

FEDERAL UNIVERSITY OF PARANÁ

MARCO ANTONIO REICHERT BOARETTO

**MACHINE LEARNING TECHNIQUES APPLIED IN HUMAN ACTIVITY
RECOGNITION USING RGB-D VIDEOS**

CURITIBA

2017

MARCO ANTONIO REICHERT BOARETTO

**MACHINE LEARNING TECHNIQUES APPLIED IN HUMAN ACTIVITY
RECOGNITION USING RGB-D VIDEOS**

Dissertation submitted to the Graduate Program in Electrical Engineering from the Federal University of Paraná, as partial fulfilment of the requirements for the degree of Master of Science in Electrical Engineering.

Supervisor: Prof. Dr. Leandro dos Santos Coelho

CURITIBA

2017

B662m

Boaretto, Marco Antonio Reichert

Machine learning techniques applied in human activity recognition using RGB-D videos / Marco Antonio Reichert Boaretto. – Curitiba, 2017.

97 f. : il. color. ; 30 cm.

Dissertação - Universidade Federal do Paraná, Setor de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, 2017.

Orientador: Leandro dos Santos Coelho.

1. RGB-D. 2. Kinect. 3. Aprendizado de máquina. I. Universidade Federal do Paraná.
II. Coelho, Leandro dos Santos. III. Título.

CDD: 005.2



UNIVERSIDADE FEDERAL DO PARANÁ
Programa de Pós-Graduação em Engenharia Elétrica, PPGE
Setor de Tecnologia

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA ELÉTRICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de MARCO ANTONIO REICHERT BOARETTO intitulada: MACHINE LEARNING TECHNIQUES APPLIED IN HUMAN ACTIVITY RECOGNITION USING RGB-D VIDEOS, após terem inquirido o aluno e realizado a avaliação do trabalho, são de parecer pela sua aprovação no rito de defesa.
A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 22 de Novembro de 2017.

LEANDRO DOS SANTOS COELHO
Presidente da Banca Examinadora (UFPR)

EVELIO MARTÍN GARCÍA FERNÁNDEZ
Avaliador Interno (UFPR)

GIDEON VILLAR LEANDRO
Avaliador Interno (UFPR)
JULIO CÉSAR NIEVOLA
Avaliador Externo (PUC/PR)

ACKNOWLEDGEMENTS

First of many thanks, I want to thank my parents Jacir Boaretto and Claudia Reichert for all the love, support and education provided to me, since my first years of school until now, and for always believing in me.

I want to thank my brother Bruno Rafael Reichert Boaretto, for all the discussions and support, for watching my presentations and specially for the patience.

To all my friends and colleagues for the friendship and memories. A special thanks to my friend Rodrigo Meira de Andrade for the all the years of support, good advice and friendship.

A special thanks to my girlfriend Kwan Pei Lu, whom gave me support and patience through these final months of my Masters.

And last but not least, I can't thank enough my supervisor Leandro dos Santos Coelho, which since from my introduction in the science with my PIBIC has giving me support, orientation and good advice through all my academic career.

Epigraph

“You can have data without
information, but you cannot have
information without data”.

Daniel Keys Moran

ABSTRACT

Given the particularities and issues on dealing with two Dimensions (2D) images, as illumination and object occlusion, one better option to counteract this matter is to work with three Dimensions (3D) images or Red, Green and Blue – Depth (RGB-D) as they are usually called. RGB-D images are invariant of illumination since mostly of its acquisition devices use infra-red or time-of-flight laser sensors. The Microsoft® Kinect developed in partnership with PrimeSense is an amazing tool for RGB-D low resolution image acquisition, which its applications vary from gaming to medical imagery. Since Kinect has an accessible cost, it has been widely used in researches on many areas that use computer vision and image classification. Several datasets have already been developed with the Kinect for RGB-D image classification, as for example the Berkeley's Multimodal Human Activity Database (MHAD) from the Tele immersion Laboratory of University of California and the Center for Imaging Science of Johns Hopkins University, which contain images of 10 subjects performing 11 activities: jumping in place (jump), jumping jacks (jack), bending-hands up all the way down (bend), punching (punch), waving two hands (wave2), waving right hand (wave1), clapping hands (clap), throwing a ball (throw), sit down and stand up (sit+stand), sit down (sit), stand up (stand). The main goal of this dissertation is to compare different machine learning approaches, (i) using a proposed ensemble learning technique with Support Vector Machines (SVM), K-Nearest Neighbors (kNN), Extreme Gradient Boosting (XGBoost) and Artificial Neural Networks (ANN) combined with three different dimensionality reduction techniques Principal Component Analysis (PCA), Factor Analysis (FA) and Nonnegative Matrix Factorization (NMF) and (ii) from the Deep Learning (DL) approach using a proposed convolutional neural network (CNN) architecture known as BOANet, using the MHAD as Dataset. The contribution of the project consists on a human activity recognition system (HAR) that uses Kinect for RGB-D image recognition and machine learning algorithm to build the model classifier. The proposed approaches have its performance compared with reference values from recent works with the MHAD of the literature. Both approaches got remarkable performance having better results than most of the reference values from the literature, the (i) approach achieved 99.93% of classification accuracy and (ii) achieved 99.05% of classification accuracy.

Key-words: RGB-D, Kinect, Machine Learning, Deep Learning, Human Activity Recognition.

RESUMO

De acordo com certas particularidades e dificuldades em lidar com imagens 2D, como por exemplo iluminação e obstrução de objetos, uma melhor opção para o problema em questão é utilizar imagens três dimensões (3D) ou *Red, Green and Blue - Depth* (RGB-D) como comumente são chamadas. Imagens RGB-D são invariantes a luz pelo fato da maioria dos seus dispositivos de aquisição utilizarem infravermelho ou sensores de laser *time-of-flight*. O Kinect da Microsoft® que foi desenvolvido em parceria com a *PrimeSense* é uma ferramenta incrível para aquisição de imagens RGB-D de baixa resolução, suas aplicações variam de jogos a imagens médicas. Como o Kinect possui um custo acessível, vem sendo muito utilizado em pesquisas de diversas áreas que fazem uso de visão computacional e classificação de imagens. Diversas base de dados para classificação de imagens RGB-D já foram desenvolvidas com o Kinect, como por exemplo a base de dados multimodal de atividade humana (MHAD) desenvolvido pelo laboratório de tele imersão da Universidade de Califórnia em parceria com o Centro de Ciências de Imagem da Universidade John Hopkins, na qual contem imagens de 10 pessoas desenvolvendo 11 atividades: pulando no lugar (pular), polichinelo (polichinelo), curvando o corpo para frente até o chão (curvar), socando (socar), acenando com as duas mãos (acenando2), acenando com a mão direita (acenando), batendo palmas (palmas), arremessando uma bola (arremessar), sentar e ficar de pé (sentar+levantar), sentando (sentar), ficando de pé (levantar). O principal objetivo da dissertação consiste em comparar duas abordagens de aprendizado de máquina, (i) usando um proposto comitê de máquina com *Support Vector Machines* (SVM), *K-Nearest Neighbors* (KNN), *Extreme Gradient Boosting* (XGBoost) e *Artificial Neural Networks* (ANN) combinado com três diferentes técnicas de redução de dimensionalidade *Principal Component Analysis* (PCA), *Factor Analysis* (FA) e *Nonnegative Matrix Factorization* (NMF) e (ii) de uma abordagem de aprendizado profundo usando uma proposta arquitetura de *Convolutional Neural Network* (CNN) chamada de BOANet, usando o MHAD como base de dados. A contribuição do projeto consiste em um sistema de reconhecimento de atividade humana que usa o Kinect para reconhecimento de imagens RGB-D e algoritmos de aprendizado de máquina para construir um modelo classificador. As abordagens propostas tiveram sua performance comparada com valores de referência de recentes trabalhos com o MHAD da literatura. Ambas abordagens tiveram ótima performance obtendo resultados melhores do que a maioria dos valores referência da literatura, a abordagem (i) conseguiu atingir um valor de 99.93% de precisão de classificação e a (ii) 99.05%.

Palavras-chave: RGB-D, Kinect, Aprendizado de máquina, Aprendizado profundo, Reconhecimento de Atividade Humana.

LIST OF FIGURES

Figure 1.1 - Proposed Methodology	16
Figure 2.1 – Kinect Sensor.	21
Figure 2.2 – Comparison Between Kinect and Minolta	22
Figure 2.3 – Color and Depth Image from Kinect.	23
Figure 2.4 - Actions Performed By One Subject With Both Color and Depth Images.....	24
Figure 2.5 - Display of The Sensors from The MHAD.....	25
Figure 3.1 - Google Trends for Machine Learning Performed in 05/06/2017.....	26
Figure 3.2 - Dimensionality Reduction	28
Figure 3.3 - PCA Method from 3D to 2D	29
Figure 3.4 - Principal Components	31
Figure 3.5 - PCA Pseudocode	31
Figure 3.6 - Common Factor Model	33
Figure 3.7 - FA Extraction Methods	34
Figure 3.8 – Factors Rotation	36
Figure 3.9 - FA Pseudocode	37
Figure 3.10 - FA Vs. PCA	39
Figure 3.11 – NMF Decomposition.....	40
Figure 3.12 – NMF Decomposition with Image	40
Figure 3.13 – Example of Classification Using Ensembles	42
Figure 3.14 – Stages of The Methodology of Creating an Ensemble.....	43
Figure 3.15 – Application of an Ensemble, in Classification (a) and in Regression (b).....	43
Figure 3.16 – Bagging	45
Figure 3.17 – Boosting.....	45
Figure 3.18 – Stacking	47
Figure 3.19 - SVM Classification Hyperplane.....	48
Figure 3.20 - Cover Theorem	49
Figure 3.21 - One-vs-All	51
Figure 3.22 – Decision Tree.....	52
Figure 3.23 – Gradient Descent Method in a Surface.....	53
Figure 3.24 – GBM Pseudocode.....	54

Figure 3.25 –Distance Comparison Related to k Value in The kNN Algorithm ..	56
Figure 3.26 – Biological Neuron.....	57
Figure 3.27 – Mathematical Representation of a Neuron.....	58
Figure 3.28 – MLP Structure	59
Figure 3.29 – Machine Learning vs. Deep Learning	60
Figure 3.30 – LeNet Architecture.	61
Figure 3.31 – Hubel and Wiesel Experiment.	62
Figure 3.32 – Example of a 4x4x3 Input Image.	63
Figure 3.33 – 2D Image Convolution.....	64
Figure 3.34 – ReLU Function Graph	65
Figure 3.35 – Leaky ReLU Function Graph	65
Figure 3.36 – Example of Max Pooling Operation	66
Figure 3.37 – Example of Flattening	67
Figure 4.1 - Methodology Flowchart.....	69
Figure 4.2 - Depth Image from the Kinect Sensor	71
Figure 4.3 - a) Original Depth Image ROI Left, b) Background Subtraction Result Image ROI Right.	72
Figure 4.4 - a) Smoothed Image Left, b) Smoothed Image + Colomar Map Change Right.	73
Figure 4.5 - Image Vectorization Example.....	74
Figure 4.6 – Cumulative Explained Variance	74
Figure 4.7 – Proposed Stacking Approach	75
Figure 4.8 - CV Algorithm	77
Figure 4.9 – Image Compression	78
Figure 5.1 - Example of Confusion Matrix.....	79

LIST OF TABLES

Table 2.1 – Comparison Between Kinect and Minolta.....	22
Table 3.1 – Comparison Between PCA and FA.....	38
Table 3.2 – NMF Cost Functions	41
Table 3.3 – Comparison Between Bagging, Boosting and Stacking	48
Table 3.4 - Kernel Functions	50
Table 3.5 – List of Activation Functions	58
Table 4.1 – Base and Meta Learners Parameters List.....	76
Table 5.1 – Results	80
Table 5.2 – Results of accuracy per class.....	81

LIST OF ACRONYMS

1M	- One Million
ANN	- Artificial Neural Network
Bagging	- Bootstrap Aggregating
CART	- Classification and Regression Tree
CFA	- Confirmatory Factor Analysis
CNN	- Convolutional Neural Network
CPU	- Central Processing Unit
CV	- Cross-Validation
DBN	- Deep Belief Networks
DESTIN	- Deep Spatiotemporal Inference Network
DL	- Deep Learning
EFA	- Exploratory Factor Analysis
eq.	- equation
FA	- Factor Analysis
FC	- Fully Connected
fps	- frames per second
GB	- Gigabyte
GBM	- Gradient Boosting Machine
G-SVM	- Gaussian SVM
HAR	- Human Activity Recognition
HARED	- HAR based on a string Edit Distance
HTM	- Hierarchical Temporal Memory
kNN	- K-Nearest Neighbors
MB	- Megabyte
MHAD	- Multimodal Human Action Database
MLE	- Maximum Likelihood Estimation
MLP	- Multi-Layer Perceptron
MNIST	- Modified National of Standards and Technology
MOCAP	- Motion Capture
NMF	- Non-Negative Matrix Factorization
PCA	- Principal Component Analysis

PF	- Principal Component
RAM	- Random Access Memories
ReLU	- Rectified Linear Unit
RGB-D	- Red, Green and Blue Depth
RNN	- Recurrent Neural Network
ROI	- Region of Interest
SVM	- Support Vector Machine
SVP	- Senior Vice President
WLS	- Weighted Least Squares
XGBoost	- Extreme Gradient Boosting Machine

CONTENTS

1	INTRODUCTION	13
1.1	RELATED WORK	16
1.2	OBJECTIVES	19
1.2.1	Specific Objectives	19
1.3	STRUCTURE OF THE DISSERTATION	20
2	PROBLEM DEFINITION	21
2.1	KINECT SENSOR	21
2.1.1	RGB-D	22
2.2	MHAD	23
3	FUNDAMENTALS OF MACHINE LEARNING	26
3.1	DIMENSIONALITY REDUCTION	28
3.1.1	PCA	29
3.1.2	FA	32
3.1.2.1	Extraction Methods	34
3.1.2.2	Factors Rotation	35
3.1.2.3	Factors Score	36
3.1.3	PCA vs. FA	37
3.1.4	Non-Negative Matrix Factorization	39
3.2	ENSEMBLE LEARNING	42
3.2.1	Bootstrap Aggregating	44
3.2.2	Boosting	45
3.2.3	Stacking Generalization	47
3.3	SUPPORT VECTOR MACHINES	48
3.3.1	Multi-Class SVM	50
3.3.1.1	One-vs-all	51
3.3.1.2	All-vs-all	51
3.4	EXTREME GRADIENT BOOSTING	52
3.4.1	Gradient Boosting Machine	52
3.4.2	XGBoost Algorithm	54
3.5	K-NEAREST NEIGHBORS	55
3.6	ARTIFICIAL NEURAL NETWORK	57
3.6.1	Deep Learning	60

3.6.2	Convolutional Neural Network	61
3.6.2.1	Input Layer.....	63
3.6.2.2	Convolution Layer.....	63
3.6.2.3	Rectified Linear Unit Layer	64
3.6.2.4	Pooling Layer.....	66
3.6.2.5	Flattening Layer	66
3.6.2.6	Fully Connected Layer.....	67
3.6.2.7	Special Operations	67
4	METHODOLOGY	69
4.1	IMAGE PROCESSING	71
4.2	DIMENSIONALITY REDUCTION	73
4.3	ENSEMBLE LEARNING CLASSIFICATION	75
4.3.1	K-fold Cross-Validation	76
4.4	DEEP LEARNING CLASSIFICATION	77
5	EXPERIMENTAL RESULTS.....	79
6	FINAL CONSIDERATIONS.....	82
6.1	CONCLUSION.....	82
6.2	FUTURE WORKS.....	82
	REFERENCES.....	84

1 INTRODUCTION

One of the main advantages on working with 2D video processing is the variety of hardware for image acquisition that is available today, and how accessible it can be, starting from a basic smartphone camera to an ultra-high resolution professional camera.

