National College of Ireland

Configuration Manual

MSc Research Project
Detecting Anomalous Insurance Claims with Hybrid Feature Optimisation and Classification Techniques

Sananda Dasgupta
Student ID: X18115781

School of Computing
National College of Ireland

Supervisor: Mr. Victor Del Rosal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sananda Dasgupta
Student ID:	X18115781
Programme:	MSc. in FinTech **Year:** 2018-19
Module:	Research Project
Lecturer:	Victor Del Rosal
Submission Due Date:	12th August 2019
Project Title:	"DETECTING ANOMALOUS INSURANCE CLAIMS WITH HYBRID FEATURE OPTIMISATION AND CLASSIFICATION TECHNIQUES"
Word Count:	1042 **Page Count:** 8

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: .....

Date: .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sananda Dasgupta
Student ID: x18115781

This research is conducted to check whether the hybrid of feature optimisation and classification can improve the current state of art detailed in the literature review section of the paper. Two feature optimisation techniques viz. Particle Swarm Optimisation (PSO) and Firefly Algorithm (FFA) are adopted for this study along with five classification methods viz. Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF), Naïve Bayes (NB) and k- Nearest Neighbours (k-NN).
This manual is designed to provide a thorough step by step guidance to reach the final outcome. Section 1 of this manual carries a detailed procedure of data extraction and analysis, whereas, section 2 provides a comprehensive overview of data preparation. Section 3 furnishes an in-depth understanding of applying the data mining algorithms and finally section 4 delivers the complete comparative evaluation of all the results.

1 Data Extraction and exploratory data analysis

- **Step1:** Extract the data from Kaggle[1] – a cloud-based platform for big data.
- **Step2:** Import the data in R studio for further analysis.
- **Step3:** Check the structure and dimension of the data for better understanding (Fig 1).

```
#upload the data

insurance_claim_updated <- read.csv("~/Desktop/Project Thesis/insurance_claim_updated.csv")
str(insurance_claim_updated)
summary(insurance_claim_updated)
dim(insurance_claim_updated)
```

Figure 1: Data extraction and understanding

There are 10211 instances and 39 variables (Fig. 2) in the data, among which 18 variables are numerical and 21 are categorical.

```
> dim(insurance_claim_updated)

[1] 10211    39
```

Figure 2: Data dimension

[1] https://www.kaggle.com/mervynakash/insurance-claim

- **Step4:** Check for missing values in the data. From the data it is quite evident that the it has "?" in few columns, viz. "police_report_available", "collision_type" and "property_damage". These "?" need to be converted as missing values while importing the data (Fig. 3).

```
> insurance_claim_updated <- read.csv("~/Desktop/Project Thesis/insurance_claim_updated.csv", header=T, na.strings=c("?"," "))
> miss_col_val <- colSums(is.na(insurance_claim_updated))
> miss_col_val <- miss_col_val[miss_col_val > 0]
> miss_col <- round(miss_col_val/nrow(insurance_claim_updated)*100,2)
> miss_names <- names(miss_col)
> miss_names

[1] "collision_type"        "property_damage"
[3] "police_report_available"
```

Figure 3: Data dimension

- **Step5:** Check for outliers in the numerical variable.
- **Step6:** Check for duplicate records.
- **Step7:** Install library 'Corrplot'. Check for multi collinearity among variables. It is evident that multi collinearity exist between variables, viz. 'wellness', 'total claim amount', 'injury claim', 'property claim' and 'vehicle claim' (Fig. 4).

```
# checking for multi collinearity

> library(corrplot)
> data_Num <- sapply(insurance_claim_updated, function(x){is.numeric(x)})
> res2 <- cor(insurance_claim_updated[,data_Num])
> corrplot(res2,tl.cex = 0.7)

#multi collinearity exist between wellness, total claim amount, injury claim, property claim and vehicle claim.
```

