Supervised Learning

Machine learning, statistical learning, deep learning and artificial intelligence

June 2021

Abstract. Organizations nowadays face a major threat in terms of cybersecurity, from malicious URLs to credential reuse, and having robust security systems can make all the difference[1].Implement algorithms on cyber attacks data is a way for detecting threats and network anomalies,such as a Denial of Service (DoS). It is an intentional attack attempted by attackers from single source which has an implicit intention of making an application unavailable to the target stakeholder.In these pages will be analysed different statistical learning techniques to find the best methodology that can predict the class of internet access, specifically whether network traffic will be classified as benign or not, then classified as a DDoS attack, meaning a malfunction due to a cyber attack in which the resources of a computer systems are deliberately exhausted for malicious purposes.
Keywords: supervised learning techniques, accuracy, prediction, ddos

Camilla Gotta 945522
Università degli Studi di Milano
Data Science and Economics

Contents

1 Introduction **2**

2 Data preprocessing **3**
- 2.1 Dataset . . . 3
- 2.2 Data cleaning . . . 3
- 2.3 Feature Engineering . . . 3

3 Methodology **5**
- 3.1 Logistic Regression . . . 5
- 3.2 Decision Tree . . . 5
- 3.3 Random Forest . . . 5
- 3.4 Artificial Neural Networks . . . 6

4 Results **7**
- 4.1 Logistic regression results . . . 7
- 4.2 Decision tree results . . . 8
- 4.3 Random forest results . . . 10
- 4.4 Artificial neural networks results . . . 12

5 Conclusion **15**

List of Figures

1 Correlation matrix of features . . . 4
2 Correlation plot of features against each other and target variable . . . 4
3 Logistic regression confusion matrix . . . 7
4 Logitic regression Auc curve . . . 8
5 Decision tree . . . 8
6 Decision tree confusion matrix . . . 9
7 Decision tree Auc curve . . . 10
8 Random forest summary . . . 10
9 Random forest variables importance . . . 11
10 Decision tree . . . 11
11 Random Forest Auc curve . . . 12
12 Artificial neural networks . . . 13
13 Artificial neural networks confusion matrix . . . 14
14 Artificial neural networks AUC curve . . . 14

List of Tables

1 Features description . . . 3

1 Introduction

Over the past years the rapid development and popularization of Internet of Things (IoT) devices has made our lives easier and faster, but at the same time it has occurred an increasing number of cyber-attacks,like the distributed denial-of-service (DDoS) attacks on Internet services and websites.
In general this kind of cyber threat is a purposeful attempt which is initiated so as to make an application or website unavailable to its legitimate users. In order to achieve this, usually, one of the several choices of attackers is to apply diversified techniques that intentionally consume huge network bandwidth, thus causing inconvenience to legitimate users. The attacker makes it by using multiple sources which may include routers, IoT devices, and computers in a distributed environment infected by malware. To make this possible, an attacker looks for availability of any compromised network. By utilizing such compromised networks, an attacker usually attacks the target system through continuously generating packet floods or requests to conquer the target system.
Therefore, computer scientists and network practitioners have tought various approaches to detect DDoS attacks and attempt to predict a given attack[8]. Rule-based algorithms have been developed to detect such attacks but have not been successful due to the complicated nature of DDoS, where many variables play important roles[9]. In this work, statistical learning-based solutions have been exploited, more specifically models were trained and tested to classify DDoS and Benign attacks. Starting with a simple logistic regression, then with a decision tree construct to go further in the analysis and find better results while performing random forests and at last the artificial neural networks.

2 Data preprocessing

2.1 Dataset

The used dataset is taken from [2]. It is about network anomalies with 500000 rows and 8 columns, describing what is shown in Table 1.

Feature	Description
Flow ID	Flow identification number
Timestamp	Time Stamp of the flow
Fwd Pkt Len Mean	Mean size of package in forward direction
Fwd Seg Size Avg	Average segment size observed in forward
Init Fwd Win Byts	Number of bytes sent in initial window in forward direction
Init Bwd Win Byts	Number of bytes sent in initial window in backward direction
Fwd Seg Size Min	Minimum segment size observed in forward
Label	Indicate whether the traffic is malicious or not

Table 1: Features description

2.2 Data cleaning

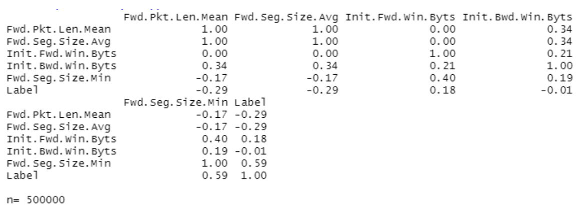
The kind of data one gets from web networks are usually unbalanced and with useless information for the analysis purpose. For that reason columns of non-interest have been dropped, i.e. ”Flow.ID” and ”Timestamp” and null values have been removed; statistical learning algorithms often require numerical data therefore categorical data must be converted to numerical features: in the ”label” column, the ”DDos” value has been converted in value 0, while value 1 correspond to ”benign” value. Moreover to speed up the training process and allows the models to be converged faster, data normalization is applied. This step refers to a process where each column representing a given feature is normalized to [-1, +1] or [0, +1] depending on statistical learning models that are used. To normalize data, different algorithms exist, such as normalization using minimum and maximum (Min-Max scalar) of data or standard deviation and average of data. In this project, the following min-max method

$$x_{scaled} = \frac{x - x_{min}}{x - x_{max}}$$

was applied to each feature column, which means the data were standardized for each feature separately.

2.3 Feature Engineering

This step refers to a process features in the dataset that were analyzed, and the importance of each feature and contribution to target variables are discovered using different methods. For instance, the canonical correlation scores between each feature and target variable among all samples demonstrate how a given feature is associated with the output, in this case the target variable ”Label”: the respective correlation values between variables is shown in Fig.1.



```
                  Fwd.Pkt.Len.Mean Fwd.Seg.Size.Avg Init.Fwd.Win.Byts Init.Bwd.Win.Byts
Fwd.Pkt.Len.Mean              1.00             1.00              0.00              0.34
Fwd.Seg.Size.Avg              1.00             1.00              0.00              0.34
Init.Fwd.Win.Byts             0.00             0.00              1.00              0.21
Init.Bwd.Win.Byts             0.34             0.34              0.21              1.00
Fwd.Seg.Size.Min             -0.17            -0.17              0.40              0.19
Label                        -0.29            -0.29              0.18             -0.01
                  Fwd.Seg.Size.Min Label
Fwd.Pkt.Len.Mean             -0.17 -0.29
Fwd.Seg.Size.Avg             -0.17 -0.29
Init.Fwd.Win.Byts             0.40  0.18
Init.Bwd.Win.Byts             0.19 -0.01
Fwd.Seg.Size.Min              1.00  0.59
Label                         0.59  1.00

n= 500000
```

Figure 1: Correlation matrix of features

Another useful tool for variables correlation analysis is shown in Fig.2 (left) which illustrates the correlation scores of 5 features and the target variable ”Label”. As seen, the scores with lighter colors offer higher scores, whereas darker areas indicate lower correlation scores.

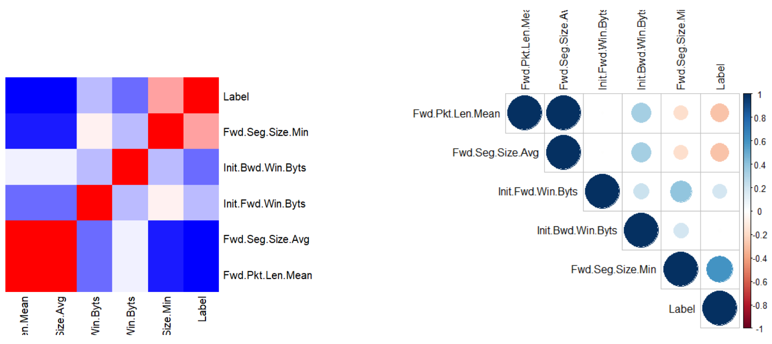


Figure 2: Correlation plot of features against each other and target variable

Indeed in Fig.2 (right) the model is recognizing that the variable Fwd.Seg.Size.Avg is so correlated with the variable Fwd.Pkt.Len.Mean. This correlation is a problem because independent variables should be independent: if the degree of correlation between variables is high enough, as in this case, it can cause problems of multiccolinearity when fitting the model and interpret the results. For this reason will be exclude the variable Fwd.Seg.Size.Avg, while performing logistic regression: since Variance Inflaction Factor returns FALSE results when delete Fwd.Seg.Size.Avg, otherwise it returns 'alias' which refers to the variables that are linearly dependent on others (i.e. cause perfect multicollinearity).
By contrary, for the other techniques no variable has been removed, because decision tree and random forest model are trained using the data where the importance of each feature is evaluated by a metric referred to as node impurity[3], such a metric is weighted by the probability of a node in the tree.