However, regardless of image quality, some application with 2D images has their efficiency depending on factors as illumination and target position. For example, in a face recognition application, the result of the acquisition will vary depending on face position, illumination, the use of accessories and facial expression (LI et al., 2013; SEGUNDO et al., 2013; HAYAT et al., 2016).

In the advent of new sensing technologies, one reliable alternative to solve such problems will be using three dimensions (3D) images acquisition. The results of a 3D image acquisition, since most sensors use infra-red light or laser sensor, are invariant on illumination, even in absence of light (LI et al., 2013) and present a higher robustness regarding target position (SEGUNDO et al., 2013).

There are several options for 3D image acquisition sensors on the market. One in particular regarding its effectiveness, noninvasive capture and low cost price is the Microsoft® Kinect sensor which since it was presented it became widely used in industry and research in several knowledge fields such as Time-Of-Flight systems (CORTI et al., 2016), pose and gesture recognition (DING; CHANG, 2015; KASTANIOTIS et al., 2015; DARBY et al., 2016; DOLATABADI et al., 2016; IBAÑEZ et al., 2016), reconstructions of chronic wounds (FILKO et al., 2016), face recognition (GOSWAMI et al., 2013; LI et al., 2013; SEGUNDO et al., 2013; HAYAT et al., 2016).

The images that are acquired by the Kinect are called RGB-D images (Red, Green and Blue with Depth images) which are the result of acquisition of both color (2D) and depth (3D) images simultaneously.

With the decrease of cost for 3D image sensors, several applications and novel datasets are getting publicly available for researchers, as in this project the Multimodal Human Action Database (MHAD) for Human Activity Recognition (HAR) from University of California with depth images proposed by (OFLI et al., 2013) will be used for the case study of HAR.

Image and video datasets are known for being large and sometimes uneasy to handle given to the number and the resolution of its frames. In order to make the

original dataset possible to work with and access the *curse of dimensionality* issue, dimensionality reduction techniques would be applied to extract relevant features and reduce the size of the dataset without losing too much information.

Principal Component Analysis (PCA) proposed by Pearson in 1901 (PEARSON, 1901) in order to find the lines and planes that fits better in a set of points in space, is a widely used dimensionality reduction technique in image applications. Recent researches using PCA has produced promising results such as (ÜZÜMCÜ et al., (2003); PATIL; MUDENGUDI , (2011); SADHASIVAM et al., (2011); and LI; TAO, (2012); and ZHOU et al. (2013)).

Factor analysis (FA) has its development commonly credited by Charles Spearman in 1904 on his work in the psychological field (SPEARMAN, 1904). It is a general scientific method for analyzing data and is one of a family of multivariate methods, as can be seen in the image applications in the works of MALINOWSKI (1978), BENALI et al. (1993), and MØRUP et al. (2006).

Non-Negative Matrix Factorization (NMF) belongs to the group of decomposition dimensionality reduction techniques and has a long history under the name of self-modeling curve resolution in chemometrics, was then introduced as the concept of Positive Matrix Factorization by Paatero and Tapper (PAATERO; TAPPER, 1994), and finally popularized by Lee and Seung (LEE; SEUNG, 1999). NMF tries to build a feasible model for learning object parts by decomposing the original dataset into two smaller matrices, given its non-negative factorization has a high representability of the results. NMF has already reached great success in real world applications such as in face recognition (OKUN, 2004; CHEN et al., 2008), motion segmentation (MO; DRAPER, 2012), Brain images (PADILLA et al., 2012) and multi-focus image fusing (XU et al., 2007).

A set of statistical techniques known as machine learning techniques must be used in order to build the classification model that will perform HAR. These techniques can be applied individually or in an ensemble form where its strength points are combined and together they achieve a better performance than used alone.

The Support Vector Machines (SVM) algorithm (VAPNIK; CORTES, 1995) consists on techniques based on statistical learning and are widely used in pattern recognition as in (GONÇALVES, (2009); JOSÉ; RIBEIRO, (2012); PRADHAN, (2012); BOUZALMAT et al., (2014); and BOARETTO et al., (2017)) .

The k-Nearest Neighbor (kNN) algorithm is a well-known instance-based method in pattern recognition proposed by (COVER; HART, 1967), and until today it has been reaching outstanding performance in many different research fields as in medical imager (MUSTAFA et al., 2012; RAMTEKE; Y, 2012), text classification (TAN, 2006), image categorization (MEJDOUB; AMAR, BEN, 2013), among others.

The Extreme Gradient Boosting (XGBoost) algorithm is a recent improvement of the Friedman's Gradient Boosting Machine (GBM) developed by (CHEN; GUESTRIN, 2016). XGBoost has already proved to be an efficient machine learning tool with several real world applications (HOLLOWAY; MARKS, 2016; BOARETTO; BUSSATO; et al., 2017; GHOSH; PURKAYASTHA, 2017; ZHANG; ZHAN, 2017) and already won several competitions hosted by the site Kaggle (a platform which hosts machine learning competitions, www.kaggle.com).

Based on the studies that developed the first mathematical model of the biological neuron performed by MCCULLOCH and PITTS, (1943), the first ANN was developed by Rosenblatt in 1957 (ROSENBLATT, 1957) with the goal to solve pattern recognition problems. ANN has gained a lot of renown since its first development and has already been applied in the most diverse areas of study (TAM; KIANG, 1992; BURRASCANO et al., 1998; GARDNER; DORLING, 1998; KALOGIROU, 2000).

In a world where information is widely spread and grow exponentially every second, the need to process and understand these huge amounts of data, usually called Big Data, has become a survival need for many tech companies. As quoted by Peter Sondergaard the SVP (Senior Vice President) and Global Head of Research at Gartner Research "Information is the oil of the 21st century, and analytics is the combustion engine", the analysis of Big Data has helped many companies to understand its customers behavior and to identify new buyer patterns, as well in managing strategies as can be seen in MCAFEE et al., (2012). However, in order to process these huge amounts of data the shallows ANN are not efficient and demand too much of computational cost, new techniques with different approaches and more complex architectures have been developed in order to counteract this issue, this new branch of machine learning is known as Deep Learning (DL).

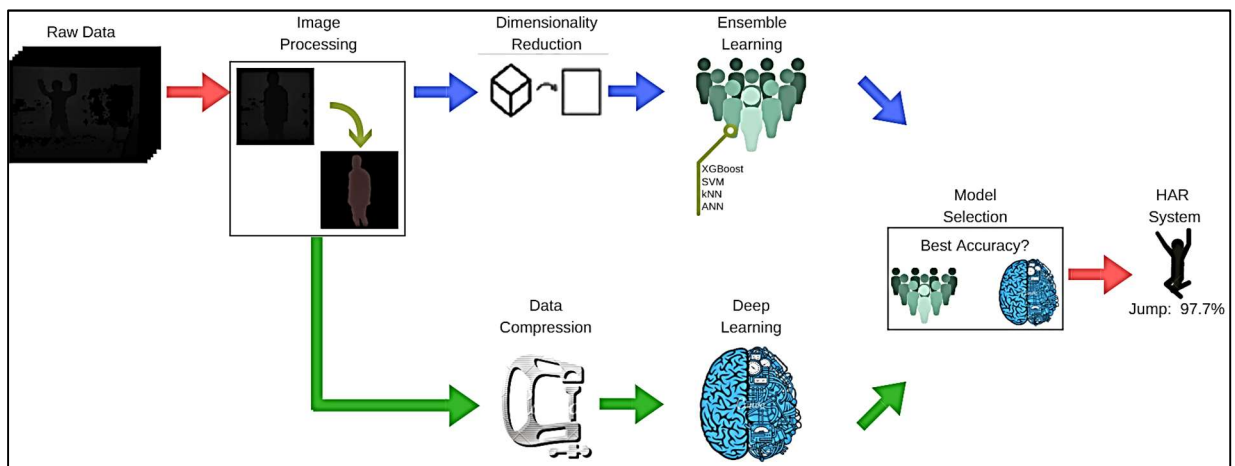
The term DL , in machine learning was introduced by (DECHTER, 1986) and later as referring to ANN used by (AIZENBERG et al., 2000). A brief description of a Deep ANN, consist on a ANN with a larger number of hidden layers, as will be

described in Section 3, where this complex architecture can identify non-linear patterns on the input data, the Deep ANN size will depend on the non-linearity of input data.

A specific group of ANN in DL called Convolutional Neural Networks (CNN) first developed by Lecun in 1998 (LECUN et al., 1998) has been widely used in machine vision applications (KRIZHEVSKY; HINTON, 2012; LECUN et al., 2015; SCHMIDHUBER, 2015; ZHOU et al., 2015; GUO et al., 2016; LINNA et al., 2016), given its powerful hierarchical architecture that mimics the human visual cortex can extract more abstract features of images.

The proposed method in this dissertation consists on comparing two machine learning approaches ensemble learning and DL, in order to develop a HAR system, that will be built based on the MHAD using only depth images acquired with the Kinect sensor, as can be seen in Figure 1.1.

Figure 1.1 - Proposed Methodology



FONT: the author, 2017.

1.1 RELATED WORK

HAR is an important field in computer vision, being used in a series of real-world applications such as gaming, human activity analysis, gait recognition, human posture, human-computer interaction, and sports.

Since the growth of the number of RGB-D human activity image datasets available as in UT Kinect (XIA et al., 2012), MSRDailyActivity3D (WANG et al., 2012), WorkoutSU-10 dataset (NEGIN et al., 2013), UCF Kinect (ELLIS et al., 2013), RGBD-

SAR Dataset (ZHAO et al., 2013) several techniques were developed in order to solve HAR problems.

The works of (TRAN; TRIVEDI, 2012) developed a system based on multi-view inputs for recognize human gesture from upper body pose, the authors obtained 90% of accuracy on the classification tests.

The ideas of (CHOUDHURY; TIAHJADI 2012) consist on a two phased silhouette-based gait recognition by combining Procrustes shape analysis and elliptic Fourier descriptors, these authors were able to get better results than other known gait recognition methods.

In gait-based gender recognition and dimensionality reduction techniques (KASTANIOTIS et al., 2015) used PCA to extract features from depth images acquired with the Kinect sensor and a Gaussian Support Vector Machine (G-SVM) trained with histogram descriptors in order to classify the computed histogram descriptor of the depth images, the authors used a real-time approach where their method got good results independently of the view angle.

In the medical research field, musculoskeletal rehabilitation of the lower limbs with images acquired with the Kinect sensor developed by (TANNOUS et al., 2016), where the authors developed a serious game system in order to improve exercise rehabilitation.

In the sports field, by using 3D laser sensors and human posture analysis (YAMAMOTO et al., 2016) were able to study the effects of drag force and lift force acted on a jumper during a take-off in ski jumping, and concluding that the position of the arms in a very low position strongly influences the flow structure.

On this project, HAR from depth images with the MHAD will be treated as the case of study. Although the MHAD has 11 different actions it is a very complete dataset for HAR, it has three action categories of movement (1) actions with movement in full body parts, (2) actions with high dynamics in upper extremities and (3) actions with high dynamics in lower extremities. The size of the MHAD is sufficient in order to explore the robustness of a machine learning algorithm in order to perform HAR, some datasets as the UTD-MAD (CHEN et al., 2015) has a huge number of actions and subjects that demands too much computational cost and are not trivial to be tested in systems with limited resources, hence the choice of working with a relatively small datasets with diverse actions as the MHAD.

Among the 156 citations of the MHAD in the literature, there are several different approaches to work with HAR with MHAD, mostly of them using Motion Capture (MOCAP) data also called Skeleton data for HAR as in (CHAUDHRY et al., 2013; IJJINA; MOHAN, 2014, 2016), and another works it can be seen the use of combining Kinect + Acceleration data as in the works of (CHEN et al., 2015).

As for this project with the goal to analyze the performance of machine learning techniques with RGB-D videos, will be considered only the Depth data of the Kinect.

Similar methodologies of non-invasive HAR using only depth images from the Kinect sensor of the MHAD with machine learning approaches can be found in the literature as in (BRUN et al., 2014) the authors by using only depth images as data input and the HAcK system managed to achieve 97.7% of classification accuracy. In (ZHANG; PARKER, 2016) the authors achieved a performance of 92.4% of classification accuracy by using the CoDe4D as feature detector, the Adaptive MCOH as descriptor and SVM as classifier. In (CHEN et al., 2015) by only using the Kinect Images with a SVM classifier and Leave-one-out cross-validation the authors obtained a classification accuracy of 92.39%. In (OFLI et al., 2013) the authors reached an accuracy of 91.24% using a Kernel-SVM with Multiple Kernel Learning for classifier. In (CHEEMA et al., 2014) the authors used KNN in order to achieve 77.73% of classification accuracy. In (BRUN et al., 2015) by using HAR based on a string Edit Distance (HARED) achieved 87.1% of classification accuracy.

