HYBRID FEATURE SELECTION METHOD FOR HEALTH DATA MINING

By

FAISAL AHMED TUSAR
(151-35-867)

A thesis submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Department of Software Engineering
DAFFODIL INTERNATIONAL UNIVERSITY

Spring – 2019

APPROVAL

This thesis titled on "**Hybrid Feature Selection Method for Health Data Mining**", submitted by **Faisal Ahmed Tusar**, **151-35-867** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

--- **Prof. Dr. Touhid Bhuiyan** **Professor and Head** Department of Software Engineering Faculty of Science and Information Technology Daffodil International University	**Chairman**
--- **Name of Internal Examiner** **Designation** Department of Software Engineering Faculty of Science and Information Technology Daffodil International University	**Internal Examiner 1**
--- **Name of Internal Examiner 2** **Designation** Department of Software Engineering Faculty of Science and Information Technology Daffodil International University	**Internal Examiner 2**
--- **Name of External Examiner** **Designation** Name of the Department Name of the University	**External Examiner**

DECLARATION

It hereby declere that this thesis has been done by **me** under the supervission of **Khandker M. Qaiduzzaman, Lecturer,** Department of Software Engineering, Daffodil International University. It also declere that nithor this thesis nor any part of this has been submitted elesewhere for award of any degree.

Faisal Ahmed Tusar
ID: 151-35-867
Batch: 16th
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

Certified by:

Khandker M. Qaiduzzaman
Lecturer
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

ACKNOWLEDGEMENT

Foremost, I would like to express my sincere gratitude to my advisor Khandker M. Qaiduzzaman for the continuous support of my research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my research.

Besides my advisor, I would like to thank the rest of my thesis committee, for their encouragement, insightful comments, and hard questions.

Last but not the least, I would like to thank my family: my parents, for giving birth to me at the first place and supporting me spiritually throughout my life.

TABLE OF CONTANT

LIST OF TABLE

LIST OF FIGURE

ABSTRACT

Background: Feature selection is one of the most important parts of machine learning for predicting the outcome. There are methods for selecting features or generating feature subset. Such as: Filter method, wrapper method. Previously features were selected using any one of these two methods. The result from the method was pretty good but it could be better.

Objective: The objective of my thesis is to get the more accurate result. Here I am emphasizing on feature selection for getting the more accurate results. There is another method for feature selection, which is: hybrid method. Hybrid method combines both the filter and wrapper methods. Here I am going to use the hybrid method for selecting features and I will show that hybrid method can get the same or more accurate result using less features.

Results: The final result is showing that, in some cases hybrid method is giving the same results as filter and wrapper method. Some other cases show that hybrid method is giving more accurate result that filter and wrapper method. In all the cases hybrid method is using less features than filter and wrapper method. Here filter and wrapper method is using four features each and hybrid method is three features.

Conclusions: Here in my thesis, I have explained all the details of the method I am using and also showed some analytical study from which it could be decided that my proposed method is giving more accurate result than other methods.

CHAPTER 1

INTRODUCTION

1.1 Background

Machine Learning (ML) is a one kind of algorithm which helps predicting the more accurate outcome without being programmed explicitly. That means, to develop such kind of algorithm which will take some data as input and after some analysis it will predict an output [1].

Feature selection is the process of reducing data dimensionality and improving the mining performance as much as possible [3]. It can be used in data pre-processing to achieve efficient data reduction by reducing of the size of the dataset in order to achieve more efficient analysis and finding accurate data models for adaptation of the dataset to best suit the selected analysis method [2]. Here the aim is to determine a subset of features and the subset should be as small as possible. This feature selection process selects the subset of original features and it does not loss any information that is necessary, removes the irrelevant and redundant features. As a result it reduces the data dimensionality, computation time and gets a better result comparatively [3].

Feature selection methods can be classified into three categories. Such as:

i. Filter Method,
ii. Wrapper Method,
iii. Hybrid Method.

Filter method selects the subset of the features based on the uniqueness of features. It can be applied to the data which has high dimensionality. To use wrapper method, a predefined algorithm is required which will select the best feature subset. The predictive accuracy of that algorithm is used for evaluation. Although the computational time of this method is comparatively high but it gives the guarantee of getting better result. Finally the hybrid method is the combination of filter method and wrapper method. By applying the hybrid method the advantages of both the method can be achieved [3].

In this paper, I am using ReliefF algorithm in filter method to remove irrelevant features and Chi Squire, Recursive Feature Elimination, Extra Tree Classifier, Sequential Floating Forward Selection (SFFS) algorithms in wrapper method for selecting features based on their accuracy.

Here I am focusing on feature selection by using the hybrid method. That means I am using both the filter and wrapper methods for selecting the features and by using the hybrid method I am showing that hybrid method gives better result than filter or wrapper method.

1.2 Motivation of the Research

Feature selection is one of the most important parts in machine learning because the final accuracy result depends a lot on this feature selection. If proper features are not selected then a good accuracy result can never be expected. On the other hand, if appropriate feature are selected then good accuracy can be achieved using less features.

Here pre-processing of the dataset is a complicated as well as an important task. That means, finding the unwanted values in the dataset and replace those values with a proper value. After that applying all the feature selection algorithms and fit all the data in those algorithms is quite challenging as different algorithms got different types parameters.

1.3 Problem Statement

- Previously they used only one method, either filter or wrapper. That's why the feature selection was not accurate all the time,
- As less accurate features gives less accurate results, so the given result was not accurate all the time,
- Sometimes they also used more features to get better results, which would increase the data redundancy.
- Even if sometimes they have got better result using only one method but the process was really slow.

1.4 Research Objectives

As feature selection is one of the most important part in machine learning, so the main aim and objective of my approach is to get the most appropriate feature subset and using the subset to get the most possible accurate result. I will achieve this using the hybrid method. The main objective could be reached by defining the sub-objectives:

- Giving priority to the feature selection. Because if I can select the appropriate features then I can get better results from those features.
- To get accurate features I have to remove irrelevant features from data set.
- Finally I have to cluster the similar features for feature selection.

1.5 Orientation of the Thesis

This thesis has been organized into six chapters. Each chapter gives distinct concept.

Chapter 1 (Introduction): Introduction of my research area has been explored in this section.

Chapter 2 (Literature Survey): This chapter presents the basic concept of feature selection.

Chapter 3 (Research Methodology): This chapter discusses about the methodology of how I implement the full process.

Chapter 4 (Results & Discussion): This chapter describes the experimental results & comparison between different feature selection methods and algorithms.

Chapter 5 (Conclusions & Recommendations): Summarization of my research work.

CHAPTER 2

LITERATURE REVIEW

2.1 Feature Selection in Machine Learning

Feature selection in machine learning plays a vital role, because if we can not select accurate features then we can not expect better results from those features. So feature selection is the part where we have to be more concerned and we have to put emphasis on it as well.

There are four key steps of feature selection process. Such as:

i. Feature subset generation,
ii. Subset Evaluation,
iii. Stopping Criteria,
iv. Result Validation.

Feature subset generation uses searching strategy like complete, sequential and random search to generate feature subset. It also follows the heuristic search process which results in the selection of a candidate subset for evaluation.

We can evaluate the goodness of generated subset using an evaluation criterion. Here we have to check which subset is better, the newly generated one or the previous one. If the newly generated subset gives better result than the previous one, it will replace the previous subset with the new one. These two processes are repeated until the stopping criterion is reached. The final best feature subset could be validated by using different tests. Figure 1 illustrates the feature selection process [3].

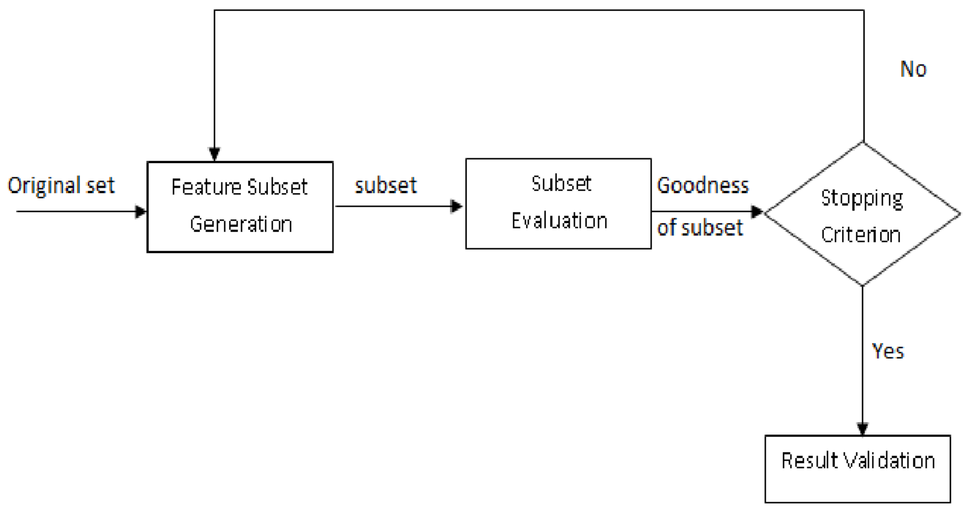

Figure 1: Feature Selection Process

2.2 Advantages of Feature Selection

Advantages of feature selection are pointed below:

- Irrelevant features could be removed using feature selection process,
- The number of redundant features could be reduced using feature selection process,
- All the similar features could clustered and shown using feature selection process,
- It can also improve the performance of machine learning algorithms,
- It reduces the training time,
- Finally it helps to get the more accurate result.

2.3 Feature Selection Methods

Feature selection methods could be classified into three categories on the basis of selection strategy [4]. Such as:

i. Filter Method,
ii. Wrapper Method,
iii. Hybrid Method.

The generality and high computation efficiency is one of the big advantages of filter method. It can reduce the high dimensionality of the data. It selects the feature based on intrinsic characteristics and independent of mining algorithm.

Wrapper method is different than filter method. To use wrapper method a predefined algorithm is required which will determine the best feature subset and this best feature subset is generated based on the predictive accuracy of the algorithm. It can give us the better result and that's a guarantee but the computational time for large dataset is slightly expensive [5].

Hybrid method is the combination of both the filter and wrapper method. That's why we can get the advantages of both algorithms from it. It can measure the goodness or accuracy of newly generated subset by applying an independent measure and mining algorithm [7]. Here, filter method is applied to remove irrelevant features and reduce search space then wrapper method is applied to get the best feature subset [6]. Figure 2 illustrates the hybrid model [3].

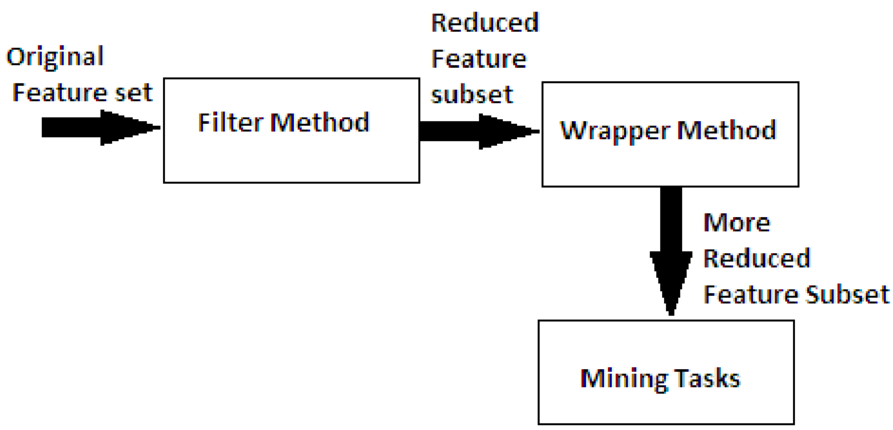

Figure 2: Hybrid Model

2.4 Filter Method

Filter method is used to reduce the irrelevant features and reduce the search space. It can reduce the high dimensionality of the data as well. It selects the features according to their performance measure regardless of the employed data modeling algorithm. Modeling algorithm can be applied only after the best features are found.

Using filter method we can create a ranking based on features accuracy [2]. Figure 3 illustrates the flow chart of filter method.

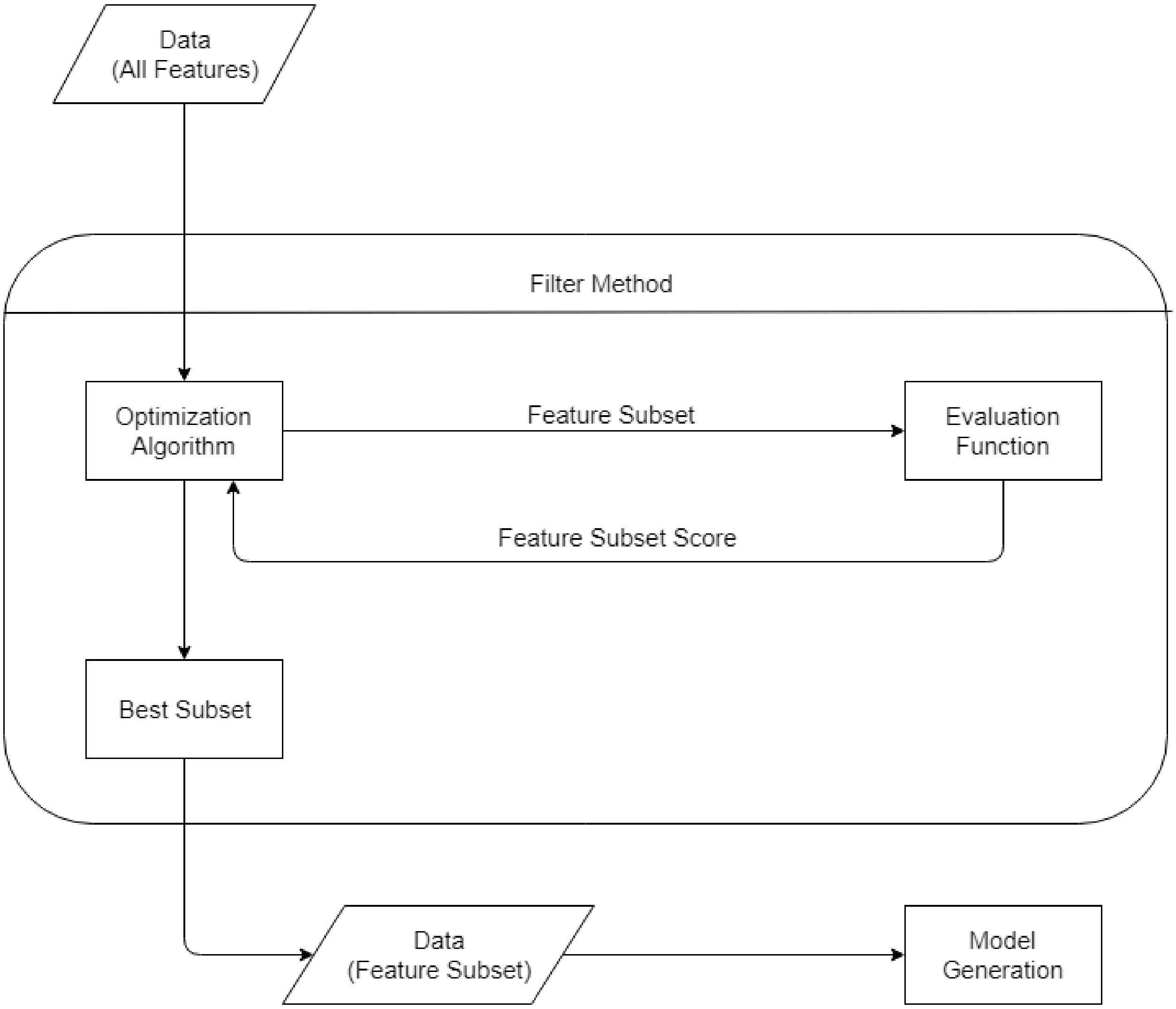

Figure 3: Flow Chat of Filter method

While there are many filter methods but not all the filter methods can be used for all classes of data mining tasks. Therefore, I am using ReliefF algorithm for the filter method. Figure 4 illustrates the pseudo code of ReliefF algorithm.

Algorithm ReliefF

Input: for each training instance a vector of attribute values and the class value

Output: the vector W of estimations of the qualities of attributes

set all weights $W[A] := 0.0$;
for i := **1 to** m **do begin**
    randomly select an instance R_i;
    find k nearest hits H_j;
    for each class $C \neq class(R_i)$ **do**
        from class C find k nearest misses $M_j(C)$;
    for A := 1 **to** a **do**
    $W[A] := W[A] - \sum_{j=1}^{k} \mathrm{diff}(A, R_i, H_j)/(m \cdot k) +$
$\sum_{C \neq class(R_i)} \left[\frac{P(C)}{1-P(class(R_i))} \sum_{j=1}^{k} \mathrm{diff}(A, R_i, M_j(C))\right]/(m \cdot k)$;
end;

Figure 4: Pseudo Code of ReliefF

2.5 Wrapper Method

Wrapper method will determine the best feature subset by applying a predefined algorithm and this best feature subset is generated based on the predictive accuracy of the algorithm. Wrapper method generates a subset based on the classifier performance for classification tasks (e.g. Naïve Bayes or SVM) [8]. Although wrapper method is little bit slower than filter method in finding the good subset but it has been proven that wrapper method finds the subset with better performance than filter method because subset is generated using a real modeling algorithm [2]. Figure 5 illustrates the flow chart of wrapper method.

