Data Mining
Project

Predict disease classes using genetic microarray data

Objective

The purpose of this project is to develop a method that uses genetic data for disease classification. Data is extracted from a DNA microarray which measures the expression levels of large numbers of genes simultaneously.

Samples in the datasets represent patients. For each patient 7070 genes expressions (values) are measured in order to classify the patient's disease into one of the following cases: EPD, JPA, MED, MGL, RHB.

Data

Gene data is in genes-in-rows format, comma-separated values. You will find on Moodle a Zip file named: final_project_data.zip file. Unzip to extract the following 3 files:

- Training dataset: pp5i_train.gr.csv
- Training data classes: pp5i_train_class.txt
- Test dataset: pp5i_test.gr.csv

Instructions

Training data: file pp5i_train.gr.csv, with 7070 genes for 69 samples. A separate file pp5i_train_class.txt has classes for each sample, in the order corresponding to the order of samples in pp5i_train.gr.csv.

Test data: file pp5i_test.gr.csv, with 23 **unlabelled** samples and same genes. You can assume that the class distribution is similar.

Your goal is to learn the best model from the training data and use it to predict the label (class) for each sample in test data. You will also need to write a paper describing your effort.

Randomization experiments showed that one can get about 10-12 (from 23) correct answers with random guessing.

Below are suggested steps for doing this experiment, but you can vary and improve on the suggested approach, as long as you produce a prediction for the test set and describe your results.

Important Hints

Be sure that you don't use the sample number as one of the predictors. Training data is ordered by class, so sample number will appear to be a good predictor on cross-validation, but it will not work on the test data!

One of the MED samples in the training data is very likely misclassified (by a human). So the best result you can expect to get on cross validation is one error (on a MED sample) out of 69. However, this should not affect your accuracy on the test set.

You can do all your experiments on Weka (https://waikato.github.io/weka-wiki/downloading_weka/). This is a software that integrates most of the mining and visualization tools needed for this project. The alternative would be to use Python and its libraries. Some steps in data cleaning will also require a spreadsheet software. Please note that these are the only development tools allowed for this project.

You can complete the project using only simple steps, but the more advanced steps will give you extra credit and probably higher accuracy.

The following steps suggest one way of finding the best model -- you are welcome to make improvements, where you think appropriate.

Step 1. Data Cleaning

Threshold both train and test data to a minimum value of 20, maximum of 16,000.

Step 2. Selecting top genes by class

- Remove from train data genes with fold differences across samples less than 2. Fold difference is defined as a ratio between maximum and minimum values (Max/Min) for a given data set.
- For each class, generate subsets with top 2,4,6,8,10,12,15,20,25, and 30 top genes with the highest absolute T-value. (See: https://www.biologyforlife.com/t-test.html) . The objective of this step is to find for each class the set of best genes to discriminate it from the other classes.
- For N=2,4,6,8,10,12,15,20,25,30 combine top genes for each class into one file (removing duplicates, if any) and call the resulting file pp5i_train.topN.gr.csv
- Add the class as the last column, remove sample no, transpose each file to "genes-in-columns" format.

Step 3. Find the best classifier/best gene set combination

- Use the following classifiers:
  - Naïve Bayes
  - Decision tree (J48 algorithm)
  - K-NN (IBk algorithm)
  - A neural network
  - One more classifier of your choice.
- For each classifier, using default settings, measure classifier accuracy on the training set using previously generated files with top N=2, 4, 6, 8, 10, 12, 15, 20, 25, 30 genes. For K-NN, test accuracy with K=2, 3 and 4.
- Select the model and the gene set with the lowest cross-validation error. Cross-validation is your main tool to measure classification accuracy. It is already built in Weka. If you are using Python, search for resources that show how it can be calculated. (e.g. https://scikit-learn.org/stable/modules/cross_validation.html).
- Once you found the gene set with the lowest cross-validation error, you can vary 1-2 additional relevant parameters for each classifier to see if the accuracy will improve.
- Use the gene names from best train gene set and extract the data corresponding to these genes from the test set.
- Convert test set to genes-in-columns format to prepare it for classification.

Step 4. Generate predictions for the test set

- You should now have the best train file, call it pp5i_train.bestN.csv, (with 69 samples and bestN number of genes for whatever best N you found) and a corresponding test file, call it pp5i_test.bestN.csv, with the same genes and 23 test samples.
- Use the best train file and the matching test file and generate predictions for the test file class.

Step 5. Write a paper describing your effort.

- Document each step.
- For each classifier used, give a paragraph describing this classifier. Give a graph showing error rate versus number of genes. Indicate which algorithm you are using in Weka or which library you have used if you opt for Python.
- Describe which classifier and which number of genes you have selected.
- Comment on the relative strengths and weaknesses of the classifiers you used for this type of data.