Some research in the literature also used DL methods in order to get a higher classification accuracy rate as in (FOGGIA et al., 2014) used a Deep Belief Network trained using a Restricted Boltzmann Machine obtained 85.8% accuracy classification. In the works of (ZHU et al., 2016) the authors used Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) called Deep LSTM and achieved 100% of classification accuracy. In (DU et al., 2015) the authors used a Hierarchical Bidirectional Recurrent Neural Network (HBRNN) and also achieved 100% of classification accuracy. In (SHAFAEI; LITTLE, 2016) by using a proposed CNN architecture the authors were able to reach for 98.1% of classification accuracy in Pose estimation.

1.2 OBJECTIVES

The general objective of this work, consist on building HAR system using machine learning techniques applied in a HAR problem comparing two machine learning approaches regarding its classification accuracy, one in an ensemble learning built with stacking and the other in deep learning using a proposed convolutional network architecture, using a dataset of depth images that were acquired with the Kinect sensor.

1.2.1 Specific Objectives

In order to achieve the general objective, the following specific objectives were outlined:

- a) Search on the literature, for efficient machine learning techniques that are used on depth video classification and feature extraction.
- b) Apply Image Processing techniques in order to perform extraction of Region of Interest and noise reduction.
- c) For the ensemble approach
 - Compare different dimensionality reduction techniques in order to achieve best reduction without too much loss.
 - Build a classifier model using different machine learning techniques in order to investigate which one has the best performance.
 - Compare the machine learning techniques individually and in an ensemble form, regarding its accuracy.
- d) For the deep learning approach
 - Given it demands too much computational cost compress the data in order to have a feasible processing time for the test.
 - Test the proposed convolutional neural network architecture in order to build a classifier model.
- e) Compare both machine learning approaches regarding its classification accuracy in order to get the best classifier model for the HAR system.
- f) Perform the integration with the Kinect sensor and the software.
- g) Apply the classifier model in the new images that are acquired with the Kinect sensor.

1.3 STRUCTURE OF THE DISSERTATION

The remainder of this dissertation is organized as follows.

In Chapter 2 a detailed definition of the problem, focusing on the sensor used to perform the data acquisition as well with a detailed presentation of the MHAD dataset is presented.

In Chapter 3, a brief description of machine learning, with the respective techniques and approaches used for developing the experiment is presented.

Chapter 4, on the methodology and the tools used to perform the experiments, described in order of application is demonstrated.

In Chapter 5, the results that were performed in the experiments, with a description of the metric use to measure both approaches performance, a detailed presentation of the results for all the test and a comparison with reference values from the literature is presented.

And finally, in Chapter 6 presents the final considerations with a conclusion of the thesis and suggestion for future works.

2 PROBLEM DEFINITION

This chapter discusses the theoretical concepts on RGB-D images and the Kinect sensor, in order to build a background for understanding the problem and the software and hardware tools used to develop this problem.

For last a description and detailed information about the MHAD dataset.

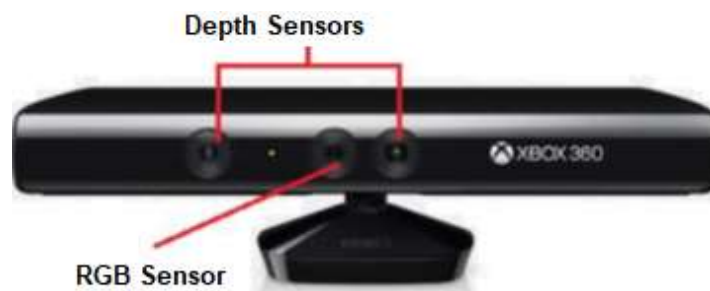
2.1 KINECT SENSOR

The *Microsoft®* Kinect sensor was developed in collaboration with PrimeSense Company and it was introduced in November 4th, 2010, as an accessory to Xbox 360 Console. In January of 2012 more 18 million units were sold (CRUZ et al., 2012).

A huge expectation was created in the computer graphics and computer vision academic communities, since the Kinect promised a new way to interact with games, completely based in gesture and voice (CRUZ et al., 2012).

The Kinect is a great sensor for low resolution RGB-D images acquisition, it captures images at 30 *fps* (frames per second) of both color and depth, in a distance between the range of 500mm and 4000mm. The color images of 32 bits are acquired with a resolution of 1280x960 *pixels*, and the depth images with a resolution of 640x480 *pixels*. The values of the depth images vary inside a range of 0 to 4095, where 0 means that the object is too close to the source and 4095 is too far away, and -1 for undefined depth (HAYAT et al., 2016), more technical data can be found in (CRUZ et al., 2012). As can be seen on Figure 2.1, the Kinect consists of two infra-red sensors (emitter and receiver) for depth and one RGB for color images.

Figure 2.1 – Kinect Sensor.



FONT: the author (2017).

Among the several options of 3D sensors the Kinect stands out in terms of low cost, high speed of acquisition and for having a compact size that is easy to be handled. Figure 2.2 and Table 2.1 show the comparison of technical and visual aspects between the Kinect and the *Minolta* 3D sensor.

Table 2.1 – Comparison Between Kinect and Minolta

Sensor	Speed (seconds)	Size (cm³)	Price (USD)	Precision (mm)
Minolta	2.5	23073	>\$50k	~0.1
Kinect	0.033	680	<\$200	~1.5 - 50

FONT: the author (2017).

Figure 2.2 – Comparison Between Kinect and *Minolta*



FONT: adapted from (LI et al., 2013).

As the image acquired with the Kinect can present some noise given its low resolution acquisition, depending the application a simple smoothing method on the image processing step can correct this flaws (WANG et al., 2016).

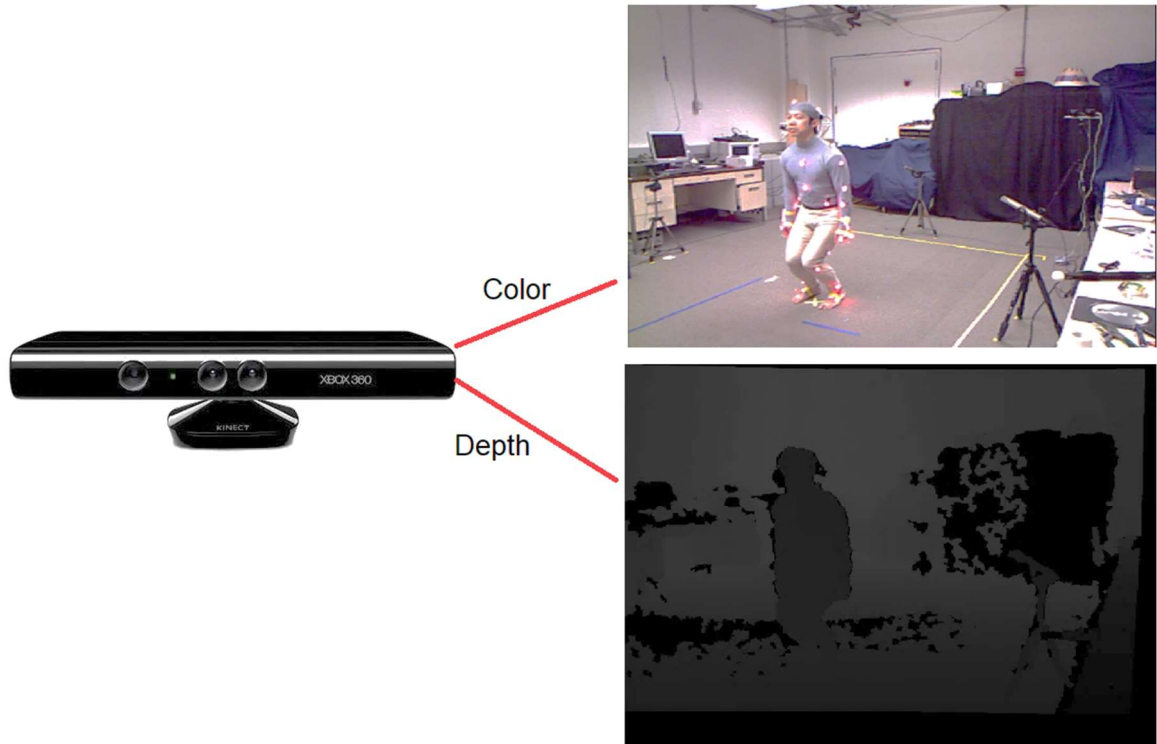
2.1.1 RGB-D

The use of 2D images, has been widely problematic in the fields of pose recognition and gait analysis, due to its severities regarding illumination, object occlusion, angle and pose ambiguity (CHATTOPADHYAY et al., 2014).

The resultant image acquired by the Kinect is called the RGB-D image, which consists on the combination of the three-color channels (red, green and blue) with an

additional depth channel. Each color channel represents a matrix with the values of 8 bits and for the depth channel a matrix with values of 16 bits, as shown in Figure 2.3.

Figure 2.3 – Color and Depth Image from Kinect.



FONT: the author (2017).

2.2 MHAD

The MHAD dataset developed in 2013 from University of California Tele immersion Lab and the Center for Imaging Science, Johns Hopkins University (OFLI et al., 2013), and publicly available, contains 11 actions performed by 7 male and 5 female subjects in the range of 23-30 years of age except for one elderly subject. All the subjects performed 5 repetitions of each action, yielding about 660 action sequences which correspond to about 82 minutes of total recording time.

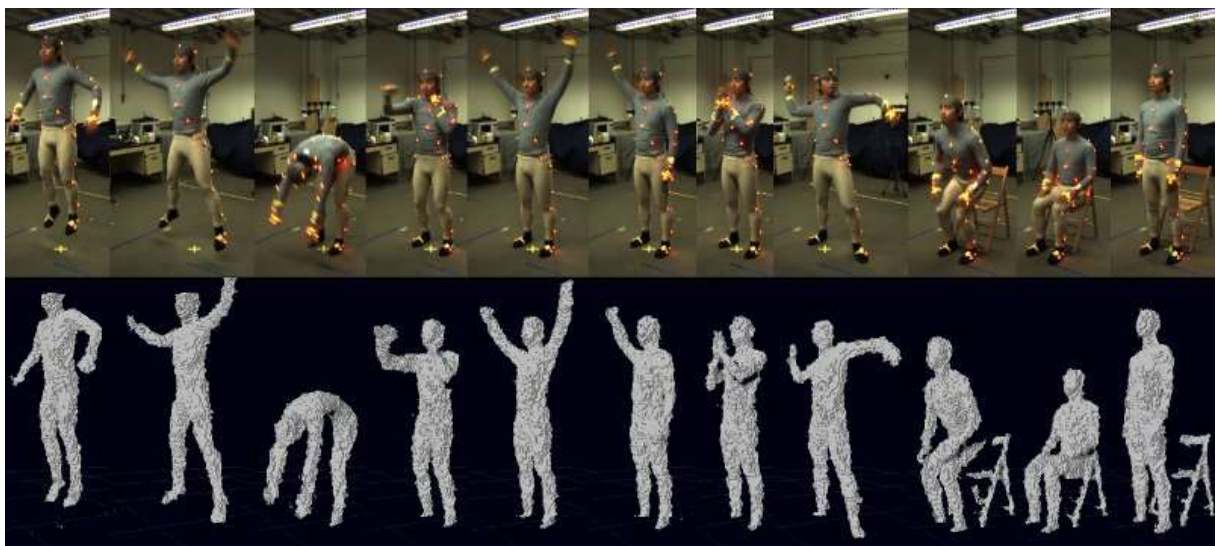
The MHAD can be used for different applications such as HAR, pose estimation, motion segmentation and dynamic 3D scene reconstruction.

The specified set of actions comprises of the following: (i) actions with movement in both upper and lower extremities, e.g., jumping in place, jumping jacks, throwing, among others, (ii) actions with high dynamics in upper extremities, e.g., waving hands, clapping hands, among others, and (iii) actions with high dynamics in

lower extremities, e.g., sit down, stand up. The subjects have incorporated different styles in performing some of the actions (e.g., punching, throwing) (OFLI et al., 2013).

The 11 actions are: jumping in place (jump), jumping jacks (jack), bending- hands up all the way down (bend), punching (punch), waving two hands (wave2), waving right hand (wave1), clapping hands (clap), throwing a ball (throw), sit down and stand up (sit +stand), sit down (sit), stand up (stand), all actions from the color images and depth images as depicted by Figure 2.4.

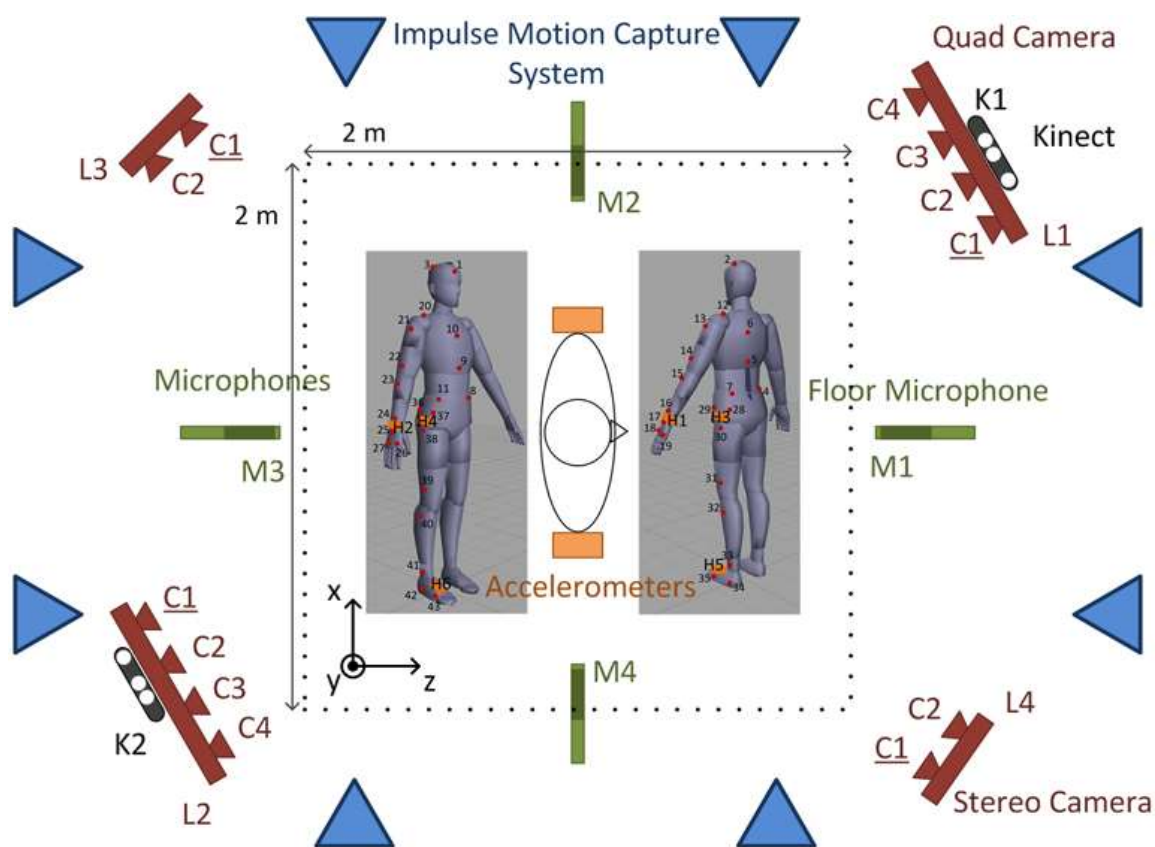
Figure 2.4 - Actions Performed By One Subject With Both Color and Depth Images.



