to create a 3x2 matrix of zeros. You could type:

B=np.zeros((3,2))  
**print**(B)

Suppose you wanted to create a matrix of zeros but then change a few entries. You could type:

B=np.zeros((3,2))  
B[0,1]=5  
B[1,1]=-4  
B[2,0]=7  
**print**(B)

Suppose you wanted to create a 2x3 matrix of the following entries. You could type:

C=np.matrix([[1,0,4],[3,-1,6]])  
**print**(C)

To find the dimensions of the matrix, you can type:

**print**(C.shape[0],C.shape[1])

Suppose you wanted to create a 3x4 matrix where each Dij entry contained the sum of the i and j indices. You could type:

D=np.zeros((3,4))  
**for** i **in** range(D.shape[0]):  
 **for** j **in** range(D.shape[1]) :  
 D[i][j]=i+j  
**print**(D)

Suppose you have the matrix P=[[1,0,2],[2,0,0],[0,1,2]] and you want to calculate its transpose and inverse. You could type:

P=np.matrix([[1,0,2],[2,0,0],[0,1,2]])  
**print**(P.T)  
**print**(linalg.inv(P))

Suppose you want to multiply P by the vector Q=[[1],[2],[3]]. You could type:

P=np.matrix([[1,0,2],[2,0,0],[0,1,2]])

Q=np.matrix([[1],[2],[3]])

print(P\*Q)

You may also want to use simple lists that are slightly different than numpy arrays.

If you want to create a blank list and then add things to it as you go, you can type:

mylist=[]

Suppose you want to add the number 7 to the list [10,2,1] and delete the number 2. Then, you want to sort it. You could type:

mylist=[1,2,10]  
mylist.append(7)  
mylist.remove(2)  
mylist.sort()  
**print**(mylist)

The scipy linear algebra package contains limitless linear algebra operations. Here are a few:

import numpy as np

from scipy import linalg

A=np.matrix([[3,0],[8,-1]])

#inverse:

print(linalg.inv(A))

#determinant:

print(linalg.det(A))

#solving Ax=b:

b = np.matrix([[1],[2]])

print(linalg.solve(A, b))

#eigenvalues:

print(linalg.eigvals(A))

#eigenvalues and eigenvectors:

print(linalg.eig(A))

**The Three Most Important Things To Remember**

1. I have given you most but probably not all of the commands that will be helpful to you with this project. Google is your friend! If I wanted to Google “numpy matrix transpose” then lots of good information would pop up.

2. Remember to start small and then increase in difficulty level. You might want to work with a smaller file before working with the full college file.

3. Patience is a virtue!! You’ll probably run into many small (and large) problems along the way. Try your best on your own, use Google when you get stuck, and leave your program alone for a while and come back to it with a fresh set of eyes. If all of that still doesn’t help, then ask for help.