3 Methodology

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs as part of the cross validation process. Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.
The following sections will explain what types of supervised learning algorithms were implemented to achieve the initial goal and present the results of their performance.

3.1 Logistic Regression

While linear regression is leveraged when dependent variables are continuous, logistical regression is selected when the dependent variable is categorical, meaning they have binary outputs, such as ”benign” and ”not benign” (i.e. ddos).
While both regression models seek to understand relationships between data inputs, logistic regression is mainly used to solve binary classification problems, such this case.

3.2 Decision Tree

One of the supervised machine learning algorithms that is used for reaching the initial aim is the decision tree model. In binary classification, this algorithm can be considered as a binary tree structure that splitting data into two subsets is performed based on specific criteria so-called decision rules. The tree is growing top-down: each internal node of the tree is an attribute, while each leaf node is a class label. To predict a class label, one has to start from the root of the tree, comparing the values of the root attribute with record’s attribute and then following the branch corresponding to that value and so jumping to the following node[4]. Comparisons between the record’s attribute values and other internal nodes of the tree continue until a leaf node with predicted class value will be reached. The Decision Trees’ algorithm used in this analysis is based on a criterium called Gini index, a measure that shows how often a randomly chosen element would be identified incorrectly: an attribute with lower Gini index is better than a one with higher index value.

3.3 Random Forest

Another supervised classification algorithm is the so called Random Forest, consisting in creating a forest made of a big number of decision trees that act as an ensemble learning method for classification, regression or other tasks. Each single tree spits out a class prediction and the one that has the most votes becomes the model’s prediction: the larger the number of trees, the more accurate will be the results.

Furthermore, predictions made by the individual trees need to have low correlations with each other in order to be more accurate, since the trees protect each other from their individual errors (as they go normally in different directions)[5]. The main difference between Decision Tree and Random Forest is that the second one finds the root node and splits the feature nodes randomly: avoiding decision trees overfitting. In classification problems each features's importance is assessed based on two criteria:

- Mean Decrease Accuracy: gives a rough estimate of the loss in prediction performance when that particular variable is omitted from the training set. If two variables are somewhat redundant, then omitting one of them may not lead to massive gains in prediction performance, but would make the second variable more important.
- Mean Decrease Gini: GINI is a measure of node impurity. Highest purity means that each node contains only elements of a single class. Assessing the decrease in GINI when that feature is omitted leads to an understanding of how important that feature is to split the data correctly.

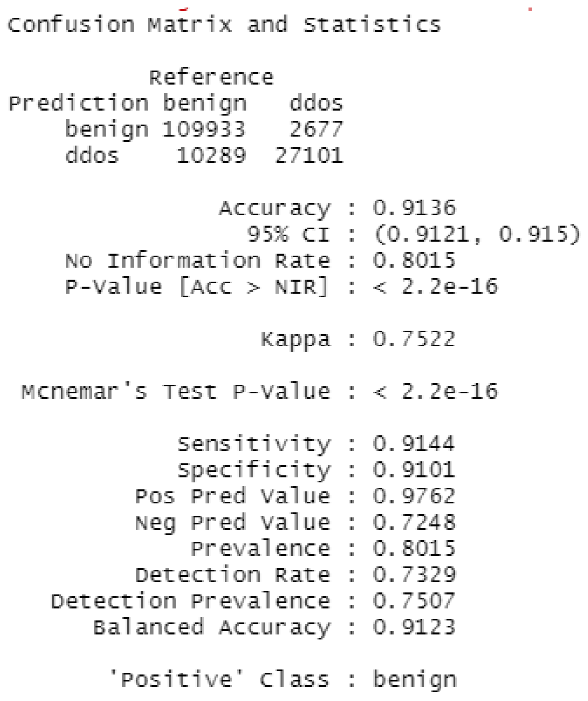
3.4 Artificial Neural Networks

Artificial Neural networks (ANN) or neural networks are computational algorithms. It intended to mimicking the interconnectivity of the human brain through layers of nodes: a system of interconnected "neurons" which can compute values from inputs.
A neural network is an oriented graph in which each neuron is a processing units, connected to each other by nodes, and each of them consists of input and output units. An input unit receives information, so that the neural network tries to learn the information to produce an output and to minimize the loss function through the so-called backpropagation[2]. So, it works "backward", from output units to input units, computing the gradient of the loss function with respect to the weights of its connection between units to obtain the error as lowest as possible.

4 Results

4.1 Logistic regression results

Before applying logistic algorithm, as already said the variable ”Fwd.Seg.Size.Avg” was removed to train this algorithm because of the multicollinearity with another variable. So the used features are: Label, Fwd.Pkt.Len.Mean, Init.Fwd.Win.Byts, Init.Bwd.Win.Byts, Fwd.Seg.Size.Min. Then it has been created the train and test set, by splitting data 70/30, where 70% of the data has been used to train the model and the remaining 30% to make predictions. Once the method has been applied to predict benign and ddos attack on the basis of network traffic, the obtained results are shown in Fig.3.
The model perform quite well, it predicts correctly 109669 benign network traffic and 27343 ddos attacks, but it misclassified 2657 benign as ddos attack. By analogy, the model misclassified 10331 network traffic as benign while they were malicious with an high accuracy of 91% and Kappa equals to 75% (quite well strength).



```
Confusion Matrix and Statistics

          Reference
Prediction benign   ddos
    benign 109933   2677
    ddos    10289  27101

               Accuracy : 0.9136
                 95% CI : (0.9121, 0.915)
    No Information Rate : 0.8015
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7522

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9144
            Specificity : 0.9101
         Pos Pred Value : 0.9762
         Neg Pred Value : 0.7248
             Prevalence : 0.8015
         Detection Rate : 0.7329
   Detection Prevalence : 0.7507
      Balanced Accuracy : 0.9123

       'Positive' Class : benign
```

Figure 3: Logistic regression confusion matrix

Sensitivity and specificity have both an high value about 91%: the first measures the proportion of positives that are correctly identified (true positive rate), the second measures the proportion of negatives that are correctly identified (true negative rate).

When it need to check or visualize the performance of the multi-class classification problem, the AUC (Area Under The Curve) is one of the most important evaluation metrics for checking any classification model's performance. Its curve represents the degree or measure of separability, by plotting on x-axes the specificity and on y-axes the sensitivity, as shown in Fig.4.
It tells how much the model is capable of distinguishing between classes: high AUC implies a better the model at distinguishing between benign and ddos traffic network.

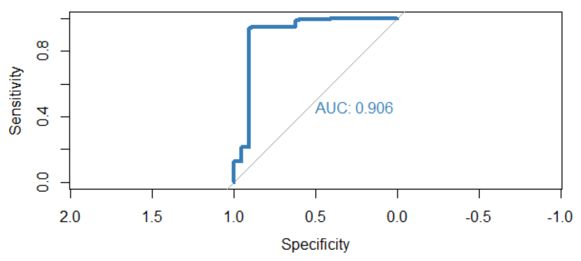


Figure 4: Logitic regression Auc curve

4.2 Decision tree results

Decision Tree has been applied to try to predict benign and malicious applications on the basis of network traffic, taking into consideration at first the most important variables that influence networking and the results are are quite satisfying,as shown in Fig. 5.

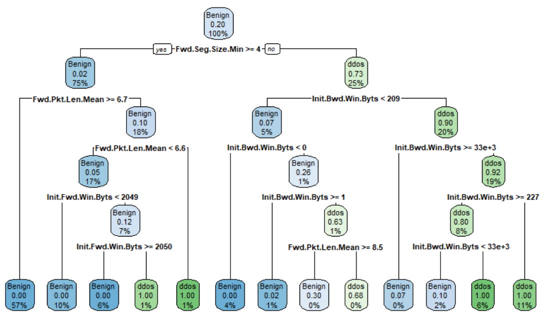


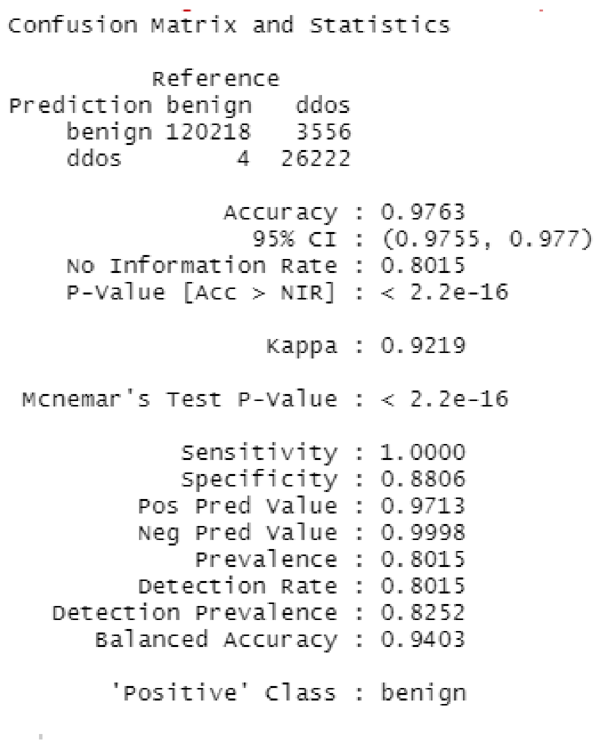
Figure 5: Decision tree

The Fwd.Seg.Size.Min is the most important factor in determining the type of application and is the predictor variable for the primary split, where the split point is