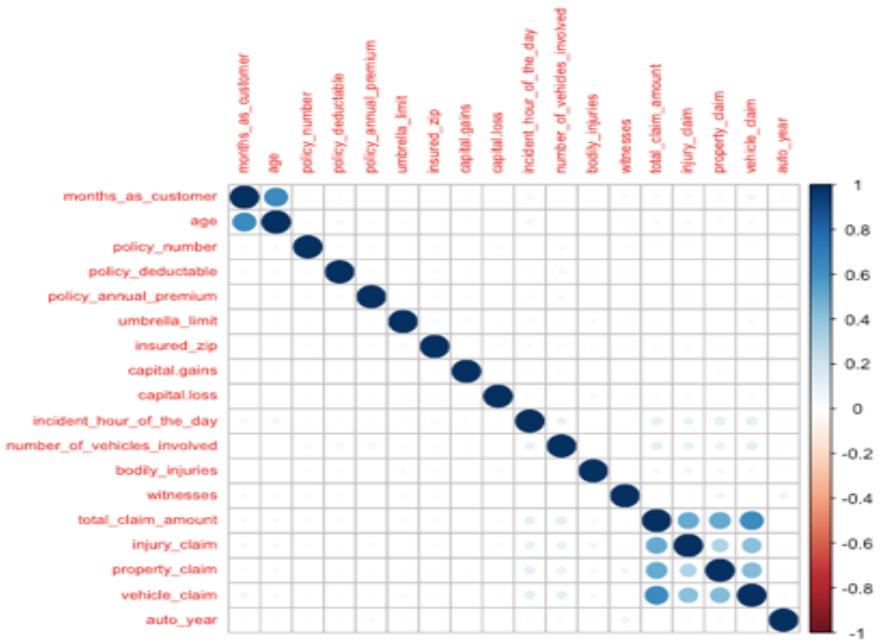


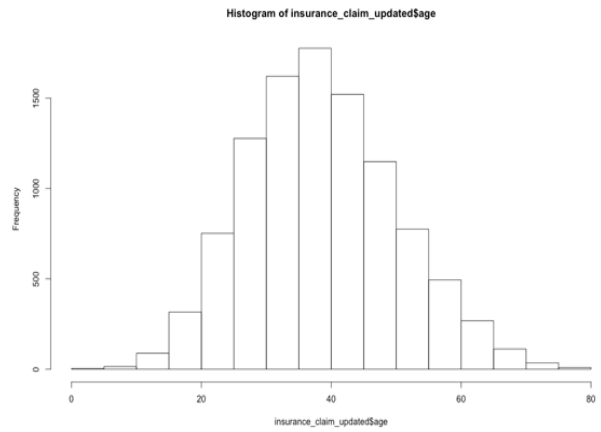
Figure 4: Multi Collinearity

- **Step8:** Install library 'ggplot2'. Install library 'MASS' for high cardinality variables. Perform some univariate and bivariate analysis for numerical and categorical variables to check their dependencies on the response variable (fraud_reported).

```
# UNIVARIATE ANALYSIS (EDA)

# NUMERICALS
#Age
summary(insurance_claim_updated_PSO$age)
hist(insurance_claim_updated_PSO$age)

#Injury Claim
summary(insurance_claim_updated_PSO$injury_claim)
hist(insurance_claim_updated_PSO$injury_claim) # right skew
```



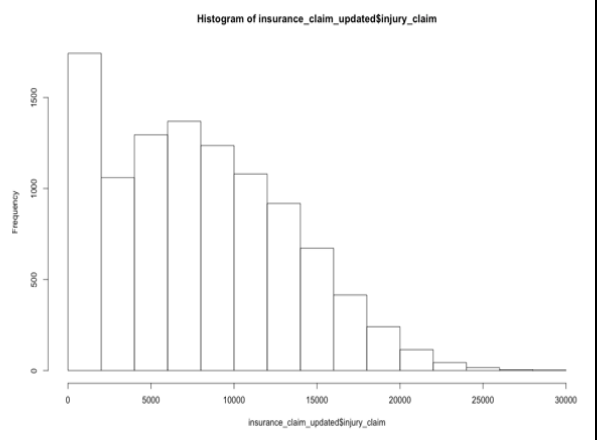


Figure 5: Univariate Analysis

2 Data Preparation

- **Step9:** Install library 'stringr'. Impute missing values. For this study, missing values exist in the categorical variables, hence, mode imputation will be performed.
- **Step10:** Treat the outliers using Inter Quartile Range (IQR) technique.
- **Step11:** Install library 'dplyr'. Convert all character variables to factors.