FONT: adapted from (OFLI et al., 2013).

There are five sensor modalities in the Berkeley MHAD, optical motion capture system, four multi-view stereo vision camera arrays (L1 through L4, with cameras C1 through C4), two Microsoft Kinect cameras (K1 and K2), six wireless accelerometers (H1 through H6) and four microphones (M1 through M4), the display of the sensors positions shown in Figure 2.5.

Figure 2.5 - Display of The Sensors from The MHAD.



FONT: adapted from (OFLI et al., 2013).

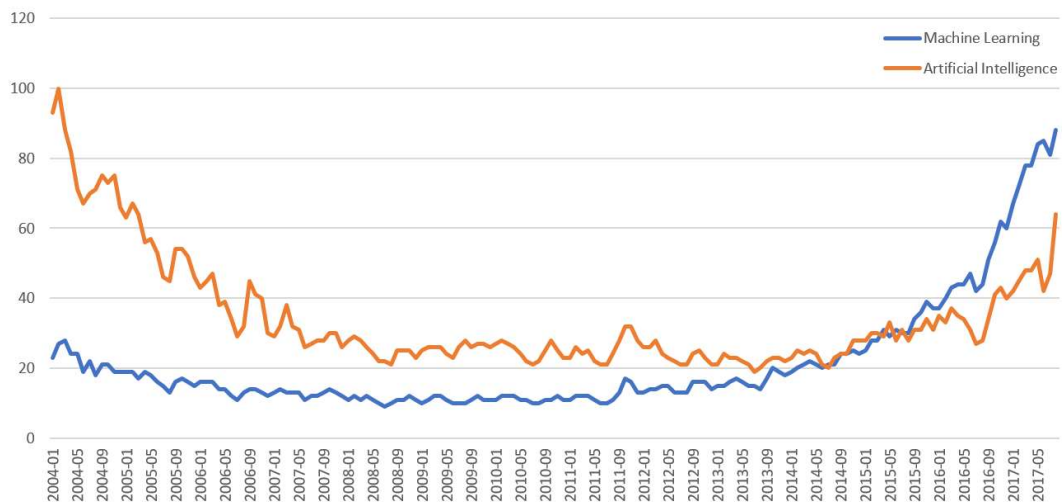
3 FUNDAMENTALS OF MACHINE LEARNING

Machine Learning in general consists of instead programming a machine to perform a task T , the computer as proposed by (MITCHELL, 1997) learn from experience E with respect to some class of tasks T and performance P , if its performance at tasks T , as measured by P , improves with experience E .

The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience (MITCHELL, 1997).

With the constant growth on data storage capacity and computational power that our civilization has been facing on the past few years the interest in machine learning has been growing a lot recently, as illustrated by Figure 3.1 a graphic from Google trends where the terms machine learning and artificial intelligence are analyzed.

Figure 3.1 - Google Trends for Machine Learning Performed in 05/06/2017.



FONT: the author, 2017.

Although it seems to be a futuristic concept, machine learning has its roots in the early 50's, when in 1959 Arthur Samuel wrote the first computer learning program, which was an IBM computer and the program was the game of checkers that improves the more it played, learning winning strategies and adding them in the program. Arthur Samuel also coined the term "Machine Learning" while on IBM in 1959 (SAMUEL, 1959).

Machine learning has many applications as in the medical field (YOO et al., 2014; DEO, 2015; MILJKOVIC et al., 2016; ROST et al., 2016), economics (GAN, 2013; WUEST et al., 2014; ATHEY; IMBENS, 2015), geology (KLUMP et al., 2014; KORUP; STOLLE, 2014), data mining (REBENTROST et al., 2014; AL-JARRAH et al., 2015; LANDSET et al., 2015).

The machine learning algorithms are divided in four groups, supervised learning, unsupervised learning, semi-supervised learning and reinforced learning.

In supervised learning both inputs and outputs are known for the algorithm, that try to learn a function that correctly describes the relation between the target prediction output and the input features, such that the generated model can generalize the predictions of the output values to new data based on those relationships which it learned from the previous data sets.

Different from supervised learning, in unsupervised are trained with unlabeled data, because only the inputs are known. Unsupervised learning tries to use techniques on the input data to mine for rules, detect patterns, summarize and group the data points which help in deriving meaningful insights and describe the data better to the users.

Semi-supervised can be seen as a combination of both supervised and unsupervised learning, where the algorithms work with either labeled and unlabeled data. When the process of obtaining labeled data is unfeasible due to either high computation cost or lack of human expertise, semi-supervised learning algorithm comes quite in hand, these methods takes advantage of the idea that even though there are unlabeled data in the dataset, this data carries important information about the input features. Semi-supervised learning is known to present higher accuracy with less effort (ZHU, 2005).

Reinforced Learning, a type of learning that continuously learns from the environment in an iterative fashion. This method focusses on using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk, depending on the application.

Supervised Learning is used in this project, since all the inputs and outputs are known and the models are generated based on this information.

This chapter will focus on the machine learning techniques that were used on this project, since dimensionality reduction, ensemble learning and classifier model selection.

3.1 DIMENSIONALITY REDUCTION

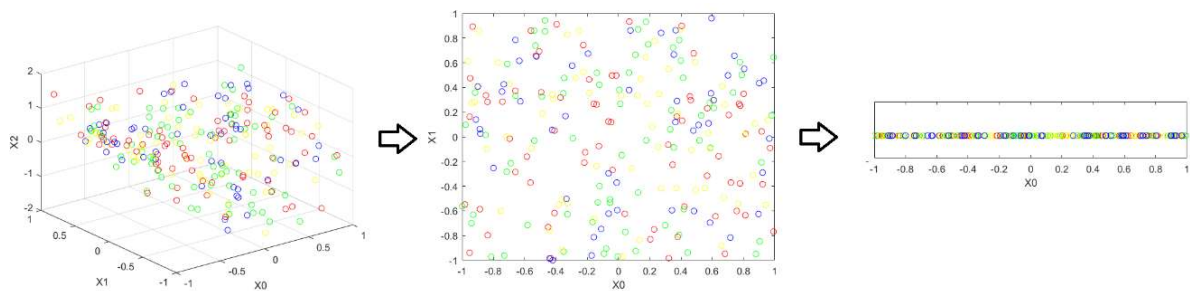
In many applications of machine learning, usually the datasets used in the learning process are huge, especially when working with image or video, which can take a lot of time and computational effort or even can't be analyzed depending the resources of the machine in case.

As Bellman coined in (BELLMAN, 1961) the term *curse of dimensionality*, which states the relation between adding more variables to a problem and the exponential increase of the dimensionality of the mathematical space, hence increasing the complexity of the problem. Given the complexity of a model $O(nd^2)$, where n is the number of samples and d is the dimension, as d increases the complexity O becomes too costly.

This phenomena is also known as *Hughes effect*, as stated by Hughes in (HUGHES, 1968) in machine learning for classification specifically “the use of too many variables in a classification procedure may decrease classification accuracy”.

So not only the computational cost increases but also the dimensionality affects directly in the classification performance, one way to counteract this matter is to use dimensionality reduction techniques as shown in Figure 3.2.

Figure 3.2 - Dimensionality Reduction



FONT: the author, 2017.

Dimensionality reduction techniques not only help to reduce the computational cost but also reduce the probability of overfitting. Dimensionality reduction can be divided in two categories, feature extraction (not-supervised) and feature selection (supervised). On feature selection a subset of the original dataset is selected, while in feature extraction a new set of features is generated from the original dataset. Feature extraction involves a transformation of the features, which often is not reversible

because some information is lost in the process of dimensionality reduction. Examples of feature selection are Pearson's correlation and ANOVA, examples for feature extraction are PCA and FA, as will be seeing in this Chapter.

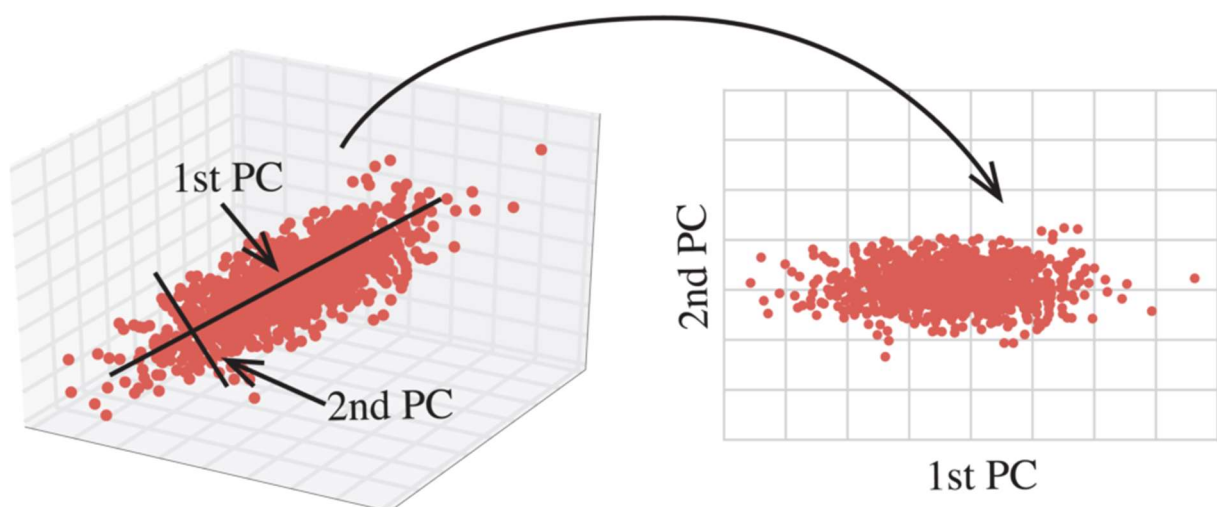
3.1.1 PCA

PCA is a remarkable technique for dimensionality reduction proposed by Pearson in 1901 (PEARSON, 1901), in order to find the lines and planes that fits better in a set of points in space. Later in the 1930 PCA was developed by Hotelling (HOTELLING, 1933) in order to find a small set of variables that could represent a bigger set of variables, he called this small set as principal components.

As quoted by (RICHARDSON, 2009) "PCA is the general name for a technique which uses sophisticated underlying mathematical principles to transforms a number of possibly correlated variables into a smaller number of variables called principal components (PC). One of PCA advantages is to work on reducing the dimension of the data without losing too much information (BOUZALMAT et al., 2014).

PCA consists on an orthogonal linear transformation of the data Figure 3.3, where the general idea is to find the eigenvalues and eigenvectors of the covariance matrix of the dataset.

Figure 3.3 - PCA Method from 3D to 2D



FONT: <http://blog.kaggle.com/2017/04/10/exploring-the-structure-of-high-dimensional-data-with-hypertools-in-kaggle-kernels/>

Given a dataset $X \leftarrow D^{m \times n}$, where m is the number of vectors x^n (observations) with n components (variables), n also defines the dimensionality of the dataset. Firstly, the PCA algorithm subtract the average μ^m (eq. 3.1) of each element of the vector x^n in X given by

$$\mu^m = \frac{\sum_{i=1}^n x_i}{n} \quad (3.1)$$

Then the covariance matrix Σ (eq. 3.2) is calculated for X , where the covariance can be thought of as a measure of how much two variables change together (RICHARDSON, 2009). Regarding the covariance signal, if outcomes a positive value it means that both observed variables grow together, if negative both observations grow in different directions, if zero both observations are independent of each other. In this context,

$$\Sigma = \begin{pmatrix} cov(x, x) & cov(x, y) & cov(x, z) \\ cov(y, x) & cov(y, y) & cov(y, z) \\ cov(z, x) & cov(z, y) & cov(z, z) \end{pmatrix} \quad (3.2)$$

where x, y and z correspond to three different observations and $cov()$ means the covariance between two observations as in

$$cov(x, y) = \frac{\sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)}{n - 1} \quad (3.3)$$

Of the covariance matrix Σ generated by the previous step on the PCA algorithm, is then calculated the eigenvalues λ (eq. 3.4) and eigenvectors e (eq. 3.5), the eigenvectors define the new space.

$$\lambda = \det(\Sigma - \lambda I) \quad (3.4)$$

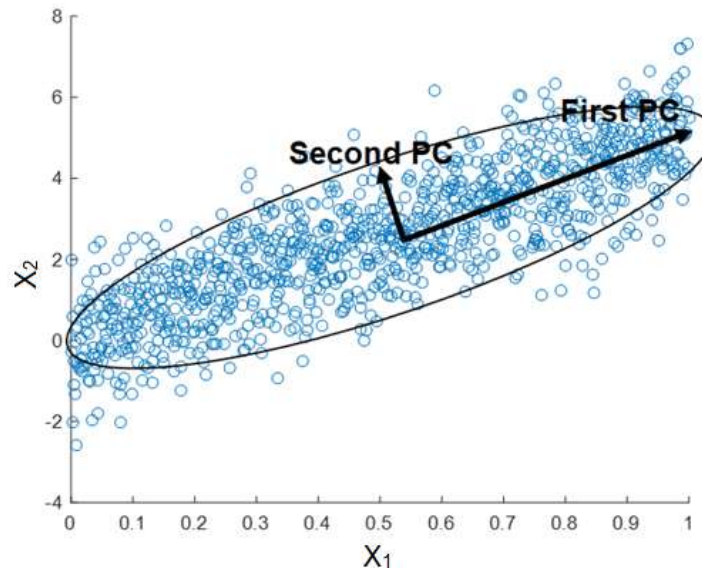
where I is the identity matrix given by

$$\Sigma e = \lambda e \quad (3.5)$$

The eigenvalues represent the largest possible degree of correlation between all variables with the principal axis, hence higher the eigenvalue higher the correlation. The principal components are sorted according with the eigenvalues,

where the i th principal component is equal to the i th eigenvalue divided by the sum of all eigenvalues presented in Figure 3.4.