4: if it is less than 4, the traffic results as benign, while if it is bigger the algorithm proceeds with further splits. The tree has 13 terminal nodes, known as leaves. After the first split, the network traffic result malicious in the following situations:

- Fwd.Seg.Size.Min ≥ 4
- Fwd.Pkt.Len.Mean ≤ 6.6
- Init.Bwd.Win.Byts $33e + 3 \leq x \leq 227$
- Init.Bwd.Win.Byts ≥ 2050

After having made predictions, it results that the model predicts correctly 120218 benign flows, but it classified 3556 benign network traffic data as malicious. By analogy, the model misclassified 4 network flows as benign while they were malicious with an accuracy of 97%, Fig. 6.



```
Confusion Matrix and Statistics

          Reference
Prediction benign   ddos
    benign 120218   3556
    ddos        4  26222

               Accuracy : 0.9763
                 95% CI : (0.9755, 0.977)
    No Information Rate : 0.8015
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9219

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 1.0000
            Specificity : 0.8806
         Pos Pred Value : 0.9713
         Neg Pred Value : 0.9998
             Prevalence : 0.8015
         Detection Rate : 0.8015
   Detection Prevalence : 0.8252
      Balanced Accuracy : 0.9403

       'Positive' Class : benign
```

Figure 6: Decision tree confusion matrix

The confusion matrix above shows the actual application type in its rows and the predicted target in its columns, showing the True positive (TP) and the True negative (TN) and the False negative (FN) and the False positive (FP) on its diagonals. False positives include all the benign traffic that has been predicted as a ddos attack, while False negatives are the ones that actually are malicious but have been predicted as benign. Let's precise that the accuracy test from the confusion matrix, that is 97%, is done by the proportion of TP and TN over the sum of the matrix:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

AUC (Area Under The Curve) curve represents the degree or measure of separability, by plotting on x-axes the specificity and on y-axes the sensitivity, as shown in Fig.7.It tells how much the model is capable of distinguishing between classes: high AUC implies a better the model at distinguishing between benign and ddos traffic network.

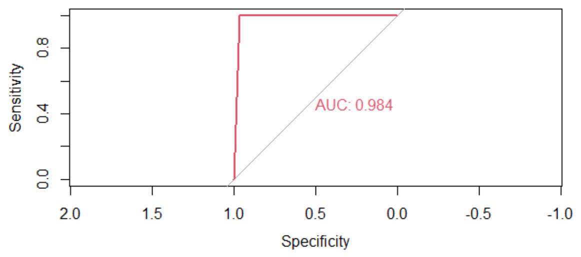
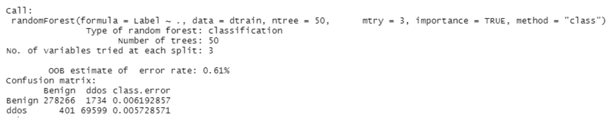


Figure 7: Decision tree Auc curve

4.3 Random forest results

Random Forest has been applied in order to try to have better predictions and to reduce the number of FP and FN, considering the same variables that have been used to apply decision tree algorithm.
For this method the OOB estimate of error rate is a useful measure to discriminate between different random forest classifiers: to select the combination that produces the smallest value for this error rate, it has been changed the number of trees and the number of variables to be considered: the resulting one is shown in Fig. 8



```
Call:
 randomForest(formula = Label ~ ., data = dtrain, ntree = 50,      mtry = 3, importance = TRUE, method = "class")
               Type of random forest: classification
                     Number of trees: 50
No. of variables tried at each split: 3

        OOB estimate of  error rate: 0.61%
Confusion matrix:
       Benign  ddos class.error
Benign 278266  1734 0.006192857
ddos      401 69599 0.005728571
```

Figure 8: Random forest summary

Random Forest presents 500 trees and the number of variables available for splitting at each node has been set to 3, since it was the one giving better model accuracy, minimizing the OOB estimate of error rate. Through the VarImp() function, it is possible to look at the most important variables.
It is not surprising because the important features are likely to appear closer to the

root of the decision tree, while the ones that are less important appear closed to the leaves, as shown in Fig.9.

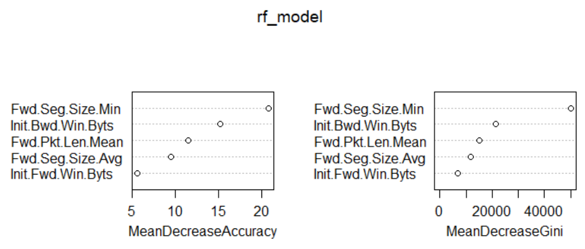
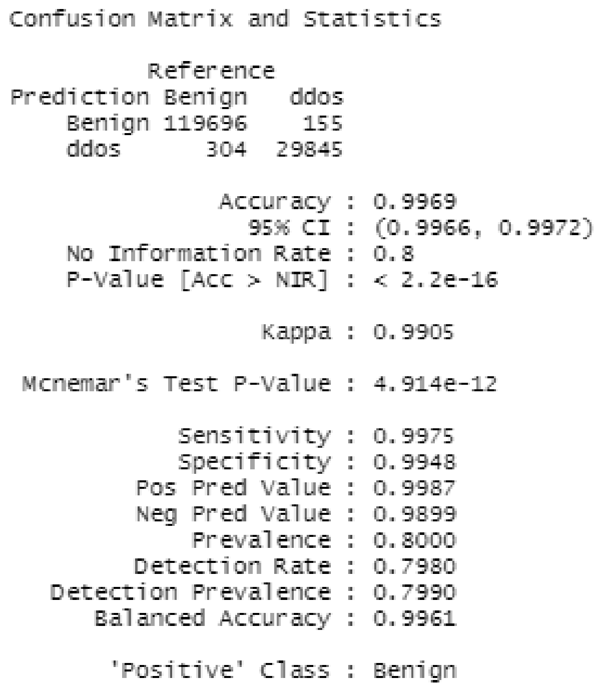


Figure 9: Random forest variables importance

Moreover, as shown in Fig.10, this algorithm perform better than the decision tree since it useful to avoid the overfitting disadvantage of the decision tree method: it correctly classifies 119696 benign traffic network flows and 29845 as ddos attack, while it misclassified 304 benign as ddos and 155 ddos as benign. It has high accuracy, almost 100% as expected better then the previous model.



```
Confusion Matrix and Statistics

          Reference
Prediction Benign   ddos
    Benign 119696    155
    ddos      304  29845

               Accuracy : 0.9969
                 95% CI : (0.9966, 0.9972)
    No Information Rate : 0.8
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9905

 Mcnemar's Test P-Value : 4.914e-12

            Sensitivity : 0.9975
            Specificity : 0.9948
         Pos Pred Value : 0.9987
         Neg Pred Value : 0.9899
             Prevalence : 0.8000
         Detection Rate : 0.7980
   Detection Prevalence : 0.7990
      Balanced Accuracy : 0.9961

       'Positive' Class : Benign
```

Figure 10: Decision tree

AUC (Area Under The Curve) curve represents the degree or measure of separability, by plotting on x-axes the specificity and on y-axes the sensitivity, as shown in Fig.12. It tells how much the model is capable of distinguishing between classes: high AUC implies a better the model at distinguishing between benign and ddos traffic network.

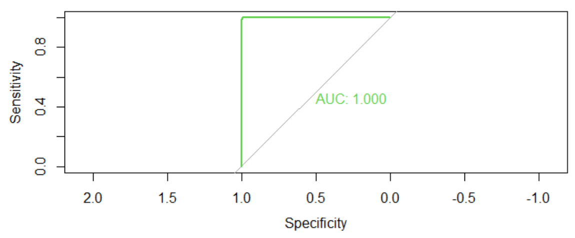


Figure 11: Random Forest Auc curve

4.4 Artificial neural networks results

Artificial neural networks algorithm faces a training phase, where the algorithm learns to recognize features in the data set and compares the output produced with the one expected. During the first step, the ANN tries to adjust the difference between the actual and expected output and to minimize the loss function. Before implementing it, a sample of instances was selected due to computational effort required in terms of time, then data has been normalized in order to be adjusted to a common scale so as to compare with more accuracy predicted and actual values. The technique is the max-min normalization one.
After this process, data has been divided into training and test sets (70/30). The threshold is set to 0.01, implying that if the change in error during an iteration is less than 0.01 then no further optimization is done by the model. After having tried different number of hidden layers to see which one has had the better accuracy of predictions, 3 and then 2 hidden layers have been chosen, Fig. 12.

