CSE 574 Introduction to Machine Learning (Fall 2017)

Instructor: Dr. Srihari

Project 1

Prasad Salvi

UBitName: prasadde

UBitNumber: 50207353

Task 1

Compute for each variable ((CS Score, Research Overhead, Admin Base Pay, Tuition)) its sample mean, variance and standard deviation. Related variables: mu1, mu2, mu3, mu4, var1, var2, var3, var4, sigma1, sigma2, sigma3, sigma4

Mean

The mean is used to summarize a data set. It is a measure of the center of a data set.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x(i)$$

Steps:

Using the numpy.mean() function we calculated the mean and stored the result in mu1, mu2, mu3, mu4 variables resp. The average is taken over the flattened array by default.

Results:

Data Column	Variable	Value
csscore	mu1	3.214
researchoverhead	mu2	53.386
adminbasepay	mu3	469178.816
tuition	mu4	29711.959

Variance

Variance is the average of the squared differences from the Mean.

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}[x(i) - \mu]^2$$

Steps:

Using the numpy.var() function we calculated the variance and stored the result in var1, var2, var3, var4 variables resp.

Results:

Data Column	Variable	Value
csscore	var1	0.448
researchoverhead	var2	12.588
adminbasepay	var3	13900134681.7
tuition	var4	30727538.733

Standard Deviation

The standard deviation is the square root of the average of the squared deviations from the mean.

std = sqrt(mean(abs(x - x.mean())**2))

Steps:

Using the numpy.std() function we calculated the standard deviation and stored the result in sigma1, sigma2, sigma3, sigma4 variables resp.

Results:

Data Column	Variable	Value
csscore	sigma1	0.669
researchoverhead	sigma2	3.548
adminbasepay	sigma3	117898.832
tuition	sigma4	5543.243

Task 2

Compute for each pair of variables their covariance and correlation. Show the results in the form of covariance and correlation matrices. Also make a plot of the pairwise data showing the label associated with each data point. Which are the most correlated and least correlated variable pair? Related variables: covarianceMat, correlationMat

Covariance

Covariance indicates the level to which two variables vary together. If we examine N-dimensional samples, $X = [x_1, x_2, ...x_N]^T$, then the covariance matrix element C_{ij} is the covariance of x_i and x_j. The element C_{ii} is the variance of x_i.

Steps:

Stack the excel column arrays vertically to make a single array using numpy.vstack() function.

Then calculate the covariance matrix using numpy.cov() function.

Results:

```
[[ 4.57000000e-01  1.10600000e+00  3.87978200e+03  1.05848000e+03]
 [ 1.10600000e+00  1.28500000e+01  7.02793760e+04  2.80578900e+03]
 [ 3.87978200e+03  7.02793760e+04  1.41897208e+10 -1.63685641e+08]
 [ 1.05848000e+03  2.80578900e+03 -1.63685641e+08  3.13676958e+07]]
```

Correlation Coefficients

Normalized covariance matrix. The relationship between the correlation coefficient matrix, *R*, and the covariance matrix, *C*, is

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}}$$

Steps:

Stack the excel column arrays vertically to make a single array using numpy.vstack() function.

Then calculate the correlation coefficients using numpy.corrcoef() function.

Results:

[[1. 0.456 0.048 0.279]
[0.456 1. 0.165 0.14]
[0.048 0.165 1. -0.245]
[0.279 0.14 -0.245 1.]]

Pairwise Plot

Steps (improvement)

