**Machine Learning**

**Assignment 4**

Or Kandabi 313321507

Elad Solomon 205955719

**Chapter 1**

**Algorithm Name**

Algorithm FSS – feature subset selection

**Reference**

New feature subset selection procedures for classification of expression profiles (part A)

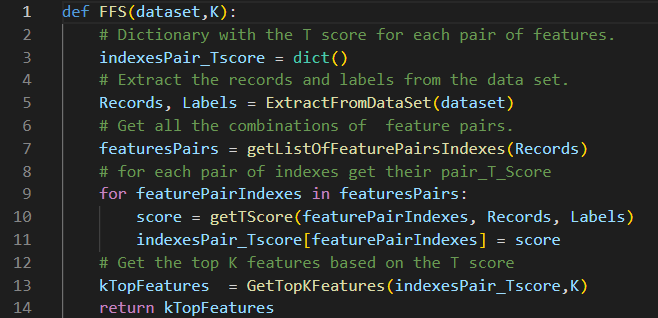
**Motivation for the algorithm**

Methods for extracting useful information from the datasets produced by microarray experiments are at present of much interest. Here we present new methods for finding gene sets that are well suited for distinguishing experiment classes, such as healthy versus diseased tissues. The methods are based on evaluating genes in pairs and evaluating how well a pair in combination distinguishes two experiment classes. The paper shows the ability of the pair-based methods to select gene sets that generalize the differences between experiment classes and compared the performance relative to two standard methods.

**Short Description**

Here we investigate new FSS methods that analyze pairs of genes, making it possible to find pairs that distinguish well between sample classes. Additionally we investigate the so-called forward selection method for FSS, where a good feature set is constructed by a 'greedy' selection procedure.

**Pseudo-Code**

****

**Algorithm explanations**

Parameters:

dataset – The data set contain the records and labels.

K – Top K feature to extract.

Algorithm Flow:

The algorithm is conducted in three stages.

First step:

We will create a dictionary containing all combinations of possible pairs of the features.

For example: if there are 3 features marked with 1,2,3,4 we will create the following pairs:

[1,2] [1,3] [1,4] [2,4] [2,3] [3,4]

Second step:

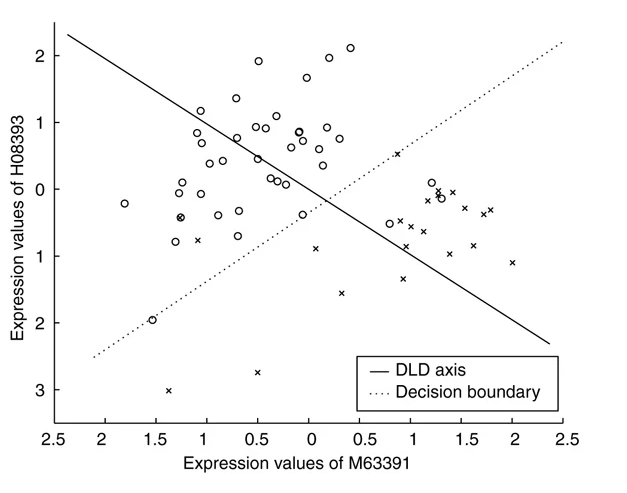
We will go through each pair and give it a T-score.

The score will be determined by how well they separate the two classes.

We did this by using SVC, which generates a linear line that separates the samples according to their class and gives a score according to how well the linear line divides the points.

Third step:

Selecting the K Best feature according to their T-Score.