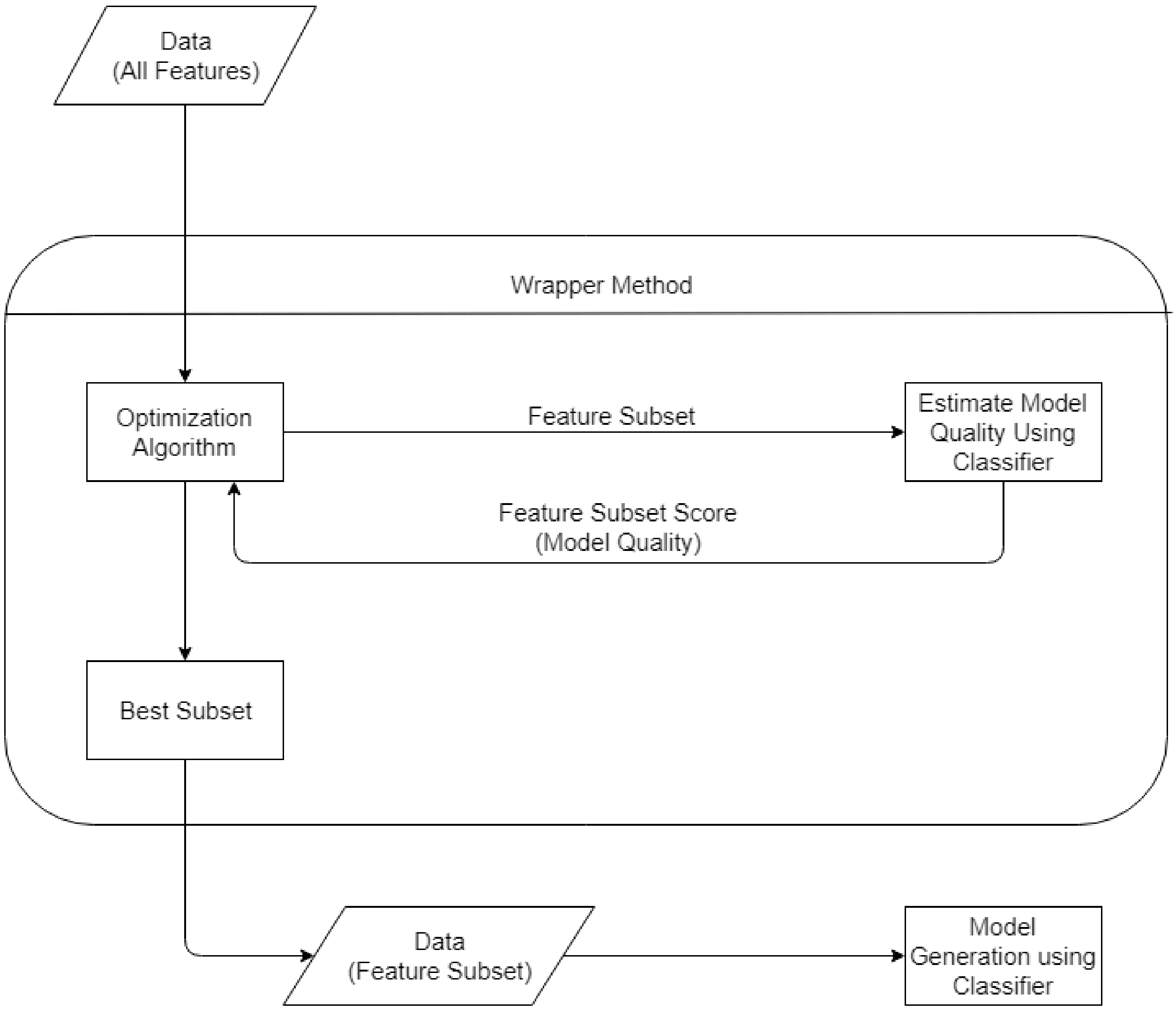

Figure 5: Flow Chart of Wrapper Method

Here I am using Chi Square, Recursive Feature Elimination, Extra Trees Classifier and Sequential Floating Forward Selection algorithms for the wrapper method. Figure 6, 7, 8 illustrate the pseudo code of Chi Square, Recursive Feature Elimination and Sequential Floating Forward Selection algorithms respectively [16] [17].

Algorithm. Improved chi-square attack on RLWE modulo $\mathfrak{q}$)

Input: K – a number field; R – the ring of integers of K; $\mathfrak{q}$ – a prime ideal in K above q with residue degree 2; $\mathcal{S}$ – a collection of M RLWE samples; $\beta > 0$ – the parameter used for comparing χ^2 values.

Output: a guess of the value $s \pmod{\mathfrak{q}}$, or **NOT-RLWE**, or **INSUFFICIENT-SAMPLES**

```
Let 𝒢 ← ∅.
for j in 1, . . . , q do
    ℰ_j ← ∅.
    for a, b in 𝒮 do
        ā, b̄ ← a (mod 𝔮), b (mod 𝔮).
        m_j ← (b̄ − b − \overline{a t_j} + a t_j) / (ā − a).
        add m_j to ℰ_j.
    end for
    Run a chi-square test for uniform distribution on ℰ_j.
    if χ²(ℰ_j) > β then
        s_0 := the element(s) in ℰ_j with highest frequency.
        s ← s_0 + t_j, add s to 𝒢.
    end if
end for
if 𝒢 = ∅ then
        return NOT-RLWE
else if 𝒢 = {s} is a singleton then
        return s
else
        return INSUFFICIENT-SAMPLES
end if
```

Figure 6: Pseudo Code of Chi Square


```
Algorithm: Recursive feature elimination
Tune/train the model on the training set using all predictors
Calculate model performance
Calculate variable importance or rankings
for Each subset size S_i, i = 1 ... S do
    Keep the S_i most important variables
    [Optional] Pre-process the data
    Tune/train the model on the training set using S_i predictors
    Calculate model performance
    [Optional] Recalculate the rankings for each predictor
end
Calculate the performance profile over the S_i
Determine the appropriate number of predictors
Use the model corresponding to the optimal S_i
```

Figure 7: Pseudo Code of Recursive Elimination

```
Algorithm: Sequential Forward Selection
// input OFM: Original feature vector model
// output NFM: New feature vector model
// newAcc: Result of classification
For i = 1 to N Do
    sAcc = ELM (OFM (i));
    AppendTo(allsAcc, (sAcc, i));
sortAcc=sort(allsAcc, sAcc);
For i = 1 to N Do
    AppendTo(FM, OFM (sortAcc (i)));
    AppendTo(allAcc, ELM (FM));
newAcc = FindMax(FM);
Return ⟨NFM, newAcc⟩;
```

Figure 8: Pseudo Code of Sequential Floating Forward Selection

2.6 Hybrid Method

Hybrid method is combining the best properties of filter and wrapper methods. First a filter method is reducing the feature space dimension by removing the irrelevant features [11]. Then, wrapper method is applied for finding the best candidate subset. We can get high accuracy using the hybrid method. Hybrid methodology can be constructed with any combination of filter and wrapper methods. There also few methodologies which was proposed. Such as:

i. Fuzzy random forest based feature selection [12],
ii. Hybrid genetic algorithms [13],
iii. Hybrid ant colony optimization [14],
iv. Mixed gravitational search algorithm [15]

Figure 9 illustrates the flow chart of hybrid method.

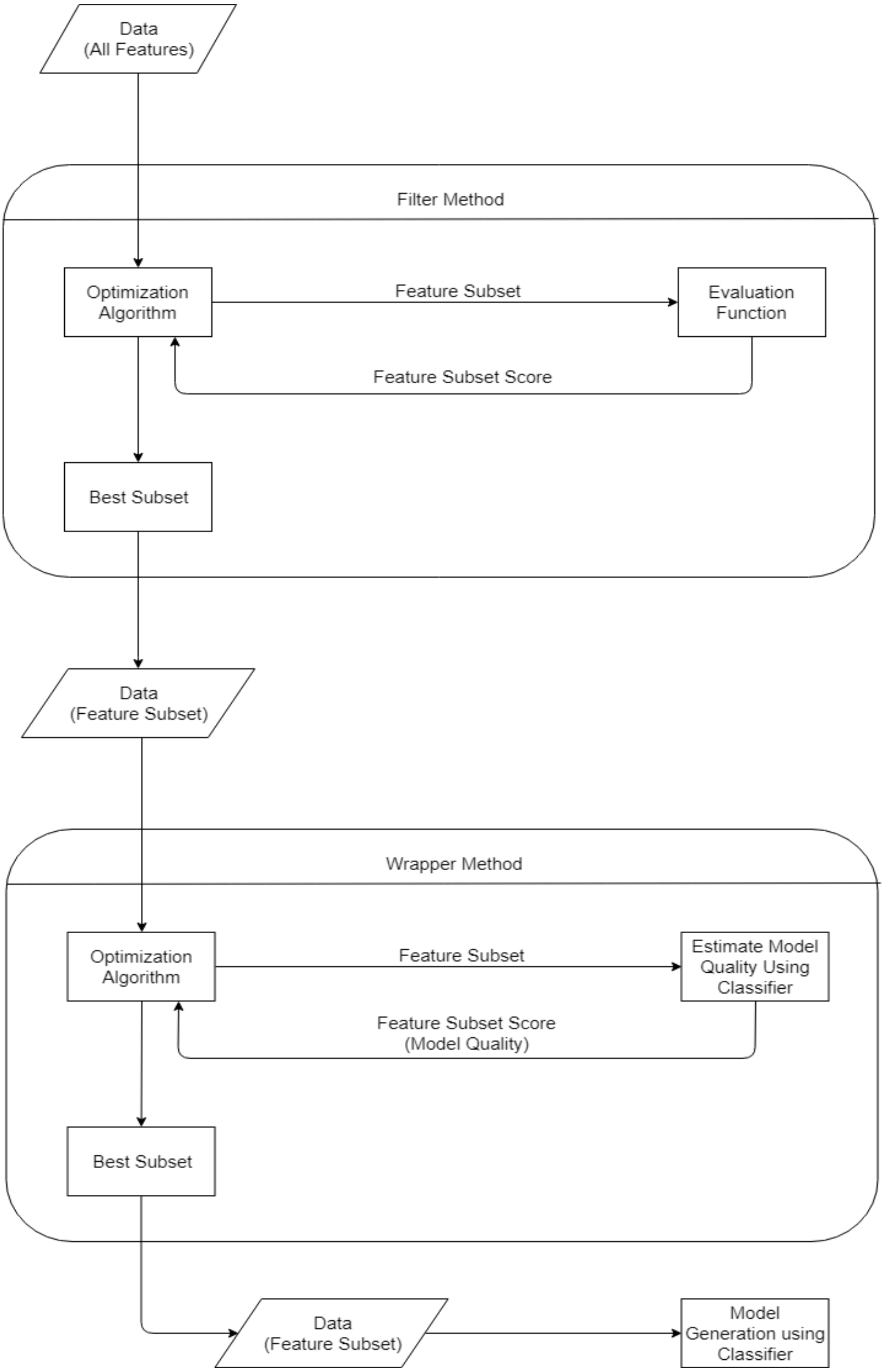

Figure 9: Flow Chart of Hybrid Method

2.7 Related Works

2.7.1 Correlation-based Feature Selection for Machine Learning [18]

In this thesis, features were selected based on correlation approach. That means, features that are highly correlated with the class are selected. Here, Correlation based Feature Selection (CFS) was experimented on artificial and natural datasets and the algorithms were used are:

i. C4.5 (A Decision Tree Learner),
ii. IB1 (An Instance based Learner),
iii. Naïve Bayes.

The result of the experiments showed that irrelevant, redundant and noisy features are quickly identified by the CFS.

2.7.2 Dragonfly Algorithm: Theory, Literature Review and Application in Feature Selection [19]

In this paper they have designed a wrapper-based feature selection algorithm which is based on the binary variant of Dragonfly Algorithm (DA). Dragonfly algorithm is a successful and well established for various optimization problems including feature selection. Here they have tested the performance of the algorithm on some special type of datasets. The datasets contain a lot of features with low number of samples. The final experimental results showed that the ability of DA to deal with this kind of datasets is better than other optimizers.

2.7.3 A Review of Feature Selection Techniques in Bioinformatics

In this paper, they discussed about several techniques for feature selection in bioinformatics. They have showed the basic taxonomy of feature selection techniques, discussed their use, variety and potential in number of both common as well as upcoming bioinformatics applications.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 The System Architecture

In this system, at first I have imported a dataset. Then I have applied filter and wrapper method respectively. From these two methods I have got the desired feature subset. After that I have divided the dataset into two parts. The ratio of the two parts is 70:30. That means one part of the dataset is having 70% of data and the other part is having 30% of data. 70% data is used for training purpose and rest 30% of data is used for testing purpose.

After dividing the data, I have applied the classifier for training purpose and to get the desired result. Finally validate the result to examine the accuracy of the result.

Here, figure 10 illustrates the basic architecture of feature selection process.

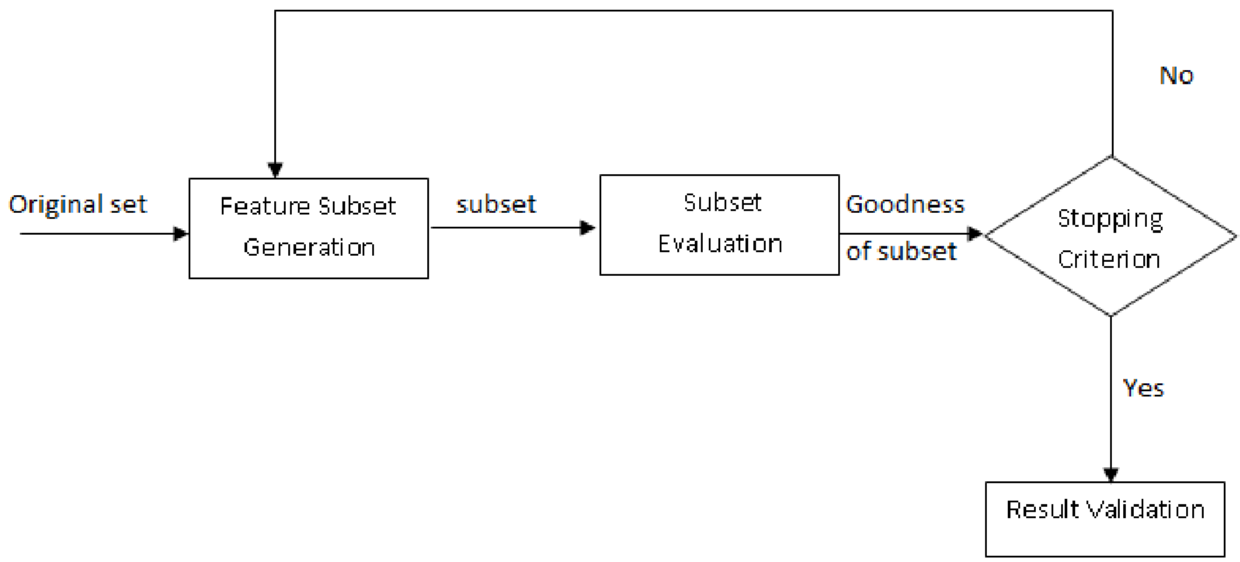

Figure 10: Basic Architecture of Feature Selection Process

3.2 Methodology

In this thesis, I have used hybrid method to find the best features from dataset, because if we can get accurate features, then we can expect accurate results.

A flow chart of our proposed system is shown in figure 11. The details of our proposed method are described below in the following steps:

Step 1: Data

In this step, I am importing the dataset as a CSV file and getting all the data and storing it in a variable named data_frame.

Step 2: Feature Acquisition

Here I am getting and separating the features from data_frame, then storing it in a variable named features.

Step 3: Feature Set

Here I am separating the test features for training purpose. I am not taking all the features for training purpose. I dividing the data into two parts, where one part is consisting of 70% of data and other part is consisting of 30% of data. 70% of data is for training purpose and rest 30% is for test purpose.

Step 4: Irrelevance Filter

Here, irrelevance filter means removing the irrelevant data from features. I am removing irrelevant features by applying the filter method. That means I am applying the ReliefF algorithm to remove irrelevant features from feature set.

Step 5: Relevant Features

After removing all the irrelevant features by applying the ReliefF algorithm in filter method, I am having a feature subset consist of only relevant features.

Step 6: Redundancy Filter

Here Redundancy filter means removing all the redundant data from features. To remove all the redundant data I am applying the wrapper method and in wrapper method I am using Chi Square, Recursive Feature Elimination, Extra Trees Classifier, Sequential Floating Forward Selection algorithms to remove the redundant features and cluster the similar features.

Step 7: Non-redundant Features

After removing the redundant features by applying all the algorithms in wrapper method, in this step I am having a feature subset containing non-redundant features only.