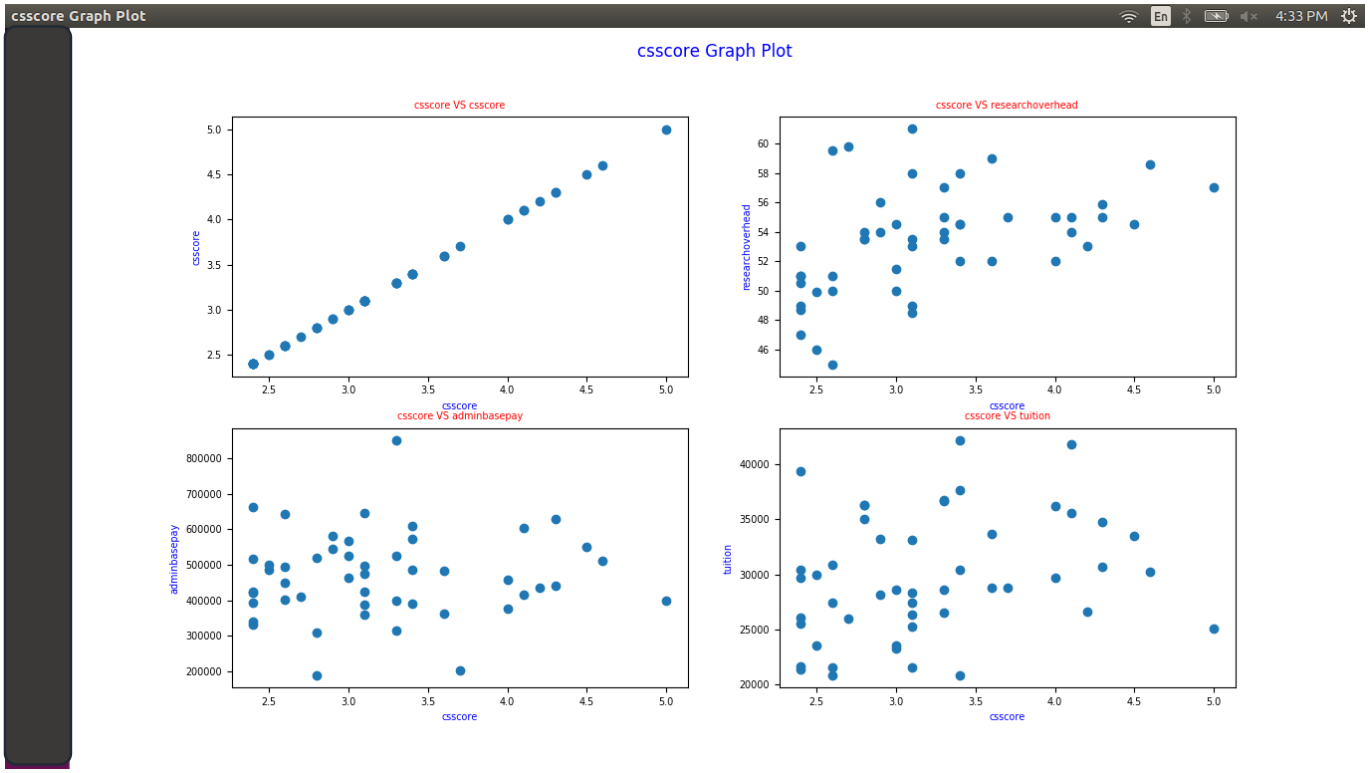
Plot the scatter graph for the one of the data variable against other data variable using matplotlib.pyplot python module.

The function *plotGraph.plot_graph(index,mainarray,names)* takes the following arguments index=[0..4] count of variables, mainarray=[data values for each variable in form of array], names=[variable names].

