Introduction

In this assignment, you shall write a series of Apache Spark programs to analyze a *gene expression* data set that was collected to study the activity (the *expression*) of individual genes of patients who are treated for various diseases. The analysis is structured into three separate tasks.

Input Data Set Description

The input data can be found in the HDFS directory /share/genedata in our cluster. There are several subdirectories with csv files named GEO.txt (gene expression data) and PatientMetaData.txt.
All these files have a comma-delimited format.
The main data set is from DNA microarrays which measure the relative activity of certain genes ('gene expression') using cell samples from patients. The corresponding files are called GEO.txt and each line representing some gene expression value for some patient:
patientid, geneid, expression value

Note that there is a header row that just gives the meaning of each column:
- The patientid column specifies the ID of the patient whose cells were analysed.
- geneid is an ID of a specific gene whose activity (*expression*) level was measured.
- expression value represents a numerical value on how much a given gene is active in the patient's cells. For the purpose of this assignment, we simply consider a gene to be expressed (or active) if its level is above 1,250,000.

The meta-data for the patients are stored in the csv file PatientMetaData.txt Each line of this file contains a record with the following comma-delimited information:
id, age, gender, postcode, diseases, drug_response

The id column specifies a patient's ID, while the age and the postcode columns give the age of a patient and in which postcode they live. The gender column specifies a patient's gender ('f' for female patients and 'm' for male patients). In the diseases column, we find one or more disease types for all diseases that a patient is treated for (a patient can have more than one disease). The individual disease types are separated by space. The last column is the drug-response which specifies how well a patient is responding to his current drug treatment (higher is better).

Analysis Task Descriptions

1. **Task 1: Number of cancer patients with certain active genes *per cancer type***

The first task is an explorative analysis of the gene expression data similar to the previous assignments. You shall write a Flink program that determines how many cancer patients have a high expression level of gene 42. We consider a gene being (strongly) expressed if its expression value is larger than 1,250,000. A cancer patient is any patient who is treated for either breast-cancer , prostate-cancer, pancreatic-cancer, leukemia, or lymphoma.
The output file should have the following format, ordered by number of expressed genes in descending order (cancer types with most gene-42-patients first); cancer types with the same number of occurrences of gene 42 should be listed alphabetically:
cancer_type \t number_of_patients_with_gene42_expressed

2. **Task 2: Frequent Itemset Mining**

In the second task, we are interested in common combinations of genes expressed with cancer patients. To do so, you shall implement the first part of the iterative Apriori algorithm for

frequent itemset mining to identify the frequent combinations (*frequent itemsets*) on our gene expression data.
We are again only interested in gene expressions from cancer patients. A cancer patient is any patient who is treated for either breast-cancer, prostate-cancer, pancreatic-cancer, leukemia, or lymphoma. Only consider strongly expressed genes, ie. such genes which have an expression value above 1,250,000.
Determine all frequent itemsets of expressed genes. A frequent itemset is any combination of expressed genes with a *support count* (number of occurrences) at least a minimum support count. This minimum support should be a program parameter. By default, check for itemsets with a support value of 30% of all patient samples. You can also define a maximum itemset size as a program parameter, which would allow you to limit the number of iterations.
The output file should list the frequent itemsets in the following tab-delimited format, ordered by descending support, and with each itemset ordered by increasing gene-ID:
support \t $item_1$ \t $item_2$ \t ... \t $item_k$

3. **Task 3: Association Rule Generation**

In the third task, you shall generate *association rules* from the frequent itemsets that you found in previous Task 2: For every frequent itemset S of size 2 or larger, that was identified in Task 2, generate all non-empty subsets R. For every such subset R output a rule of the form R => (S - R) if the *confidence* value for this rule is at least a *confidence threshold*. This confidence threshold should be a parameter of your program, with a default value of 60%.
The confidence of a rule is defined as: confidence = $\text{support}(S) / \text{support}(R)$.
The output should have the following format (ordered by descending confidence value):
R \t (S - R) \t confidence_value
where R and (S - R) are the items from the right-hand respectively left-hand side of an association rule, separated by space.