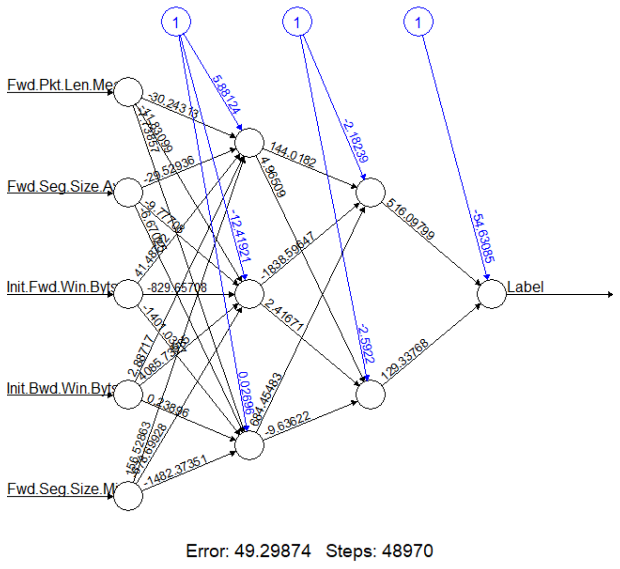
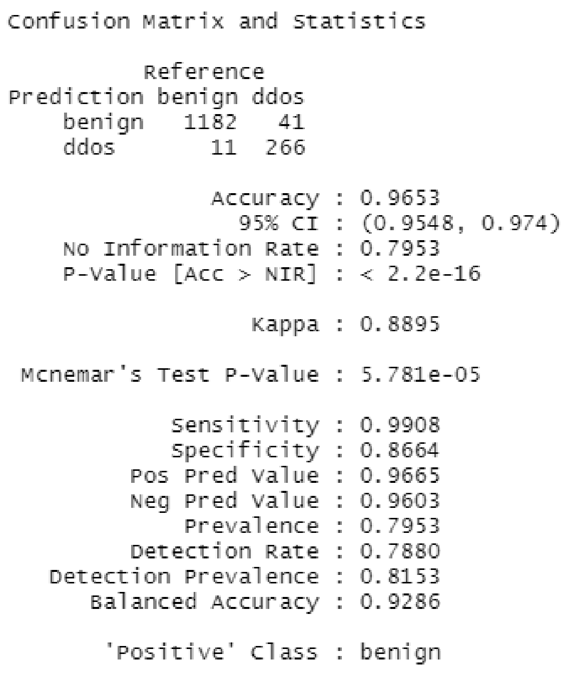


Figure 12: Artificial neural networks

The confusion matrix presents, in Fig.13, that the model predicts correctly 1182 (over the sample of 5000) benign network traffic and 266 ddos attacks, but it misclassified 11 benign as ddos attack. By analogy, the model misclassified 41 network traffic as benign while they were malicious with an high accuracy of 96% and Kappa equals to 89% (well strength).



```
Confusion Matrix and Statistics

          Reference
Prediction benign ddos
    benign   1182   41
    ddos       11  266

               Accuracy : 0.9653
                 95% CI : (0.9548, 0.974)
    No Information Rate : 0.7953
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8895

 Mcnemar's Test P-Value : 5.781e-05

            Sensitivity : 0.9908
            Specificity : 0.8664
         Pos Pred Value : 0.9665
         Neg Pred Value : 0.9603
             Prevalence : 0.7953
         Detection Rate : 0.7880
   Detection Prevalence : 0.8153
      Balanced Accuracy : 0.9286

       'Positive' Class : benign
```

Figure 13: Artificial neural networks confusion matrix

AUC (Area Under The Curve) curve represents the degree or measure of separability, by plotting on x-axes the specificity and on y-axes the sensitivity, as shown in Fig.14. It tells how much the model is capable of distinguishing between classes: high AUC implies a better the model at distinguishing between benign and ddos traffic network.

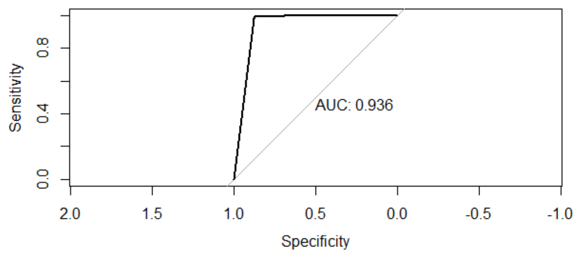


Figure 14: Artificial neural networks AUC curve

5 Conclusion

Nowadays the rapid development and popularization of Internet of Things (IoT) devices has made our lives easier and faster, but at the same time it has occurred an increasing number of cyber-attacks, thus leading to face a major threat in terms of cybersecurity and having robust security systems can make the difference. The chosen dataset presents information about the flow of data in the network: with 500000 records and with data mainly about the size, the segment, time and ID number of packets sent both in forward and backward direction in the network. For this reason in this work, a number of statistical learning-based algorithm solutions were used to classify the data flow in the network in the dataset as malicious or not, more specifically models were trained and tested to classify DDoS and Benign attacks. Starting with a simple logistic regression, then with a decision tree construct to go further in the analysis and find better results while performing random forests and at last the artificial neural networks. It was seen from the results that all the methods chosen to be applied to the dataset reveal very good results in predicting the type of traffic, so any technique one wishes to implement on this dataset will all lead to a good prediction of network traffic.

References

[1] Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture, Soe, Feng, Santosa, Hartanto and Sakurai

[2] https://github.com/PacktPublishing/Machine-Learning-for-Cybersecurity-Cookbook

[3] Sugumaran, V. Muralidharan and K. Ramachandran, Feature selection using decision tree and classification through proximal support vector machine for fault diagnostics of roller bearing. Mechanical systems and signal processing, 2007. 21(2): p. 930-942.

[4] https://www.ibm.com/cloud/learn/supervised-learning

[5] https://en.wikipedia.org/

[6] https://www.datasciencecentral.com/

[7] Analysis and Detection of DDoS Attacks Using Machine Learning Techniques, Saman Sarraf