**Illustration**

Given the dataset (we generate random values for the example):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Gen1 | Gen2 | Gen3 | Gen4 | Gen5 | class |
| 2 | 1 | 2 | 2 | 1 | 0 |
| 2 | 2 | 1 | 1 | 2 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

We want to extract the best k features where K=2

First step

Num of features =5, we create all permutation of pairs:

[1,2], [1,3], [1,4], [1,5], [2,3], [2,4], [2,5], [3,4], [3,5], [4,5]

Second step:

Calculate the t\_score for each pair:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Pair | [1,2] | [1,3] | [1,4] | [1,5] | [2,3] | [2,4] | [2,5] | [3,4] | [3,5] | [4,5] |
| t\_score | 0.3 | 0.4 | 0.21 | 0.45 | 0.23 | 0.54 | 0.8 | 0.3 | 0.44 | 0.78 |

Third step:

Get the best k features with the highest t\_score: in our case k=2, so the features that will be chosen are Gen2, Gen5.

**Taxonomy**

Search strategy – Wrapper:

Because it’s based on a specific machine learning algorithm. High computation time for data set with a lot of features.

**Strengths**

* The algorithm is based on the score of pair of features instead of each feature alone, that approach was proven to be more accurate.
* Check all the pair combinations in the data set.
* Simplicity.

**Drawbacks**

* Run time – Going over each pair can take a long time, for example, if there are 1000 features the algorithm need to check 500 500 pairs.
* 3000 features have become 4,501,500 pairs!
* Based on only one specific machine learning algorithm.
* Over Fitting – The dependency to use only one algorithm can cause overfitting.

**Experimental Results**

all the results can be seen in the Results.csv file that contains the output of the algorithms with the following fields:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset Name | Number of samples | Original Number of features | Filtering Algorithm | Learning algorithm | Number of features selected (K) | CV Method | Fold | Measure Type | Measure Value | List of Selected Features Names (Long STRING) | Selected Features scores |

**Conclusions**

* We can see that the algorithm provides us strong and steady results when we examine the accuracy level of each table.
* We can see that for some tables the algorithm provides us small number of features that gets high accuracy in terms of predictions.
* The algorithm though may have struggled with data sets with large number of features, since it examines each permutation of features pairs. The runtime of the algorithm can increase drastically so we had to filter the larger DB with the f\_classif FS method.

**Chapter 2**

**Algorithm Name**

Algorithm AHFS – Adaptive Hybrid Feature Selection

**Reference**

Viharos, Zsolt János, et al. "Adaptive, hybrid feature selection (AHFS)." Pattern Recognition 116 (2021) (Part B)

**Motivation for the algorithm**

Most real-world modeling problems can be formulated as estimation of some numerical value or classifying a given number of samples. These problems, more often than not, are very complex and can be defined by tens, hundreds and even thousands of variables. The [high dimensional data](https://www.sciencedirect.com/topics/computer-science/high-dimensional-data) is hard to handle by [soft computing methods](https://www.sciencedirect.com/topics/computer-science/soft-computing-method), but in most of the cases many variables are highly redundant, noisy and/or irrelevant when solving a specific estimation or [classification task](https://www.sciencedirect.com/topics/engineering/classification-task). The need arises to reduce the dimension of a problem by selecting only the relevant features for a given assignment with relatively fast methods compared to the highly accurate but costly estimation models.

**Description**

This paper deals with the problem of integrating the most suitable feature selection methods for a given problem in order to achieve the best feature order. A new, adaptive and hybrid feature selection approach is proposed, which combines and utilizes multiple individual methods in order to achieve a more generalized solution. Various state-of-the-art feature selection methods are presented in detail with examples of their applications and an exhaustive evaluation is conducted to measure and compare their performance with the proposed approach. Results prove that while the individual feature selection methods may perform with high variety on the test cases, the combined algorithm steadily provides noticeably better solution.

**Pseudo-Code**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**

**Algorithm explanations**

Parameters:

dataset – The data set contain the records and labels.

K – Top K feature to extract.

Models – List of models who are in charge to choose the best feature.

Algorithm Flow:

התרגום ארוך מדי לשמירה

The algorithm is conducted in three stages.

First step:

The first step is the feature selection part, which iterates over the given model’s algorithms, invoking them with the already selected feature set S.