Figure 3.4 - Principal Components



Font: the author 2017.

In Figure 3.5 can be seen the PCA pseudocode.

Figure 3.5 - PCA Pseudocode

PCA pseudocode

- 1) Given a dataset $X \leftarrow D^{m \times n}$
- 2) subtract the average μ^m (eq. 1)
- 3) calculate the covariance matrix Σ (eq. 2)
- 4) calculate the eigenvalues λ (eq. 4) and eigenvectors e (eq. 5)
- 5) select PC

FONT: author, 2017.

In order to reduce the dimension to a feasible number of components that represents the original dataset without losing too much information, there are a few methods that can be used. One simple way to solve is by a scree plot, first the proportion of variance of each component is calculated and then plotted in a descending order on a graphic, then is analyzed the n components that has the higher

proportion and that the sum of this n components' variance proportion do not result below 90%. Another method is to use only the components whose eigenvalues is above the average. Although both methods seem simple and practical, sometimes valuable information is lost leading to a bad representation of the original dataset that could result in overfitting, as shown in (WOLD et al., 1987; ABDI; WILLIAMS, 2010) the authors suggest more complex methods can serve as a better criteria for this problem using Cross-Validation and *Boosting*.

3.1.2 FA

Factor analysis (FA), has its development commonly credited by Charles Spearman in 1904 on his work in the psychological field (SPEARMAN, 1904). In (SPEARMAN, 1927) was developed the two-factor model and later extended to a multiple factor model by THURSTONE (1935).

As cited by DECOSTER; HALL (1998) is a collection of methods used to examine how underlying constructs influence the responses on a number of measured variables.

The concept of FA consist on that the variables can be grouped by their correlations, it may be assumed that variables within a particular group are highly correlated among themselves, but they have relatively small correlations with variables in a different group (KHOSLA, 2004). By grouping all variables according to its correlations, the number of groups formed are the number of factors.

One key on learning FA is to first understand the concept factors. Factors as formal concept, are coordinates defining the boundaries of the space and within it the location and magnitude of all vectors, also defined as symbolic terms within a mathematical function linking vectors and parameters to mathematical rules for their combination. Factor as a theoretical concept, measures the inner process of "black-box" which observed inputs are transformed into observed outputs, factor define the causal nexus underlying the observed patterns. The third concept is factor as an empirical concept, which defines factor as characteristics that classify phenomena according to their inter-relationships (RUMMEL, 1988).

The value of a factor analysis is dependent on the meaningfulness of the variability in the data, hence if the data has lower or none variability than a lower number of factors will be derived from the data (RUMMEL, 1988).

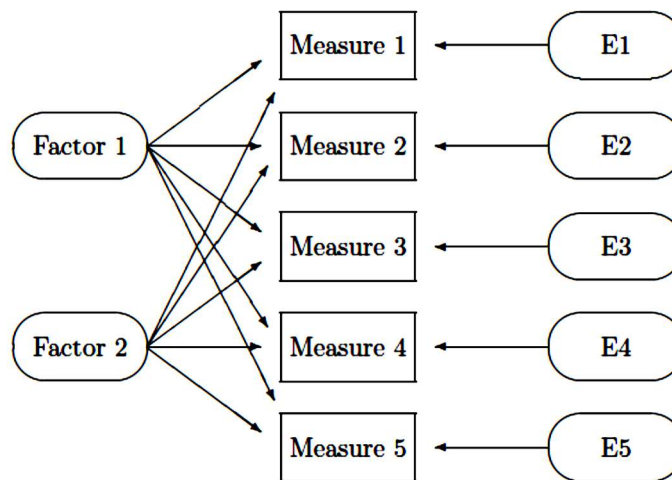
FA can be divided into two categories Confirmatory FA (CFA) and, Exploratory FA (EFA).

CFA, the objective is to determine the ability of a predefined factor model to fit an observed set of data (DECOSTER; HALL, 1998), It is more difficult to perform CFA than EFA, given the strong links to structural equation modeling (DECOSTER; HALL, 1998).

EFA is concerned on finding the number of common factors influencing a set of measures and the strength of correlation between each factor and each observed measure (DECOSTER; HALL, 1998). EFA is often confused with PCA.

The common factor model, as can be seen in Figure 3.6, represents that each observed response (Measure 1 through 5) is influenced partially by underlying common factors (Factors 1 and 2) and partially by unique factors (E1 through E5). The strength of the link between each factor and each measure varies, such that a given factor influences some measures more than others (DECOSTER; HALL, 1998).

Figure 3.6 - Common Factor Model



FONT: adapted from (DECOSTER; HALL, 1998).

Factor analysis are performed by examining the pattern of correlations (or covariances) between the observed measures. Measures that are highly correlated (either positively or negatively) are likely influenced by the same factors, while those that are relatively uncorrelated are likely influenced by different factors (DECOSTER; HALL, 1998).

The common factor model, is a partial correlation approach to the data, it determines the minimum number of independent coordinate axes (dimension) necessary to reproduce the variation in vectors in the space (RUMMEL, 1988).

The FA model is defined, as cited by (PAISLEY; CARIN, 2009) of modeling a data matrix $X \in \mathbb{R}^{N \times D}$, where N and D represents respectively the number of samples and the dimension of the data matrix, as the multiplication, (eq. 3.6), of two matrix $\phi \in \mathbb{R}^{K \times D}$ and $Z \in \mathbb{R}^{N \times K}$, where K is the dimension of factors.

$$X = Z\phi + E \quad (3.6)$$

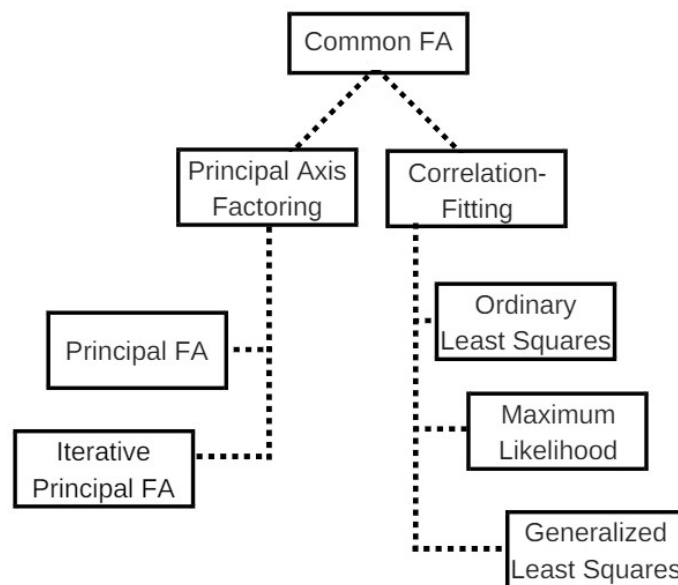
where E is the error matrix also called specific factors.

The steps of the FA algorithm, starts similar with the PCA algorithm, until the extraction step, while PCA uses the eigenvalues and eigenvectors approach, FA generally uses the Maximum Likelihood Estimation (MLE) method or the Principal Factor (PF) method.

3.1.2.1 Extraction Methods

There are several factor extraction methods applied in FA, as can be seen in Figure 3.7, however this topic will focus on the most commonly used methods PF and MLE.

Figure 3.7 - FA Extraction Methods



FONT: the author, 2017.

a) MLE

The MLE method was popularized by R. A. Fisher between the years of 1912 and 1922 which introduced the term likelihood in 1921 and later the method name in 1922 (HALD, 1999), is an approach for parameter estimation in classical statistics, it views the parameters θ as quantities whose values are fixed but unknown, the best estimate of the value is defined to be the one that maximizes the probability (likelihood) of obtaining the samples actually observed (KHOSLA, 2004), the basic idea of likelihood function $L(\theta)$ method can be seen in eq. 3.7.

$$L(\theta) = f(x_1; \theta) \cdot f(x_2; \theta) \dots f(x_n; \theta) = \prod_{i=1}^n f(x_i | \theta) \quad (3.7)$$

where x_n is a random sample with n samples for which the probability density function of each sample x_i is $f(x_i; \theta)$.

MLE is a simple method that has good convergence properties as the number of training samples increase, the estimate has the smallest variance and are usually consistent and unbiased. However, drawback of MLE is that the correct probability distribution for the problem must be known.

b) PF

The PF method, also called principal axis method, resemble the principal component analysis approach, is better able to recover weak factors and that the maximum likelihood estimator is asymptotically efficient, as in PCA can analyze not only correlations but also covariances.

As cited in (FABRIGAR et al., 1999) if data are relatively normally distributed, maximum likelihood is the best choice because “it allows for the computation of a wide range of indexes of the goodness of fit of the model and permits statistical significance testing of factor loadings and correlations among factors and the computation of confidence intervals.”

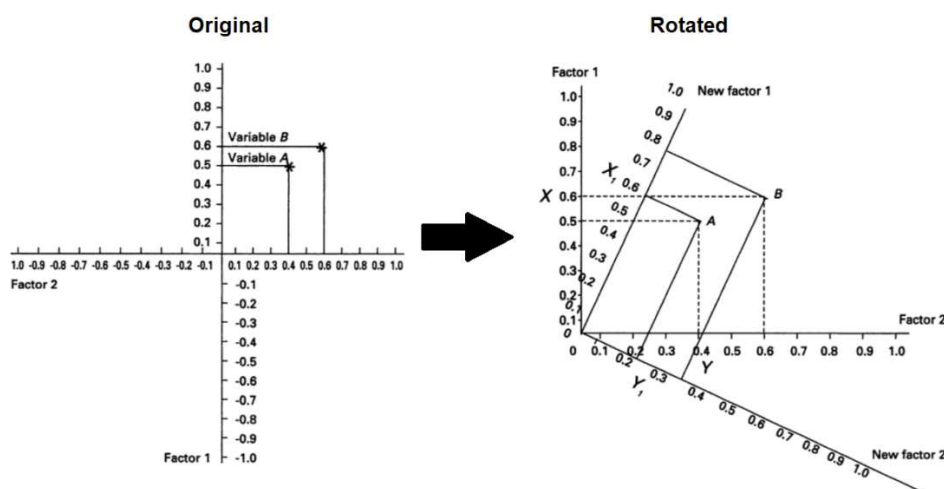
3.1.2.2 Factors Rotation

Sometimes the first solution produced by FA does not reveal the hypothesized structure of the factor loadings, one way to solve this issue is to apply a rotation

strategy to find another set of factor loadings, that can similarly fit the observations well and also can be easily interpreted.

Although rotating the factors (Figure 3.8) can change the factor loadings and thus the meaning of the factors, the new rotated factors can reproduce precisely the original correlations of the unrotated factors, because they are mathematically equivalent they can explain the same amount of variance in each variable of the original unrotated factors (KLINE, 1994).

Figure 3.8 – Factors Rotation



FONT: the author, 2017.

One common and widely used rotation strategy is the varimax method proposed by Kaiser in 1958 (KAISER, 1958), this method gives us the smaller of variables which high factor loadings for each factor and this helps us to interpret the component in a clear way (EDITOR IJSMI, 2017). The goal is to make some of these loadings as large as possible and the rest as small as possible in absolute value. The varimax method encourages the detection of factors each of which is related to few variables, it discourages the detection of factors influencing all variables.

3.1.2.3 Factors Score

Factor scores in an important step to perform further analysis of the identified factors, one easy method for scoring the factors, is to add together the scores on the variables which load most highly on the factor.

However more complex methods can also be applied in order to get a better evaluation of the factors estimates, as the Regression method or Weighted Least Squares (WLS).

a) Regression

The regression method or exact factor score methods, are mostly used when the MLE method is used to extract the factors, use the estimated parameters from a factor analysis to define linear combinations of observed variables that generate factor scores (EDITOR IJSMI, 2017).

b) WLS

Bartlett's WLS method can be used to estimate factor scores if multivariate normality assumption is valid. Here original variables are considered as dependent variable and factors are treated as independent variable and factor scores are the unknown coefficients.

Figure 3.9, shows the FA pseudocode.

Figure 3.9 - FA Pseudocode

<p>FA pseudocode</p> <ol style="list-style-type: none"> 1) Given a dataset $X \leftarrow D^{m \times n}$ 2) Calculate the covariance matrix Σ 3) Select the number of factors f 4) Extract factors (MLE or PF) 5) Rotate factors (varimax rotation) 6) Interpret the factors 7) Factor Scores (Regression or WLS)

FONT: the author, 2017

3.1.3 PCA vs. FA

As cited above "FA is often confused with PCA", so it's important to highlight some differences between both algorithms.

Both PCA and FA resemble in:

- the fact that both can be applied in dimensionality reduction by capturing the variance of the variables in a smaller set;

- both outputs look very similar;

Although, they vary in many ways as can be seen in Table 3.1 as stated in (KHOSLA, 2004).

Table 3.1 – Comparison Between PCA and FA.

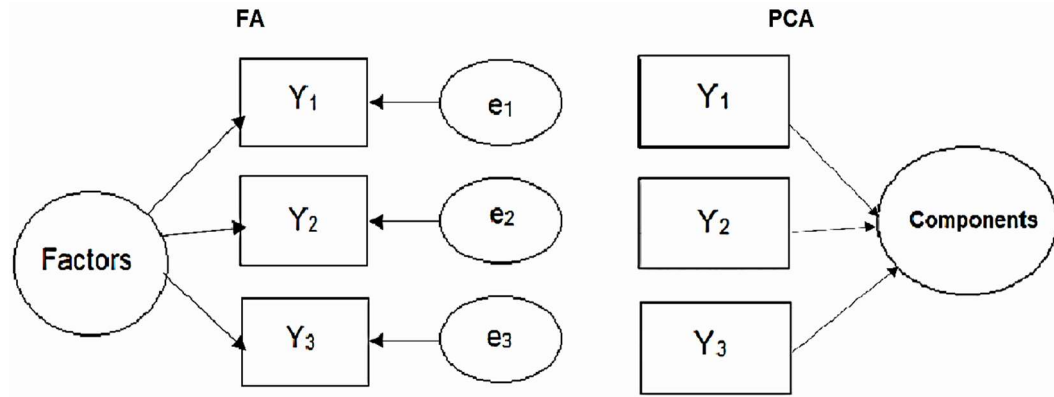
PCA	FA
<ul style="list-style-type: none"> • more of a dimensionality reduction technique. 	<ul style="list-style-type: none"> • more of a latent variable technique.
<ul style="list-style-type: none"> • decomposes the total variance and in the case of standardized variables, it produces a decomposition of correlation matrix. 	<ul style="list-style-type: none"> • analyzes the decomposition of the reduced correlation matrix and the diagonal matrix of the unique variances associated with the variables.
<ul style="list-style-type: none"> • Can be expressed as linear functions of the variables or the variables can be expressed as linear functions of the principal components. 	<ul style="list-style-type: none"> • Concentrates on defining the variables as a linear combination of common factors and unique factors.
<ul style="list-style-type: none"> • Emphasis in expressing the principal components as a linear function of the variable set x. 	<ul style="list-style-type: none"> • Emphasis on explaining the variable set x as a linear function of unobservable common factors.
<ul style="list-style-type: none"> • Regarding error, PCA will be linear composites of unreliable variables, and will contain measurement error. 	<ul style="list-style-type: none"> • Common factor is uncontaminated by measurement error because measurement error is part of the unique variance.
<ul style="list-style-type: none"> • the principal components are based on the measured responses. 	<ul style="list-style-type: none"> • assumes that the measured responses are based on the underlying factors.