[8] Weiss, DDoS Detection Using Deep Neural Networks on Packet Flows. 2019

[9] Holl, Exploring DDoS defense mechanisms. Network, 2015. 25

Appendix

```
#upload necessary libraries
library(stats)
library(rsample)
library(ggplot2)
library(neuralnet)
library(dplyr)
library(stats)
library(plyr)
library(caret)
library(boot)
library(car)
library("Hmisc")
library(corrplot)
library(rpart)
library(rpart.plot)
library(ggplot2)
library(ROCR)
library(e1071)
library(randomForest)
library(pROC)
library(tree)
library(ISLR)

#upload balanced dataset
data<-read.csv(file = "C:/Users/USER/Downloads/ddos_dataset/ddos_dataset.csv")
summary(data)
count(data, data$Label)

#take only columns of interest
data<- data[c(3:8)]
colnames(data)

#remove NA values
data <- na.omit(data)
data

#which variables are more correlated?
res<- cor(data)
round(res, 2)

#pvalue
symnum(res, abbr.colnames = FALSE)
```

```
rcorr(as.matrix(data))

flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
  )
}
res1<-rcorr(as.matrix(data))

flattenCorrMatrix(res1$r, res1$P)

#enlarging margins
par("mar")
par(mar=c(1,1,1,1))

#heatmap
col<- colorRampPalette(c("blue", "white", "red"))(20)
heatmap(x = res, col = col, symm = TRUE, Colv = NA, Rowv = NA)

corrplot(res, type = "upper", method= "circle",
         tl.col = "black",tl.srt = 90,tl.cex = 1)

#Factor
data$Label<-as.factor(data$Label)
data$Label <- ifelse(data$Label=='benign',0,1)

# split data in train and test set (70%)
set.seed(123)
split_train_test = createDataPartition(data$Label, p = 0.7, list = FALSE)

dtrain = data[split_train_test, ]
dtest = data[-split_train_test, ]

#logistic regression
lr_model <- glm(Label~., data = dtrain, family = binomial(link = "logit"))
summary(lr_model)

#check for multicollinearity
vif(lr_model)
alias(lr_model)
```

```
#the model is recognizing that Fwd.Seg.Size.Avg is identical,
#or perfectly predicted by the combination of other predictors
#(highly correlated variables).
#This means that we cannot include that terms in the model.

lr_model <- glm(Label~Fwd.Pkt.Len.Mean+
                 Init.Fwd.Win.Byts+Init.Bwd.Win.Byts+Fwd.Seg.Size.Min,
                data = dtrain, family = binomial(link = "logit"))
summary(lr_model)

#prediction & test error
lr_prob <- predict(lr_model, dtest, type = "response")
lr_pred <- ifelse(lr_prob > 0.5,"benign","ddos")
table_test = table(Predicted = lr_pred, Actual = dtest$Label)
table_test

#confusion matrix

confusionMatrix(factor(lr_pred),
                factor(dtest$Label),
                positive = "Benign")

#AUC curve
test_roc = roc(dtest$Label~lr_prob,plot = TRUE,
               print.auc = TRUE, percentage=TRUE,
               col="#377eb8", lwd=4)

#Decision Tree
dt_model <- tree(Label~.,data = dtrain, method="class")
summary(dt_model)

#plot decision tree
binary.model <- rpart(Label~., data = dtrain,cp=0.001)
rpart.plot(binary.model)

#prediction & test error
dt_prob <- predict(dt_model, dtest, method = "class")
dt_pred <- ifelse(dt_prob  > 0.5,"benign","ddos")
table_test_dt = table(Predicted = dt_pred, Actual = dtest$Label)
table_test_dt

#confusion matrix
confusionMatrix(as.factor(dt_pred),
                as.factor(dtest$Label),
                  positive = "Benign")

#AUC curve
dt_results <- cbind.data.frame(attacktype1=dtest$Label,dt_pred)
res.roc1 <- roc(as.numeric(dt_results$attacktype1),
                  as.numeric(dt_results$dt_pred))
plot.roc(res.roc1, col=2, print.auc=TRUE)

# Random Forest
rf_model <- randomForest(Label~.,ntree=50, mtry=3, importance=TRUE,
                            data=dtrain, method="class")
rf_model

#List of the most important variables used in our Random Forest
varImp(rf_model)
varImpPlot(rf_model)

#prediction & test error
rf_prob <- predict(rf_model, dtest, method = "class")
rf_pred <- ifelse(rf_prob > 0.5,"benign","ddos")
table_test_dt = table(Predicted = rf_prob, Actual = dtest$Label)
table_test_dt

#confusion matrix
confusionMatrix(as.factor(rf_prob),
                  as.factor(dtest$Label),
                  positive = "Benign")


#ROC curve
rf_results <- cbind.data.frame(attacktype=dtest$Label,rf_pred)
res.roc <- roc(as.numeric(rf_results$attacktype),
                 as.numeric(rf_results$rf_pred))
plot.roc(res.roc, col=3, print.auc = TRUE)

#artificial neural networks
#take a sample
data_s <- sample_n(data,5000)
data_s
count(data_s, data_s$Label)

#ddos=0, benign=1
#data$Label<- ifelse(data$Label=='ddos',0,1)
#normalized values
#data has been normalized in order to be adjusted
```

```
#to a common scale so as to compare with more accuracy
#predicted and actual values.
#The technique is the max-min normalization one

normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
maxmindf <- as.data.frame(lapply(data_s, normalize))
maxmindf

#Split data into Train and Test dataset (70% training)

set.seed(33)
dt = sort(sample(nrow(maxmindf), nrow(maxmindf)*.7))
dtrain<- maxmindf[dt,]
dtest<- maxmindf[-dt,]

#count(dtest, dtest$Label)
#count(dtrain, dtrain$Label)

prop.table(table(dtrain$Label))
prop.table(table(dtest$Label))

#Implement ANN on data
#The threshold is set to 0.01, implying that if the change
#in error during an iteration is less than 0.01 then
#no further optimization is done by the model

nn_model <- neuralnet(Label~.,data=dtrain,
                      hidden=c(3,2), linear.output=FALSE,threshold=0.01)
ann_pred <- predict(nn_model, dtest)

nn_prediction <- ifelse(ann_pred > 0.5, 1, 0)
table_test = table(Predicted = nn_prediction, Actual = dtest$Label)
results <- data.frame(actual = dtest$Label, prediction = nn_prediction)
nn$result.matrix
results

nn <- neuralnet(Gender~.,data=dtrain,hidden=c(9),
               linear.output=FALSE,threshold=0.01)
pred <- predict(nn, dtest)
nn_prediction <- ifelse(pred > 0.5, 1, 0)
table_test = table(Predicted = nn_prediction, Actual = dtest$Gender)
results <- data.frame(actual = dtest$Gender, prediction = nn_prediction)
nn$result.matrix
```

```
results


#confusion matrix
confusionMatrix(ann_pred, positive = "benign")
confusionMatrix(factor(mapvalues(nn_prediction,
                                 from=c("0", "1"),
                                 to=c("ddos", "benign"))),
                factor(mapvalues(dtest$Label,
                                 from=c("0", "1"),
                                 to=c("ddos", "benign"))))

#roc curve
test_roc_nn = roc(dtest$Label~ann_pred,plot = TRUE, print.auc = TRUE)
test_roc_nn
as.numeric(test_roc_nn$auc)

# plot the results
plot(nn_model)
```

Unsupervised Learning

Machine learning, statistical learning, deep learning and artificial intelligence

June 2021

Abstract. The World Happiness Report is a landmark survey of the state of global happiness and report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. Leading experts across fields – economics, psychology, survey analysis, national statistics, health, public policy and more – describe how measurements of well-being can be used effectively to assess the progress of nations. The reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness. In this analysis, their data are used to show correlations of the variables used in this Index, furthermore the countries will be analyzed with the help of the Principal Component Analysis (PCA) techniques and with the clustering method to agglomerate similar countries based on Happiness score. **Keywords:** Unsupervised learning, cluster, components, happiness

Camilla Gotta 945522
Università degli Studi di Milano
Data Science and Economics

Contents

1 Introduction **2**

2 Data preprocessing **3**
2.1 Dataset 3
2.2 Data cleaning 4

3 Methodology **5**
3.1 Principal Component analysis 5
3.2 Clustering 5
3.2.1 K-means clustering 5
3.2.2 Hierarchical clustering 6

4 Results **7**
4.1 Principal Component results 7
4.2 Clustering results 10

5 Conclusion **12**

List of Figures

1 Dataset summary 3
2 Correlation matrix of features 4
3 Principal components 7
4 variance explained by principal components 7
5 Variables contribution 8
6 Loading matrix 9
7 Individuals contribution 9
8 Optimal number of clusters 10
9 Cluster plot 10
10 Complete Linkage 11
11 Single Linkage 11
12 Average Linkage 11
13 Ward Linkage 11
14 Hierarchical Cluster box plot 11

1 Introduction

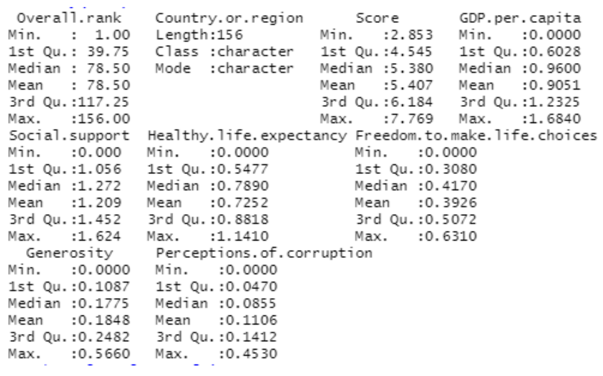
The World Happiness Report is a landmark survey of the state of global happiness. The World Happiness 2019, which ranks 156 countries by their happiness levels, was released at the United Nations at an event celebrating International Day of Happiness on March 20th. The report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. Leading experts across fields – economics, psychology, survey analysis, national statistics, health, public policy and more – describe how measurements of well-being can be used effectively to assess the progress of nations. The reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness.
The variables scores are based on answers to the main life evaluation question asked respondents to think of a ladder with the best possible life for them being a 10 and the worst possible life being a 0 and to rate their own current lives on that scale. The columns following the happiness score estimate the extent to which each of six factors – economic production, social support, life expectancy, freedom, absence of corruption, and generosity – contribute to making life evaluations higher in each country than they are in Dystopia, a hypothetical country that has values equal to the world's lowest national averages for each of the six factors. They have no impact on the total score reported for each country, but they do explain why some countries rank higher than others. In this analysis, their data are used to show correlations of the variables used in this Index, furthermore it will be analyzed the countries with the help of the Principal Component Analysis technique and with the clustering method to agglomerate similar countries, both based on Happiness score and/or based on the other features.

