Homework 0: Preliminary exercises in Data Science

Data understanding and visualization

1. Download survey_multiple_choice.tsv located here. (hint: use wget or curl to pull the remote data from the server to your computer) This data represents the answers to the multiple choice survey issued prior to the course. A brief description of this dataset is given here.

2. How many students have completed the survey? Hint: use the wc unix utility.

   bash-3.2$ wc -l survey_multiple_choice.tsv

   39 survey_multiple_choice.tsv

3. How many students claim to have each skill level for the unix shell, databases, and programming, respectively?

   Unix :

   bash-3.2$ cut -f1 survey_multiple_choice.tsv|sort |uniq -c|sort -nr

   12 I have written simple terminal commands or done some system work on the terminal

   11 I have issued a few commands in a terminal based on given instructions

   8 I have no experience working in a terminal

   4 I have written complex commands done or have done deep system work

   4 I dont even understand the question

   Databases :

   bash-3.2$ cut -f2 survey_multiple_choice.tsv|sort |uniq -c|sort -nr

   12 I can write simple queries and issue them to a database

   10 I can write very complex queries when needed

   9 I have issued simple queries to a relational database based on given instructions

   7 I have never directly accessed a database

   1 I am a database hacker

   Programming

   bash-3.2$ cut -f3 survey_multiple_choice.tsv|sort |uniq -c|sort -nr

   13 I have written simple programs, based on instructions or a tutorial

10 I can write simple programs to accomplish tasks I encounter

9 I can write complex programs, am familiar with programming design patterns, software testing, system design, and algorithms.

4 I have never programmed before.

3 I am a hacker or have senior-level programming experience

4. Which discipline (unix, database, or programming) would you say has the highest overall skill level amongst the students responding to the survey? Which discipline has the lowest?

   unixScore : 82

   databaseScore : 67 (Lowest)

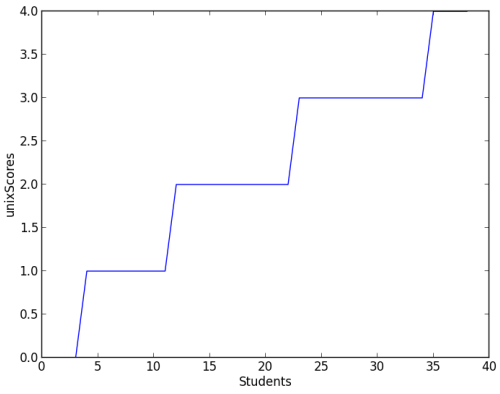
   programmingScore : 72

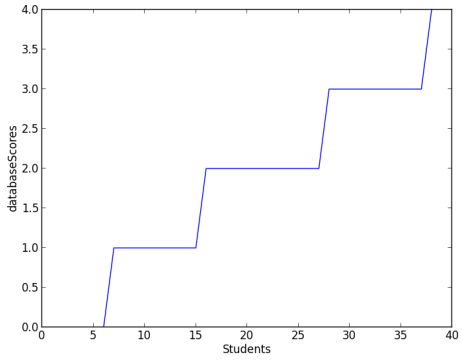
5. Write a simple python program to consume this data. (hint: look at the string split method to get at the data in the individual rows)

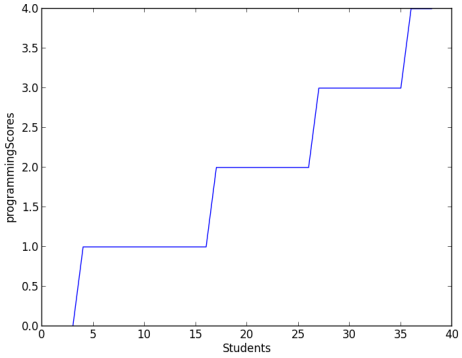
   Please see attached python code

6. Using matplotlib, make a plot of the distribution of skill levels, starting from the lowest and going to the highest for each of the three disciplines. You may wish to substitute the strings describing the skill level with a numeric value.