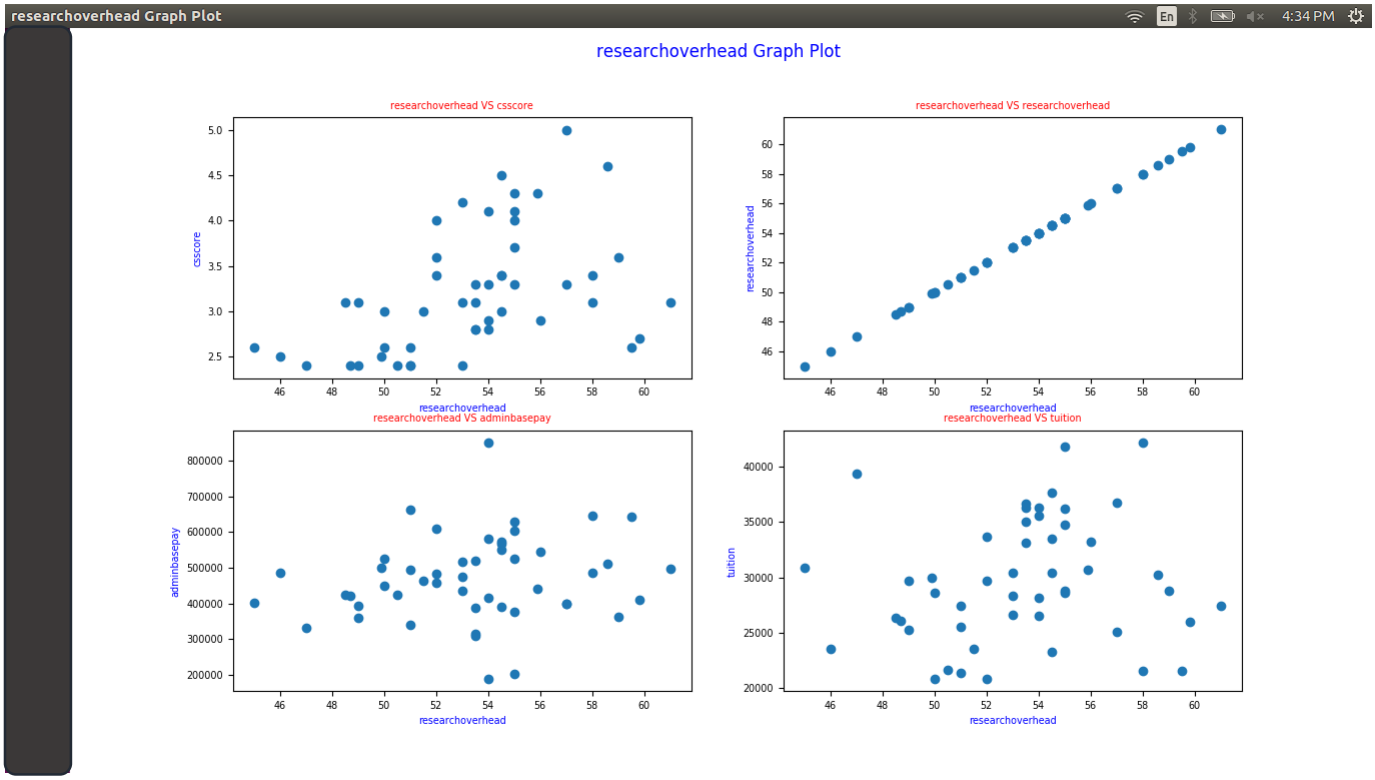
Each graph is plot using the same configuration like font size, label size, figure size, etc.

This function can be used even if the variable arguments changes