2 Data preprocessing

2.1 Dataset

The used dataset is taken from [1] and it looks like Fig.1.
This file contains the Happiness Score for 153 countries along with the factors used to explain the score.
The Happiness Score is a national average of the responses to the main life evaluation question asked in the Gallup World Poll (GWP), which uses the Cantril Ladder[2].
The Happiness Score is explained by the following factors:

- GDP per capita: GDP per capita is a measure of a country's economic output that accounts for its number of people.
- Social support: Social support means having friends and other people, including family, to turn to in times of need or crisis to give you a broader focus and positive self-image. Social support enhances quality of life and provides a buffer against adverse life events.
- Healthy life expectancy: Healthy Life Expectancy is the average number of years that a newborn can expect to live in "full health"—in other words, not hampered by disabling illnesses or injuries.
- Freedom to make life choices: Freedom of choice describes an individual's opportunity and autonomy to perform an action selected from at least two available options, unconstrained by external parties.
- Generosity: the quality of being kind and generous.
- Perceptions of corruption:The Corruption Perceptions Index (CPI) is an index published annually by Transparency International since 1995 which ranks countries "by their perceived levels of public sector corruption, as determined by expert assessments and opinion surveys.



```
  Overall.rank    Country.or.region      Score       GDP.per.capita
 Min.    :  1.00   Length:156         Min.   :2.853   Min.   :0.0000
 1st Qu.: 39.75   Class :character   1st Qu.:4.545   1st Qu.:0.6028
 Median : 78.50   Mode  :character   Median :5.380   Median :0.9600
 Mean   : 78.50                      Mean   :5.407   Mean   :0.9051
 3rd Qu.:117.25                      3rd Qu.:6.184   3rd Qu.:1.2325
 Max.   :156.00                      Max.   :7.769   Max.   :1.6840
 Social.support  Healthy.life.expectancy Freedom.to.make.life.choices
 Min.   :0.000   Min.   :0.0000          Min.   :0.0000
 1st Qu.:1.056   1st Qu.:0.5477          1st Qu.:0.3080
 Median :1.272   Median :0.7890          Median :0.4170
 Mean   :1.209   Mean   :0.7252          Mean   :0.3926
 3rd Qu.:1.452   3rd Qu.:0.8818          3rd Qu.:0.5072
 Max.   :1.624   Max.   :1.1410          Max.   :0.6310
   Generosity     Perceptions.of.corruption
 Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.1087   1st Qu.:0.0470
 Median :0.1775   Median :0.0855
 Mean   :0.1848   Mean   :0.1106
 3rd Qu.:0.2482   3rd Qu.:0.1412
 Max.   :0.5660   Max.   :0.4530
```

Figure 1: Dataset summary

2.2 Data cleaning

Statistical learning algorithms often require numerical data therefore ”Country or region” column was not used in the analysis, but only for labelling data results. as data cleaning always required, another step was the detection of null values thus removing them. To reach the purpose data have been scaled in order to let them be comparable with each other. Subsequently, it has been detect features importance and if it was the case of collinearity between variables: the respective correlation values between variables is shown in Fig.2.

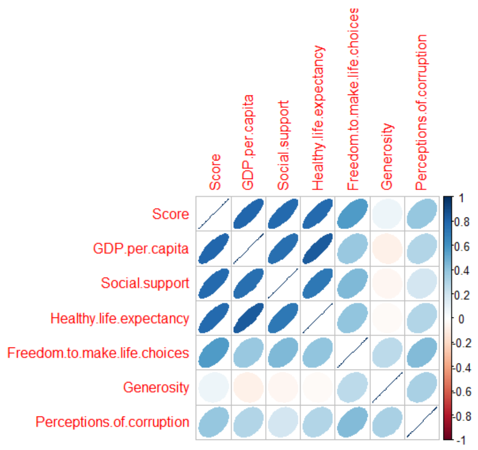


Figure 2: Correlation matrix of features

As seen, the scores with lighter colors offer higher scores, whereas darker areas indicate lower correlation scores. For instance, we see that the Economy/ GDP score and Health, Life expectancy are highly correlated as well as a good correlation score is shown between GDP score and Social support while there is literally no variables that are negatively correlated one-another.

3 Methodology

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, and image recognition. In this project unsupervised learning algorithms are utilized for clustering and dimensionality reduction purposes.

3.1 Principal Component analysis

Principal components analysis (PCA) in an unsupervised approach which involves only a set of features and no associated response.
It refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data: when faced with a large set of correlated variables, principal components allow to summarize this set with a smaller number of representative variables that correctively explain most of the variability in the original set[3].

3.2 Clustering

Clustering is a data mining technique which groups unlabeled data based on their similarities or differences. Clustering algorithms are used to process raw, unclassified data objects into groups represented by structures or patterns in the information. This section is focus on perhaps the two best-known clustering approaches: k-means clustering and hierchical clustering. K-means clustering requests to partition the observations into a prespecified number of clusters. On the other hand in hierarchical clustering we do not have a measure to know in advance how many clusters are needed; in fact one ends up with a tree-like visual representation of the observations, called a dendogram, that allows to view at once the clusterings obtained in each possible number of clusters, from 1 to n.

3.2.1 K-means clustering

K-means clustering is a common example of an exclusive clustering method where data points are assigned into K distinct and non-overlapping groups, where K represents the number of clusters based on the distance from each group's centroid. The data points closest to a given centroid will be clustered under the same category. A larger K value will be indicative of smaller groupings with more granularity whereas a smaller K value will have larger groupings and less granularity.
The idea behind is that a good clustering is one for which the within-cluster variation is as small as possible: the within-cluster variation for cluster is a measure of the amount by which the observations within a cluster differ from each other[3].

3.2.2 Hierarchical clustering

Hierarchical clustering, also known as hierarchical cluster analysis (HCA), is an unsupervised clustering algorithm that can be categorized in two ways; they can be agglomerative or divisive. Agglomerative clustering is considered a "bottoms-up approach": its data points are isolated as separate groupings initially, and then they are merged together iteratively on the basis of similarity until one cluster has been achieved. Four different methods are commonly used to measure similarity, also known as linkage:

- Ward's linkage: This method states that the distance between two clusters is defined by the increase in the sum of squared after the clusters are merged.
- Average linkage: This method is defined by the mean distance between two points in each cluster
- Complete (or maximum) linkage: This method is defined by the maximum distance between two points in each cluster
- Single (or minimum) linkage: This method is defined by the minimum distance between two points in each cluster

Euclidean distance is the most common metric used to calculate these distance, defined as: $d\left(p,q\right) = \sqrt{\sum_{i=1}^{n} \left(q_i - p_i\right)^2}$, Divisive clustering instead can be defined as the opposite of agglomerative clustering; instead it takes a "top-down" approach. In this case, a single data cluster is divided based on the differences between data points. Divisive clustering is not commonly used, but it is still worth noting in the context of hierarchical clustering. These clustering processes are usually visualized using a dendrogram, a tree-like diagram that documents the merging or splitting of data points at each iteration.

4 Results

4.1 Principal Component results

PCA is a procedure for identifying a smaller number of uncorrelated variables, called “principal components”, from a large set of data. The goal of principal components analysis is to explain the maximum amount of variance with the minimum number of principal components. The percentages on each bar indicate the proportion of total variance explained by the respective principal component. The first PC corresponds to the maximum amount of variation in the data set, the scree plot in Fig.3 shows which components explain most of the variability in the data: in this case, almost 85% of the variances contained in the data are retained by the first two principal components.

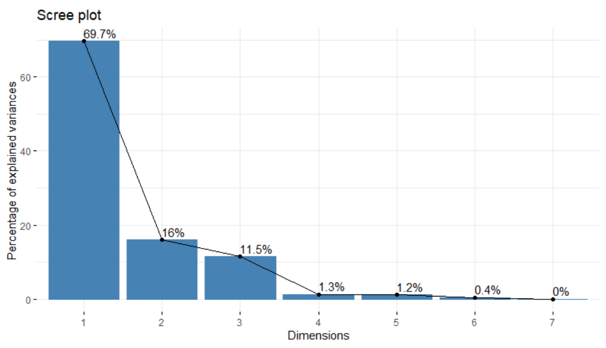


Figure 3: Principal components

A second step concerns the plot of the proportion of variance explained by each component as well as the cumulative proportion of variability explained, as shown in Fig. 5

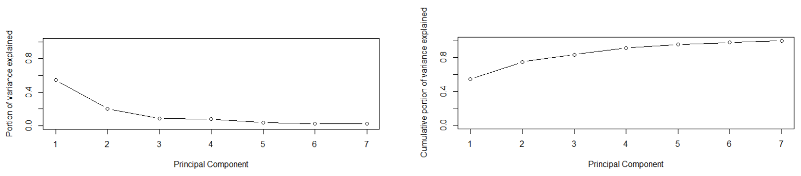


Figure 4: variance explained by principal components

Variables that are correlated with PC1 and PC2 are the most important in explaining the variability in the data set: the contribution of variables was extracted: the larger the value of the contribution, the more the variable contributes to the component.