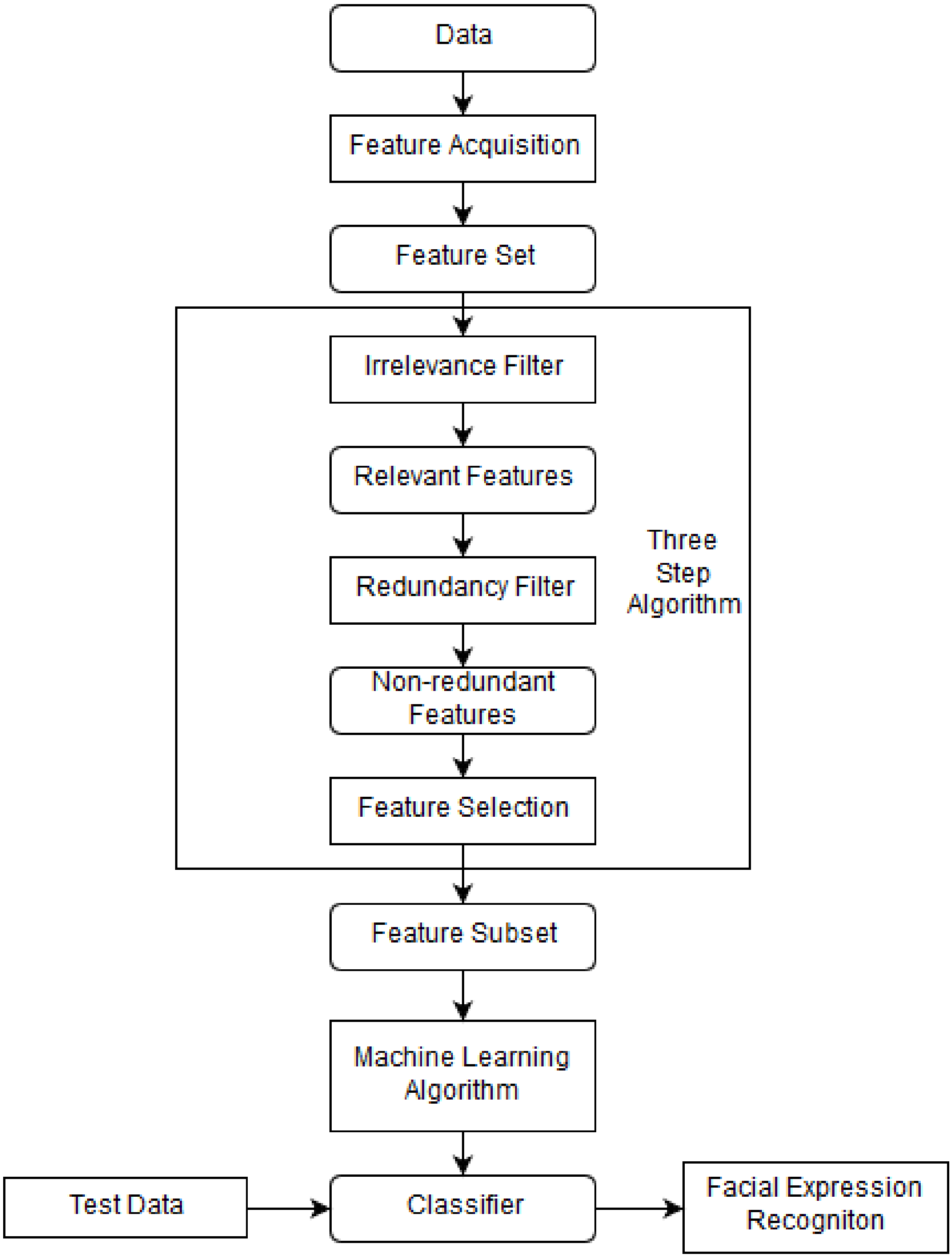

Figure 11: Flow Chart of the System

Step 8: Feature Selection

In every algorithm I am getting the accuracy of all the features. From that accuracy I can select which feature is more related and which feature is less related. So from all the algorithms I am getting four features each. Finally I am taking the common features among all and these are the final features for applying the classifier.

Step 9: Feature Subset

After having final features with the combination of final features I am making a feature subset which is the final feature subset for applying the classifier.

Step 10: Machine Learning Algorithm

Machine learning algorithm means classifier algorithm. That means which algorithm I am using in the classifier algorithm. In this step I am applying several algorithms by which I am having different results. Such as: Gaussian, Decision Tree, Random Forest, SVM, SVM (Kernel='Linear').

Step 11: Classifier

Finally in this step, I am applying the classifier and getting the accuracy of that classifier. Based on that accuracy I am deciding which one is giving me more accurate results and which one is giving less accurate results.

3.3 Challenges

- Pre processing of the datasets that means if there is any other value like empty or null in the dataset then I have to remove that,
- Selecting the algorithms for filter and wrapper methods that means I have search on which is algorithm is going to be good for filter and wrapper methods and which is going to give me better result,
- Applying all the algorithms accurately that means different algorithms got different ways of setting their parameters. I have to set up all the parameters accurate in order to run the algorithms accurately.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Implementation

In my thesis, I have used two datasets for getting the experimental result. Such as:

- Pima Diabetes Dataset
- Breast Cancer Dataset

For using these datasets I have to use some tools, which have been mentioned below. After applying all the tools I have got my desired results. The details of the results have been given below.

4.1.1 Dataset

Here I have used two datasets. Such as:

- **Pima Diabetes Dataset:** Pima Diabetes dataset is used for predicting the diabetes, that means according to the value of the features it could be predicted whether there is diabetes or not. There are different features in the dataset such as: pregnancies, glucose, diastolic blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, age, diabetes. I have applied several algorithms on this dataset then according to the results I have selected four specific features but for hybrid algorithm I have selected three most common features. Using those features I have predicted whether there is diabetes or not. Finally I have applied the classifier on those selected features and got the accuracy. Accuracy could be 0 to 1 and it indicates how much accurate result I have got. The algorithms I have applied on this dataset are: ReliefF, Chi Square, Recursive Feature Elimination, Sequential Floating Forward Selection.

- **Breast Cancer Dataset:** Breast Cancer dataset is used for predicting the breast cancer, that means according to the value of the features it could predicted whether there is breast cancer or not. There are different features in the dataset, such as: clump thickness, size uniformity, shape uniformity, marginal adhesion, epithelial size, bare nucleoli, bland chromatin, normal nucleoli, mitoses, class. I have applied several algorithms on this dataset then according to the results I have selected four specific features but for hybrid algorithm I have selected three most common features. Using those features I have predicted whether there is breast cancer or not. Finally I have applied the classifier on those selected features and got the accuracy. Accuracy could be 0 to 1 and it indicates how much accurate result I have got. The algorithms I have applied on this dataset are: ReliefF, Chi Square, Recursive Feature Elimination, Sequential Floating Forward Selection.

4.1.2 Tools Used for My Thesis

I have used the following tools and technologies for my thesis:

- **Programming Language:** Python
- **Python Libraries:** Sci-kit Learn, Pip
- **Text Editor:** Atom

4.2 Experimental Results

As it is very obvious that if I take more features I will get better result, so here I have tried to show that we could get better result taking as less features as possible. To do that I have taken four features for applying each algorithm but for applying the hybrid algorithm I have taken three most common features among all the algorithms and showed that hybrid algorithm gives almost the same result as other algorithms are giving. Detail of the result is given below.

4.2.1 Comparison

Here I have created two tables for comparison. One is for diabetes prediction and another one is for breast cancer prediction. Each table has got the results of several algorithms and showing the comparison of those results.

Table 1 shows the diabetes prediction accuracy.

Classifier	ReliefF	Chi Square	Recursive Feature Elimination	Extra Trees Classifier	SFFS	Hybrid
Guassian	0.7273	0.7403	0.7706	0.7489	0.7143	0.7403
DecisionTree	0.6970	0.6797	0.7186	0.7143	0.6104	0.6753
RandomForest	0.7403	0.7316	0.7403	0.7446	0.7013	0.7316
SVM	0.6537	0.6494	0.6494	0.6364	0.6407	0.6450
SVM(kernel='linear')	0.7316	0.7359	0.7576	0.7446	0.6494	0.7359

Table 1: Diabetes Prediction Accuracy

Table 2 shows breast cancer prediction accuracy.

Classifier	ReliefF	Chi Square	Recursive Feature Elimination	Extra Trees Classifier	SFFS	Hybrid
Guassian	0.9512	0.9512	0.9317	0.9463	0.9366	0.9463
DecisionTree	0.9415	0.9317	0.9220	0.9463	0.9122	0.9366
RandomForest	0.9415	0.9463	0.9268	0.9366	0.9220	0.9463
SVM	0.9561	0.9659	0.9561	0.9659	0.9415	0.9512

SVM(kernel='linear')	0.9366	0.9366	0.9463	0.9463	0.9415	0.9366

Table 2: Breast Cancer Prediction Accuracy

4.3 Analysis

According to the found experimental results I can do an analysis and come to the following conclusion that-

Hybrid algorithm is giving almost the same result as other algorithms but with less features than other algorithms. That means according to the result of each algorithm I am selecting four specific features but in the hybrid algorithm I am selecting the most three common features among all the algorithms. But still hybrid algorithm is giving almost the same result as other algorithms.

4.4 Observations

By implementing different algorithms for feature selection and classifier, I have found some observations. Some of the observations are discussed below-

- From the comparison we can see that, for the diabetes dataset if I apply Gaussian algorithm for the classifier, I am getting the highest accuracy for almost every algorithm.
- For the breast cancer dataset if I apply SVM algorithm for the classifier, I am getting the highest accuracy for almost every algorithm.
- On the other hand, for the diabetes dataset I am getting the lowest accuracy for almost every algorithm if I apply SVM algorithm for the classifier.
- For the breast cancer dataset, I am getting the lowest accuracy for almost every algorithm if I apply Decision Tree algorithm for the classifier.
- From the comparison we can notice another thing that for every classifier hybrid algorithm is giving almost the highest accuracy, although hybrid algorithm is using only three features whereas other algorithms are using four features.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusion

In my thesis, I have explained the architecture of my method in details and conducted an analytical study of different algorithms. Here I have designed my own hybrid method and executed an experimental results. From my analytical study, I have shown that the prediction of hybrid method is more accurate rather than the filter and wrapper method individually although hybrid method is having less features than other two methods. Here hybrid method is using three features and filter and wrapper methods are using four features.

5.2 Recommendations for Future Works

By keeping the similarity I will experiment different algorithms for hybrid method by representing the problem in different way in future. I will also use the same method for recognizing the facial expression of human and execute an experimental result.

REFERENCES

[1] “Machine Learning”[Online].

Available: https://searchenterpriseai.techtarget.com/definition/machine-learning-ML. [Accessed: Jan 15, 2019].

[2] A. Jović, K. Brkić, “A review of feature selection methods with applications” 2015.

[3] K.Sutha, Dr.J. Jebamalar, “A Review of Feature Selection Algorithms for Data Mining Techniques”.

[4] Lei Yu, Huan Liu, “Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution”, Department of Computer Science & Engineering, Arizone State University, Tempe, AZ 85287-5406, USA, 2003.

[5] A.Blum and P.Langley, “Selection of relevant features and examples in machine learning”, Artificial Intelligence, vol 97, pp 245-271,1997.

[6] Qinbao Song, Jingjie Ni and Guangtao Wang, “A Fast Clustering-Based Feature Subset Selection Algorithm for High-Dimensional Data”, IEEE Transactions on Knowledge and Data Engineering, Vol 25, No.1, 2013.

[7] Huan Liu and Lei Yu, “Towards Integrating Feature Selection Algorithms for Classification and Clustering”, IEEE Transactions on Knowledge and Data Engineering, Vol.17 No.4 2005.

[8] P. S. Bradley and O. L. Mangasarian, "Feature selection via concave minimization and support vector machines," in: Proc. 15th International Conference on Machine Learning (ICML-1998), Madison, Wisconsin, USA, Morgan Kaufmann, pp. 82–90, 1998.

[9] S. Maldonado, R. Weber, and F. Famili, "Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines," Information Sciences, vol. 286, pp. 228–246, 2014.

[10] Y. S. Kim, W. N. Street, and F. Menczer, "Evolutionary model selection in unsupervised learning," Intelligent Data Analysis, vol. 6, no. 6, pp. 531–556, 2002.

[11] S. Das, "Filters, wrappers and a boosting-based hybrid for feature selection," in: Proc. 18th International Conference on Machine Learning (ICML-2001), San Francisco, CA, USA, Morgan Kaufmann, pp. 74–81, 2001.

[12] J. M. Cadenas, M. C. Garrido, and R. Martínez, "Feature subset selection Filter–Wrapper based on low quality data," Expert Systems with Applications, vol. 40, pp. 6241–6252, 2013.

[13] I. S. Oh, J. S. Lee, and B. R. Moon, "Hybrid genetic algorithms for feature selection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 11, pp. 1424–1437, 2004.

[14] S. I. Ali and W. Shahzad, "A Feature Subset Selection Method based on Conditional Mutual Information and Ant Colony Optimization," International Journal of Computer Applications, vol. 60, no. 11, pp. 5–10, 2012.

[15] S. Sarafrazi and H. Nezamabadi-pour, "Facing the classification of binary problems with a GSA-SVM hybrid system," Mathematical and Computer Modelling, vol. 57, issues 1-2, pp. 270–278, 2013.

[16] “Recursive Feature Elimination” [Online].

Available: https://topepo.github.io/caret/recursive-feature-elimination.html. [Accessed: Apr 13, 2019].

[17] “Sequential Floating Forward Selection” [Online].

Available: https://link.springer.com/article/10.1007/s00521-014-1764-0. [Accessed: Apr 13, 2019].

[18] Mark A. Hall, “Correlation-based Feature Selection for Machine Learning”, April, 1999.

[19] M. Mafarja, A. A. Heidari, H. Faris, I Aljarah, “Dragonfly Algorithm: Theory, Literature Review and Application in Feature Selection”, February, 2019.

Appendix – A

Data Sets

Table 3 shows Pima Indian Diabetes Dataset

Num_preg	Glucose_conc	Diastolic_bp	Skin_thickness	Insulin	BMI	Diab_pred	Age	Diabetes
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1
5	166	72	19	175	25.8	0.587	51	1
7	100	0	0	0	30	0.484	32	1
0	118	84	47	230	45.8	0.551	31	1
7	107	74	0	0	29.6	0.254	31	1
1	103	30	38	83	43.3	0.183	33	0
1	115	70	30	96	34.6	0.529	32	1
3	126	88	41	235	39.3	0.704	27	0
8	99	84	0	0	35.4	0.388	50	0
7	196	90	0	0	39.8	0.451	41	1
9	119	80	35	0	29	0.263	29	1
11	143	94	33	146	36.6	0.254	51	1
10	125	70	26	115	31.1	0.205	41	1
7	147	76	0	0	39.4	0.257	43	1
1	97	66	15	140	23.2	0.487	22	0
13	145	82	19	110	22.2	0.245	57	0

5	117	92	0	0	34.1	0.337	38	0
5	109	75	26	0	36	0.546	60	0
3	158	76	36	245	31.6	0.851	28	1
3	88	58	11	54	24.8	0.267	22	0
6	92	92	0	0	19.9	0.188	28	0
10	122	78	31	0	27.6	0.512	45	0
4	103	60	33	192	24	0.966	33	0
11	138	76	0	0	33.2	0.42	35	0
9	102	76	37	0	32.9	0.665	46	1
2	90	68	42	0	38.2	0.503	27	1
4	111	72	47	207	37.1	1.39	56	1
3	180	64	25	70	34	0.271	26	0
7	133	84	0	0	40.2	0.696	37	0
7	106	92	18	0	22.7	0.235	48	0
9	171	110	24	240	45.4	0.721	54	1
7	159	64	0	0	27.4	0.294	40	0
0	180	66	39	0	42	1.893	25	1
1	146	56	0	0	29.7	0.564	29	0
2	71	70	27	0	28	0.586	22	0
7	103	66	32	0	39.1	0.344	31	1
7	105	0	0	0	0	0.305	24	0
1	103	80	11	82	19.4	0.491	22	0
1	101	50	15	36	24.2	0.526	26	0
5	88	66	21	23	24.4	0.342	30	0
8	176	90	34	300	33.7	0.467	58	1
7	150	66	42	342	34.7	0.718	42	0
1	73	50	10	0	23	0.248	21	0
7	187	68	39	304	37.7	0.254	41	1
0	100	88	60	110	46.8	0.962	31	0
0	146	82	0	0	40.5	1.781	44	0
0	105	64	41	142	41.5	0.173	22	0
2	84	0	0	0	0	0.304	21	0
8	133	72	0	0	32.9	0.27	39	1
5	44	62	0	0	25	0.587	36	0
2	141	58	34	128	25.4	0.699	24	0
7	114	66	0	0	32.8	0.258	42	1

5	99	74	27	0	29	0.203	32	0
0	109	88	30	0	32.5	0.855	38	1
2	109	92	0	0	42.7	0.845	54	0
1	95	66	13	38	19.6	0.334	25	0
4	146	85	27	100	28.9	0.189	27	0
2	100	66	20	90	32.9	0.867	28	1
5	139	64	35	140	28.6	0.411	26	0
13	126	90	0	0	43.4	0.583	42	1
4	129	86	20	270	35.1	0.231	23	0
1	79	75	30	0	32	0.396	22	0
1	0	48	20	0	24.7	0.14	22	0
7	62	78	0	0	32.6	0.391	41	0
5	95	72	33	0	37.7	0.37	27	0
0	131	0	0	0	43.2	0.27	26	1
2	112	66	22	0	25	0.307	24	0
3	113	44	13	0	22.4	0.14	22	0
2	74	0	0	0	0	0.102	22	0
7	83	78	26	71	29.3	0.767	36	0
0	101	65	28	0	24.6	0.237	22	0
5	137	108	0	0	48.8	0.227	37	1
2	110	74	29	125	32.4	0.698	27	0
13	106	72	54	0	36.6	0.178	45	0
2	100	68	25	71	38.5	0.324	26	0
15	136	70	32	110	37.1	0.153	43	1
1	107	68	19	0	26.5	0.165	24	0
1	80	55	0	0	19.1	0.258	21	0
4	123	80	15	176	32	0.443	34	0
7	81	78	40	48	46.7	0.261	42	0
4	134	72	0	0	23.8	0.277	60	1
2	142	82	18	64	24.7	0.761	21	0
6	144	72	27	228	33.9	0.255	40	0
2	92	62	28	0	31.6	0.13	24	0
1	71	48	18	76	20.4	0.323	22	0
6	93	50	30	64	28.7	0.356	23	0
1	122	90	51	220	49.7	0.325	31	1
1	163	72	0	0	39	1.222	33	1