Every algorithm proposes one feature as the next potential best feature to be selected according to its own measure.

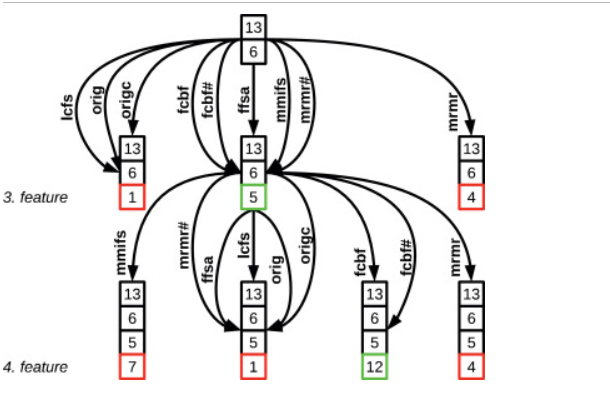
Second level:

We derived an evaluation for all the options given, and we chose the list of features that received the highest score, we used SVC for the evaluation, but this is dynamic, and we can choose which model we want.

Third phase:

We will return to the first step with the current list selected.

Example:



In the beginning, the selected features are 13 and 6.

After that, each algorithm chooses its best feature.

There are 3 options now: [13,6,1] [13,6,5] [13,6,4]

We will evaluate the score that each feature subset can get.

The beast feature subset is [13,6,5] so we will choose to proceed with those features and keep repeating this procedure until the number of features will be equal to K.

**Illustration**

Given the dataset (we generate random values for the example):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Gen1 | Gen2 | Gen3 | Gen4 | Gen5 | class |
| 2 | 1 | 2 | 2 | 1 | 0 |
| 2 | 2 | 1 | 1 | 2 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

We want to extract the best k features where K=2

**Iteration 1**

First step

We will get the best features according to 3 different FS algorithms:

RandomForestRegressor = Gen3

RandomForestClassifier = Gen4

ExtraTreesClassifier = Gen3

Second step:

We will evaluate the score of each feature with a chosen model evaluation.

|  |  |  |
| --- | --- | --- |
| Feature | Gen3 | Gen4 |
| score | 0.8 | 0.2 |

**The feature selected in iteration 1 is Gen3**

**Iteration 2**

First step

We will get the best features according to 3 different FS algorithms without the selected features in the previous iterations (Gen3):

RandomForestRegressor = Gen2

RandomForestClassifier = Gen4

ExtraTreesClassifier = Gen1

Second step:

We will evaluate the score of each feature with a chosen model evaluation including the chosen features from the previous iterations:

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Gen3,Gen1 | Gen3,Gen2 | Gen3,Gen4 |
| score | 0.3 | 0.6 | 0.14 |

The feature selected in iteration 2 is Gen2 that will be added to the other chosen features (Gen3)

**The algorithm selected features will be [Gen2, Gen3]**

**Taxonomy**

Search strategy – Embedded method:

Because each feature selection is separated by iterations. The time complexity is better than the wrapper method, which uses several machine learning algorithms.

**Strengths**

* The algorithm is based on multiple algorithms and can take advantage of each one.
* Modularity- We can choose the model for evaluation and feature selections.
* Runtime – The max of Evaluation is: (NumOfModels X Knumber)
* Simplicity.

**Drawbacks**

* Greedy search approach – Always choosing the best feature can take some time.
* Based on the model’s algorithms – If the models are similar can cause overfitting and lose the advantage of using several models.

**Experimental Results**

all the results can be seen in the Results.csv file that contains the output of the algorithms with the following fields:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset Name | Number of samples | Original Number of features | Filtering Algorithm | Learning algorithm | Number of features selected (K) | CV Method | Fold | Measure Type | Measure Value | List of Selected Features Names (Long STRING) | Selected Features scores |

**Conclusions**

* We can see that the algorithm provides us strong and steady results when we examine the accuracy level of each table.
* The runtime of the algorithm better than the first algorithm and pretty close to the compared algorithm that we used.