FONT: the author, 2017.

In summary, PCA has advantage over FA when there is no underlying model of the data in mind. Although FA is better when working with a dataset of variables that

contain measurement error, Figure 3.10 shows the main difference between PCA and FA in a graphical way.

Figure 3.10 - FA Vs. PCA



FONT: the author, 2017.

3.1.4 Non-Negative Matrix Factorization

As pointed in the previous topics, PCA and FA share a respective set of properties, one in particular, that was not mentioned previously, is that there is no constraint in the sign of the elements in the resultant matrices, hence, the negative component or the subtractive combination is allowed in the representation matrix resulting in a lower representability of the results (WANG, 2013).

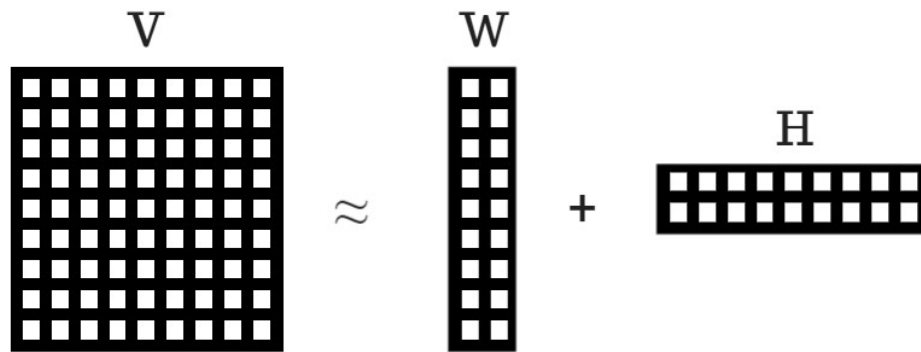
In the group of decomposition methods, NMF as its name says, constrains the values of the factorization matrix to only non-negative elements, which leads to a part-based representation because they allow only additive combinations (OKUN, 2004).

Given a non-negative matrix $\mathbf{V}^{m \times n}$ (m corresponds to the number of observations in a dataset and n number of features of each observation in m , as can be seen in Figure 3.11), the NMF algorithm decomposes the original $\mathbf{V}^{n \times m}$ matrix in two smaller matrices $\mathbf{W}^{n \times r}$ and $\mathbf{H}^{r \times m}$ such in (eq. 3.8).

$$\mathbf{V}^{n \times m} \approx \mathbf{W}^{n \times r} \mathbf{H}^{r \times m} \quad (3.8)$$

where r corresponds to the number of elements that will be selected for the dimensionality reduction following the rule $r < \frac{nm}{n+m}$.

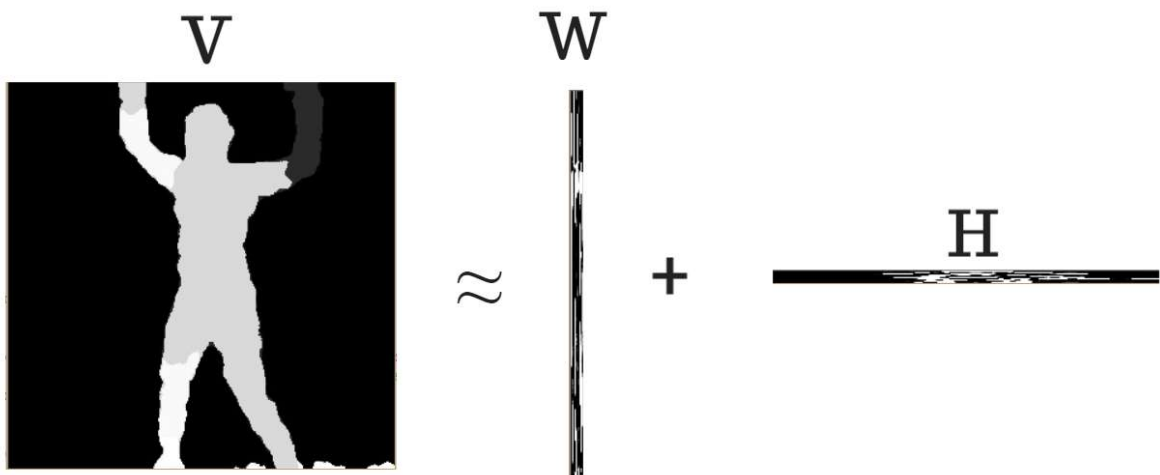
Figure 3.11 – NMF Decomposition



FONT: the author, 2017.

W and H correspond respectively to the basis and weight matrices. The learnt features from of each column in the original matrix V are stored in the columns of the basis matrix W with a corresponding reduced representation of the same column in V in a column in the weight matrix H (Figure 3.12), in the words of (LEE; SEUNG, 2001), V remains of being a linear representation of W and H , and W can be regarded as containing a basis that is optimized for the linear approximation of the data in V .

Figure 3.12 – NMF Decomposition with Image



FONT: the author, 2017.

In a way to measure the quality of the approximation $V \approx WH$ a cost function F must be defined, a good and useful way to do this is to simply use the square of the Euclidean distance between the matrix V and WH as in eq. 3.9.

$$F = \sum_{i=1}^n \sum_{j=1}^m [V_{ij} - (WH)_{ij}]^2 \quad (3.9)$$

By using the cost function F , the value of F describes the likelihood of the original images in V and the generated images from combining W and H .

In order to reach the local minima of the cost function, a certain procedure that consists on that first W and H are initialized with random non-negative values, then they must be updated simultaneously iteratively as described by eq. 3.10 and 3.12, until a stop criterion is met (usually a pre-defined number of iterations or a value of F).

$$W_{ia} = W_{ia} \sum_j \frac{V_{ij}}{(WH)_{ij}} H_{aj} \quad (3.10)$$

$$W_{ia} = \frac{W_{ia}}{\sum_j W_{aj}} \quad (3.11)$$

$$H_{aj} = H_{aj} \sum_i \frac{V_{ij}}{(WH)_{ij}} W_{ai} \quad (3.12)$$

Not restricted to the cost function described in eq. 3.9, the NMF algorithm allows other different cost functions that could present better results depending of the respective application. However, choosing the right cost-function could be a tricky task, as seen in (QUINTANILHA, 2016) a descriptive table with different kinds of application that provide useful information when choosing the right cost function is presented in Table 3.2.

Table 3.2 – NMF Cost Functions

Data Type	Distribution	$f(W,H)$	Example
Real	Gaussian	Frobenius	Images
Integer	Multinomial	KL	Word Counter
Integer	Poisson	Generalized KL	Photons Counter
Non-Negative	Multiplicative Gamma	Itakura-Saito	Spectral Data
Non-Negative	Tweedie	β -divergent	Generalization of the models above

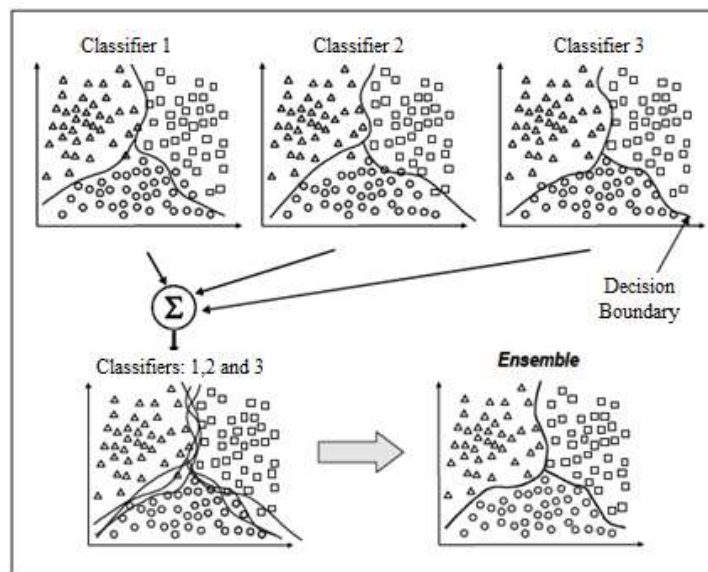
FONT: adapted from (QUINTANILHA, 2016).

3.2 ENSEMBLE LEARNING

Ensembles, consists on a group of machine learning techniques that seeks to aggregate knowledge gathered by the models that compose it, aiming to reach a global solution that results in a more efficient model than its components applied alone (WILFREDO; VILLANUEVA, 2006). In other words, an ensemble, also called a multiple classifier or committee, is a set of individual component classifiers whose predictions are combined to predict new incoming instances.

The application of ensembles seeks to improve the generalization capacity using the advantages of each component on solving the same problem (WANG et al., 2012; KANG et al., 2015; CHOI et al., 2016; LOCHTER et al., 2016; REN et al., 2016; XU et al., 2016), which makes it more preferable to choose diversified models that have more distinct characteristics as shown in Figure 3.13.

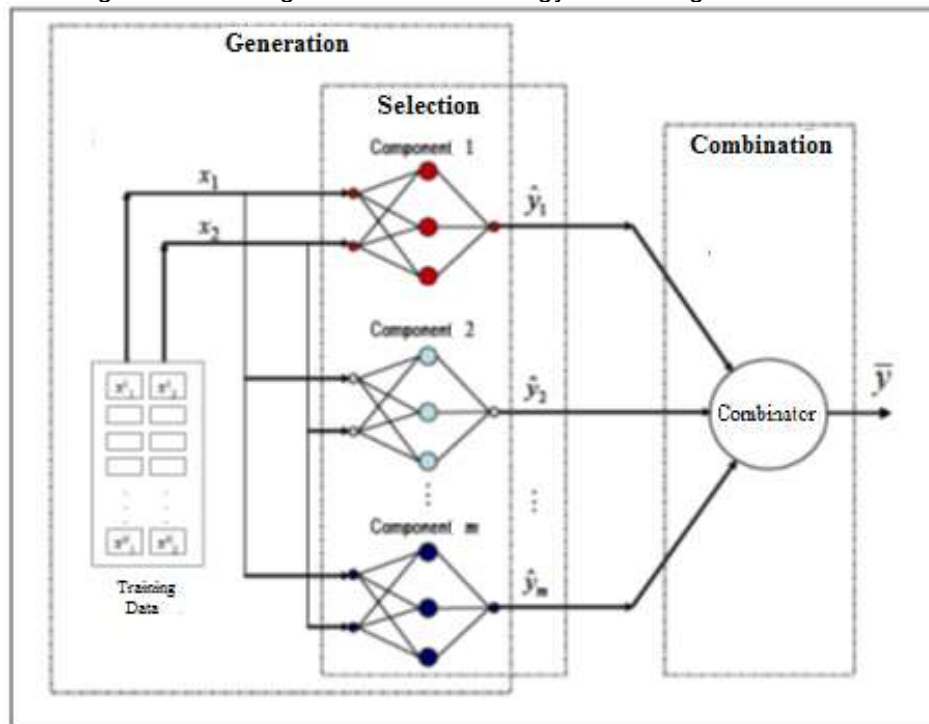
Figure 3.13 – Example of Classification Using Ensembles



Font: adapted from (WILFREDO; VILLANUEVA, 2006)

The ensemble techniques are composed by at least three components, and most of the cases adopt a three-step methodology, training (generation), selection and combination (LIMA, 2004), as illustrated in Figure 3.14.

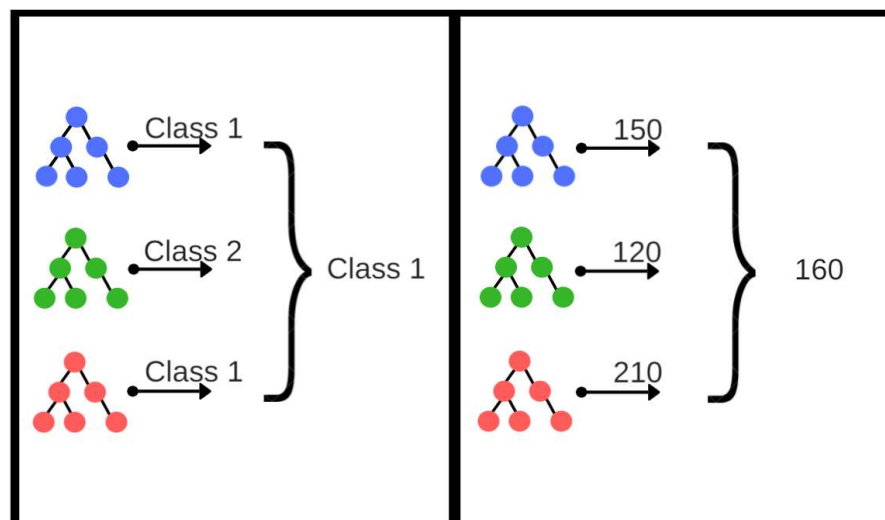
Figure 3.14 – Stages of The Methodology of Creating an Ensemble.



Font:, adapted from (WILFREDO; VILLANUEVA, 2006).

On the training step, the ensemble models are generated. The combination step, the combination method will differ depending on what kind of problem it would be applied. In a classification problem, it can be used a voting technique, Figure 3.15a, but in case of a regression problem it is generally used an average of the resulting outcomes of each component, Figure 3.15b.

Figure 3.15 – Application of an Ensemble, in Classification (a) and in Regression (b).



FONT: the author, 2017.