1	151	60	0	0	26.1	0.179	22	0
0	125	96	0	0	22.5	0.262	21	0
1	81	72	18	40	26.6	0.283	24	0
2	85	65	0	0	39.6	0.93	27	0
1	126	56	29	152	28.7	0.801	21	0
1	96	122	0	0	22.4	0.207	27	0
4	144	58	28	140	29.5	0.287	37	0
3	83	58	31	18	34.3	0.336	25	0
0	95	85	25	36	37.4	0.247	24	1
3	171	72	33	135	33.3	0.199	24	1
8	155	62	26	495	34	0.543	46	1
1	89	76	34	37	31.2	0.192	23	0
4	76	62	0	0	34	0.391	25	0
7	160	54	32	175	30.5	0.588	39	1
4	146	92	0	0	31.2	0.539	61	1
5	124	74	0	0	34	0.22	38	1
5	78	48	0	0	33.7	0.654	25	0
4	97	60	23	0	28.2	0.443	22	0
4	99	76	15	51	23.2	0.223	21	0
0	162	76	56	100	53.2	0.759	25	1
6	111	64	39	0	34.2	0.26	24	0
2	107	74	30	100	33.6	0.404	23	0
5	132	80	0	0	26.8	0.186	69	0
0	113	76	0	0	33.3	0.278	23	1
1	88	30	42	99	55	0.496	26	1
3	120	70	30	135	42.9	0.452	30	0
1	118	58	36	94	33.3	0.261	23	0
1	117	88	24	145	34.5	0.403	40	1
0	105	84	0	0	27.9	0.741	62	1
4	173	70	14	168	29.7	0.361	33	1
9	122	56	0	0	33.3	1.114	33	1
3	170	64	37	225	34.5	0.356	30	1
8	84	74	31	0	38.3	0.457	39	0
2	96	68	13	49	21.1	0.647	26	0
2	125	60	20	140	33.8	0.088	31	0
0	100	70	26	50	30.8	0.597	21	0

0	93	60	25	92	28.7	0.532	22	0
0	129	80	0	0	31.2	0.703	29	0
5	105	72	29	325	36.9	0.159	28	0
3	128	78	0	0	21.1	0.268	55	0
5	106	82	30	0	39.5	0.286	38	0
2	108	52	26	63	32.5	0.318	22	0
10	108	66	0	0	32.4	0.272	42	1
4	154	62	31	284	32.8	0.237	23	0
0	102	75	23	0	0	0.572	21	0
9	57	80	37	0	32.8	0.096	41	0
2	106	64	35	119	30.5	1.4	34	0
5	147	78	0	0	33.7	0.218	65	0
2	90	70	17	0	27.3	0.085	22	0
1	136	74	50	204	37.4	0.399	24	0
4	114	65	0	0	21.9	0.432	37	0
9	156	86	28	155	34.3	1.189	42	1
1	153	82	42	485	40.6	0.687	23	0
8	188	78	0	0	47.9	0.137	43	1
7	152	88	44	0	50	0.337	36	1
2	99	52	15	94	24.6	0.637	21	0
1	109	56	21	135	25.2	0.833	23	0
2	88	74	19	53	29	0.229	22	0
17	163	72	41	114	40.9	0.817	47	1
4	151	90	38	0	29.7	0.294	36	0
7	102	74	40	105	37.2	0.204	45	0
0	114	80	34	285	44.2	0.167	27	0
2	100	64	23	0	29.7	0.368	21	0
0	131	88	0	0	31.6	0.743	32	1
6	104	74	18	156	29.9	0.722	41	1
3	148	66	25	0	32.5	0.256	22	0
4	120	68	0	0	29.6	0.709	34	0
4	110	66	0	0	31.9	0.471	29	0
3	111	90	12	78	28.4	0.495	29	0
6	102	82	0	0	30.8	0.18	36	1
6	134	70	23	130	35.4	0.542	29	1
2	87	0	23	0	28.9	0.773	25	0

1	79	60	42	48	43.5	0.678	23	0
2	75	64	24	55	29.7	0.37	33	0
8	179	72	42	130	32.7	0.719	36	1
6	85	78	0	0	31.2	0.382	42	0
0	129	110	46	130	67.1	0.319	26	1
5	143	78	0	0	45	0.19	47	0
5	130	82	0	0	39.1	0.956	37	1
6	87	80	0	0	23.2	0.084	32	0
0	119	64	18	92	34.9	0.725	23	0
1	0	74	20	23	27.7	0.299	21	0
5	73	60	0	0	26.8	0.268	27	0
4	141	74	0	0	27.6	0.244	40	0
7	194	68	28	0	35.9	0.745	41	1
8	181	68	36	495	30.1	0.615	60	1
1	128	98	41	58	32	1.321	33	1
8	109	76	39	114	27.9	0.64	31	1
5	139	80	35	160	31.6	0.361	25	1
3	111	62	0	0	22.6	0.142	21	0
9	123	70	44	94	33.1	0.374	40	0
7	159	66	0	0	30.4	0.383	36	1
11	135	0	0	0	52.3	0.578	40	1
8	85	55	20	0	24.4	0.136	42	0
5	158	84	41	210	39.4	0.395	29	1
1	105	58	0	0	24.3	0.187	21	0
3	107	62	13	48	22.9	0.678	23	1
4	109	64	44	99	34.8	0.905	26	1
4	148	60	27	318	30.9	0.15	29	1
0	113	80	16	0	31	0.874	21	0
1	138	82	0	0	40.1	0.236	28	0
0	108	68	20	0	27.3	0.787	32	0
2	99	70	16	44	20.4	0.235	27	0
6	103	72	32	190	37.7	0.324	55	0
5	111	72	28	0	23.9	0.407	27	0
8	196	76	29	280	37.5	0.605	57	1
5	162	104	0	0	37.7	0.151	52	1
1	96	64	27	87	33.2	0.289	21	0

7	184	84	33	0	35.5	0.355	41	1
2	81	60	22	0	27.7	0.29	25	0
0	147	85	54	0	42.8	0.375	24	0
7	179	95	31	0	34.2	0.164	60	0
0	140	65	26	130	42.6	0.431	24	1
9	112	82	32	175	34.2	0.26	36	1
12	151	70	40	271	41.8	0.742	38	1
5	109	62	41	129	35.8	0.514	25	1
6	125	68	30	120	30	0.464	32	0
5	85	74	22	0	29	1.224	32	1
5	112	66	0	0	37.8	0.261	41	1
0	177	60	29	478	34.6	1.072	21	1
2	158	90	0	0	31.6	0.805	66	1
7	119	0	0	0	25.2	0.209	37	0
7	142	60	33	190	28.8	0.687	61	0
1	100	66	15	56	23.6	0.666	26	0
1	87	78	27	32	34.6	0.101	22	0
0	101	76	0	0	35.7	0.198	26	0
3	162	52	38	0	37.2	0.652	24	1
4	197	70	39	744	36.7	2.329	31	0
0	117	80	31	53	45.2	0.089	24	0
4	142	86	0	0	44	0.645	22	1
6	134	80	37	370	46.2	0.238	46	1
1	79	80	25	37	25.4	0.583	22	0
4	122	68	0	0	35	0.394	29	0
3	74	68	28	45	29.7	0.293	23	0
4	171	72	0	0	43.6	0.479	26	1
7	181	84	21	192	35.9	0.586	51	1
0	179	90	27	0	44.1	0.686	23	1
9	164	84	21	0	30.8	0.831	32	1
0	104	76	0	0	18.4	0.582	27	0
1	91	64	24	0	29.2	0.192	21	0
4	91	70	32	88	33.1	0.446	22	0
3	139	54	0	0	25.6	0.402	22	1
6	119	50	22	176	27.1	1.318	33	1
2	146	76	35	194	38.2	0.329	29	0

9	184	85	15	0	30	1.213	49	1
10	122	68	0	0	31.2	0.258	41	0
0	165	90	33	680	52.3	0.427	23	0
9	124	70	33	402	35.4	0.282	34	0
1	111	86	19	0	30.1	0.143	23	0
9	106	52	0	0	31.2	0.38	42	0
2	129	84	0	0	28	0.284	27	0
2	90	80	14	55	24.4	0.249	24	0
0	86	68	32	0	35.8	0.238	25	0
12	92	62	7	258	27.6	0.926	44	1
1	113	64	35	0	33.6	0.543	21	1
3	111	56	39	0	30.1	0.557	30	0
2	114	68	22	0	28.7	0.092	25	0
1	193	50	16	375	25.9	0.655	24	0
11	155	76	28	150	33.3	1.353	51	1
3	191	68	15	130	30.9	0.299	34	0
3	141	0	0	0	30	0.761	27	1
4	95	70	32	0	32.1	0.612	24	0
3	142	80	15	0	32.4	0.2	63	0
4	123	62	0	0	32	0.226	35	1
5	96	74	18	67	33.6	0.997	43	0
0	138	0	0	0	36.3	0.933	25	1
2	128	64	42	0	40	1.101	24	0
0	102	52	0	0	25.1	0.078	21	0
2	146	0	0	0	27.5	0.24	28	1
10	101	86	37	0	45.6	1.136	38	1
2	108	62	32	56	25.2	0.128	21	0
3	122	78	0	0	23	0.254	40	0
1	71	78	50	45	33.2	0.422	21	0
13	106	70	0	0	34.2	0.251	52	0
2	100	70	52	57	40.5	0.677	25	0
7	106	60	24	0	26.5	0.296	29	1
0	104	64	23	116	27.8	0.454	23	0
5	114	74	0	0	24.9	0.744	57	0
2	108	62	10	278	25.3	0.881	22	0
0	146	70	0	0	37.9	0.334	28	1

10	129	76	28	122	35.9	0.28	39	0
7	133	88	15	155	32.4	0.262	37	0
7	161	86	0	0	30.4	0.165	47	1
2	108	80	0	0	27	0.259	52	1
7	136	74	26	135	26	0.647	51	0
5	155	84	44	545	38.7	0.619	34	0
1	119	86	39	220	45.6	0.808	29	1
4	96	56	17	49	20.8	0.34	26	0
5	108	72	43	75	36.1	0.263	33	0
0	78	88	29	40	36.9	0.434	21	0
0	107	62	30	74	36.6	0.757	25	1
2	128	78	37	182	43.3	1.224	31	1
1	128	48	45	194	40.5	0.613	24	1
0	161	50	0	0	21.9	0.254	65	0
6	151	62	31	120	35.5	0.692	28	0
2	146	70	38	360	28	0.337	29	1
0	126	84	29	215	30.7	0.52	24	0
14	100	78	25	184	36.6	0.412	46	1
8	112	72	0	0	23.6	0.84	58	0
0	167	0	0	0	32.3	0.839	30	1
2	144	58	33	135	31.6	0.422	25	1
5	77	82	41	42	35.8	0.156	35	0
5	115	98	0	0	52.9	0.209	28	1
3	150	76	0	0	21	0.207	37	0
2	120	76	37	105	39.7	0.215	29	0
10	161	68	23	132	25.5	0.326	47	1
0	137	68	14	148	24.8	0.143	21	0
0	128	68	19	180	30.5	1.391	25	1
2	124	68	28	205	32.9	0.875	30	1
6	80	66	30	0	26.2	0.313	41	0
0	106	70	37	148	39.4	0.605	22	0
2	155	74	17	96	26.6	0.433	27	1
3	113	50	10	85	29.5	0.626	25	0
7	109	80	31	0	35.9	1.127	43	1
2	112	68	22	94	34.1	0.315	26	0
3	99	80	11	64	19.3	0.284	30	0

3	182	74	0	0	30.5	0.345	29	1
3	115	66	39	140	38.1	0.15	28	0
6	194	78	0	0	23.5	0.129	59	1
4	129	60	12	231	27.5	0.527	31	0
3	112	74	30	0	31.6	0.197	25	1
0	124	70	20	0	27.4	0.254	36	1
13	152	90	33	29	26.8	0.731	43	1
2	112	75	32	0	35.7	0.148	21	0
1	157	72	21	168	25.6	0.123	24	0
1	122	64	32	156	35.1	0.692	30	1
10	179	70	0	0	35.1	0.2	37	0
2	102	86	36	120	45.5	0.127	23	1
6	105	70	32	68	30.8	0.122	37	0
8	118	72	19	0	23.1	1.476	46	0
2	87	58	16	52	32.7	0.166	25	0
1	180	0	0	0	43.3	0.282	41	1
12	106	80	0	0	23.6	0.137	44	0
1	95	60	18	58	23.9	0.26	22	0
0	165	76	43	255	47.9	0.259	26	0
0	117	0	0	0	33.8	0.932	44	0
5	115	76	0	0	31.2	0.343	44	1
9	152	78	34	171	34.2	0.893	33	1
7	178	84	0	0	39.9	0.331	41	1
1	130	70	13	105	25.9	0.472	22	0
1	95	74	21	73	25.9	0.673	36	0
1	0	68	35	0	32	0.389	22	0
5	122	86	0	0	34.7	0.29	33	0
8	95	72	0	0	36.8	0.485	57	0
8	126	88	36	108	38.5	0.349	49	0
1	139	46	19	83	28.7	0.654	22	0
3	116	0	0	0	23.5	0.187	23	0
3	99	62	19	74	21.8	0.279	26	0
5	0	80	32	0	41	0.346	37	1
4	92	80	0	0	42.2	0.237	29	0
4	137	84	0	0	31.2	0.252	30	0
3	61	82	28	0	34.4	0.243	46	0

1	90	62	12	43	27.2	0.58	24	0
3	90	78	0	0	42.7	0.559	21	0
9	165	88	0	0	30.4	0.302	49	1
1	125	50	40	167	33.3	0.962	28	1
13	129	0	30	0	39.9	0.569	44	1
12	88	74	40	54	35.3	0.378	48	0
1	196	76	36	249	36.5	0.875	29	1
5	189	64	33	325	31.2	0.583	29	1
5	158	70	0	0	29.8	0.207	63	0
5	103	108	37	0	39.2	0.305	65	0
4	146	78	0	0	38.5	0.52	67	1
4	147	74	25	293	34.9	0.385	30	0
5	99	54	28	83	34	0.499	30	0
6	124	72	0	0	27.6	0.368	29	1
0	101	64	17	0	21	0.252	21	0
3	81	86	16	66	27.5	0.306	22	0
1	133	102	28	140	32.8	0.234	45	1
3	173	82	48	465	38.4	2.137	25	1
0	118	64	23	89	0	1.731	21	0
0	84	64	22	66	35.8	0.545	21	0
2	105	58	40	94	34.9	0.225	25	0
2	122	52	43	158	36.2	0.816	28	0
12	140	82	43	325	39.2	0.528	58	1
0	98	82	15	84	25.2	0.299	22	0
1	87	60	37	75	37.2	0.509	22	0
4	156	75	0	0	48.3	0.238	32	1
0	93	100	39	72	43.4	1.021	35	0
1	107	72	30	82	30.8	0.821	24	0
0	105	68	22	0	20	0.236	22	0
1	109	60	8	182	25.4	0.947	21	0
1	90	62	18	59	25.1	1.268	25	0
1	125	70	24	110	24.3	0.221	25	0
1	119	54	13	50	22.3	0.205	24	0
5	116	74	29	0	32.3	0.66	35	1
8	105	100	36	0	43.3	0.239	45	1
5	144	82	26	285	32	0.452	58	1

3	100	68	23	81	31.6	0.949	28	0
1	100	66	29	196	32	0.444	42	0
5	166	76	0	0	45.7	0.34	27	1
1	131	64	14	415	23.7	0.389	21	0
4	116	72	12	87	22.1	0.463	37	0
4	158	78	0	0	32.9	0.803	31	1
2	127	58	24	275	27.7	1.6	25	0
3	96	56	34	115	24.7	0.944	39	0
0	131	66	40	0	34.3	0.196	22	1
3	82	70	0	0	21.1	0.389	25	0
3	193	70	31	0	34.9	0.241	25	1
4	95	64	0	0	32	0.161	31	1
6	137	61	0	0	24.2	0.151	55	0
5	136	84	41	88	35	0.286	35	1
9	72	78	25	0	31.6	0.28	38	0
5	168	64	0	0	32.9	0.135	41	1
2	123	48	32	165	42.1	0.52	26	0
4	115	72	0	0	28.9	0.376	46	1
0	101	62	0	0	21.9	0.336	25	0
8	197	74	0	0	25.9	1.191	39	1
1	172	68	49	579	42.4	0.702	28	1
6	102	90	39	0	35.7	0.674	28	0
1	112	72	30	176	34.4	0.528	25	0
1	143	84	23	310	42.4	1.076	22	0
1	143	74	22	61	26.2	0.256	21	0
0	138	60	35	167	34.6	0.534	21	1
3	173	84	33	474	35.7	0.258	22	1
1	97	68	21	0	27.2	1.095	22	0
4	144	82	32	0	38.5	0.554	37	1
1	83	68	0	0	18.2	0.624	27	0
3	129	64	29	115	26.4	0.219	28	1
1	119	88	41	170	45.3	0.507	26	0
2	94	68	18	76	26	0.561	21	0
0	102	64	46	78	40.6	0.496	21	0
2	115	64	22	0	30.8	0.421	21	0
8	151	78	32	210	42.9	0.516	36	1