**Toy example Iterations**

We run each algorithm 3 times with 3 different K – [5,10,20]

Here are the following results

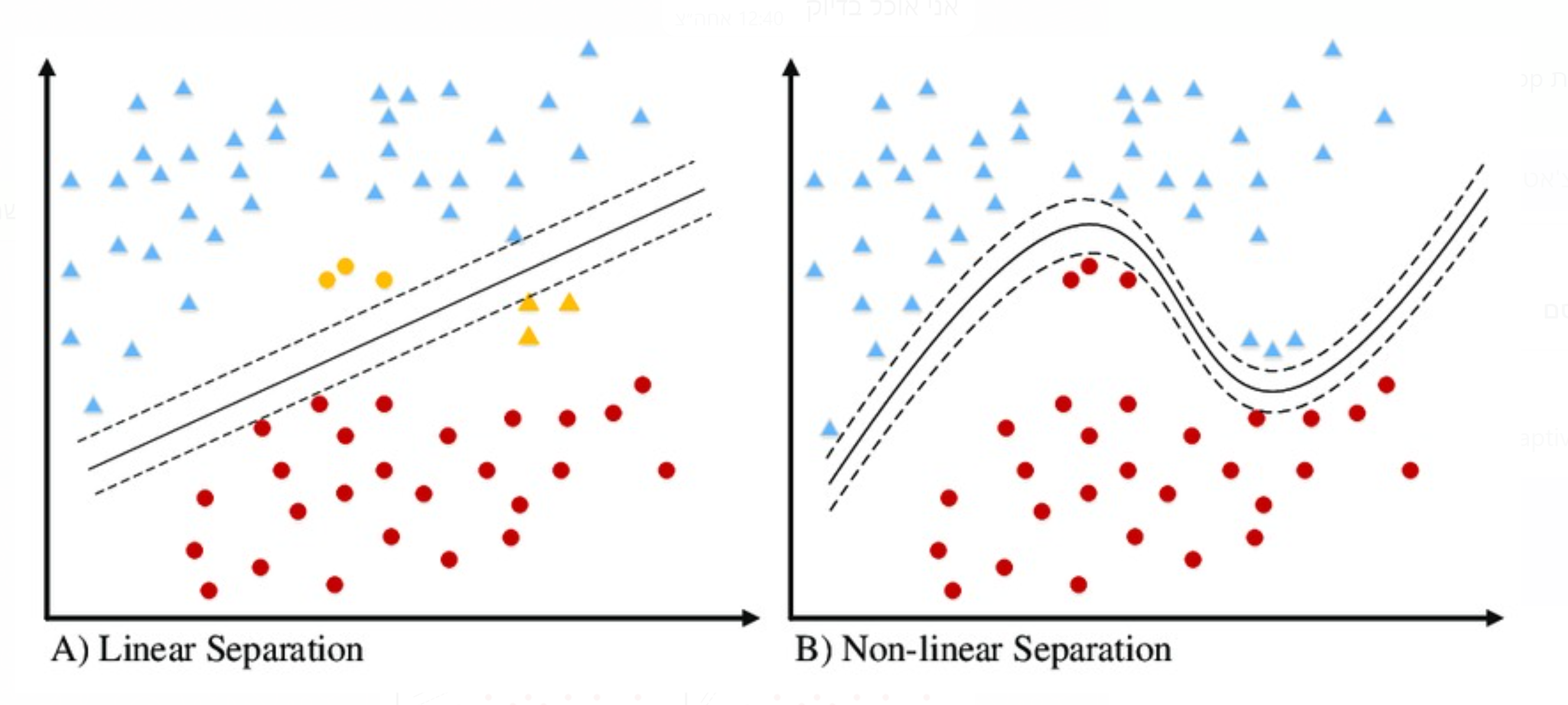
|  |  |  |
| --- | --- | --- |
| **Algorithm** | **K** | **Selected Features Indexes** |
| **FSS (Algorithm 1)** | 5 | 29, 30, 38, 39, 24 |
| **FSS (Algorithm 1)** | 10 | 29, 30, 38, 39, 24, 25, 27, 28, 8, 9 |
| **FSS (Algorithm 1)** | 20 | 29, 30, 38, 39, 24, 25, 27, 28, 8, 9, 21, 22, 31, 32, 41, 42, 1, 2, 14, 15 |
|  |  |  |
| **AHFS (Algorithm 2)** | 5 | 39, 29, 25, 3, 20 |
| **AHFS (Algorithm 2)** | 10 | 39, 29, 25, 3, 20, 34, 24, 7, 33, 6 |
| **AHFS (Algorithm 2)** | 20 | 39, 29, 25, 3, 20, 24, 35, 2, 35, 6, 5, 19, 31, 30, 29, 3, 10, 16, 19, 11 |