This highlights the most important variables in explaining the variations retained by

the principal components. Firstly, how variables relate to the dimensions, and how they relate to each other. The main indicator here is the distance, distance between a variable and the axis, distance between two variables' arrows and the distance between the tip of the arrow and the circle. PCA plots are interpreted as follows: sites that are close together in the diagram have a similar composition; sites 5, 6,7, and 8 are quite similar.
The origin (0,0) is the averages. Points near the origin are either average or poorly explained.
If an arrow is close to dimension 1 (horizontal axis), it correlates well with it. In this case, Generosity seems to move together with the right side of the first dimension whereas Freedom to make life choices is a better indicator for the second dimension, as well as Happiness, Social support, Gdp per capita and Healthy life expectancy.
Moreover If two variables are close to each other, they have little distance, thus similar. If the tip of the arrow of a variable is close to the circle, this variable is well-explained by the first two dimensions (example most of variables) while if the length of the arrow is shorter, it is not as well explained (perception of corruption), as shown in Fig. 5.

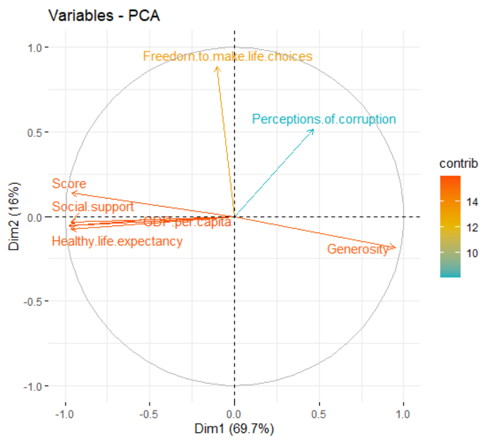
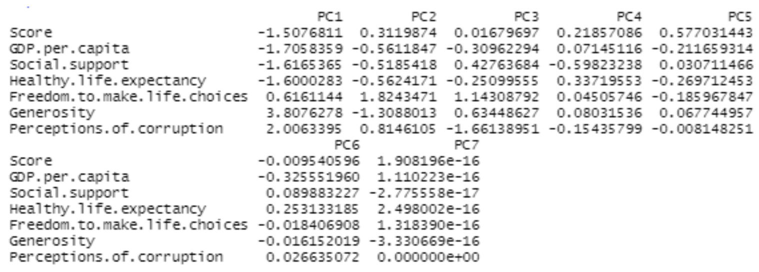


Figure 5: Variables contribution

Moreover, as in Fig.6, for the first component the biggest variability proportion is explained by Generosity and Perception of corruption, while the second component in the major part by the freedom to life choices and secondly the remaining variables.



```
                                   PC1        PC2         PC3         PC4          PC5
Score                       -1.5076811  0.3119874  0.01679697  0.21857086  0.577031443
GDP.per.capita              -1.7058359 -0.5611847 -0.30962294  0.07145116 -0.211659314
Social.support              -1.6165365 -0.5185418  0.42763684 -0.59823238  0.030711466
Healthy.life.expectancy     -1.6000283 -0.5624171 -0.25099555  0.33719553 -0.269712453
Freedom.to.make.life.choices 0.6161144  1.8243471  1.14308792  0.04505746 -0.185967847
Generosity                   3.8076278 -1.3088013  0.63448627  0.08031536  0.067744957
Perceptions.of.corruption    2.0063395  0.8146105 -1.66138951 -0.15435799 -0.008148251
                                     PC6           PC7
Score                       -0.009540596  1.908196e-16
GDP.per.capita              -0.325551960  1.110223e-16
Social.support               0.089883227 -2.775558e-17
Healthy.life.expectancy      0.253133185  2.498002e-16
Freedom.to.make.life.choices -0.018406908  1.318390e-16
Generosity                  -0.016152019 -3.330669e-16
Perceptions.of.corruption    0.026635072  0.000000e+00
```

Figure 6: Loading matrix

The plot about the contribution of individuals, as shown in Fig 7 shows in the right side countries that are close to each other, meaning that they show similar characteristics in the variables.
So why are they different from the first block of countries? Well, first of all they manage to be less good on the first dimension, and as they are below the first dimensional axis, they score worse even in the second dimension.
So knowing all that already said about the contributions of each variable to the dimensions and the quality of representation, the more they are on the right side of the second dimension, the highest is the Happiness score, the GDP per capita and the Health/ Life expectancy and the Family variables. The more high a country is on the first dimensional axis, the more corruption is in his country. If case of a difficult interpretation of the dimensions, one can see observations that are characteristically different from others, and one can see again it is not always possible to interpret the dimensions of a PCA.

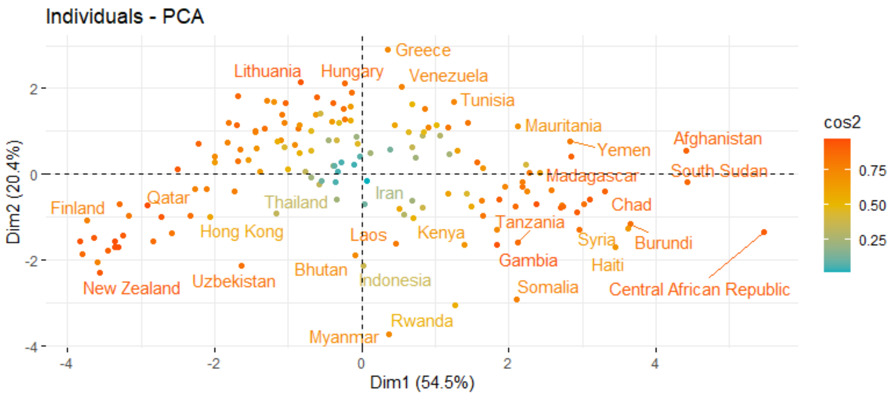


Figure 7: Individuals contribution

4.2 Clustering results

The aim is to make clusters from the data that can segment similar countries together, and thanks to the Silhouette method it was find that the optimal number of k cluster, in order to minimize the within-cluster variability, has to set at 3, as shown in Fig. 8

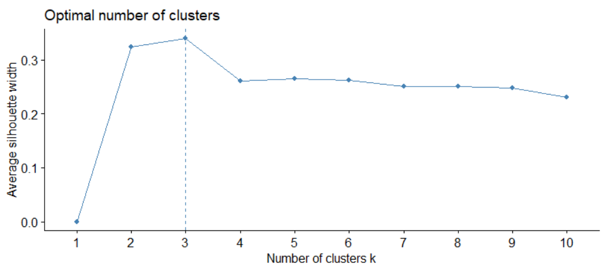


Figure 8: Optimal number of clusters

Moreover the results of K-means clustering are shown in Fig. 9 and the it seems to perform as expected: from the clusters is possible to see three different groups of countries with similar characteristics with respect to the analysed features.

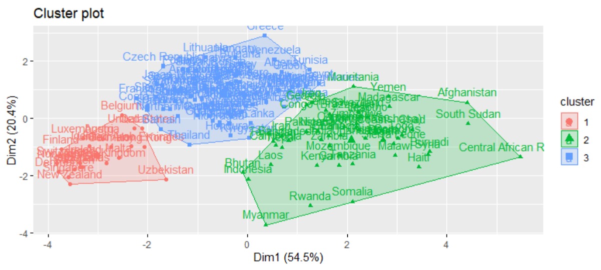


Figure 9: Cluster plot

Lastly, as already mentioned, was performed the hierarchical cluster which has been performed after the standardization on variables considering Euclidean distance measure, $d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$ and four agglomeration methods (complete, average, single, Ward linkages) in order to check for the robustness of cluster analysis.
The results shown in Fig.10-13 do not change so much thanks to a built-in function it is possible to verify that same countries are grouped together most of cases and that's not happened just for the single linkage, so one could conclude that the clusters are quite well defined.

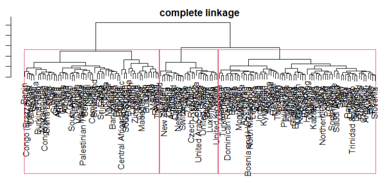


Figure 10: Complete Linkage

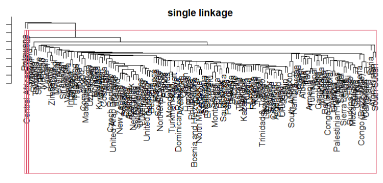


Figure 11: Single Linkage

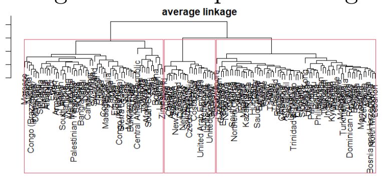


Figure 12: Average Linkage

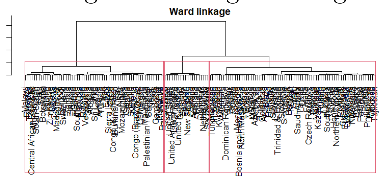


Figure 13: Ward Linkage

Another way to understand the clusters is to produce boxplots for each variable within each cluster: for example, in Fig.14 cluster 1 is strong on perception of corruption and happiness score, cluster number 2 in social support and healthy life expectancy and the cluster in generosity and freedom to make life choices.