2.1 PSO Optimisation

- **Step12:** Perform Particle Swarm Optimisation (PSO) through WEKA, to select most relevant features.
  - Install WEKA explorer.
  - Install library 'PSOSearch' in the 'tool' section.
  - Import the data using 'Open file' section.
  - In the 'Select attributes' section choose 'WrapperSubsetEval' as the 'Attribute Evaluator' and choose 'PSOSearch' as the 'Search Method'.

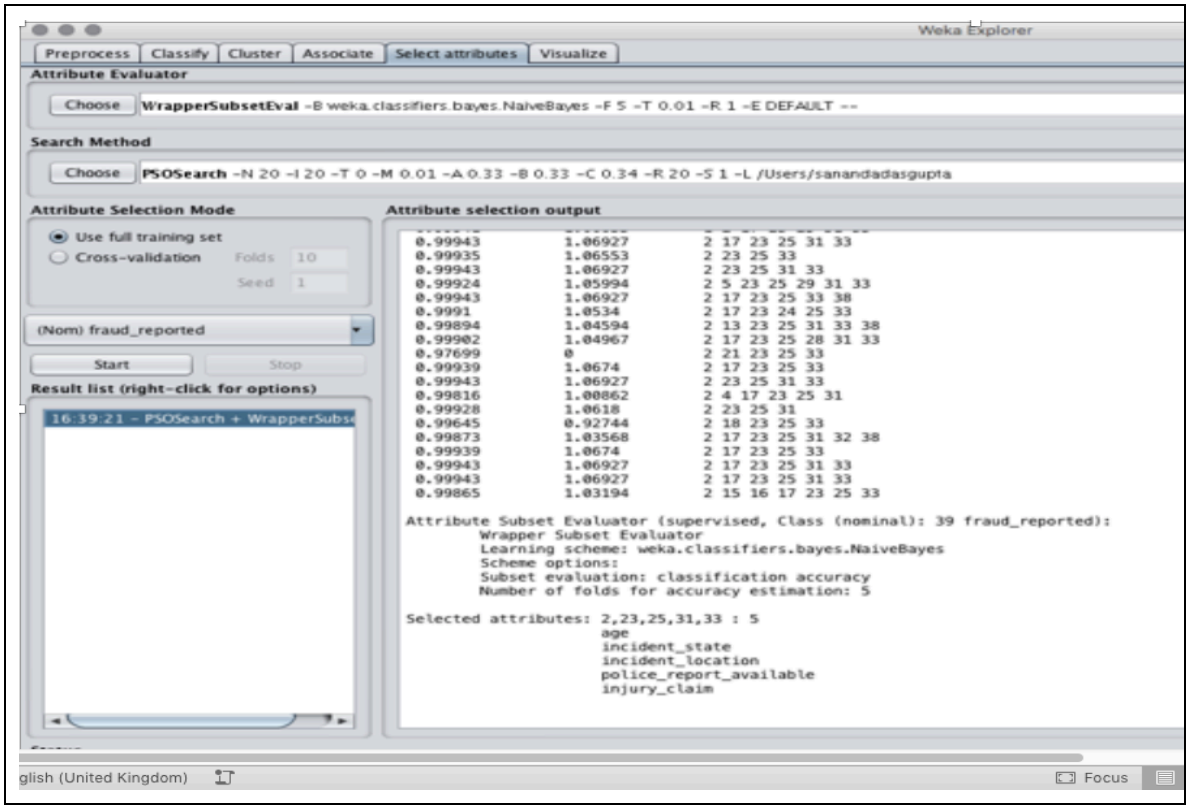


Figure 6: PSO using WEKA

- **Step 13:** Exclude all those variables which are not selected by PSOSearch and create a new data using the ones chosen by PSO (Fig. 6) along with the response.
- **Step 14:** Exclude those attributes having more than 52 levels as R is unable to handle those high cardinality categorical variables.
- **Step 15:** Split the data into 70:30 proportion. 70% of the data is considered as training purpose and 30% is kept for validation or testing purpose.
- **Step 16:** Further, normalisation is used to re-scale the numerical data values. It scales the data values between 0 and 1 thus enabling the same range of values for each of the inputs.

```
#splitting the data
insurance_claim_updated_PSO <-  insurance_claim_updated_PSO[, -3]
nrows <- nrow(insurance_claim_updated_PSO)
set.seed(1234)
index <- sample(1:nrow(insurance_claim_updated_PSO), 0.7 * nrows)
#separate train and validation set
train = insurance_claim_updated_PSO[index,]
validation =  insurance_claim_updated_PSO[-index,]
head(train)
dim(train)
#standardizing new data (Normalise the variables)
trainTask <- normalizeFeatures(train,method = "standardize")
testTask <- normalizeFeatures(validation,method = "standardize")
dim(trainTask)
str(trainTask)
```