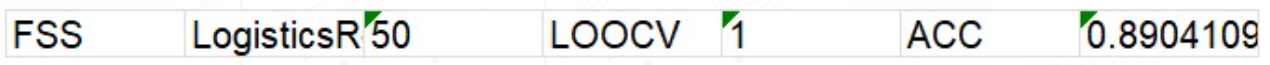
**Chapter 3**

**Improvement of FSS algorithm**

The FSS algorithm making feature selection by calculating for each pair of feature its pair t\_score. The t\_score is calculating by examine the affect and prediction probability for the right classification. This is done with isolation of the classification with a straight line and making two groups of classification.

The improvement method can be done by isolate the two classifications with a non-straight line- polynomial method. With this approach we can see a better accuracy of the real classifications:



For example, in the dataset "lung\_small" we can see an improvement with the ACC score.

****

This result shows us that most likely the samples was separated in a way the polemically line can separate them in more accuracy way then a linear line

**Chapter 4**

**Conclusions**

We extracted the best configuration for each dataset that provides the best AUC value:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Data Base Name** | **Filtering algorithem** | **Num of features** | **Learning algorithem** | **CV Type** | **Measure Type** | **Measure Value** | **Aug\_Algorithm** |
| Colon1 | FSS | 4 | SVM | LOOCV | ACC | 0.93 | 0.54 |
| DLBCL | AHFS | 20 | Random forest | 10Folds | AUC | 0.71 | 0.89 |
| Isolate | MRMR | 3 | NB | 5Folds | AUC | 0.65 | 0.54 |
| Lung small | MRMR | 20 | Logistics Regression | LOOCV | ACC | 0.93 | 0.93 |
| NCI60 | AHFS | 100 | SVM | LOOCV | ACC | 0.8 | 0.78 |
| Nutt-2003-v2\_BrainCancer | MRMR | 10 | SVM | LPO | ACC | 0.99 | 0.82 |
| Orli1 | FSS | 4 | Logistics Regression | 10Folds | AUC | 0.67 | 0.51 |
| pone.0246039.s001 | FSS | 2 | SVM | LOOCV | AUC | 0.98 | 0.54 |
| Risinger\_Endometrial Cancer | RFE | 3 | SVM | LPO | AUC | 0.95 | 0.9 |
| USPS | FSS | 1 | Logistics Regression | 5Folds | AUC | 0.72 | 0.55 |
| warpAR10P1 | relief F | 5 | Logistics Regression | 10Folds | AUC | 0.71 | 0.47 |
| alon\_inputs | FSS | 15 | Random forest | LOOCV | ACC | 0.91 | 0.61 |
| christensen\_inputs | AHFS | 10 | SVM | 10Folds | AUC | 1 | 0.92 |
| COIL20 | FSS | 1 | Logistics Regression | 5Folds | AUC | 0.67 | 0.74 |
| curatedOvarianData | FSS | 4 | Random forest | 10Folds | AUC | 0.92 | 0.875 |
| pone.0202167.s013 | FSS | 4 | SVM | LOOCV | ACC | 0.93 | 0.85 |
| sorlie\_inputs | MRMR | 100 | SVM | LOOCV | ACC | 0.92 | 0.83 |
| Singh-2002\_ProstateCancer | MRMR | 15 | NB | 10Folds | ACC | 0.95 | 0.91 |
| madelon | FSS | 30 | K\_neighbors | 5Folds | ACC | 0.76 | 0.48 |
| Yale1 | FSS | 4 | Logistics Regression | 10Folds | AUC | 0.84 | 0.7 |