4	184	78	39	277	37	0.264	31	1
0	94	0	0	0	0	0.256	25	0
1	181	64	30	180	34.1	0.328	38	1
0	135	94	46	145	40.6	0.284	26	0
1	95	82	25	180	35	0.233	43	1
2	99	0	0	0	22.2	0.108	23	0
3	89	74	16	85	30.4	0.551	38	0
1	80	74	11	60	30	0.527	22	0
2	139	75	0	0	25.6	0.167	29	0
1	90	68	8	0	24.5	1.138	36	0
0	141	0	0	0	42.4	0.205	29	1
12	140	85	33	0	37.4	0.244	41	0
5	147	75	0	0	29.9	0.434	28	0
1	97	70	15	0	18.2	0.147	21	0
6	107	88	0	0	36.8	0.727	31	0
0	189	104	25	0	34.3	0.435	41	1
2	83	66	23	50	32.2	0.497	22	0
4	117	64	27	120	33.2	0.23	24	0
8	108	70	0	0	30.5	0.955	33	1
4	117	62	12	0	29.7	0.38	30	1
0	180	78	63	14	59.4	2.42	25	1
1	100	72	12	70	25.3	0.658	28	0
0	95	80	45	92	36.5	0.33	26	0
0	104	64	37	64	33.6	0.51	22	1
0	120	74	18	63	30.5	0.285	26	0
1	82	64	13	95	21.2	0.415	23	0
2	134	70	0	0	28.9	0.542	23	1
0	91	68	32	210	39.9	0.381	25	0
2	119	0	0	0	19.6	0.832	72	0
2	100	54	28	105	37.8	0.498	24	0
14	175	62	30	0	33.6	0.212	38	1
1	135	54	0	0	26.7	0.687	62	0
5	86	68	28	71	30.2	0.364	24	0
10	148	84	48	237	37.6	1.001	51	1
9	134	74	33	60	25.9	0.46	81	0
9	120	72	22	56	20.8	0.733	48	0

1	71	62	0	0	21.8	0.416	26	0
8	74	70	40	49	35.3	0.705	39	0
5	88	78	30	0	27.6	0.258	37	0
10	115	98	0	0	24	1.022	34	0
0	124	56	13	105	21.8	0.452	21	0
0	74	52	10	36	27.8	0.269	22	0
0	97	64	36	100	36.8	0.6	25	0
8	120	0	0	0	30	0.183	38	1
6	154	78	41	140	46.1	0.571	27	0
1	144	82	40	0	41.3	0.607	28	0
0	137	70	38	0	33.2	0.17	22	0
0	119	66	27	0	38.8	0.259	22	0
7	136	90	0	0	29.9	0.21	50	0
4	114	64	0	0	28.9	0.126	24	0
0	137	84	27	0	27.3	0.231	59	0
2	105	80	45	191	33.7	0.711	29	1
7	114	76	17	110	23.8	0.466	31	0
8	126	74	38	75	25.9	0.162	39	0
4	132	86	31	0	28	0.419	63	0
3	158	70	30	328	35.5	0.344	35	1
0	123	88	37	0	35.2	0.197	29	0
4	85	58	22	49	27.8	0.306	28	0
0	84	82	31	125	38.2	0.233	23	0
0	145	0	0	0	44.2	0.63	31	1
0	135	68	42	250	42.3	0.365	24	1
1	139	62	41	480	40.7	0.536	21	0
0	173	78	32	265	46.5	1.159	58	0
4	99	72	17	0	25.6	0.294	28	0
8	194	80	0	0	26.1	0.551	67	0
2	83	65	28	66	36.8	0.629	24	0
2	89	90	30	0	33.5	0.292	42	0
4	99	68	38	0	32.8	0.145	33	0
4	125	70	18	122	28.9	1.144	45	1
3	80	0	0	0	0	0.174	22	0
6	166	74	0	0	26.6	0.304	66	0
5	110	68	0	0	26	0.292	30	0

2	81	72	15	76	30.1	0.547	25	0
7	195	70	33	145	25.1	0.163	55	1
6	154	74	32	193	29.3	0.839	39	0
2	117	90	19	71	25.2	0.313	21	0
3	84	72	32	0	37.2	0.267	28	0
6	0	68	41	0	39	0.727	41	1
7	94	64	25	79	33.3	0.738	41	0
3	96	78	39	0	37.3	0.238	40	0
10	75	82	0	0	33.3	0.263	38	0
0	180	90	26	90	36.5	0.314	35	1
1	130	60	23	170	28.6	0.692	21	0
2	84	50	23	76	30.4	0.968	21	0
8	120	78	0	0	25	0.409	64	0
12	84	72	31	0	29.7	0.297	46	1
0	139	62	17	210	22.1	0.207	21	0
9	91	68	0	0	24.2	0.2	58	0
2	91	62	0	0	27.3	0.525	22	0
3	99	54	19	86	25.6	0.154	24	0
3	163	70	18	105	31.6	0.268	28	1
9	145	88	34	165	30.3	0.771	53	1
7	125	86	0	0	37.6	0.304	51	0
13	76	60	0	0	32.8	0.18	41	0
6	129	90	7	326	19.6	0.582	60	0
2	68	70	32	66	25	0.187	25	0
3	124	80	33	130	33.2	0.305	26	0
6	114	0	0	0	0	0.189	26	0
9	130	70	0	0	34.2	0.652	45	1
3	125	58	0	0	31.6	0.151	24	0
3	87	60	18	0	21.8	0.444	21	0
1	97	64	19	82	18.2	0.299	21	0
3	116	74	15	105	26.3	0.107	24	0
0	117	66	31	188	30.8	0.493	22	0
0	111	65	0	0	24.6	0.66	31	0
2	122	60	18	106	29.8	0.717	22	0
0	107	76	0	0	45.3	0.686	24	0
1	86	66	52	65	41.3	0.917	29	0

6	91	0	0	0	29.8	0.501	31	0
1	77	56	30	56	33.3	1.251	24	0
4	132	0	0	0	32.9	0.302	23	1
0	105	90	0	0	29.6	0.197	46	0
0	57	60	0	0	21.7	0.735	67	0
0	127	80	37	210	36.3	0.804	23	0
3	129	92	49	155	36.4	0.968	32	1
8	100	74	40	215	39.4	0.661	43	1
3	128	72	25	190	32.4	0.549	27	1
10	90	85	32	0	34.9	0.825	56	1
4	84	90	23	56	39.5	0.159	25	0
1	88	78	29	76	32	0.365	29	0
8	186	90	35	225	34.5	0.423	37	1
5	187	76	27	207	43.6	1.034	53	1
4	131	68	21	166	33.1	0.16	28	0
1	164	82	43	67	32.8	0.341	50	0
4	189	110	31	0	28.5	0.68	37	0
1	116	70	28	0	27.4	0.204	21	0
3	84	68	30	106	31.9	0.591	25	0
6	114	88	0	0	27.8	0.247	66	0
1	88	62	24	44	29.9	0.422	23	0
1	84	64	23	115	36.9	0.471	28	0
7	124	70	33	215	25.5	0.161	37	0
1	97	70	40	0	38.1	0.218	30	0
8	110	76	0	0	27.8	0.237	58	0
11	103	68	40	0	46.2	0.126	42	0
11	85	74	0	0	30.1	0.3	35	0
6	125	76	0	0	33.8	0.121	54	1
0	198	66	32	274	41.3	0.502	28	1
1	87	68	34	77	37.6	0.401	24	0
6	99	60	19	54	26.9	0.497	32	0
0	91	80	0	0	32.4	0.601	27	0
2	95	54	14	88	26.1	0.748	22	0
1	99	72	30	18	38.6	0.412	21	0
6	92	62	32	126	32	0.085	46	0
4	154	72	29	126	31.3	0.338	37	0

0	121	66	30	165	34.3	0.203	33	1
3	78	70	0	0	32.5	0.27	39	0
2	130	96	0	0	22.6	0.268	21	0
3	111	58	31	44	29.5	0.43	22	0
2	98	60	17	120	34.7	0.198	22	0
1	143	86	30	330	30.1	0.892	23	0
1	119	44	47	63	35.5	0.28	25	0
6	108	44	20	130	24	0.813	35	0
2	118	80	0	0	42.9	0.693	21	1
10	133	68	0	0	27	0.245	36	0
2	197	70	99	0	34.7	0.575	62	1
0	151	90	46	0	42.1	0.371	21	1
6	109	60	27	0	25	0.206	27	0
12	121	78	17	0	26.5	0.259	62	0
8	100	76	0	0	38.7	0.19	42	0
8	124	76	24	600	28.7	0.687	52	1
1	93	56	11	0	22.5	0.417	22	0
8	143	66	0	0	34.9	0.129	41	1
6	103	66	0	0	24.3	0.249	29	0
3	176	86	27	156	33.3	1.154	52	1
0	73	0	0	0	21.1	0.342	25	0
11	111	84	40	0	46.8	0.925	45	1
2	112	78	50	140	39.4	0.175	24	0
3	132	80	0	0	34.4	0.402	44	1
2	82	52	22	115	28.5	1.699	25	0
6	123	72	45	230	33.6	0.733	34	0
0	188	82	14	185	32	0.682	22	1
0	67	76	0	0	45.3	0.194	46	0
1	89	24	19	25	27.8	0.559	21	0
1	173	74	0	0	36.8	0.088	38	1
1	109	38	18	120	23.1	0.407	26	0
1	108	88	19	0	27.1	0.4	24	0
6	96	0	0	0	23.7	0.19	28	0
1	124	74	36	0	27.8	0.1	30	0
7	150	78	29	126	35.2	0.692	54	1
4	183	0	0	0	28.4	0.212	36	1

1	124	60	32	0	35.8	0.514	21	0
1	181	78	42	293	40	1.258	22	1
1	92	62	25	41	19.5	0.482	25	0
0	152	82	39	272	41.5	0.27	27	0
1	111	62	13	182	24	0.138	23	0
3	106	54	21	158	30.9	0.292	24	0
3	174	58	22	194	32.9	0.593	36	1
7	168	88	42	321	38.2	0.787	40	1
6	105	80	28	0	32.5	0.878	26	0
11	138	74	26	144	36.1	0.557	50	1
3	106	72	0	0	25.8	0.207	27	0
6	117	96	0	0	28.7	0.157	30	0
2	68	62	13	15	20.1	0.257	23	0
9	112	82	24	0	28.2	1.282	50	1
0	119	0	0	0	32.4	0.141	24	1
2	112	86	42	160	38.4	0.246	28	0
2	92	76	20	0	24.2	1.698	28	0
6	183	94	0	0	40.8	1.461	45	0
0	94	70	27	115	43.5	0.347	21	0
2	108	64	0	0	30.8	0.158	21	0
4	90	88	47	54	37.7	0.362	29	0
0	125	68	0	0	24.7	0.206	21	0
0	132	78	0	0	32.4	0.393	21	0
5	128	80	0	0	34.6	0.144	45	0
4	94	65	22	0	24.7	0.148	21	0
7	114	64	0	0	27.4	0.732	34	1
0	102	78	40	90	34.5	0.238	24	0
2	111	60	0	0	26.2	0.343	23	0
1	128	82	17	183	27.5	0.115	22	0
10	92	62	0	0	25.9	0.167	31	0
13	104	72	0	0	31.2	0.465	38	1
5	104	74	0	0	28.8	0.153	48	0
2	94	76	18	66	31.6	0.649	23	0
7	97	76	32	91	40.9	0.871	32	1
1	100	74	12	46	19.5	0.149	28	0
0	102	86	17	105	29.3	0.695	27	0

4	128	70	0	0	34.3	0.303	24	0
6	147	80	0	0	29.5	0.178	50	1
4	90	0	0	0	28	0.61	31	0
3	103	72	30	152	27.6	0.73	27	0
2	157	74	35	440	39.4	0.134	30	0
1	167	74	17	144	23.4	0.447	33	1
0	179	50	36	159	37.8	0.455	22	1
11	136	84	35	130	28.3	0.26	42	1
0	107	60	25	0	26.4	0.133	23	0
1	91	54	25	100	25.2	0.234	23	0
1	117	60	23	106	33.8	0.466	27	0
5	123	74	40	77	34.1	0.269	28	0
2	120	54	0	0	26.8	0.455	27	0
1	106	70	28	135	34.2	0.142	22	0
2	155	52	27	540	38.7	0.24	25	1
2	101	58	35	90	21.8	0.155	22	0
1	120	80	48	200	38.9	1.162	41	0
11	127	106	0	0	39	0.19	51	0
3	80	82	31	70	34.2	1.292	27	1
10	162	84	0	0	27.7	0.182	54	0
1	199	76	43	0	42.9	1.394	22	1
8	167	106	46	231	37.6	0.165	43	1
9	145	80	46	130	37.9	0.637	40	1
6	115	60	39	0	33.7	0.245	40	1
1	112	80	45	132	34.8	0.217	24	0
4	145	82	18	0	32.5	0.235	70	1
10	111	70	27	0	27.5	0.141	40	1
6	98	58	33	190	34	0.43	43	0
9	154	78	30	100	30.9	0.164	45	0
6	165	68	26	168	33.6	0.631	49	0
1	99	58	10	0	25.4	0.551	21	0
10	68	106	23	49	35.5	0.285	47	0
3	123	100	35	240	57.3	0.88	22	0
8	91	82	0	0	35.6	0.587	68	0
6	195	70	0	0	30.9	0.328	31	1
9	156	86	0	0	24.8	0.23	53	1

0	93	60	0	0	35.3	0.263	25	0
3	121	52	0	0	36	0.127	25	1
2	101	58	17	265	24.2	0.614	23	0
2	56	56	28	45	24.2	0.332	22	0
0	162	76	36	0	49.6	0.364	26	1
0	95	64	39	105	44.6	0.366	22	0
4	125	80	0	0	32.3	0.536	27	1
5	136	82	0	0	0	0.64	69	0
2	129	74	26	205	33.2	0.591	25	0
3	130	64	0	0	23.1	0.314	22	0
1	107	50	19	0	28.3	0.181	29	0
1	140	74	26	180	24.1	0.828	23	0
1	144	82	46	180	46.1	0.335	46	1
8	107	80	0	0	24.6	0.856	34	0
13	158	114	0	0	42.3	0.257	44	1
2	121	70	32	95	39.1	0.886	23	0
7	129	68	49	125	38.5	0.439	43	1
2	90	60	0	0	23.5	0.191	25	0
7	142	90	24	480	30.4	0.128	43	1
3	169	74	19	125	29.9	0.268	31	1
0	99	0	0	0	25	0.253	22	0
4	127	88	11	155	34.5	0.598	28	0
4	118	70	0	0	44.5	0.904	26	0
2	122	76	27	200	35.9	0.483	26	0
6	125	78	31	0	27.6	0.565	49	1
1	168	88	29	0	35	0.905	52	1
2	129	0	0	0	38.5	0.304	41	0
4	110	76	20	100	28.4	0.118	27	0
6	80	80	36	0	39.8	0.177	28	0
10	115	0	0	0	0	0.261	30	1
2	127	46	21	335	34.4	0.176	22	0
9	164	78	0	0	32.8	0.148	45	1
2	93	64	32	160	38	0.674	23	1
3	158	64	13	387	31.2	0.295	24	0
5	126	78	27	22	29.6	0.439	40	0
10	129	62	36	0	41.2	0.441	38	1

0	134	58	20	291	26.4	0.352	21	0
3	102	74	0	0	29.5	0.121	32	0
7	187	50	33	392	33.9	0.826	34	1
3	173	78	39	185	33.8	0.97	31	1
10	94	72	18	0	23.1	0.595	56	0
1	108	60	46	178	35.5	0.415	24	0
5	97	76	27	0	35.6	0.378	52	1
4	83	86	19	0	29.3	0.317	34	0
1	114	66	36	200	38.1	0.289	21	0
1	149	68	29	127	29.3	0.349	42	1
5	117	86	30	105	39.1	0.251	42	0
1	111	94	0	0	32.8	0.265	45	0
4	112	78	40	0	39.4	0.236	38	0
1	116	78	29	180	36.1	0.496	25	0
0	141	84	26	0	32.4	0.433	22	0
2	175	88	0	0	22.9	0.326	22	0
2	92	52	0	0	30.1	0.141	22	0
3	130	78	23	79	28.4	0.323	34	1
8	120	86	0	0	28.4	0.259	22	1
2	174	88	37	120	44.5	0.646	24	1
2	106	56	27	165	29	0.426	22	0
2	105	75	0	0	23.3	0.56	53	0
4	95	60	32	0	35.4	0.284	28	0
0	126	86	27	120	27.4	0.515	21	0
8	65	72	23	0	32	0.6	42	0
2	99	60	17	160	36.6	0.453	21	0
1	102	74	0	0	39.5	0.293	42	1
11	120	80	37	150	42.3	0.785	48	1
3	102	44	20	94	30.8	0.4	26	0
1	109	58	18	116	28.5	0.219	22	0
9	140	94	0	0	32.7	0.734	45	1
13	153	88	37	140	40.6	1.174	39	0
12	100	84	33	105	30	0.488	46	0
1	147	94	41	0	49.3	0.358	27	1
1	81	74	41	57	46.3	1.096	32	0
3	187	70	22	200	36.4	0.408	36	1

6	162	62	0	0	24.3	0.178	50	1
4	136	70	0	0	31.2	1.182	22	1
1	121	78	39	74	39	0.261	28	0
3	108	62	24	0	26	0.223	25	0
0	181	88	44	510	43.3	0.222	26	1
8	154	78	32	0	32.4	0.443	45	1
1	128	88	39	110	36.5	1.057	37	1
7	137	90	41	0	32	0.391	39	0
0	123	72	0	0	36.3	0.258	52	1
1	106	76	0	0	37.5	0.197	26	0
6	190	92	0	0	35.5	0.278	66	1
2	88	58	26	16	28.4	0.766	22	0
9	170	74	31	0	44	0.403	43	1
9	89	62	0	0	22.5	0.142	33	0
10	101	76	48	180	32.9	0.171	63	0
2	122	70	27	0	36.8	0.34	27	0
5	121	72	23	112	26.2	0.245	30	0
1	126	60	0	0	30.1	0.349	47	1
1	93	70	31	0	30.4	0.315	23	0

Table 3: Pima Indian Diabetes Dataset

Table 4 shows Breast Cancer Analysis Dataset.