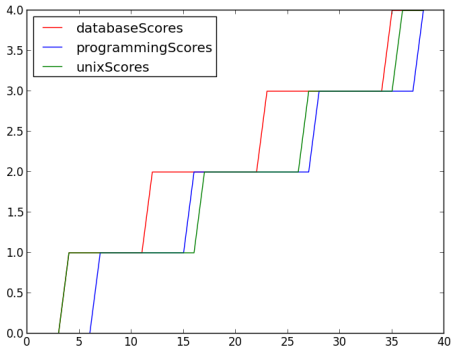
   Please see attached python code



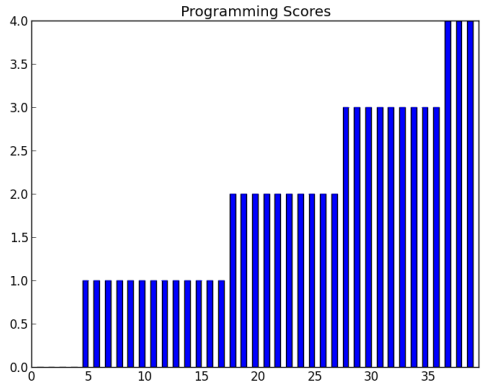


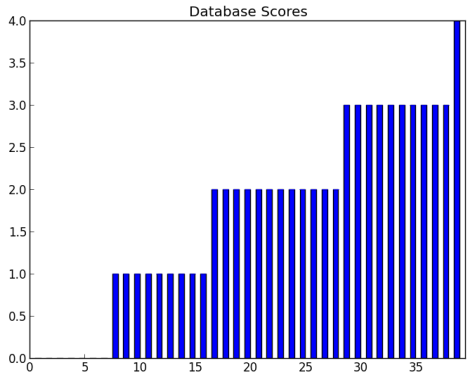


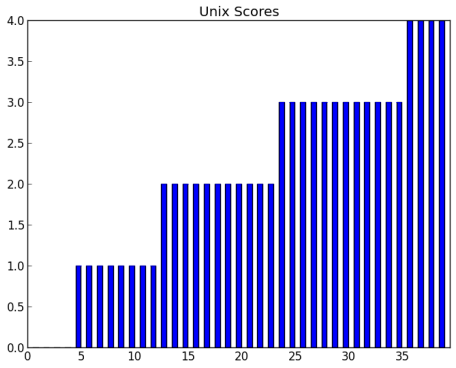
7. Combine these three plots, overlaying them on a single graph. Make sure each line is a different color for each line, and make a legend to tell the different colors apart.



8. Repeat question 6 using a bar plot.







Question 2

1. How many lines are in the file?

   Vamsee-Jastis-MacBook-Pro:second zudec$ wc -l marketing.data

   8994 marketing.data

2. Notice that many lines have some fields unavailable (NA). Remove any lines without complete data. How many lines remain?

Vamsee-Jastis-MacBook-Pro:second zudec$ grep -v "NA" marketing.data | wc -l

6877

3. The fifth column corresponds to education level. What is the most common education level?

Vamsee-Jastis-MacBook-Pro:second zudec$ clear

Vamsee-Jastis-MacBook-Pro:second zudec$ python incomeSurvey.py marketing.data

Reading the survey file

{'1': 176, '3': 1479, '2': 787, '5': 1207, '4': 2407, '6': 820}

The most common education level is 4 - 2407

4. What is the income distribution for households with some graduate school? (hint: use a python dict data structure to store income level counts)

Based on 3 above, combining both 5 and 6 -> 1207+ 820 = 2027

6. Consider the following simple model of income level using only education level.

Let 4 be the nominal income level, with the following adjustments in income level being made according to education:

education level	income modifier
1	-3
2	-1
3	0
4	+1
5	+3
6	+4

What is the total difference between actual and predicted income level using the above model? What about the average difference per user? (Hint: again use a dict data structure, this time to store the modifiers of the model)

Total Entries : 6876
Total differenceFirst : 759
Average differenceFirst per user : 0.110383944154

See source below

7. Consider the following modification to the model presented in question 6 that additionally incorporates the following information about a person's occupation:

occupation	income modifier
1	+2.5
2	+.6
3	0
4	+.2
5	-.5
6	-1.5
7	+.3

8 +.8
9 -2.5

In this setting, we are using a two-factor estimate an individual's income, according to both occupation and education level. What is the total difference between actual and predicted income level using the above model? What about the average difference per user? Is this better or worse than the model presented in question 6? Is this model more likely to overestimate or underestimate an individual's income level?

Total Entries : 6876

Total differenceSecond : 5011.9

Average differenceSecond per user : 0.728897614892

The model is worse off than the previous one because the difference in salary levels is much greater than the first one.

The model is more likely to over estimate because the salary levels are much higher than the actual salary levels.