Figure 7: Splitting and normalising the data

2.2 FFA Optimisation

- **Step 12:** Repeat step 1 to step 11 and perform Firefly Algorithm (FFA) to select most relevant features according to FFA.
  - Install library 'MetaphorSearchMethods' in the 'tool' section of WEKA.
  - Import the data using 'Open file' section.
  - In the 'Select attributes' section choose 'WrapperSubsetEval' as the 'Attribute Evaluator' and choose 'FireFlySearch' as the 'Search Method'.

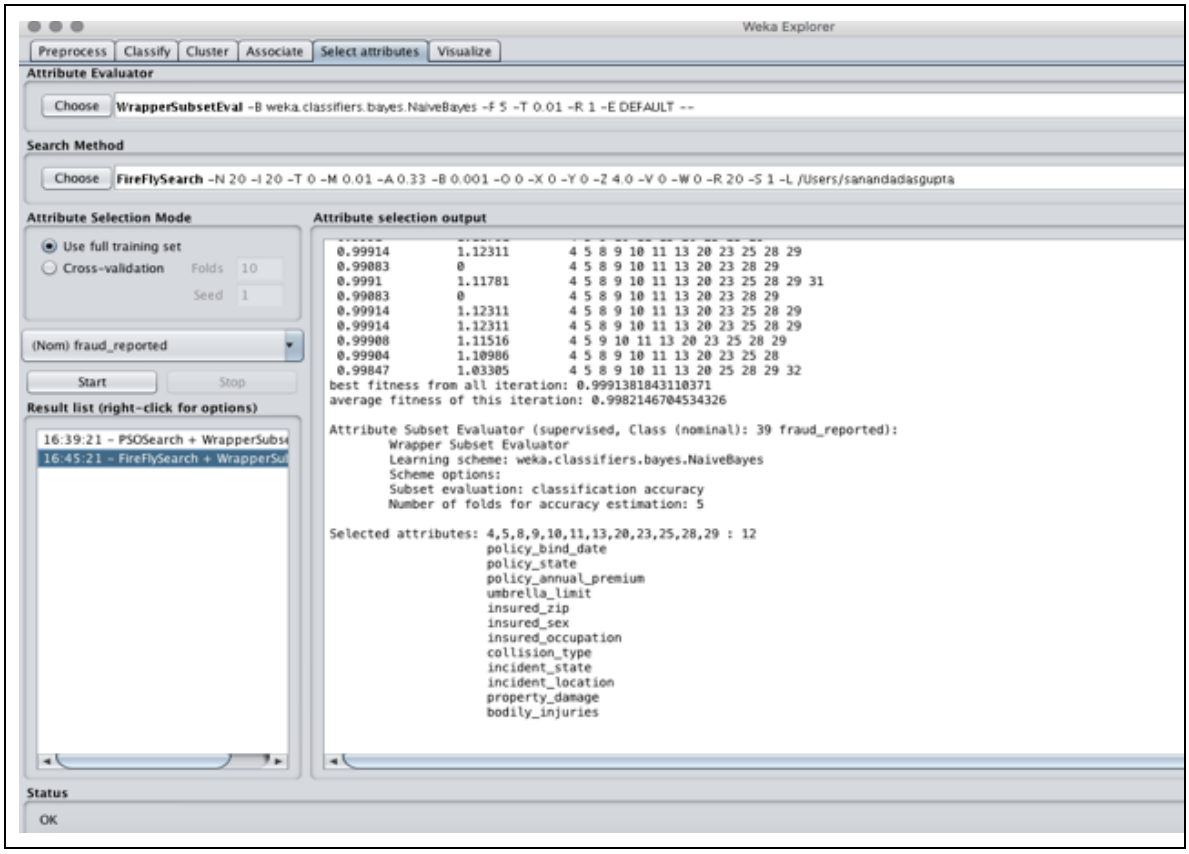


Figure 8: FFA using WEKA

- **Step 13:** Exclude all those variables which are not selected by FireFlySearch and create a new data using the ones chosen by FFA (Fig. 8) along with the response.
- Repeat step 14, 15 and step 16 after building the new data with the attributes chosen by FFA.