Id	Clump_thickness	Size_uniformity	Shape_uniformity	Marginal_adhesion	Epithelial_size	Bare_nucleoli	Bland_chromatin	Normal_nucleoli	Mitoses	Class
1000025	5	1	1	1	2	1	3	1	1	2
1002945	5	4	4	5	7	10	3	2	1	2

1015425	3	1	1	1	2	2	3	1	1	2
1016277	6	8	8	1	3	4	3	7	1	2
1017023	4	1	1	3	2	1	3	1	1	2
1017122	8	10	10	8	7	10	9	7	1	4
1018099	1	1	1	1	2	10	3	1	1	2
1018561	2	1	2	1	2	1	3	1	1	2
1033078	2	1	1	1	2	1	1	1	5	2
1033078	4	2	1	1	2	1	2	1	1	2
1035283	1	1	1	1	1	1	3	1	1	2
1036172	2	1	1	1	2	1	2	1	1	2
1041801	5	3	3	3	2	3	4	4	1	4
1043999	1	1	1	1	2	3	3	1	1	2
1044572	8	7	5	10	7	9	5	5	4	4
1047630	7	4	6	4	6	1	4	3	1	4
1048672	4	1	1	1	2	1	2	1	1	2
1049815	4	1	1	1	2	1	3	1	1	2
1050670	10	7	7	6	4	10	4	1	2	4
1050718	6	1	1	1	2	1	3	1	1	2
1054590	7	3	2	10	5	10	5	4	4	4
1054593	10	5	5	3	6	7	7	10	1	4
1056784	3	1	1	1	2	1	2	1	1	2
1057013	8	4	5	1	2	?	7	3	1	4
1059552	1	1	1	1	2	1	3	1	1	2
1065726	5	2	3	4	2	7	3	6	1	4
1066373	3	2	1	1	1	1	2	1	1	2
1066979	5	1	1	1	2	1	2	1	1	2
1067444	2	1	1	1	2	1	2	1	1	2
1070935	1	1	3	1	2	1	1	1	1	2
1070935	3	1	1	1	1	1	2	1	1	2
1071760	2	1	1	1	2	1	3	1	1	2
1072179	10	7	7	3	8	5	7	4	3	4
1074610	2	1	1	2	2	1	3	1	1	2
1075123	3	1	2	1	2	1	2	1	1	2
1079304	2	1	1	1	2	1	2	1	1	2
1080185	10	10	10	8	6	1	8	9	1	4
1081791	6	2	1	1	1	1	7	1	1	2

1084584	5	4	4	9	2	10	5	6	1	4
1091262	2	5	3	3	6	7	7	5	1	4
1096800	6	6	6	9	6	?	7	8	1	2
1099510	10	4	3	1	3	3	6	5	2	4
1100524	6	10	10	2	8	10	7	3	3	4
1102573	5	6	5	6	10	1	3	1	1	4
1103608	10	10	10	4	8	1	8	10	1	4
1103722	1	1	1	1	2	1	2	1	2	2
1105257	3	7	7	4	4	9	4	8	1	4
1105524	1	1	1	1	2	1	2	1	1	2
1106095	4	1	1	3	2	1	3	1	1	2
1106829	7	8	7	2	4	8	3	8	2	4
1108370	9	5	8	1	2	3	2	1	5	4
1108449	5	3	3	4	2	4	3	4	1	4
1110102	10	3	6	2	3	5	4	10	2	4
1110503	5	5	5	8	10	8	7	3	7	4
1110524	10	5	5	6	8	8	7	1	1	4
1111249	10	6	6	3	4	5	3	6	1	4
1112209	8	10	10	1	3	6	3	9	1	4
1113038	8	2	4	1	5	1	5	4	4	4
1113483	5	2	3	1	6	10	5	1	1	4
1113906	9	5	5	2	2	2	5	1	1	4
1115282	5	3	5	5	3	3	4	10	1	4
1115293	1	1	1	1	2	2	2	1	1	2
1116116	9	10	10	1	10	8	3	3	1	4
1116132	6	3	4	1	5	2	3	9	1	4
1116192	1	1	1	1	2	1	2	1	1	2
1116998	10	4	2	1	3	2	4	3	10	4
1117152	4	1	1	1	2	1	3	1	1	2
1118039	5	3	4	1	8	10	4	9	1	4
1120559	8	3	8	3	4	9	8	9	8	4
1121732	1	1	1	1	2	1	3	2	1	2
1121919	5	1	3	1	2	1	2	1	1	2
1123061	6	10	2	8	10	2	7	8	10	4
1124651	1	3	3	2	2	1	7	2	1	2
1125035	9	4	5	10	6	10	4	8	1	4

1126417	10	6	4	1	3	4	3	2	3	4
1131294	1	1	2	1	2	2	4	2	1	2
1132347	1	1	4	1	2	1	2	1	1	2
1133041	5	3	1	2	2	1	2	1	1	2
1133136	3	1	1	1	2	3	3	1	1	2
1136142	2	1	1	1	3	1	2	1	1	2
1137156	2	2	2	1	1	1	7	1	1	2
1143978	4	1	1	2	2	1	2	1	1	2
1143978	5	2	1	1	2	1	3	1	1	2
1147044	3	1	1	1	2	2	7	1	1	2
1147699	3	5	7	8	8	9	7	10	7	4
1147748	5	10	6	1	10	4	4	10	10	4
1148278	3	3	6	4	5	8	4	4	1	4
1148873	3	6	6	6	5	10	6	8	3	4
1152331	4	1	1	1	2	1	3	1	1	2
1155546	2	1	1	2	3	1	2	1	1	2
1156272	1	1	1	1	2	1	3	1	1	2
1156948	3	1	1	2	2	1	1	1	1	2
1157734	4	1	1	1	2	1	3	1	1	2
1158247	1	1	1	1	2	1	2	1	1	2
1160476	2	1	1	1	2	1	3	1	1	2
1164066	1	1	1	1	2	1	3	1	1	2
1165297	2	1	1	2	2	1	1	1	1	2
1165790	5	1	1	1	2	1	3	1	1	2
1165926	9	6	9	2	10	6	2	9	10	4
1166630	7	5	6	10	5	10	7	9	4	4
1166654	10	3	5	1	10	5	3	10	2	4
1167439	2	3	4	4	2	5	2	5	1	4
1167471	4	1	2	1	2	1	3	1	1	2
1168359	8	2	3	1	6	3	7	1	1	4
1168736	10	10	10	10	10	1	8	8	8	4
1169049	7	3	4	4	3	3	3	2	7	4
1170419	10	10	10	8	2	10	4	1	1	4
1170420	1	6	8	10	8	10	5	7	1	4
1171710	1	1	1	1	2	1	2	3	1	2
1171710	6	5	4	4	3	9	7	8	3	4

1171795	1	3	1	2	2	2	5	3	2	2
1171845	8	6	4	3	5	9	3	1	1	4
1172152	10	3	3	10	2	10	7	3	3	4
1173216	10	10	10	3	10	8	8	1	1	4
1173235	3	3	2	1	2	3	3	1	1	2
1173347	1	1	1	1	2	5	1	1	1	2
1173347	8	3	3	1	2	2	3	2	1	2
1173509	4	5	5	10	4	10	7	5	8	4
1173514	1	1	1	1	4	3	1	1	1	2
1173681	3	2	1	1	2	2	3	1	1	2
1174057	1	1	2	2	2	1	3	1	1	2
1174057	4	2	1	1	2	2	3	1	1	2
1174131	10	10	10	2	10	10	5	3	3	4
1174428	5	3	5	1	8	10	5	3	1	4
1175937	5	4	6	7	9	7	8	10	1	4
1176406	1	1	1	1	2	1	2	1	1	2
1176881	7	5	3	7	4	10	7	5	5	4
1177027	3	1	1	1	2	1	3	1	1	2
1177399	8	3	5	4	5	10	1	6	2	4
1177512	1	1	1	1	10	1	1	1	1	2
1178580	5	1	3	1	2	1	2	1	1	2
1179818	2	1	1	1	2	1	3	1	1	2
1180194	5	10	8	10	8	10	3	6	3	4
1180523	3	1	1	1	2	1	2	2	1	2
1180831	3	1	1	1	3	1	2	1	1	2
1181356	5	1	1	1	2	2	3	3	1	2
1182404	4	1	1	1	2	1	2	1	1	2
1182410	3	1	1	1	2	1	1	1	1	2
1183240	4	1	2	1	2	1	2	1	1	2
1183246	1	1	1	1	1	?	2	1	1	2
1183516	3	1	1	1	2	1	1	1	1	2
1183911	2	1	1	1	2	1	1	1	1	2
1183983	9	5	5	4	4	5	4	3	3	4
1184184	1	1	1	1	2	5	1	1	1	2
1184241	2	1	1	1	2	1	2	1	1	2
1184840	1	1	3	1	2	?	2	1	1	2

1185609	3	4	5	2	6	8	4	1	1	4
1185610	1	1	1	1	3	2	2	1	1	2
1187457	3	1	1	3	8	1	5	8	1	2
1187805	8	8	7	4	10	10	7	8	7	4
1188472	1	1	1	1	1	1	3	1	1	2
1189266	7	2	4	1	6	10	5	4	3	4
1189286	10	10	8	6	4	5	8	10	1	4
1190394	4	1	1	1	2	3	1	1	1	2
1190485	1	1	1	1	2	1	1	1	1	2
1192325	5	5	5	6	3	10	3	1	1	4
1193091	1	2	2	1	2	1	2	1	1	2
1193210	2	1	1	1	2	1	3	1	1	2
1193683	1	1	2	1	3	?	1	1	1	2
1196295	9	9	10	3	6	10	7	10	6	4
1196915	10	7	7	4	5	10	5	7	2	4
1197080	4	1	1	1	2	1	3	2	1	2
1197270	3	1	1	1	2	1	3	1	1	2
1197440	1	1	1	2	1	3	1	1	7	2
1197510	5	1	1	1	2	?	3	1	1	2
1197979	4	1	1	1	2	2	3	2	1	2
1197993	5	6	7	8	8	10	3	10	3	4
1198128	10	8	10	10	6	1	3	1	10	4
1198641	3	1	1	1	2	1	3	1	1	2
1199219	1	1	1	2	1	1	1	1	1	2
1199731	3	1	1	1	2	1	1	1	1	2
1199983	1	1	1	1	2	1	3	1	1	2
1200772	1	1	1	1	2	1	2	1	1	2
1200847	6	10	10	10	8	10	10	10	7	4
1200892	8	6	5	4	3	10	6	1	1	4
1200952	5	8	7	7	10	10	5	7	1	4
1201834	2	1	1	1	2	1	3	1	1	2
1201936	5	10	10	3	8	1	5	10	3	4
1202125	4	1	1	1	2	1	3	1	1	2
1202812	5	3	3	3	6	10	3	1	1	4
1203096	1	1	1	1	1	1	3	1	1	2
1204242	1	1	1	1	2	1	1	1	1	2

1204898	6	1	1	1	2	1	3	1	1	2
1205138	5	8	8	8	5	10	7	8	1	4
1205579	8	7	6	4	4	10	5	1	1	4
1206089	2	1	1	1	1	1	3	1	1	2
1206695	1	5	8	6	5	8	7	10	1	4
1206841	10	5	6	10	6	10	7	7	10	4
1207986	5	8	4	10	5	8	9	10	1	4
1208301	1	2	3	1	2	1	3	1	1	2
1210963	10	10	10	8	6	8	7	10	1	4
1211202	7	5	10	10	10	10	4	10	3	4
1212232	5	1	1	1	2	1	2	1	1	2
1212251	1	1	1	1	2	1	3	1	1	2
1212422	3	1	1	1	2	1	3	1	1	2
1212422	4	1	1	1	2	1	3	1	1	2
1213375	8	4	4	5	4	7	7	8	2	2
1213383	5	1	1	4	2	1	3	1	1	2
1214092	1	1	1	1	2	1	1	1	1	2
1214556	3	1	1	1	2	1	2	1	1	2
1214966	9	7	7	5	5	10	7	8	3	4
1216694	10	8	8	4	10	10	8	1	1	4
1216947	1	1	1	1	2	1	3	1	1	2
1217051	5	1	1	1	2	1	3	1	1	2
1217264	1	1	1	1	2	1	3	1	1	2
1218105	5	10	10	9	6	10	7	10	5	4
1218741	10	10	9	3	7	5	3	5	1	4
1218860	1	1	1	1	1	1	3	1	1	2
1218860	1	1	1	1	1	1	3	1	1	2
1219406	5	1	1	1	1	1	3	1	1	2
1219525	8	10	10	10	5	10	8	10	6	4
1219859	8	10	8	8	4	8	7	7	1	4
1220330	1	1	1	1	2	1	3	1	1	2
1221863	10	10	10	10	7	10	7	10	4	4
1222047	10	10	10	10	3	10	10	6	1	4
1222936	8	7	8	7	5	5	5	10	2	4
1223282	1	1	1	1	2	1	2	1	1	2
1223426	1	1	1	1	2	1	3	1	1	2

1223793	6	10	7	7	6	4	8	10	2	4
1223967	6	1	3	1	2	1	3	1	1	2
1224329	1	1	1	2	2	1	3	1	1	2
1225799	10	6	4	3	10	10	9	10	1	4
1226012	4	1	1	3	1	5	2	1	1	4
1226612	7	5	6	3	3	8	7	4	1	4
1227210	10	5	5	6	3	10	7	9	2	4
1227244	1	1	1	1	2	1	2	1	1	2
1227481	10	5	7	4	4	10	8	9	1	4
1228152	8	9	9	5	3	5	7	7	1	4
1228311	1	1	1	1	1	1	3	1	1	2
1230175	10	10	10	3	10	10	9	10	1	4
1230688	7	4	7	4	3	7	7	6	1	4
1231387	6	8	7	5	6	8	8	9	2	4
1231706	8	4	6	3	3	1	4	3	1	2
1232225	10	4	5	5	5	10	4	1	1	4
1236043	3	3	2	1	3	1	3	6	1	2
1241232	3	1	4	1	2	?	3	1	1	2
1241559	10	8	8	2	8	10	4	8	10	4
1241679	9	8	8	5	6	2	4	10	4	4
1242364	8	10	10	8	6	9	3	10	10	4
1243256	10	4	3	2	3	10	5	3	2	4
1270479	5	1	3	3	2	2	2	3	1	2
1276091	3	1	1	3	1	1	3	1	1	2
1277018	2	1	1	1	2	1	3	1	1	2
128059	1	1	1	1	2	5	5	1	1	2
1285531	1	1	1	1	2	1	3	1	1	2
1287775	5	1	1	2	2	2	3	1	1	2
144888	8	10	10	8	5	10	7	8	1	4
145447	8	4	4	1	2	9	3	3	1	4
167528	4	1	1	1	2	1	3	6	1	2
169356	3	1	1	1	2	?	3	1	1	2
183913	1	2	2	1	2	1	1	1	1	2
191250	10	4	4	10	2	10	5	3	3	4
1017023	6	3	3	5	3	10	3	5	3	2
1100524	6	10	10	2	8	10	7	3	3	4

1116116	9	10	10	1	10	8	3	3	1	4
1168736	5	6	6	2	4	10	3	6	1	4
1182404	3	1	1	1	2	1	1	1	1	2
1182404	3	1	1	1	2	1	2	1	1	2
1198641	3	1	1	1	2	1	3	1	1	2
242970	5	7	7	1	5	8	3	4	1	2
255644	10	5	8	10	3	10	5	1	3	4
263538	5	10	10	6	10	10	10	6	5	4
274137	8	8	9	4	5	10	7	8	1	4
303213	10	4	4	10	6	10	5	5	1	4
314428	7	9	4	10	10	3	5	3	3	4
1182404	5	1	4	1	2	1	3	2	1	2
1198641	10	10	6	3	3	10	4	3	2	4
320675	3	3	5	2	3	10	7	1	1	4
324427	10	8	8	2	3	4	8	7	8	4
385103	1	1	1	1	2	1	3	1	1	2
390840	8	4	7	1	3	10	3	9	2	4
411453	5	1	1	1	2	1	3	1	1	2
320675	3	3	5	2	3	10	7	1	1	4
428903	7	2	4	1	3	4	3	3	1	4
431495	3	1	1	1	2	1	3	2	1	2
432809	3	1	3	1	2	?	2	1	1	2
434518	3	1	1	1	2	1	2	1	1	2
452264	1	1	1	1	2	1	2	1	1	2
456282	1	1	1	1	2	1	3	1	1	2
476903	10	5	7	3	3	7	3	3	8	4
486283	3	1	1	1	2	1	3	1	1	2
486662	2	1	1	2	2	1	3	1	1	2
488173	1	4	3	10	4	10	5	6	1	4
492268	10	4	6	1	2	10	5	3	1	4
508234	7	4	5	10	2	10	3	8	2	4
527363	8	10	10	10	8	10	10	7	3	4
529329	10	10	10	10	10	10	4	10	10	4
535331	3	1	1	1	3	1	2	1	1	2
543558	6	1	3	1	4	5	5	10	1	4
555977	5	6	6	8	6	10	4	10	4	4