On the selection step, the components that have the best performance are selected. When using a higher number of components, it is possible that not every component contributes for the global performance of the ensemble, hence refining technique are recommended to prune the ensemble according to a selection criterion which could be an error measure over a subsample of data (WILFREDO; VILLANUEVA, 2006).

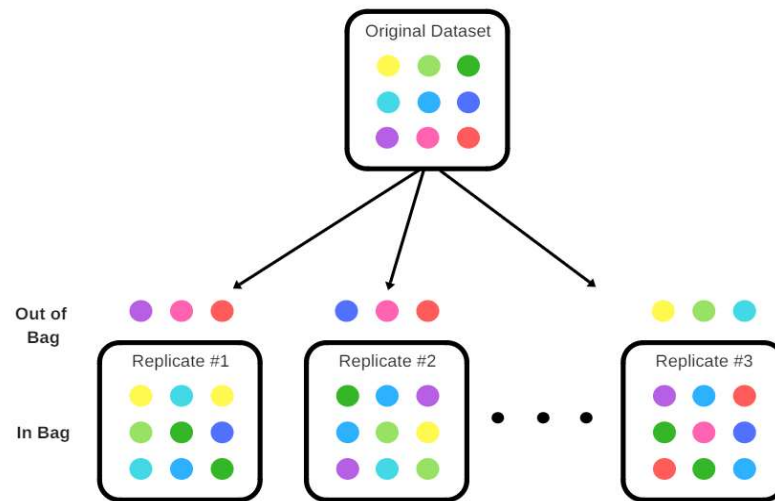
Whenever dealing with a highly complex real-world problem, the application of an ensemble learning by itself could not be sufficient to deal with tricky patterns and non-linearities of the feature space. In that case is recommended to use one of the meta-algorithms in order to decrease the variance (Bootstrap Aggregating), bias (Boosting) or improving the predictive force (Stacking Generalization).

3.2.1 Bootstrap Aggregating

The Bootstrap Aggregating (Bagging) algorithm developed by Breiman in 1996 (BREIMAN, 1996) votes classifiers generated by different bootstrap samples. A bootstrap sample is generated by uniformly sampling m instances from the training set with replacement, Figure 3.16. N bootstrap samples B_1, B_2, \dots, B_N are generated and a classifier C_i is built from each bootstrap sample B_i . A final classifier C_i is built from C_1, C_2, \dots, C_N whose output is selected by majority vote from the output classes of the sub-classifiers (BAUER et al., 1999). In Bagging approximately 63% of observations from the sample occurs at least once and the remaining observations are called out-of-bag (CUTLER et al., 2007).

As (BREIMAN, 1996) point out, that improvements will occur with the bagging algorithm for unstable procedures (like artificial neural networks, linear regression and regression trees) where a small change in the dataset can result in a large change in the predictor model.

Figure 3.16 – Bagging



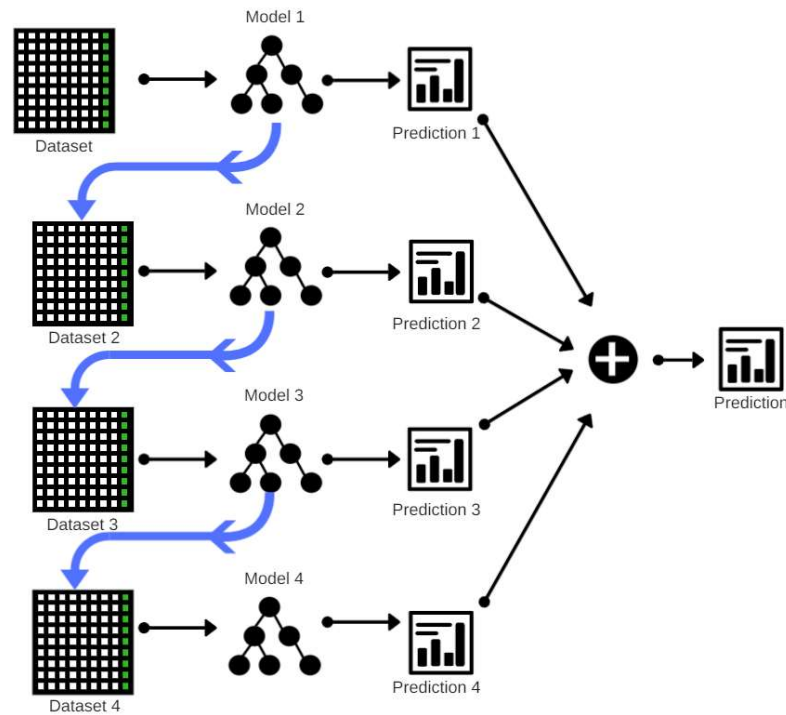
FONT: from (IZMIRLIAN, 2004).

3.2.2 Boosting

Boosting is a general and provably effective method to improve the accuracy of any learning algorithm for both classification and regression applications (SCHAPIRE, 1999). Unlike the parallel fitting of the base models of the bagging algorithm, boosting build models sequentially (XIA et al., 2017).

Boosting was based on the questions of (KEARNS; VALIANT, 1988, 1994) “can a set of weak learners create a single strong one?”. The first boosting algorithm was originally proposed by (SCHAPIRE, 1990).

Figure 3.17 – Boosting



FONT: from author, 2017.

The boosting algorithm, Figure 3.17, consists on a combination of weak mathematical models that are built iteratively each one is being trained with different subsets of the original dataset without substitution (ABNEY et al., 1999), it uses weights to each training set, setting a higher weight to poorly predicted examples so the probability of this example to be chosen in the next subset is higher. At the end of the process all of the models are weighted according to their score and then a final model is created by combined using voting

The major difference between bagging and boosting methods is that the boosting method strategically resamples the training data to provide the most useful information for each consecutive model. The adjusted distribution during each step of training is based on the error produced by the previous models. Unlike the bagging method where each sample is uniformly selected to produce a training dataset, the probability of selecting an individual sample is not equal for the boosting algorithm. Samples that are misclassified or incorrectly estimated have more chances to be selected with higher weight. Therefore, each newly created model places emphasis on the samples that have been misclassified by previous models (ZHANG; HAGHANI, 2015).

3.2.3 Stacking Generalization

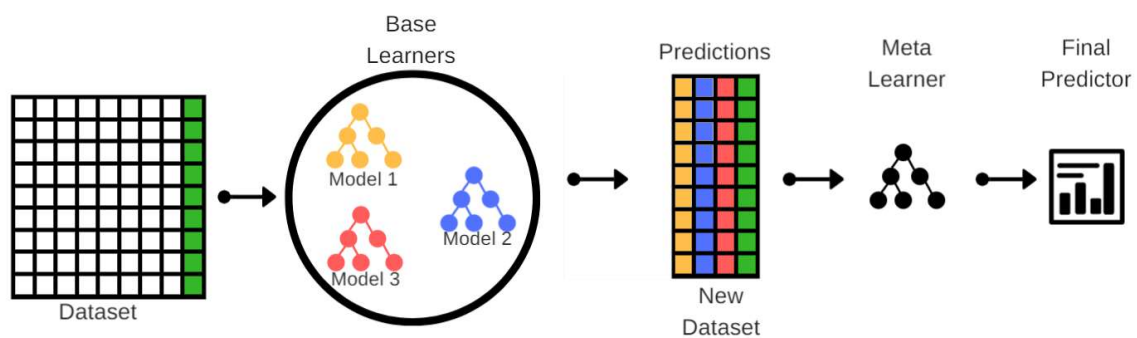
Stacking Generalization also called Stacking, introduced by Wolpert in 1992 (WOLPERT, 1992). Stacking has already won several competitions from the site Kaggle including the Netflix 1M (one million) prize (<http://netflixprize.com>) with an implementation of a Feature-Weighted Linear Stacking (SILL et al., 2009).

Different from bagging and boosting, stacking use the concept of meta learner instead of voting algorithms to combine the base learners, since when using voting it is not clear which learner to trust (WITTEN et al., 2017).

First a set of base learners is used to learn part of the dataset that is left for training, then these same base learners make predictions on the other part of the dataset that is left to testing. A higher-level learner, called meta learner is trained using the predictions from the previous step as input, the process can be seen in Figure 3.18.

Once the base learners are built, they do the predictions in an unseen part of the original dataset, hence their predictions are unbiased, therefore the data that will be trained in the meta learner reflects the true performance of the base learners (WITTEN et al., 2017)

Figure 3.18 – Stacking



FONT: the author, 2017.

In Table 3.3, a comparison between the three meta-algorithms is presented.

Table 3.3 – Comparison Between Bagging, Boosting and Stacking

Property	Bagging	Boosting	Stacking
Partitioning of the data into subsets	Random	Giving misclassified samples higher preference	Various
Goal to achieve	Minimize variance	Increase predictive force	Minimize variance and Increase predictive force
Methods where this is using	Random subspace	Gradient descent	Blending
Function to combine single models	Weighted average	Weighted majority vote	Logistic regression

FONT: the author, 2017.

3.3 SUPPORT VECTOR MACHINES

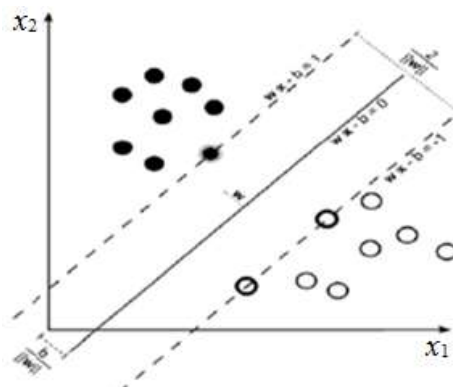
SVM is a technique based on statistical learning developed by Vapnik in 1995 (Vapnik, 1995), aiming to solve pattern classification problems. Since then it has been widely used in either classification (BURGES, 1998; GONÇALVES, 2009; JOSÉ; RIBEIRO, 2012; BOUZALMAT et al., 2014) and regression problems (Camps-Valls et al., 2006; Dutta, Pal, & Sen, 2016; Ghaedi et al., 2016).

SVM is considered a technique easier to be applied than a neural network (BOUZALMAT et al., 2014). SVM in a classification problem, which presents linearly separable characteristics, performs a class separation with a hyperplane (eq. 3.13) positioned in a way that the distance (Euclidean distance) between the hyperplane and the classes are the largest as possible (GONÇALVES, 2009), Figure 3.19.

$$\mathbf{w}^T \mathbf{x} = 0 \quad (3.13)$$

where \mathbf{w} and \mathbf{x} are vectors, the vector \mathbf{w} will always be normal to the hyperplane because is the vector that will define the hyperplane.

Figure 3.19 - SVM Classification Hyperplane



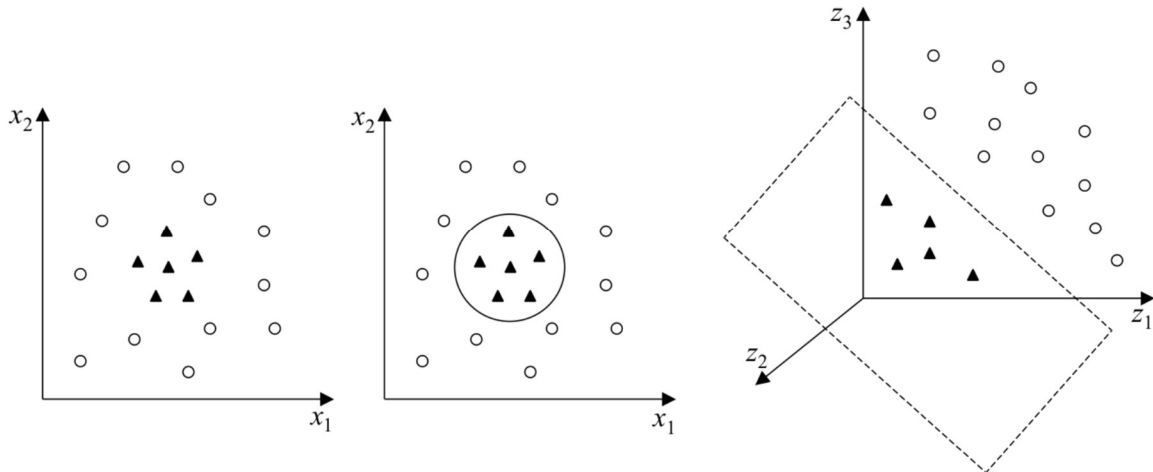
FONT: adapted from (Bouzalmat et al., 2014).

In order to the distance between the hyperplane and the margin m is the largest as possible, the m must be maximized (eq. 3.14), hence the $\|w\|$ must be minimized.

$$m = \frac{2}{\|w\|} \quad (3.14)$$

In case of a non-linearly separation of the characteristics, by its complexity is not trivial to perform a linear hyperplane application. Hence, according to the Cover's theorem (Cover, 1965), in which a non-linear problem by having its dimensionality increased it has a better chance to become linearly separable, Figure 3.20, Kernel functions are used to map the characteristics which makes the algorithm more efficient (GONÇALVES, 2009). Some of the basic Kernel functions used in SVM are linear, polynomial, sigmoid and radial basis (BOUZALMAT et al., 2014), as can be seen in Table 3.4.

Figure 3.20 - Cover Theorem



FONT: adapted from (Lorena & de Carvalho, 2007).

The new mapped feature space is separated with a hyperplane as well, as seen in eq. 3.15.

$$w^T \Phi(x) = 0 \quad (3.15)$$

where points of $\Phi(x)$ that satisfy the condition $w^T \Phi(x) = 1$ are called support vectors.

Table 3.4 - Kernel Functions

Kernel Functions	
Linear	$K(x_i, x_j) = x_i^T \times x_j$
Polynomial	$K(x_i, x_j) = (\gamma x_i^T \times x_j + r)^d, \gamma > 0$
Radial-Basis Function	$K(x_i, x_j) = \exp(-\gamma \ x_i - x_j\ ^2), \gamma > 0$
Sigmoid	$K(x_i, x_j) = \tanh(\gamma x_i^T \times x_j + r), \gamma > 0$

FONT: the author, 2017.

where x_i and x_j represent the 2D vectors, γ is the kernel parameter, d is the degree of polynomial function, and r is coefficient of interception.