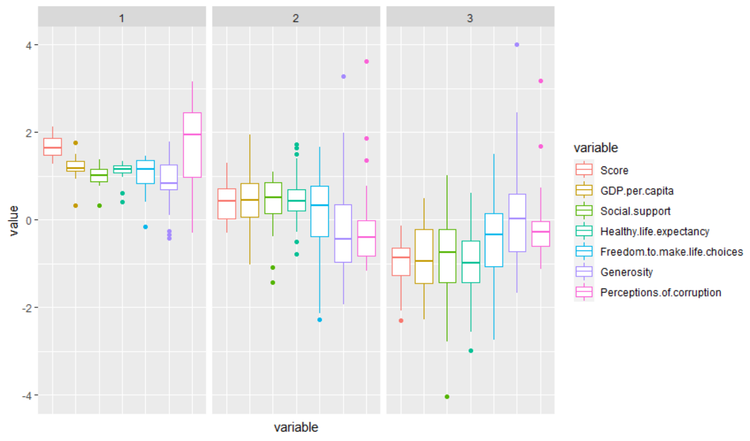


Figure 14: Hierarchical Cluster box plot

5 Conclusion

The World Happiness Report is a landmark survey of the state of global happiness. The report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. It reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness.
The variables scores are based on answers to the main life evaluation question asked respondents in a range from o to 10: the columns following the happiness score estimate the extent to which each of six factors – economic production, social support, life expectancy, freedom, absence of corruption, and generosity. To reach the initial purpose of the analysis, data are used to show correlations of the variables used in this Index, furthermore it has been analyzed the countries with the help of the Principal Component Analysis technique and with the clustering method to agglomerate similar countries, both based on Happiness score and/or based on the other features. Clustering results show mainly three groups around the Happiness level: one can be defined made of developed countries, such as Europe, U.S.A. but also Canada and so on; the second by the developed countries but with some low levels in corruption/happiness score and at last the biggest cluster show mainly the last 50 countries with low happiness level probably due to low life condition, it is not surprisingly that the most are part of Africa and East Asia. One can conclude that the aim was reached and more detailed analysis could be done to explore the life condition and economic status of the world, and consequently to increase the efficiency of helps in needed countries.

References

[1] https://www.kaggle.com/unsdsn/world-happiness?select=2019.csv

[2] Helliwell, John F., Richard Layard, Jeffrey Sachs, and Jan-Emmanuel De Neve, eds. 2020. World Happiness Report 2020. New York: Sustainable Development Solutions Network

[3] An Introduction to Statistical Learning by Daniela Witten, Robert Tibshirani e Trevor Hastie

Appendix

```
#upload libraries
library(factoextra)
library(reshape2)
library(cluster)
library(ggplot2)
library(FactoMineR)
library(corrplot)
library(dplyr)
library(plotly)
library(stringr)
library(ggthemes)
library(NbClust)

#upload balanced dataset
mydata<-read.csv(file = "C:/Users/USER/Downloads/2019/2019.csv")
summary(mydata)
rownames(mydata)<-mydata$Country.or.region
data<- mydata[c(2:8)]
colnames(data)

#remove NA values
data <- na.omit(data)

#enlarging margins
par("mar")
par(mar=c(1,1,1,1))

#top 10
hpi_sort <- data[with(data, order(-Score)), ]
hpi_top_10 <- head(hpi_sort, 10)
hpi_top_10 <- hpi_top_10[, c(1:7)]
hpi_top_10

#last 10
hpi_bottom_10 <- tail(hpi_sort, 10)
hpi_bottom_10 <- hpi_bottom_10[, c(1:7)]
hpi_bottom_10

#correlation pca plot
data<- cor(data)
corrplot(data, method="ellipse")

#implementing pca
```

```
pr.out=prcomp(data, scale=TRUE)
pr.out$center
pr.out$rotation
pr.out$scale
pr.out$x
pr.out$sdev

pr.var=pr.out$sdev^2
pve=pr.var/sum(pr.var)

# principal components
fviz_screeplot(pr.out, addlabels = TRUE)

#pve by each component
plot(pve, xlab="Prncipal Component", ylab="Proportion of variance explained",
     ylim=c(0,1), type='b')

#cumulative pve
plot(cumsum(pve), xlab="Principal Component",
     ylab="Cumulative proportion of variance explained",
     ylim=c(0,1), type='b')

#pca variables
var$contrib
var <- get_pca_var(pr.out)
fviz_pca_var(pr.out, col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE # Avoid text overlapping
)

#pca individuals
ind <-get_pca_ind(pr.out)
fviz_pca_ind(pr.out, col.ind = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE
)


#CLUSTER - KMEANS
# K-Means

#To standarize the variables
data.stand <- scale(mydata[,-1])

#select number of K
```

```
set.seed(123)
wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")
  wss
}

wssplot(data.stand)

fviz_nbclust(data.stand, kmeans,nstart=15, method = "silhouette")

# k = 3
k.means.fit <- kmeans(data.stand, 3)
str(k.means.fit)
clusplot(data.stand, k.means.fit$cluster,
         main='2D representation of the Cluster solution',
         color=TRUE,
         labels=2, lines=0)
fviz_cluster(k.means.fit, data = data.stand)

#CLUSTER-HIERARCHICAL

rownames(mydata)<-mydata$Country.or.region
data<-mydata[,-1]

#choose the distances
d1 <- dist(data, method="euclidean", diag=F, upper=F)

#function to generate the agglomeration program
agglo <- function(hc){
  data.frame(row.names=paste0("Cluster",seq_along(hc$height)),
             height=hc$height,
             components=ifelse(hc$merge<0,
                               hc$labels[abs(hc$merge)],
                               paste0("Cluster",hc$merge)),
             stringsAsFactors=FALSE) }

#with complete linkage
h1 <- hclust(d1, method="complete"); h1
agglo(h1)
plot(h1, main="complete linkage")
```

```
complete <- cutree(h1, k=3)
rect.hclust(h1, 3)
h1cluster <- cutree(h1, k=3)
h1cluster

#with avg linkage
h2<-hclust(d1,method="average");h2
agglo(h2)
plot(h2, main="average␣linkage")
average <- cutree(h2, k=3)
rect.hclust(h2, 3)
h2cluster <- cutree(h2, k=3)
h2cluster

#with single linkage
h3<-hclust(d1,method="single");h3
agglo(h3)
plot(h3, main="single␣linkage")
single<- cutree(h3, k=3)
rect.hclust(h3, 3)
h3cluster <- cutree(h3, k=3)
h3cluster

#with ward linkage
h4<-hclust(d1,method="ward.D");h4
agglo(h4)
plot(h4, main="Ward␣linkage")
ward<- cutree(h4, k=3)
rect.hclust(h4, 3)
h4cluster <- cutree(h4, k=3)
h4cluster

plot(data, col=h4cluster , main="ward␣likage")

#Add the cluster to the dataset
h4cluster<- cutree(h4, k=3)
data <- as.data.frame(data)
data$clu<-h4cluster
data$clu<-h4cluster

#Means for variables
medie<-aggregate(data, list(h4cluster), mean)
medie

#Calculus of R^2 for each variables
```

```
mydata<-data
R2 <- rep(NA, (ncol(mydata)-1))
for(i in 1:(ncol(mydata)-1))
  R2[i] <- anova(aov(mydata[,i] ~
                          mydata[,ncol(mydata)]))[1,2]/
  (anova(aov(mydata[,i] ~ mydata[,ncol(mydata)]))[1,2]+
     anova(aov(mydata[,i] ~ mydata[,ncol(mydata)]))[2,2])
R2

mydata<-mydata[,-ncol(mydata)]
col<-colnames(mydata)
finali<-cbind(col,R2)
finali

# Plots for cluster interpretation
col<-colnames(mydata)
mydataz<-data.frame(scale(mydata))
mydataz$clu<-h4cluster
dati <- melt(mydataz, measure.vars=col)
ggplot(dati, aes(x = variable, y = value, color=variable)) +
  geom_boxplot() +
  facet_wrap(~ mydataz$clu) +
  theme(axis.text.x=element_blank(),
        axis.ticks.x=element_blank())
```