560680	1	1	1	1	2	1	1	1	1	2
561477	1	1	1	1	2	1	3	1	1	2
563649	8	8	8	1	2	?	6	10	1	4
601265	10	4	4	6	2	10	2	3	1	4
606140	1	1	1	1	2	?	2	1	1	2
606722	5	5	7	8	6	10	7	4	1	4
616240	5	3	4	3	4	5	4	7	1	2
61634	5	4	3	1	2	?	2	3	1	2
625201	8	2	1	1	5	1	1	1	1	2
63375	9	1	2	6	4	10	7	7	2	4
635844	8	4	10	5	4	4	7	10	1	4
636130	1	1	1	1	2	1	3	1	1	2
640744	10	10	10	7	9	10	7	10	10	4
646904	1	1	1	1	2	1	3	1	1	2
653777	8	3	4	9	3	10	3	3	1	4
659642	10	8	4	4	4	10	3	10	4	4
666090	1	1	1	1	2	1	3	1	1	2
666942	1	1	1	1	2	1	3	1	1	2
667204	7	8	7	6	4	3	8	8	4	4
673637	3	1	1	1	2	5	5	1	1	2
684955	2	1	1	1	3	1	2	1	1	2
688033	1	1	1	1	2	1	1	1	1	2
691628	8	6	4	10	10	1	3	5	1	4
693702	1	1	1	1	2	1	1	1	1	2
704097	1	1	1	1	1	1	2	1	1	2
704168	4	6	5	6	7	?	4	9	1	2
706426	5	5	5	2	5	10	4	3	1	4
709287	6	8	7	8	6	8	8	9	1	4
718641	1	1	1	1	5	1	3	1	1	2
721482	4	4	4	4	6	5	7	3	1	2
730881	7	6	3	2	5	10	7	4	6	4
733639	3	1	1	1	2	?	3	1	1	2
733639	3	1	1	1	2	1	3	1	1	2
733823	5	4	6	10	2	10	4	1	1	4
740492	1	1	1	1	2	1	3	1	1	2
743348	3	2	2	1	2	1	2	3	1	2

752904	10	1	1	1	2	10	5	4	1	4
756136	1	1	1	1	2	1	2	1	1	2
760001	8	10	3	2	6	4	3	10	1	4
760239	10	4	6	4	5	10	7	1	1	4
76389	10	4	7	2	2	8	6	1	1	4
764974	5	1	1	1	2	1	3	1	2	2
770066	5	2	2	2	2	1	2	2	1	2
785208	5	4	6	6	4	10	4	3	1	4
785615	8	6	7	3	3	10	3	4	2	4
792744	1	1	1	1	2	1	1	1	1	2
797327	6	5	5	8	4	10	3	4	1	4
798429	1	1	1	1	2	1	3	1	1	2
704097	1	1	1	1	1	1	2	1	1	2
806423	8	5	5	5	2	10	4	3	1	4
809912	10	3	3	1	2	10	7	6	1	4
810104	1	1	1	1	2	1	3	1	1	2
814265	2	1	1	1	2	1	1	1	1	2
814911	1	1	1	1	2	1	1	1	1	2
822829	7	6	4	8	10	10	9	5	3	4
826923	1	1	1	1	2	1	1	1	1	2
830690	5	2	2	2	3	1	1	3	1	2
831268	1	1	1	1	1	1	1	3	1	2
832226	3	4	4	10	5	1	3	3	1	4
832567	4	2	3	5	3	8	7	6	1	4
836433	5	1	1	3	2	1	1	1	1	2
837082	2	1	1	1	2	1	3	1	1	2
846832	3	4	5	3	7	3	4	6	1	2
850831	2	7	10	10	7	10	4	9	4	4
855524	1	1	1	1	2	1	2	1	1	2
857774	4	1	1	1	3	1	2	2	1	2
859164	5	3	3	1	3	3	3	3	3	4
859350	8	10	10	7	10	10	7	3	8	4
866325	8	10	5	3	8	4	4	10	3	4
873549	10	3	5	4	3	7	3	5	3	4
877291	6	10	10	10	10	10	8	10	10	4
877943	3	10	3	10	6	10	5	1	4	4

888169	3	2	2	1	4	3	2	1	1	2
888523	4	4	4	2	2	3	2	1	1	2
896404	2	1	1	1	2	1	3	1	1	2
897172	2	1	1	1	2	1	2	1	1	2
95719	6	10	10	10	8	10	7	10	7	4
160296	5	8	8	10	5	10	8	10	3	4
342245	1	1	3	1	2	1	1	1	1	2
428598	1	1	3	1	1	1	2	1	1	2
492561	4	3	2	1	3	1	2	1	1	2
493452	1	1	3	1	2	1	1	1	1	2
493452	4	1	2	1	2	1	2	1	1	2
521441	5	1	1	2	2	1	2	1	1	2
560680	3	1	2	1	2	1	2	1	1	2
636437	1	1	1	1	2	1	1	1	1	2
640712	1	1	1	1	2	1	2	1	1	2
654244	1	1	1	1	1	1	2	1	1	2
657753	3	1	1	4	3	1	2	2	1	2
685977	5	3	4	1	4	1	3	1	1	2
805448	1	1	1	1	2	1	1	1	1	2
846423	10	6	3	6	4	10	7	8	4	4
1002504	3	2	2	2	2	1	3	2	1	2
1022257	2	1	1	1	2	1	1	1	1	2
1026122	2	1	1	1	2	1	1	1	1	2
1071084	3	3	2	2	3	1	1	2	3	2
1080233	7	6	6	3	2	10	7	1	1	4
1114570	5	3	3	2	3	1	3	1	1	2
1114570	2	1	1	1	2	1	2	2	1	2
1116715	5	1	1	1	3	2	2	2	1	2
1131411	1	1	1	2	2	1	2	1	1	2
1151734	10	8	7	4	3	10	7	9	1	4
1156017	3	1	1	1	2	1	2	1	1	2
1158247	1	1	1	1	1	1	1	1	1	2
1158405	1	2	3	1	2	1	2	1	1	2
1168278	3	1	1	1	2	1	2	1	1	2
1176187	3	1	1	1	2	1	3	1	1	2
1196263	4	1	1	1	2	1	1	1	1	2

1196475	3	2	1	1	2	1	2	2	1	2
1206314	1	2	3	1	2	1	1	1	1	2
1211265	3	10	8	7	6	9	9	3	8	4
1213784	3	1	1	1	2	1	1	1	1	2
1223003	5	3	3	1	2	1	2	1	1	2
1223306	3	1	1	1	2	4	1	1	1	2
1223543	1	2	1	3	2	1	1	2	1	2
1229929	1	1	1	1	2	1	2	1	1	2
1231853	4	2	2	1	2	1	2	1	1	2
1234554	1	1	1	1	2	1	2	1	1	2
1236837	2	3	2	2	2	2	3	1	1	2
1237674	3	1	2	1	2	1	2	1	1	2
1238021	1	1	1	1	2	1	2	1	1	2
1238464	1	1	1	1	1	?	2	1	1	2
1238633	10	10	10	6	8	4	8	5	1	4
1238915	5	1	2	1	2	1	3	1	1	2
1238948	8	5	6	2	3	10	6	6	1	4
1239232	3	3	2	6	3	3	3	5	1	2
1239347	8	7	8	5	10	10	7	2	1	4
1239967	1	1	1	1	2	1	2	1	1	2
1240337	5	2	2	2	2	2	3	2	2	2
1253505	2	3	1	1	5	1	1	1	1	2
1255384	3	2	2	3	2	3	3	1	1	2
1257200	10	10	10	7	10	10	8	2	1	4
1257648	4	3	3	1	2	1	3	3	1	2
1257815	5	1	3	1	2	1	2	1	1	2
1257938	3	1	1	1	2	1	1	1	1	2
1258549	9	10	10	10	10	10	10	10	1	4
1258556	5	3	6	1	2	1	1	1	1	2
1266154	8	7	8	2	4	2	5	10	1	4
1272039	1	1	1	1	2	1	2	1	1	2
1276091	2	1	1	1	2	1	2	1	1	2
1276091	1	3	1	1	2	1	2	2	1	2
1276091	5	1	1	3	4	1	3	2	1	2
1277629	5	1	1	1	2	1	2	2	1	2
1293439	3	2	2	3	2	1	1	1	1	2

1293439	6	9	7	5	5	8	4	2	1	2
1294562	10	8	10	1	3	10	5	1	1	4
1295186	10	10	10	1	6	1	2	8	1	4
527337	4	1	1	1	2	1	1	1	1	2
558538	4	1	3	3	2	1	1	1	1	2
566509	5	1	1	1	2	1	1	1	1	2
608157	10	4	3	10	4	10	10	1	1	4
677910	5	2	2	4	2	4	1	1	1	2
734111	1	1	1	3	2	3	1	1	1	2
734111	1	1	1	1	2	2	1	1	1	2
780555	5	1	1	6	3	1	2	1	1	2
827627	2	1	1	1	2	1	1	1	1	2
1049837	1	1	1	1	2	1	1	1	1	2
1058849	5	1	1	1	2	1	1	1	1	2
1182404	1	1	1	1	1	1	1	1	1	2
1193544	5	7	9	8	6	10	8	10	1	4
1201870	4	1	1	3	1	1	2	1	1	2
1202253	5	1	1	1	2	1	1	1	1	2
1227081	3	1	1	3	2	1	1	1	1	2
1230994	4	5	5	8	6	10	10	7	1	4
1238410	2	3	1	1	3	1	1	1	1	2
1246562	10	2	2	1	2	6	1	1	2	4
1257470	10	6	5	8	5	10	8	6	1	4
1259008	8	8	9	6	6	3	10	10	1	4
1266124	5	1	2	1	2	1	1	1	1	2
1267898	5	1	3	1	2	1	1	1	1	2
1268313	5	1	1	3	2	1	1	1	1	2
1268804	3	1	1	1	2	5	1	1	1	2
1276091	6	1	1	3	2	1	1	1	1	2
1280258	4	1	1	1	2	1	1	2	1	2
1293966	4	1	1	1	2	1	1	1	1	2
1296572	10	9	8	7	6	4	7	10	3	4
1298416	10	6	6	2	4	10	9	7	1	4
1299596	6	6	6	5	4	10	7	6	2	4
1105524	4	1	1	1	2	1	1	1	1	2
1181685	1	1	2	1	2	1	2	1	1	2

1211594	3	1	1	1	1	1	2	1	1	2
1238777	6	1	1	3	2	1	1	1	1	2
1257608	6	1	1	1	1	1	1	1	1	2
1269574	4	1	1	1	2	1	1	1	1	2
1277145	5	1	1	1	2	1	1	1	1	2
1287282	3	1	1	1	2	1	1	1	1	2
1296025	4	1	2	1	2	1	1	1	1	2
1296263	4	1	1	1	2	1	1	1	1	2
1296593	5	2	1	1	2	1	1	1	1	2
1299161	4	8	7	10	4	10	7	5	1	4
1301945	5	1	1	1	1	1	1	1	1	2
1302428	5	3	2	4	2	1	1	1	1	2
1318169	9	10	10	10	10	5	10	10	10	4
474162	8	7	8	5	5	10	9	10	1	4
787451	5	1	2	1	2	1	1	1	1	2
1002025	1	1	1	3	1	3	1	1	1	2
1070522	3	1	1	1	1	1	2	1	1	2
1073960	10	10	10	10	6	10	8	1	5	4
1076352	3	6	4	10	3	3	3	4	1	4
1084139	6	3	2	1	3	4	4	1	1	4
1115293	1	1	1	1	2	1	1	1	1	2
1119189	5	8	9	4	3	10	7	1	1	4
1133991	4	1	1	1	1	1	2	1	1	2
1142706	5	10	10	10	6	10	6	5	2	4
1155967	5	1	2	10	4	5	2	1	1	2
1170945	3	1	1	1	1	1	2	1	1	2
1181567	1	1	1	1	1	1	1	1	1	2
1182404	4	2	1	1	2	1	1	1	1	2
1204558	4	1	1	1	2	1	2	1	1	2
1217952	4	1	1	1	2	1	2	1	1	2
1224565	6	1	1	1	2	1	3	1	1	2
1238186	4	1	1	1	2	1	2	1	1	2
1253917	4	1	1	2	2	1	2	1	1	2
1265899	4	1	1	1	2	1	3	1	1	2
1268766	1	1	1	1	2	1	1	1	1	2
1277268	3	3	1	1	2	1	1	1	1	2

1286943	8	10	10	10	7	5	4	8	7	4
1295508	1	1	1	1	2	4	1	1	1	2
1297327	5	1	1	1	2	1	1	1	1	2
1297522	2	1	1	1	2	1	1	1	1	2
1298360	1	1	1	1	2	1	1	1	1	2
1299924	5	1	1	1	2	1	2	1	1	2
1299994	5	1	1	1	2	1	1	1	1	2
1304595	3	1	1	1	1	1	2	1	1	2
1306282	6	6	7	10	3	10	8	10	2	4
1313325	4	10	4	7	3	10	9	10	1	4
1320077	1	1	1	1	1	1	1	1	1	2
1320077	1	1	1	1	1	1	2	1	1	2
1320304	3	1	2	2	2	1	1	1	1	2
1330439	4	7	8	3	4	10	9	1	1	4
333093	1	1	1	1	3	1	1	1	1	2
369565	4	1	1	1	3	1	1	1	1	2
412300	10	4	5	4	3	5	7	3	1	4
672113	7	5	6	10	4	10	5	3	1	4
749653	3	1	1	1	2	1	2	1	1	2
769612	3	1	1	2	2	1	1	1	1	2
769612	4	1	1	1	2	1	1	1	1	2
798429	4	1	1	1	2	1	3	1	1	2
807657	6	1	3	2	2	1	1	1	1	2
8233704	4	1	1	1	1	1	2	1	1	2
837480	7	4	4	3	4	10	6	9	1	4
867392	4	2	2	1	2	1	2	1	1	2
869828	1	1	1	1	1	1	3	1	1	2
1043068	3	1	1	1	2	1	2	1	1	2
1056171	2	1	1	1	2	1	2	1	1	2
1061990	1	1	3	2	2	1	3	1	1	2
1113061	5	1	1	1	2	1	3	1	1	2
1116192	5	1	2	1	2	1	3	1	1	2
1135090	4	1	1	1	2	1	2	1	1	2
1145420	6	1	1	1	2	1	2	1	1	2
1158157	5	1	1	1	2	2	2	1	1	2
1171578	3	1	1	1	2	1	1	1	1	2

1174841	5	3	1	1	2	1	1	1	1	2
1184586	4	1	1	1	2	1	2	1	1	2
1186936	2	1	3	2	2	1	2	1	1	2
1197527	5	1	1	1	2	1	2	1	1	2
1222464	6	10	10	10	4	10	7	10	1	4
1240603	2	1	1	1	1	1	1	1	1	2
1240603	3	1	1	1	1	1	1	1	1	2
1241035	7	8	3	7	4	5	7	8	2	4
1287971	3	1	1	1	2	1	2	1	1	2
1289391	1	1	1	1	2	1	3	1	1	2
1299924	3	2	2	2	2	1	4	2	1	2
1306339	4	4	2	1	2	5	2	1	2	2
1313658	3	1	1	1	2	1	1	1	1	2
1313982	4	3	1	1	2	1	4	8	1	2
1321264	5	2	2	2	1	1	2	1	1	2
1321321	5	1	1	3	2	1	1	1	1	2
1321348	2	1	1	1	2	1	2	1	1	2
1321931	5	1	1	1	2	1	2	1	1	2
1321942	5	1	1	1	2	1	3	1	1	2
1321942	5	1	1	1	2	1	3	1	1	2
1328331	1	1	1	1	2	1	3	1	1	2
1328755	3	1	1	1	2	1	2	1	1	2
1331405	4	1	1	1	2	1	3	2	1	2
1331412	5	7	10	10	5	10	10	10	1	4
1333104	3	1	2	1	2	1	3	1	1	2
1334071	4	1	1	1	2	3	2	1	1	2
1343068	8	4	4	1	6	10	2	5	2	4
1343374	10	10	8	10	6	5	10	3	1	4
1344121	8	10	4	4	8	10	8	2	1	4
142932	7	6	10	5	3	10	9	10	2	4
183936	3	1	1	1	2	1	2	1	1	2
324382	1	1	1	1	2	1	2	1	1	2
378275	10	9	7	3	4	2	7	7	1	4
385103	5	1	2	1	2	1	3	1	1	2
690557	5	1	1	1	2	1	2	1	1	2
695091	1	1	1	1	2	1	2	1	1	2

695219	1	1	1	1	2	1	2	1	1	2
824249	1	1	1	1	2	1	3	1	1	2
871549	5	1	2	1	2	1	2	1	1	2
878358	5	7	10	6	5	10	7	5	1	4
1107684	6	10	5	5	4	10	6	10	1	4
1115762	3	1	1	1	2	1	1	1	1	2
1217717	5	1	1	6	3	1	1	1	1	2
1239420	1	1	1	1	2	1	1	1	1	2
1254538	8	10	10	10	6	10	10	10	1	4
1261751	5	1	1	1	2	1	2	2	1	2
1268275	9	8	8	9	6	3	4	1	1	4
1272166	5	1	1	1	2	1	1	1	1	2
1294261	4	10	8	5	4	1	10	1	1	4
1295529	2	5	7	6	4	10	7	6	1	4
1298484	10	3	4	5	3	10	4	1	1	4
1311875	5	1	2	1	2	1	1	1	1	2
1315506	4	8	6	3	4	10	7	1	1	4
1320141	5	1	1	1	2	1	2	1	1	2
1325309	4	1	2	1	2	1	2	1	1	2
1333063	5	1	3	1	2	1	3	1	1	2
1333495	3	1	1	1	2	1	2	1	1	2
1334659	5	2	4	1	1	1	1	1	1	2
1336798	3	1	1	1	2	1	2	1	1	2
1344449	1	1	1	1	1	1	2	1	1	2
1350568	4	1	1	1	2	1	2	1	1	2
1352663	5	4	6	8	4	1	8	10	1	4
188336	5	3	2	8	5	10	8	1	2	4
352431	10	5	10	3	5	8	7	8	3	4
353098	4	1	1	2	2	1	1	1	1	2
411453	1	1	1	1	2	1	1	1	1	2
557583	5	10	10	10	10	10	10	1	1	4
636375	5	1	1	1	2	1	1	1	1	2
736150	10	4	3	10	3	10	7	1	2	4
803531	5	10	10	10	5	2	8	5	1	4
822829	8	10	10	10	6	10	10	10	10	4
1016634	2	3	1	1	2	1	2	1	1	2