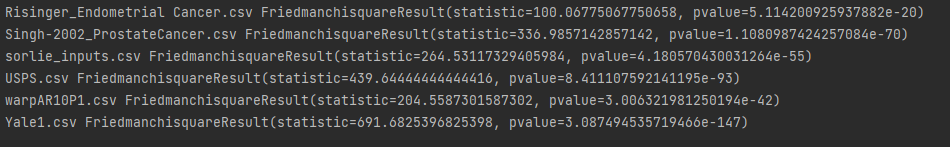
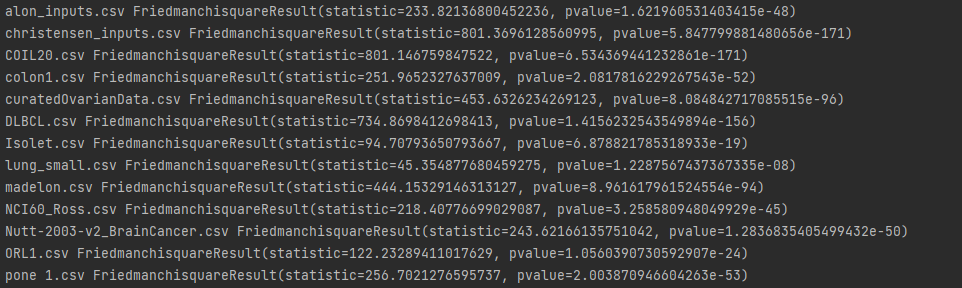
* We Can see that for most of the datasets, the FSS algorithm provides us the best results with the least number of features selected.
* We come to conclusions that the Augmentation methods does not provide us better results – all of the accuracy scores were affected.
* At some rare cases we can see that there are datasets that one feature can predict the records classification with the best probability.

**Chapter 5**

**Friedman Test**

For performing the Friedmantest on each dataset according to the ACC we extracted for each algorithm its ACC value that was calculated in the 1 part of the assignment. The ACC score where in an array and divided by the FS algorithm.

The results are divided for each DB:



**Chapter 6**

**LOOCV – underestimate**

Q: Why LOOCV may be underestimated?

How does LOOCV work?

1. Split the data set into training and test sets, where the test set includes only one observation.
2. Train the model on the train set and evaluate the test set.
3. Repeat the process n times when n is the data set size.

Now let’s look at the following example:

Data set with 100 observations, with 50 observations identified with 0 and 50 observations identified with 1.

Now we will use a dummy classifier that uses frequency as our model.

Each time when we will split the data to train and test, we will have in the train test 50 observation of Type X and 49 observations of type Y while the test observation is classed as Y.

Because we used a frequency dummy classifier, we will predict the observation to be X (because most samples are classified as X), and this is wrong.

When we will repeat this procedure, we will be wrong each time of evaluations, and we will receive acc of 0%!

Q: How Leave-Pair-Out can handle the issue mentioned above?

The difference between LOOCV to LPO is that LPO is choosing each time pair of observations.

In this case, we will have multiple splits that the pair that choose is included both classes X, and Y. Therefore, the train set will stay equal (49 X, 49 Y) and in this case, the LOP solves the problem that mention above.