2.3 Random Forest Important Variable Selection

- **Step 12:** Repeat step 1 to step 11. In a new R script install library 'RandomForest' and then perform Random Forest Important Variable Selection technique to identify top 12 most important features. This has been done to compare the results with current state of art as mentioned in the literature review section of the research.

```
#Removing variables with more than 52 levels
insurance_claim_updated <- insurance_claim_updated[, c(-4, -18, -25)]
insurance_claim_updated <- mutate_if(insurance_claim_updated, is.character, as.factor)
str(insurance_claim_updated)
rf.model <- randomForest(fraud_reported ~ .,data=insurance_claim_updated,
importance=TRUE,ntree=500)
rf.model
rf.1.var_imp <- varImpPlot(rf.model)
# Top 12 => Insured_hobbies, incident_severity, auto_model, insured_occupation,
insured_education_level, incident_city, auto_make, insured_relationship, incident_state,
auhorities_contacted, policy_annual_premium, fraud_reported
```

Figure 9: Random Forest Important Variable Selection using R

- **Step 13:** Create a new data using only top 12 variables (Fig. 9) along with the response, selected by Random Forest Important Variable Selection method.
- Repeat step 14, 15 and step 16.

3 Applying Data Mining Algorithms

- **Step 17:** Apply Machine Learning technique – Random Forest (RF) on training data.
- **Step 18:** Validate the results using the test data.
- **Step 19:** Install library 'caret' to generate confusion matrix.
- **Step 20:** Apply Support Vector Machine (SVM) on training data.
- **Step 21:** Validate the results using the test data and generate confusion matrix.
- **Step 22:** Install library (e1071) for Naïve Bayes (NB). Apply on training data.
- **Step 23:** Validate the results using the test data and generate confusion matrix.
- **Step 24:** Apply k-Nearest Neighbour (kNN) algorithm on training data.
- **Step 25:** Validate the results using the test data and generate confusion matrix.
- **Step 26:** Install library 'nnet' and 'caretEnsemble'. Apply Artificial Neural Network (ANN).
- **Step 27:** Validate the results using the test data and generate confusion matrix.

4 Evaluation

- **Step 28:** Compare the confusion matrix generated from PSO_RF, PSO_SVM, PSO_NB, PSO-kNN, PSO_ANN, FFA_RF, FFA_SVM, FFA_NB, FFA_kNN and FFA_ANN (eg: Fig. 10 and Fig. 11).

- **Step 29:** Three performance metrices are chosen to conduct this study, viz. Accuracy, Sensitivity and Specificity.

- **Step 30:** Choose the hybrid model that performs the best among all other models in terms of accuracy, sensitivity and specificity.

```
# Random Forest
library("randomForest")
library(e1071)

rf_model <- randomForest(fraud_reported ~ ., data =
trainTask, mtry = 5,ntree = 500, importance = TRUE,
do.trace = 100)
rf_model

# Predicting on train set

rf_pred_train <- predict(rf_model, trainTask, type = "class")

# Checking classification accuracy
table(rf_pred_train, trainTask$fraud_reported)

# Predicting on test set

rf_pred_test <- predict(rf_model, testTask, type = "class")
rf_pred_test
table(rf_pred_test,testTask$fraud_reported)
result_rf <- confusionMatrix(testTask$fraud_reported,
rf_pred_test)
result_rf
```

```
          Reference
Prediction   0   1
         0 897 599
         1 699 869

               Accuracy : 0.5764
                 95% CI : (0.5586, 0.594)
    No Information Rate : 0.5209
    P-Value [Acc > NIR] : 3.997e-10

                  Kappa : 0.1536

 Mcnemar's Test P-Value : 0.005998

            Sensitivity : 0.5620
            Specificity : 0.5920
         Pos Pred Value : 0.5996
         Neg Pred Value : 0.5542
             Prevalence : 0.5209
         Detection Rate : 0.2928
   Detection Prevalence : 0.4883
      Balanced Accuracy : 0.5770
```

