Biostatistics project Report

Ahmed Mohamed
ahmedbadra29@gmail.com BN: 7, Sec: 1

Mohamed Khaled Galloul
eng.mohamedgalloul@gmail.com BN: 14, Sec: 2

kareem Moustafa
karemyassen98@gmail.com BN: 9, Sec: 2

Moamen Gamal
moameneisa@yahoo.com BN: 10, Sec: 2

1. Introduction and Motivation

Cancer classification based on molecular level investigation has gained the interest of researches as it provides a systematic, accurate and objective diagnosis for different cancer types. Several recent researches have been studying the problem of cancer classification using data mining methods, machine learning algorithms and statistical methods to reach an efficient analysis for gene expression profiles.

Studying the characteristics of thousands of genes simultaneously offered a deep insight into cancer classification problem. in this paper we will walk through conducting a study to analyze gene expression (GE) data for the cancer type **Lung Squamous Cell Carcinoma (LUSC)**.

we have two data files for the GE data for this cancer type with two particular cases, the first is for affected tissues and the second is for healthy tissues. our objective is to find each gene correlation to determine how first samples is associated with the other samples, also we want to Infer the differentially expressed genes (DEGs); the genes whose expression level differ from one condition (healthy) to another (diseased).

2. Methods

packages used in the implementation are **pandas ,matplotlib, scipy and statsmodels**

2.1. Correlation

- we begin with loading the datasets into two dataframes: healthy and cancerous using read_csv from pandas package
- Filtration process: As the data provided will have missing and undefined values due to various problems, it is important to filter them before going to the next step to acquire accurate results but unfortunately we couldn't complete the processing due to technical issues in the programming part
- Acquiring the correlation coffiecnts: First we created a new data frame to store the gene id and both the pearson and spearman coefficient .then we will go through a loop on each row form the healthy and cancerous data frame to get the coefficients and store them in the new data frame.
- Ranking and sorting: After completing the loop, we will rank the data frame through the ".rank()"method then sort them according to the Spearman coefficient value
- Getting the max and min: After ranking and sorting, we get the max and min value of both Pearson and spearman coefficients by getting their id and symbol then getting their gene expressions from the original data frame
- Figure below represent the max and min pearson coefficients genes
-

	Gene_Hugo_Symbol	Entrez_Gene_Id
8433	GAGE2A	2574

	Gene_Hugo_Symbol	Entrez_Gene_Id
14706	FAM222B	55731

- Figure below represent the max and min spearman genes
-

	Gene_Hugo_Symbol	Entrez_Gene_Id
631	NPIPA8	101059938

	Gene_Hugo_Symbol	Entrez_Gene_Id
9207	TRIM15	89870

2.2. Hypothesis testing

for paired samples case:

- set Null Hypothesis (Ho): Samples are not paired
- set Alternative Hypothesis (Ha): Samples are paired
- we set out confidence level to be 99 perc.
- iterate across each gene row and compute it's p _value using ttest _rel Method from scipy.stats
- store the p _value in new csv file as before FDR correction
- check the p _value : if it's smaller than alpha then rejecting null hypothesis, thus we set is _paired variable to True in this gene
- if it's not, then accepting null hypothesis, thus we set is _paired variable to False in this gene
- we get the new csv file with each gene's p _value before FDR
- then we apply FDR correction to the p _values of each gene and recompare with alpha to fill a new variable is _paired after FDR correction
- the resulting csv file is the is _paired variable before and after FDR

for independent samples case:

- set Null Hypothesis (Ho): Samples are paired
- set Alternative Hypothesis (Ha): Samples are independent
- then redo all the steps listed in paired sample case

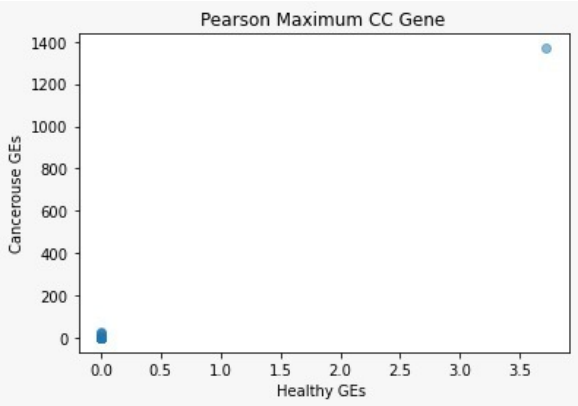
to get common and distinct independent and paired genes:

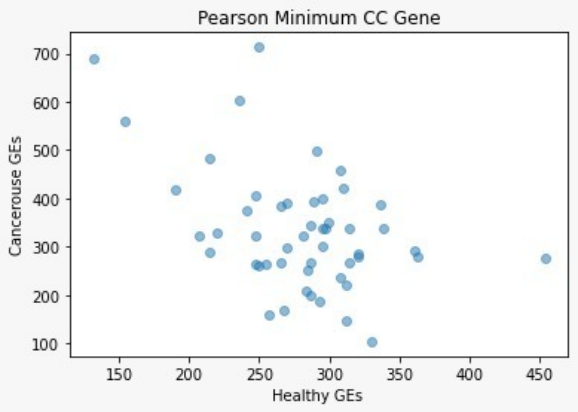
- we merge the paired genes dataframe and independent dataframe by merge method of pandas to get the common, and set is _common variable to True we merge the common dataframe with the independent dataframe and retrieve only the genes which it's is _common variable is not True, we get the distinct independent genes we merge the common dataframe with the paired dataframe and retrieve only the genes which it's is _common variable is not True, we get the distinct paired genes

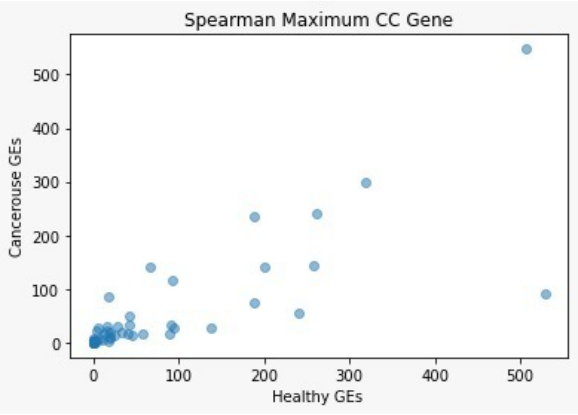
3. Results and Discussions

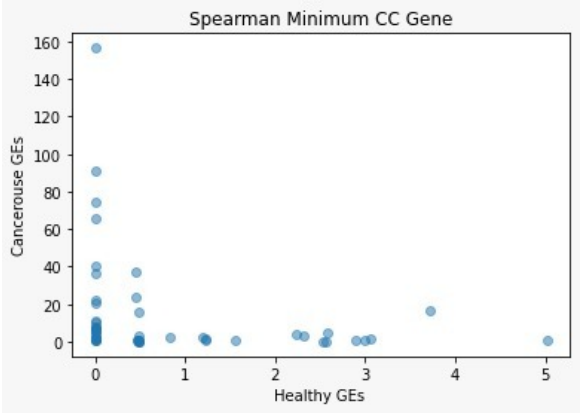
3.1. Correlation

In the plotting of the data we put the healthy gene expression on the x-axis and the cancerous gene expression on the y-axis

- 

- 

- 

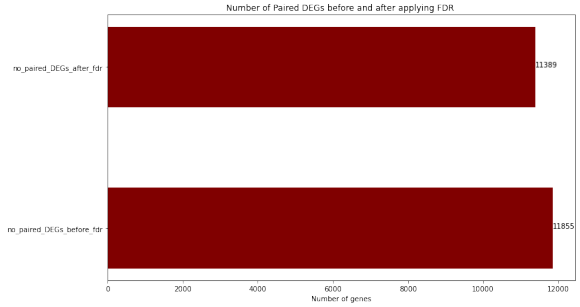
- 

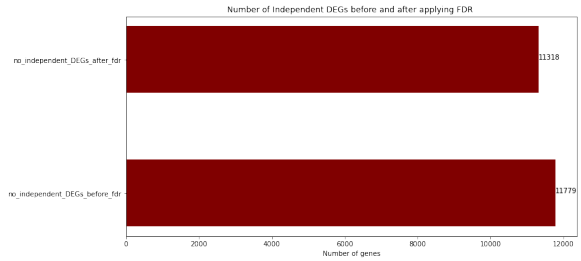
Pearson vs spearman coefficient:

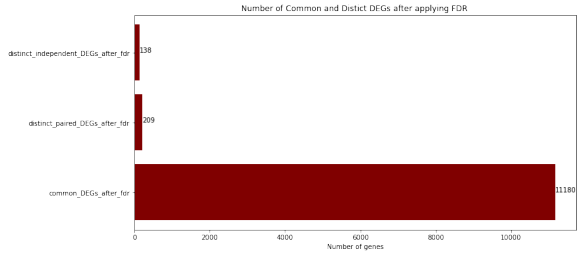
The Pearson correlation evaluates the linear relationship between two continuous variables. A relationship is linear when a change in one variable is associated with a proportional change in the other variable. So in the case of the Pearson correlation we are trying to find whether there is a relation between the change in the value of both the healthy and cancerous expression. The Spearman correlation evaluates the relationship between two continuous or ordinal variables. In a monotonic relationship, the variables tend to change together, but not necessarily at a constant rate. The Spearman correlation coefficient is based on the ranked values for each variable rather than the raw data. So the spearman correlation can be used to determine if the rank of the gene effects its change.

3.2. Hypothesis test

Setting our confidence level to be 99 perc. (alpha = 0.01) meaning that we are 99 perc. confident about the genes we found to be paired or independent or both. Our analysis leads to the following:

- **First result:**
- 

- We found the number of paired DEGs before we apply the FDR correction method to be 11,855 genes, while the number of paired DEGs after we apply FDR correction is 11,389 which corresponds to almost 60.3 perc. and 58 perc. of all the genes (19,648), respectively.
- This means that the FDR correction method helped us reject 457 genes that we thought paired DEGs which represent about 3.85 perc. of paired DEGs before we apply FDR.
- **Second result:**
- 

- We found the number of independent DEGs before we apply the FDR correction method to be 11,779 genes, while the number of independent DEGs after we apply FDR correction is 11,318 which corresponds to almost 59.95 perc. and 57.6 perc. of all the genes (19,648), respectively.
- This means that the FDR correction method helped us reject 461 genes that we thought were independent DEGs which represent about 3.91 perc. of independent DEGs before we apply FDR.
- **Third result:**
- 

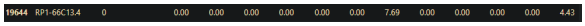
- We found that there are 11,180 happen to be paired and independent at the same time DEGs after applying the FDR correction method which corresponds to almost 56.9 perc. of all the genes (19,648) where the distinct paired DEGs and distinct independent DEGs found to be 209 and 138 respectively which corresponds to 1.06 perc. and 0.7 perc. respectively all the genes (19,648).

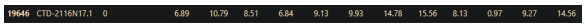
4. Conclusions

After observing the data we can see that a lot of genes tend to have liner relationship with its change according to Pearson correlation as the value of max Pearson was nearly 1 and according to spearman correlation we can see that the variable don't really have a strong monotonic relationship.

How to improve our results?

By preprocessing our data to:

- Drop rows (genes) that have zero value in the majority of its 50 GE columns
- 

- Drop rows (fake genes) that have zero ‘Enterz _Gene _Id ‘ and/or any weird ‘Hugo _Symbol‘
- 

These kinds of issues in the datasets could be cleaned using two approaches:

- Manually cleaning and droping the rows with the specified issues above. Which would be a hard and long job especially with this kind of long data. (almost 20K rows)
- Programmatically cleaning the data: This sounds like the nicer and gentle approach to take, unfortunately, this approach would cost us high Big O notation as we will iterate on the whole dataset gene by gene and then check the number of zeros in the 50 GE columns for each gene would produce Big O notation of O(n2) (nested for loop for example) So in this case we are talking about execution time of hours to apply this nested drop the rows (genes) with the issues.

5. Contributions

- Mohamed Khaled Galloul and Ahmed Mohamed contributed in the part of **hypothesis testing** code and report
- Moamen gamal and kareem mostafa contributed in the part of **Correlation** code and report

References