See source below

Source for Problem 1

```
from sys import argv
import matplotlib.pyplot as plt
import numpy as np
import csv
import pylab as p
from pylab import *
# Initilize a static class with answers and corresponding scores
class Data:

   unix = {'I dont even understand the question': 0 ,
                'I have no experience working in a terminal':1 ,
         'I have issued a few commands in a terminal based on given instructions':2,
         'I have written simple terminal commands or done some system work on the terminal':3,
         'I have written complex commands done or have done deep system work':4
        };

   database ={'I have never directly accessed a database': 0 ,
          'I have issued simple queries to a relational database based on given instructions':1 ,
          'I can write simple queries and issue them to a database':2,
          'I can write very complex queries when needed':3,
          'I am a database hacker':4
                 };

   programming = {'I have never programmed before.': 0 ,
              'I have written simple programs, based on instructions or a tutorial':1 ,
              'I can write simple programs to accomplish tasks I encounter':2,
              'I can write complex programs, am familiar with programming design patterns, software testing, system design, and
algorithms.':3,
              'I am a hacker or have  senior-level programming experience':4
                           };

print("Reading the survey file\n");

script, filename= argv

unixScore = 0;
databaseScore = 0;
programmingScore = 0;
student = 0;
```

```
answers = {}

def plotBarChart(data, chartName) :
   error = [0] * len(data)

   xlocations = np.array(range(len(data)))+0.5
   width = 0.5
   bar(xlocations, data, yerr=error, width=width)
   #yticks(range(0, 8))
   #xticks(xlocations+ width/2, labels)
   xlim(0, xlocations[-1]+width*2)
   title(chartName)
   gca().get_xaxis().tick_bottom()
   gca().get_yaxis().tick_left()
   p.show()


def convertDictToNumArr(answers) :
   myarray = np.empty((len(answers.keys()), 3), dtype=int)
   for student in range(len(answers.keys())):
      for question in range(3):
         myarray[student, question] = answers[student+1][question]

   return myarray

def import_text(filename, separator):
   for line in csv.reader(open(filename), delimiter=separator,
                  skipinitialspace=True):
      if line:
         yield line

# Reading the input file

for data in import_text(filename, '\t'):
   student = student + 1
   entry = data[0]
   localUnixScore = Data.unix[entry.strip()]
   unixScore = unixScore+localUnixScore

   entry = data[1]
   localDatabaseScore = Data.database[entry.strip()]
   databaseScore = databaseScore + localDatabaseScore


   entry = data[2]
   localProgrammingScore = Data.programming[entry.strip()]
   programmingScore = programmingScore + localProgrammingScore

   studentScore = [localUnixScore, localDatabaseScore, localProgrammingScore]
   answers[student]=studentScore

print("Cummulative scores :");

print("unixScore : %d" %unixScore)
print("databaseScore : %d "%databaseScore)
print("programmingScore : %d "%programmingScore)


# Converting the dictionary with the student and his scores to a numPyArray

numPyArray = convertDictToNumArr(answers)
studentsArray = np.arange(39)

# Get sorted unix scores

unixScores = sorted(numPyArray[:,0])
```

```
# Get sorted database scores

databaseScores = sorted(numPyArray[:,1])

# Get sorted programming scores

programmingScores = sorted(numPyArray[:,2])

#Display individual graphs

plt.plot(unixScores)
plt.ylabel('unixScores')
plt.xlabel('Students')
plt.show()

plt.plot(databaseScores)
plt.ylabel('databaseScores')
plt.xlabel('Students')
plt.show()

plt.plot(programmingScores)
plt.ylabel('programmingScores')
plt.xlabel('Students')
plt.show()

#Display the combined graph

plt.plot(studentsArray, unixScores, 'r',studentsArray, databaseScores,'b', studentsArray, programmingScores,'g')
plt.legend( ('databaseScores', 'programmingScores', 'unixScores'), loc='upper left')
plt.show()

# Bar plot

plotBarChart(unixScores, "Unix Scores")
plotBarChart(databaseScores, "Database Scores")
plotBarChart(programmingScores, "Programming Scores")
```

Source for Problem 2

```
import csv

educationModifier = {"1" :-3,
                     "2" :-1,
                "3" :0,
                "4" :1,
                "5" :3,
                "6" :4}

occupationModifier = {"1" :2.5,
                 "2" :.6,
                 "3" :0,
                 "4" :.2,
                 "5" :-.5,
                 "6" :-1.5,
                 "7" :.3,
                 "8" :.8,
                 "9" : -2.5}

nominalModifier = 4
actualIncomeLevel = 0
occupationLevel = 0
runningSalaryFirst = 0
runningSalarySecond = 0
counter =0
reader = csv.reader(open("marketing.data","rb"),delimiter=' ',quoting=csv.QUOTE_NONE)
```

```
for entry in reader :
    educationLevel = entry[4]
    actualIncomeLevel = entry[0]
    occupationLevel = entry[5]
    individualEducationModifier = educationModifier.get(educationLevel)
    individualOccupationModifier = occupationModifier.get(occupationLevel)
    predictedSalaryFirst = nominalModifier +individualEducationModifier
    predictedSalarySecond = nominalModifier + individualEducationModifier+individualOccupationModifier
    differenceFirst = predictedSalaryFirst - int(actualIncomeLevel)
    differenceSecond = predictedSalarySecond - int(actualIncomeLevel)
    runningSalaryFirst = runningSalaryFirst + differenceFirst
    runningSalarySecond = runningSalarySecond + differenceSecond
    counter = counter + 1


print "Total Entries :", counter
print "Total differenceFirst : ", runningSalaryFirst
print "Total differenceSecond : ", runningSalarySecond
print "Average differenceFirst per user :", float(runningSalaryFirst)/float(counter)
print "Average differenceSecond per user :", float(runningSalarySecond)/float(counter)
```