Figure 10: PSO-RF in R-studio

```
# Random Forest
library("randomForest")
library(e1071)

rf_model <- randomForest(fraud_reported ~ ., data =
trainTask, mtry = 5,ntree = 500, importance = TRUE,
do.trace = 100)
rf_model

# Predicting on train set

rf_pred_train <- predict(rf_model, trainTask, type = "class")

# Checking classification accuracy
table(rf_pred_train, trainTask$fraud_reported)

# Predicting on test set

rf_pred_test <- predict(rf_model, testTask, type = "class")
rf_pred_test
table(rf_pred_test,testTask$fraud_reported)
result_rf <- confusionMatrix(testTask$fraud_reported,
rf_pred_test)
result_rf
```

```
          Reference
Prediction    0    1
         0 1445   51
         1   39 1529

               Accuracy : 0.9706
                 95% CI : (0.964, 0.9763)
    No Information Rate : 0.5157
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9412

 Mcnemar's Test P-Value : 0.2463

            Sensitivity : 0.9737
            Specificity : 0.9677
         Pos Pred Value : 0.9659
         Neg Pred Value : 0.9751
             Prevalence : 0.4843
         Detection Rate : 0.4716
   Detection Prevalence : 0.4883
      Balanced Accuracy : 0.9707

       'Positive' Class : 0
```

Figure 11: FFA-RF in R-studio

- **Step 31:** Additionally step 17 to step 27 can be performed along with the attributes chosen with the help of Random Forest Variable Selection Method to check whether the result is at par or outperforming the current state of art as detailed in the literature review section of the paper.

- From the results obtained, it can be concluded that FFA_RF outperforms all other hybrid models in terms of accuracy, sensitivity and specificity, even though ANN combined with both PSO and FFA has generated an unrealistic result (Fig. 12). This may be a result of overfitting of the model which can be left for further consideration as a future work of this research.

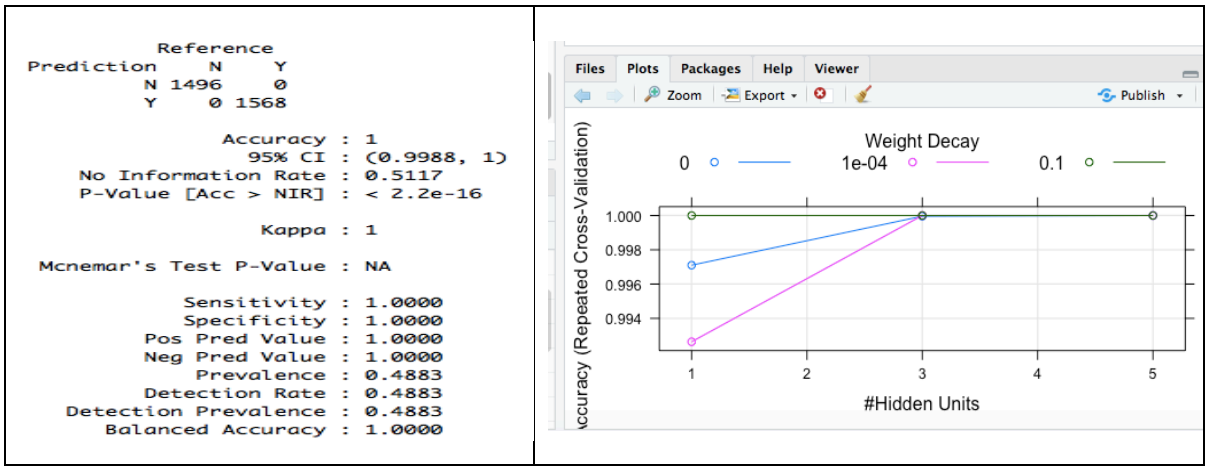


Figure 12: Result of PSO_ANN and FFA_ANN

References

https://www.cs.waikato.ac.nz/ml/weka/downloading.html

https://www.java.com/en/download/

https://www.rstudio.com/products/rstudio/download/