in the given data set and can be used as generalized graph plot for n number of data variables without any code modification. (Refer: plotGraph.py)

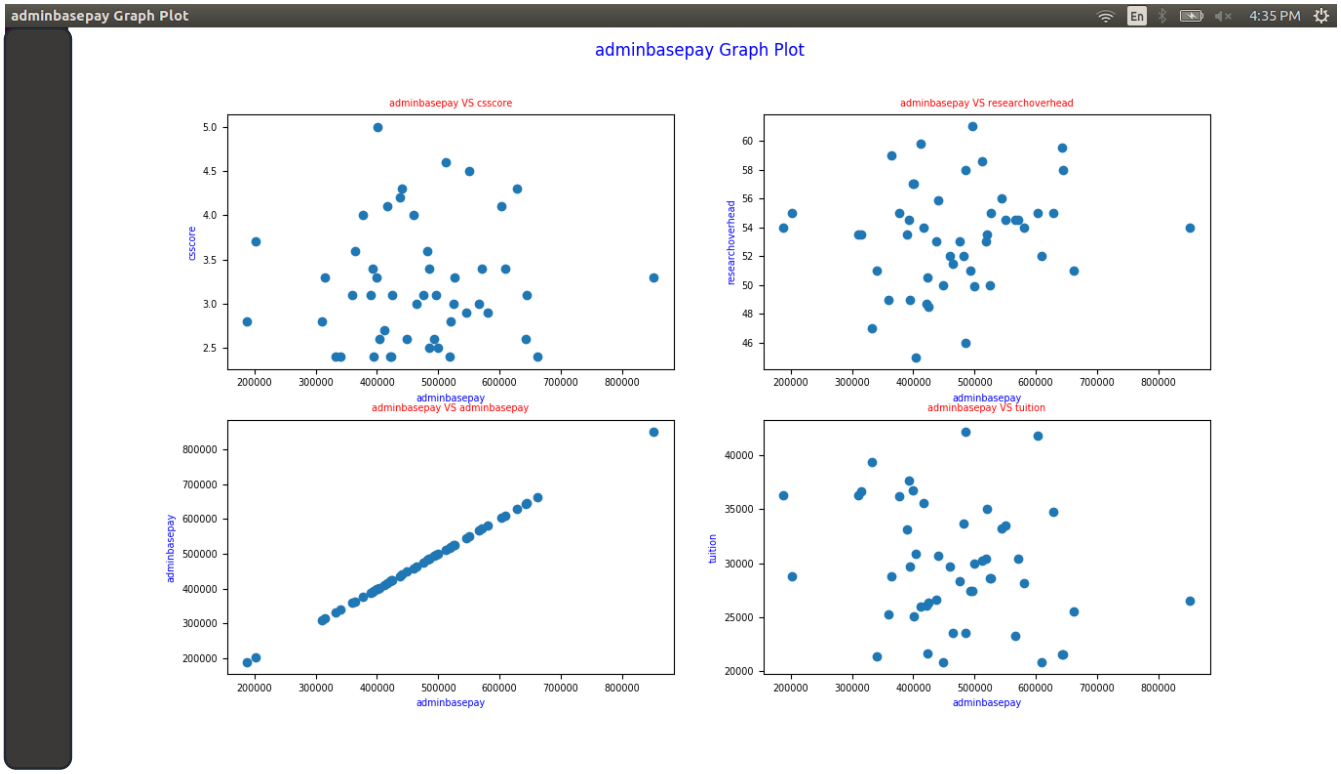
Results:



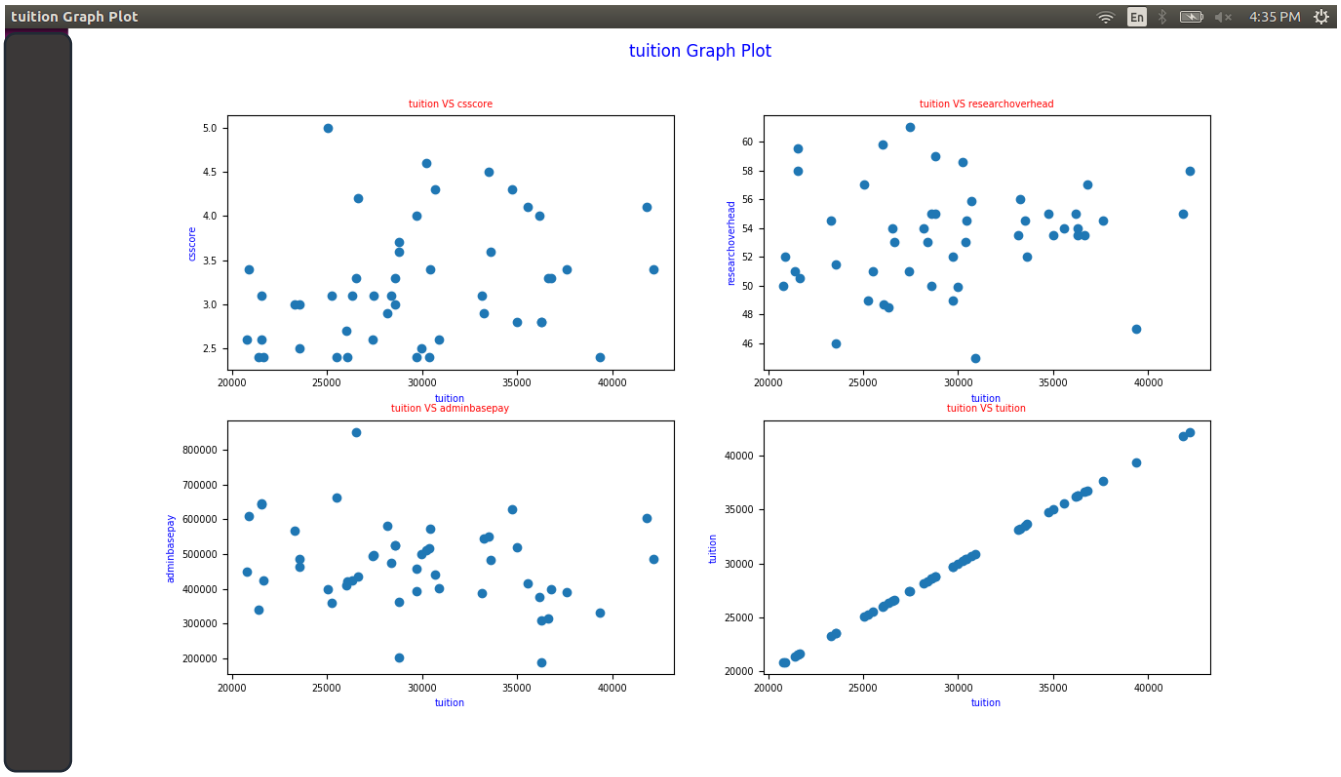
Graph csscore



Graph research overhead



Graph admins base pay



Graph tuition

Task 3

Assuming that each variable is normally distributed and that they are independent of each other, determine the log-likelihood of the data (Use the means and variances computed earlier to determine the likelihood of each data value.) Related variables: logLikelihood

LogLikelihood

Likelihood is used after data are available to describe a function of a parameter (or parameter vector) for a given outcome. We use LogLikelihood because the logarithm is a monotonically increasing function, the logarithm of a function achieves its maximum value at the same points as the function itself, and hence the log-likelihood can be used in place of the likelihood in maximum likelihood estimation and related techniques.

$$\mathbf{L}(\mathbf{x}_1, ..\mathbf{x}_N) = \sum_{i=1}^{N} log\ p(\mathbf{x}_i)$$

Steps (improvement):

We have calculated mean and variance for each variable. We will make a list of these calculated values and called them as mean{} and sigma{}. Using the logpdf() function we can then calculate logLikelihood for each variable independently. And to find logLikelihood for sample data we will add these calculated likelihoods for each variable. (Refer: calculateLogLikelihood.py)

This function can be used even if the variable arguments changes in the given data set and can be used as to calculate log likelihood for n number of data variables without any code modification.

Result:

Individual Log likelihood

[-49.86437156 -131.58053354 -641.7295151 -491.92439915]

LogLikelihood=

-1315.099

Task 4

Using the correlation values construct a Bayesian network which results in a higher loglikelihood than in task 3. Related variables: BNgraph, BNlogLikelihood

The Bayesian network was created using the combination of variable by trial and error which led to the best possible logLikelihood for the network for the below combination of Bayesian network.

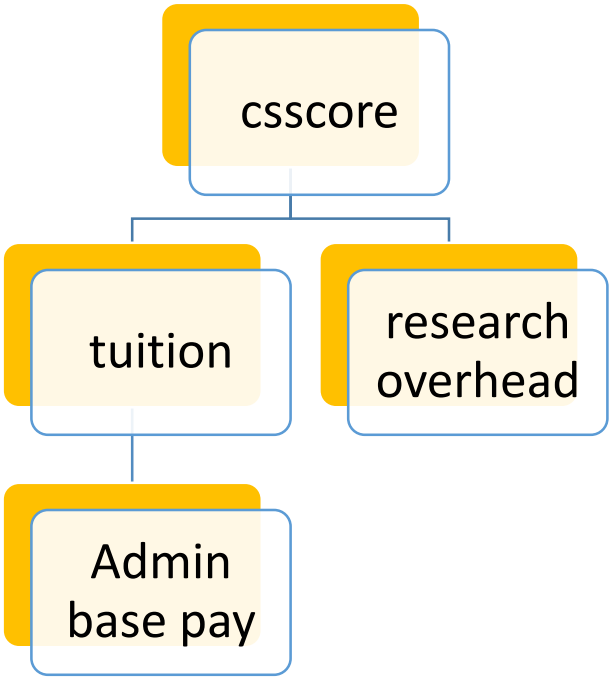
The calculated logLikelihood was greater than the one derived by mathematical computation for independent variables earlier in Task 3 (-1315.099).

Results:

BNgraph=
[[0. 1. 0. 1.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 1. 0.]]

BNlogLikelihood=
-1306.263

Bayesian Network:
Bayesian network giving maximum likelihood.



Console Output