1031608	2	1	1	1	1	1	2	1	1	2
1041043	4	1	3	1	2	1	2	1	1	2
1042252	3	1	1	1	2	1	2	1	1	2
1057067	1	1	1	1	1	?	1	1	1	2
1061990	4	1	1	1	2	1	2	1	1	2
1073836	5	1	1	1	2	1	2	1	1	2
1083817	3	1	1	1	2	1	2	1	1	2
1096352	6	3	3	3	3	2	6	1	1	2
1140597	7	1	2	3	2	1	2	1	1	2
1149548	1	1	1	1	2	1	1	1	1	2
1174009	5	1	1	2	1	1	2	1	1	2
1183596	3	1	3	1	3	4	1	1	1	2
1190386	4	6	6	5	7	6	7	7	3	4
1190546	2	1	1	1	2	5	1	1	1	2
1213273	2	1	1	1	2	1	1	1	1	2
1218982	4	1	1	1	2	1	1	1	1	2
1225382	6	2	3	1	2	1	1	1	1	2
1235807	5	1	1	1	2	1	2	1	1	2
1238777	1	1	1	1	2	1	1	1	1	2
1253955	8	7	4	4	5	3	5	10	1	4
1257366	3	1	1	1	2	1	1	1	1	2
1260659	3	1	4	1	2	1	1	1	1	2
1268952	10	10	7	8	7	1	10	10	3	4
1275807	4	2	4	3	2	2	2	1	1	2
1277792	4	1	1	1	2	1	1	1	1	2
1277792	5	1	1	3	2	1	1	1	1	2
1285722	4	1	1	3	2	1	1	1	1	2
1288608	3	1	1	1	2	1	2	1	1	2
1290203	3	1	1	1	2	1	2	1	1	2
1294413	1	1	1	1	2	1	1	1	1	2
1299596	2	1	1	1	2	1	1	1	1	2
1303489	3	1	1	1	2	1	2	1	1	2
1311033	1	2	2	1	2	1	1	1	1	2
1311108	1	1	1	3	2	1	1	1	1	2
1315807	5	10	10	10	10	2	10	10	10	4
1318671	3	1	1	1	2	1	2	1	1	2

1319609	3	1	1	2	3	4	1	1	1	2
1323477	1	2	1	3	2	1	2	1	1	2
1324572	5	1	1	1	2	1	2	2	1	2
1324681	4	1	1	1	2	1	2	1	1	2
1325159	3	1	1	1	2	1	3	1	1	2
1326892	3	1	1	1	2	1	2	1	1	2
1330361	5	1	1	1	2	1	2	1	1	2
1333877	5	4	5	1	8	1	3	6	1	2
1334015	7	8	8	7	3	10	7	2	3	4
1334667	1	1	1	1	2	1	1	1	1	2
1339781	1	1	1	1	2	1	2	1	1	2
1339781	4	1	1	1	2	1	3	1	1	2
13454352	1	1	3	1	2	1	2	1	1	2
1345452	1	1	3	1	2	1	2	1	1	2
1345593	3	1	1	3	2	1	2	1	1	2
1347749	1	1	1	1	2	1	1	1	1	2
1347943	5	2	2	2	2	1	1	1	2	2
1348851	3	1	1	1	2	1	3	1	1	2
1350319	5	7	4	1	6	1	7	10	3	4
1350423	5	10	10	8	5	5	7	10	1	4
1352848	3	10	7	8	5	8	7	4	1	4
1353092	3	2	1	2	2	1	3	1	1	2
1354840	2	1	1	1	2	1	3	1	1	2
1354840	5	3	2	1	3	1	1	1	1	2
1355260	1	1	1	1	2	1	2	1	1	2
1365075	4	1	4	1	2	1	1	1	1	2
1365328	1	1	2	1	2	1	2	1	1	2
1368267	5	1	1	1	2	1	1	1	1	2
1368273	1	1	1	1	2	1	1	1	1	2
1368882	2	1	1	1	2	1	1	1	1	2
1369821	10	10	10	10	5	10	10	10	7	4
1371026	5	10	10	10	4	10	5	6	3	4
1371920	5	1	1	1	2	1	3	2	1	2
466906	1	1	1	1	2	1	1	1	1	2
466906	1	1	1	1	2	1	1	1	1	2
534555	1	1	1	1	2	1	1	1	1	2

536708	1	1	1	1	2	1	1	1	1	2
566346	3	1	1	1	2	1	2	3	1	2
603148	4	1	1	1	2	1	1	1	1	2
654546	1	1	1	1	2	1	1	1	8	2
654546	1	1	1	3	2	1	1	1	1	2
695091	5	10	10	5	4	5	4	4	1	4
714039	3	1	1	1	2	1	1	1	1	2
763235	3	1	1	1	2	1	2	1	2	2
776715	3	1	1	1	3	2	1	1	1	2
841769	2	1	1	1	2	1	1	1	1	2
888820	5	10	10	3	7	3	8	10	2	4
897471	4	8	6	4	3	4	10	6	1	4
897471	4	8	8	5	4	5	10	4	1	4

Table 4: Breast Cancer Analysis Dataset

Appendix – B

Implementation Code

Pima Indian Diabetes Prediction Implementation Code:

```
from __future__ import division

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np
```

```
import scipy as sp
from sklearn import datasets
from sklearn import metrics
from sklearn.pipeline import make_pipeline
from skrebate import ReliefF
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import ExtraTreesClassifier
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB

data_frame = pd.read_csv("pimadataorig.csv")

#Feature Selection: Relief
print("\n\nReliefF:")
features, labels = data_frame.drop('diabetes', axis=1).values, data_frame['diabetes'].values
X_train, X_test, y_train, y_test = train_test_split(features, labels)
fs = ReliefF()
fs.fit(X_train, y_train)
for feature_name, feature_score in zip(data_frame.drop('diabetes', axis=1).columns, fs.feature_importances_):
    # print(feature_name, '\t', feature_score)
    print("{} \t {}".format(feature_name, feature_score))


#Feature Selection: Chi Squire
print("\n\nChi Squire:")
array = data_frame.values
X = array[:,0:8]
```

```
Y = array[:,8]
# feature extraction
test = SelectKBest(score_func=chi2, k=4)
fit = test.fit(X, Y)
# summarize scores
np.set_printoptions(precision=3)
print(fit.scores_)
features = fit.transform(X)
# summarize selected features
print(features[0:5,:])

#Feature Selection using Recursive Feature Elimination
print("\n\nRecursive Feature Elimination:")
array = data_frame.values
X = array[:,0:8]
Y = array[:,8]
# feature extraction
model = LogisticRegression()
rfe = RFE(model, 4)
fit = rfe.fit(X, Y)
print("Num Features: %d" % (fit.n_features_,))
print("Selected Features: %s" % (fit.support_,))
print("Feature Ranking: %s" % (fit.ranking_,))

#Feature Selection using Extra Trees Classifier
print("\n\nExtra Trees Classifier:")
array = data_frame.values
X = array[:,0:8]
Y = array[:,8]
# feature extraction
model = ExtraTreesClassifier()
model.fit(X, Y)
print(model.feature_importances_)

# Here size means plot-size
def corr_heatmap(data_frame, size=10):
```

```
  # Getting correlation using Pandas
  correlation = data_frame.corr()

  # Dividing the plot into subplots for increasing size of plots
  fig, heatmap = plt.subplots(figsize=(size, size))

  # Plotting the correlation heatmap
  heatmap.matshow(correlation)

  # Adding xticks and yticks
  plt.xticks(range(len(correlation.columns)), correlation.columns)
  plt.yticks(range(len(correlation.columns)), correlation.columns)

  # Displaying the graph
  plt.show()

# corr_heatmap(data_frame, 8)

# SFFS
print("\n\nSFFS")
features, labels = data_frame.drop('diabetes', axis=1).values, data_frame['diabetes'].values
X_train, X_test, y_train, y_test = train_test_split(features, labels)

knn = KNeighborsClassifier(n_neighbors=4)
# knn = RandomForestClassifier()
sfs = SFS(knn,
        k_features=8,
        forward=True,
        floating=False,
        verbose=2,
        scoring='accuracy',
        cv=0)

feature_names = ('num_preg', 'glucose_conc', 'diastolic_bp', 'skin_thickness', 'insulin', 'bmi', 'diab_pred', 'age')
sfs = sfs.fit(X_train, y_train, custom_feature_names=feature_names)
# sfs.subsets_
```

```
print("\n{}".format(sfs.subsets_))

# print("\n{}".format(sfs.k_feature_idx_))

# print("\n{}".format(sfs.k_feature_names_))

# print("\nScore: {}".format(sfs.k_score_))


# Pythonic Way

print("\n\nPythonic Way:")

num_true = len(data_frame.loc[data_frame['diabetes'] == True])

num_false = len(data_frame.loc[data_frame['diabetes'] == False])

print("Number of True Cases: {0} ({1:2.2f}%)".format(num_true, (num_true / (num_true + num_false)) * 100))

print("Number of False Cases: {0} ({1:2.2f}%)".format(num_false, (num_false / (num_true + num_false)) * 100))


# Training and Split Data

print("\n\nTraining and Split Data:")


# feature_column_names = ['num_preg', 'glucose_conc', 'diastolic_bp', 'skin_thickness', 'insulin', 'bmi', 'diab_pred', 'age']

# feature_column_names = ['glucose_conc', 'age', 'num_preg', 'skin_thickness'] # ReliefF

# feature_column_names = ['insulin', 'glucose_conc', 'age', 'bmi'] # Chi2

# feature_column_names = ['num_preg', 'glucose_conc', 'bmi', 'diab_pred'] # Recursive

# feature_column_names = ['glucose_conc', 'bmi', 'age', 'diab_pred'] # Extra

# feature_column_names = ['bmi', 'diab_pred', 'age', 'insulin'] # SFFS

feature_column_names = ['glucose_conc', 'bmi', 'age'] # Hybrid


predicted_class_name = ['diabetes']


# Getting feature variable values

X = data_frame[feature_column_names].values

# X_test.columns = data_frame.columns

y = data_frame[predicted_class_name].values


# Saving 30% for testing

split_test_size = 0.30


# Splitting using scikit-learn train_test_split function

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = split_test_size, random_state = 42)
```

```
print("{0:0.2f}% in training set".format((len(X_train)/len(data_frame.index)) * 100))
print("{0:0.2f}% in test set".format((len(X_test)/len(data_frame.index)) * 100))

# Classifier
print("\n\nClassifier:")
# classifier = tree.DecisionTreeClassifier()     # Best
# classifier = svm.SVC()
classifier = svm.SVC(kernel='linear')
# classifier = RandomForestClassifier()
# classifier = GaussianNB()
classifier.fit(X_train, y_train.ravel())

# this returns array of predicted results from test_data
prediction_from_test_data = classifier.predict(X_test)
accuracy = metrics.accuracy_score(y_test, prediction_from_test_data)

print ("Accuracy: {0:0.4f}".format(accuracy))
```

Breast Cancer Prediction Implementation Code:

```
from __future__ import division
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import scipy as sp
# from sklearn.impute import SimpleImputer
from sklearn.preprocessing.imputation import Imputer as SimpleImputer
from sklearn import datasets
```

```
from sklearn import metrics
from sklearn.pipeline import make_pipeline
from skrebate import ReliefF
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import ExtraTreesClassifier
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB


data_frame = pd.read_csv("breastCancer_new.csv")

data_frame.drop(['id'], axis = 1, inplace = True)
# print(data_frame.head(3))

#Feature Selection: Relief
print("\n\nReliefF:")
features, labels = data_frame.drop('class', axis=1).values, data_frame['class'].values
# print(data_frame.head(3))
X_train, X_test, y_train, y_test = train_test_split(features, labels)
fs = ReliefF()
fs.fit(X_train, y_train)
for feature_name, feature_score in zip(data_frame.drop('class', axis=1).columns, fs.feature_importances_):
    # print(feature_name, '\t', feature_score)
    print("{} \t {}".format(feature_name, feature_score))


#Feature Selection: Chi Squire
```

```
print("\n\nChi Squire:")
array = data_frame.values
X = array[:,0:9]
Y = array[:,9]
# feature extraction
test = SelectKBest(score_func=chi2, k=4)
fit = test.fit(X, Y)
# summarize scores
np.set_printoptions(precision=3)
print(fit.scores_)
features = fit.transform(X)
# summarize selected features
print(features[0:5,:])

#Feature Selection using Recursive Feature Elimination
print("\n\nRecursive Feature Elimination:")
array = data_frame.values
X = array[:,0:9]
Y = array[:,9]
# feature extraction
model = LogisticRegression()
rfe = RFE(model, 4)
fit = rfe.fit(X, Y)
print("Num Features: %d" % (fit.n_features_,))
print("Selected Features: %s" % (fit.support_,))
print("Feature Ranking: %s" % (fit.ranking_,))


#Feature Selection using Extra Trees Classifier
print("\n\nExtra Trees Classifier:")
array = data_frame.values
X = array[:,0:9]
Y = array[:,9]
# feature extraction
model = ExtraTreesClassifier()
model.fit(X, Y)
```

```
print(model.feature_importances_)

# Here size means plot-size
def corr_heatmap(data_frame, size=10):
  # Getting correlation using Pandas
  correlation = data_frame.corr()

  # Dividing the plot into subplots for increasing size of plots
  fig, heatmap = plt.subplots(figsize=(size, size))

  # Plotting the correlation heatmap
  heatmap.matshow(correlation)

  # Adding xticks and yticks
  plt.xticks(range(len(correlation.columns)), correlation.columns)
  plt.yticks(range(len(correlation.columns)), correlation.columns)

  # Displaying the graph
  plt.show()

# corr_heatmap(data_frame, 9)

# SFFS
print("\n\nSFFS")
features, labels = data_frame.drop('class', axis=1).values, data_frame['class'].values
X_train, X_test, y_train, y_test = train_test_split(features, labels)

knn = KNeighborsClassifier(n_neighbors=4)
# knn = RandomForestClassifier()
sfs = SFS(knn,
          k_features=9,
          forward=True,
          floating=False,
          verbose=2,
          scoring='accuracy',
          cv=0)
```



```
feature_names =
('clump_thickness','size_uniformity','shape_uniformity','marginal_adhesion','epithelial_size','bare_nucleoli','bland_chromatin','normal_nucleoli','mitoses')

sfs = sfs.fit(X_train, y_train, custom_feature_names=feature_names)

# sfs.subsets_

# print("\n{}".format(sfs.subsets_))

# print("\n{}".format(sfs.k_feature_idx_))

# print("\n{}".format(sfs.k_feature_names_))

# print("\nScore: {}".format(sfs.k_score_))



# Pythonic Way

# print("\n\nPythonic Way:")

# num_true = len(data_frame.loc[data_frame['class'] == True])

# num_false = len(data_frame.loc[data_frame['class'] == False])

# print("Number of True Cases: {0} ({1:2.2f}%)".format(num_true, (num_true / (num_true + num_false)) * 100))

# print("Number of False Cases: {0} ({1:2.2f}%)".format(num_false, (num_false / (num_true + num_false)) * 100))


# Training and Split Data

print("\n\nTraining and Split Data:")



# feature_column_names =
['clump_thickness','size_uniformity','shape_uniformity','marginal_adhesion','epithelial_size','bare_nucleoli','bland_chromatin','normal_nucleoli','mitoses'] # All Features

# feature_column_names = ['bare_nucleoli', 'size_uniformity', 'shape_uniformity', 'epithelial_size'] # ReliefF

# feature_column_names = ['bare_nucleoli', 'size_uniformity', 'shape_uniformity', 'normal_nucleoli'] # Chi Squire

# feature_column_names = ['shape_uniformity', 'bare_nucleoli', 'bland_chromatin', 'mitoses'] # Recursive

# feature_column_names = ['bare_nucleoli', 'size_uniformity', 'epithelial_size', 'bland_chromatin'] # Extra

# feature_column_names = ['marginal_adhesion', 'bare_nucleoli', 'epithelial_size', 'bland_chromatin'] # SFS

feature_column_names = ['bare_nucleoli', 'size_uniformity', 'epithelial_size'] # Hybrid


predicted_class_name = ['class']


# Getting feature variable values

X = data_frame[feature_column_names].values

# X_test.columns = data_frame.columns

y = data_frame[predicted_class_name].values
```

```
# Saving 30% for testing
split_test_size = 0.30

# Splitting using scikit-learn train_test_split function
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = split_test_size, random_state = 42)

print("{0:0.2f}% in training set".format((len(X_train)/len(data_frame.index)) * 100))
print("{0:0.2f}% in test set".format((len(X_test)/len(data_frame.index)) * 100))



# Classifier
print("\n\nClassifier:")
# classifier = tree.DecisionTreeClassifier()     # Best
# classifier = svm.SVC()
classifier = svm.SVC(kernel='linear')
# classifier = RandomForestClassifier()
# classifier = GaussianNB()
classifier.fit(X_train, y_train.ravel())

# this returns array of predicted results from test_data
prediction_from_test_data = classifier.predict(X_test)
accuracy = metrics.accuracy_score(y_test, prediction_from_test_data)

print ("Accuracy: {0:0.4f}".format(accuracy))
```