The best function to be chosen when working with SVM is the one that outcomes the lowest empirical risk $Remp(f)$ (eq. 3.16), which is the difference between the expected output y and the produced output $f(x)$, for a given input x .

$$Remp(f) = \frac{1}{m} \sum_{i=1}^m L(f(x_i), y_i) \quad (3.16)$$

where m is the number of observations in the set, and $L(.)$ represents the cost function chosen as in eq. 3.17 also known as loss function, where

$$L(f(x), y) = \begin{cases} 0 & \text{if } y = f(x) \\ 1 & \text{if } y \neq f(x) \end{cases} \quad (3.17)$$

The SVM is a very robust technique on working with a high dimension of data, presents a convexity of the optimization problem that is formulated on its training process which implies on a existence of a single global minima (LORENA; CARVALHO, 2007), despite of its advantages the SVM algorithm has a higher computational cost compared with other techniques regarding the classification problem and the model generated by the technique have a lower interpretability (LORENA; CARVALHO, 2007).

3.3.1 Multi-Class SVM

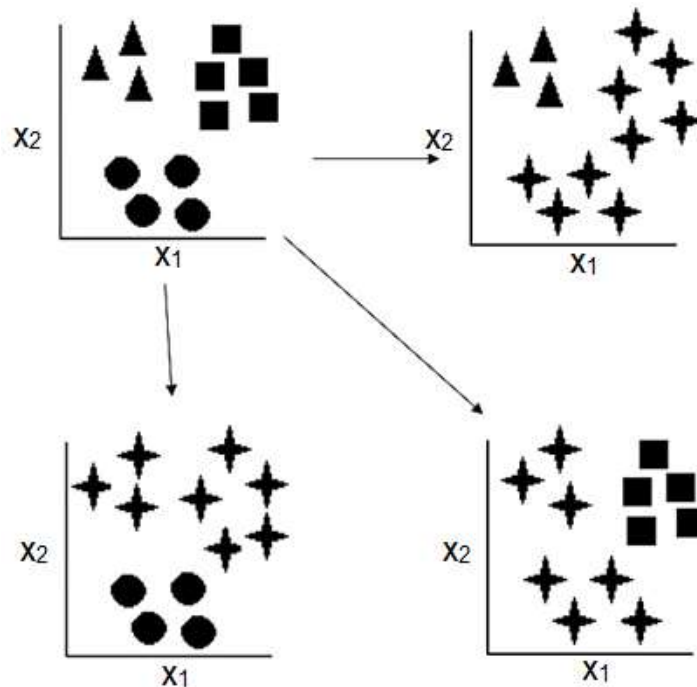
SVM not only have a good performance in binary classification but also in multi-class classification. Though, the approach for dealing with a higher number of classes are different with the use of decomposition methods as in the one-vs-all and all-vs-all.

3.3.1.1 One-vs-all

The one-vs-all approach, builds the number of k binary classifiers, k is the number of classes. Then each f_i classifier handles each i class. As can be seen in eq. 3.18, given a new observation x , the class which this new observation belongs is the class represented by the classifier f_i that has the maximum value between the k classifiers (GONÇALVES, 2009), as shown in Figure 3.21.

$$f(x) = \arg \max_{1 \leq i \leq k} (f_i(x)) \quad (3.18)$$

Figure 3.21 - One-vs-All



FONT: the author, 2017.

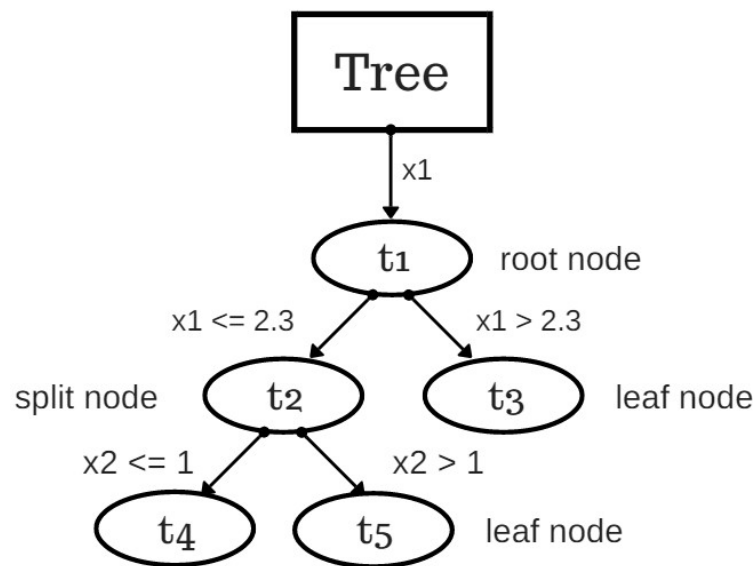
3.3.1.2 All-vs-all

The all-vs-all approach, consists on comparing classes in pairs being needed $\frac{k \times (k-1)}{2}$ SVMs, where k is the number of classes. In order to attribute a class to a new observation x , instead of using a maximization function as in one-vs-all approach, all-vs-all uses majority vote to classify the new observation (GONÇALVES, 2009). So, each one of the SVMs outcomes a result for the new observation, and the class in which has the majority number of votes is the one attributed to the new observation.

3.4 EXTREME GRADIENT BOOSTING

The decision tree is a non-linear and non-parametric supervised classification algorithm where the nonterminal nodes indicate the features and terminal nodes are outcomes, as shown in Figure 3.22.

Figure 3.22 – Decision Tree



FONT: the author, 2017.

The training of decision trees for both classification and regression problems, as defined by the term CART (Classification And Regression Trees) introduced by Breinman in 2001 (BREIMAN, 2001) starts with the root node, and binarily divides the nodes into branches until it reaches the leaves, where the nodes represents the test over a feature, the branch (split node) represents the value of the results from the test and the leaf node represent the class.

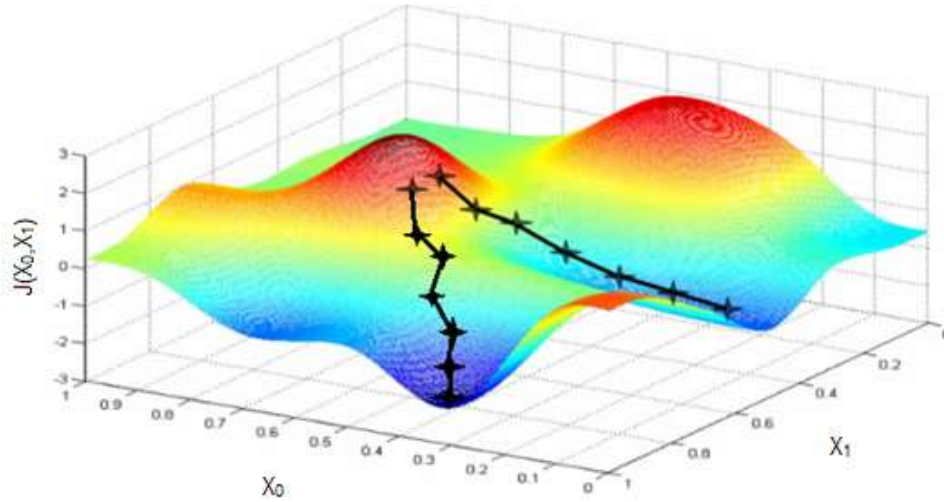
3.4.1 Gradient Boosting Machine

Gradient boosting machine (GBM) produces a competitive, highly robust, interpretable procedures for both classification and regression (FRIEDMAN, 2001). In other words, GBM is an ensemble of boosted regression trees, regression trees differs from decision trees (Figure 3.22) on the fact that regression trees contains a continuous score on each of the leaf (CHEN; GUESTRIN, 2016).

GBM uses a gradient-descent method (

Figure 3.23) to build a tree which decrease the objective on the direction of the gradient, in consequence data-points which are hard to classify gain influence during the training (KECK, 2016).

Figure 3.23 – Gradient Descent Method in a Surface



FONT: the author, 2017.

GBM works with the idea of additive training, by combining several weak learners to develop a strong learner (Urraca et al., 2017). Given a dataset $(x_i, y_i)_{i=1}^N$, where N is the number of samples, x_i is the i th set of features and y_i corresponds to its respective response variable. To determine the optimal model f , GBM calculates the optimal base learner f_i given by eq. 3.19, by transforming f_i in a parametrized function $\rho h(x, \theta)$, where ρ given by eq. 3.20 and θ represents respectively the best scale coefficient. The best parameters in the base learner $h(x, \theta)$ where,

$$f_i = f_{i-1} + \rho_i h(x_i, \theta_i) \quad (3.19)$$

GBM uses the gradient-descent method to build a tree that decreases the objective value towards the gradient, assuring that the loss-function L will decrease on each iteration, where

$$\rho_i = \operatorname{argmin}_\rho \sum_{j=1}^N L[y_j, f_{j-1}(x_j) + \rho h(x_j, \theta_j)] \quad (3.20)$$

Figure 3.24 shows a representation of the GBM pseudocode.

Figure 3.24 – GBM Pseudocode

GBM Pseudocode	
Input:	
•	dataset $(x_i, y_i)_{i=1}^N$
•	maximum number of iterations M
•	loss-Function $L(y, f)$
•	base-learner model $h(x, \theta)$
Training:	
Initialize a tree f_0	
for $i \leftarrow 1$ to M do	
i)	Construct a new tree model $h(x, \theta_i)$
ii)	Find the best gradient descent step ρ_i (3.20)
iii)	Update the function f_i (3.19)
end for	
Output:	
•	Best f

FONT: the author, 2017.

3.4.2 XGBoost Algorithm

Extreme Gradient Boosting (XGBoost) is new implementation of the GBM algorithm, as GBM's additive learning nature have a higher risk of overfitting, XGBoost aims on preventing overfitting without jeopardizing the computation efficiency of the algorithm (Urraca et al., 2017).

XGBoost algorithm have a high scalability due to its several important systems and algorithmic optimizations and also are faster than other methods due to its parallel and distributed computing (CHEN; GUESTRIN, 2016). According to (ALER et al., 2017) XGBoost automatically takes advantage of the available computes processor cores, CPUs (Central Processing Unit) and random access memory (RAM).

Similar to GBM, XGBoost also uses an ensemble of i functions f_i to create a strong optimized model f . As can be seen in (XIA et al., 2017) XGBoost uses CART form of base-learner $\omega q(x)$, $q \in \{1, 2, \dots, T\}$, where T denotes the number of leaves, q the decision rules of the tree and ω the leaf weight of each node of the tree.

In XGBoost a regularization term $\Omega(f)$ is added to the loss function given by

$$\Omega(f) = \gamma T + 0.5 * \lambda \|\omega\|^2 \quad (3.21)$$

where, γ is the complexity parameter, λ is a fixed coefficient and $\|\omega\|^2$ is the L2 norm of leaf weights.

The core problem of XGBoost as said by (XIA et al., 2017) is to determine the optimal tree structure, in order to tackle this problem the algorithm uses a greedy search

algorithm in finding an optimal tree structure called exact greedy algorithm for splitting finding (CHEN; GUESTRIN, 2016).

Since the XGBoost algorithm is controlled by various hyperparameters, a proper selection of their values is important toward obtaining accurate results. To find the best hyper-parameters in each case, an exhaustive search over a grid of values is carried out and the best performing case is retained (ALER et al., 2017).

The most common hyper-parameters of the XGBoost are the **maximum depth** of the tree which range from zero to infinity and by increasing its value will make the model more complex. The **learning rate**, which is the step size shrinkage used in update to prevents overfitting, the learning rate actually shrinks the feature weights to make the boosting process more conservative and varies from 0 to 1. The **number of estimators** which consists on the number of boosted trees for the XGBoost algorithm and it's an integer number varying from 1 to infinity, by increasing this value it can influence in the processing time of the algorithm as well with its complexity.

3.5 K-NEAREST NEIGHBORS

The conventional kNN algorithm simply uses the k training samples (observations) that are closest to (nearest neighbors) the test sample according to a distance metric to classify it. It is called a lazy learning algorithm because the generalization occurs only when a new observation beyond the training data needs to be classified. The kNN can be seen as a nonparametric classification technique based on an empirical Bayes decision rule that can achieve high classification accuracy in problems that have unknown and non-normal distribution.

Its learning process consists on saving all the training instances with its class labels, to classify an unknown instance the classifier ranks the instance's neighbors among the training instances and use the class label of k most similar neighbors to predict the class of the new instance (TAN, 2006).

The distance between the new instance and the k nearest neighbors is measured (eq. 3.22) throughout all dataset and the closest neighbors indicates the class for the new instance, as shown in Figure 3.25, where k is a parameter set in the beginning of the algorithm that represents the number of neighbors that will be compared. The value of k is preferable to be an odd number since an even number of neighbors can result in a tie decision.

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \dots + (x_n - x'_n)^2} \quad (3.22)$$

where d is the distance to be calculated between the two vectors x and x' , and n is the number of elements of the vectors.

Then the algorithm estimates the conditional probability P (eq. 3.23) for each class, that is, the fraction of points in the set of k closest neighbors with that given class label.

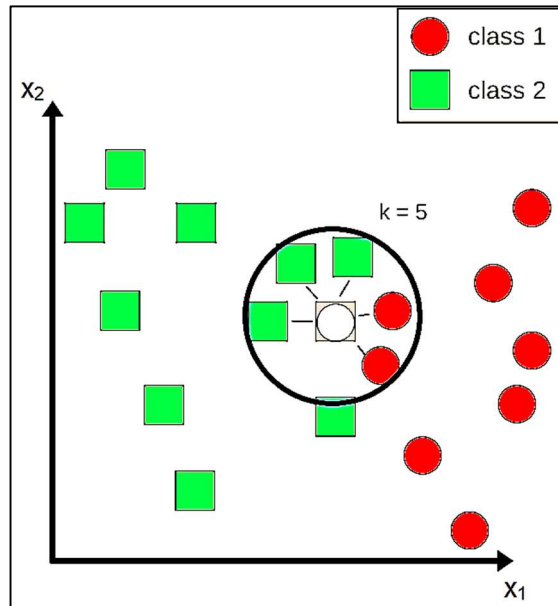
$$P(y = j|X = x) = \frac{1}{k} \sum_{i \in \mathcal{A}} I(y^{(i)} = j) \quad (3.23)$$

where, x is the input, y is the output, \mathcal{A} is the set of closest neighbors, I is the function which evaluates j to 1 if when the argument x is true and 0 otherwise.

Different distance functions have been adopted in kNN design such as Euclidean, Hamming, Manhattan, Tanimoto, Jaccard, Mahalanobis, cosine, and Minkowski distance.

In this paper, the Euclidean distance measured is used to distance metric to the adopted case study of fire incident classification.

Figure 3.25 –Distance Comparison Related to k Value in The kNN Algorithm



FONT: the author, 2017.

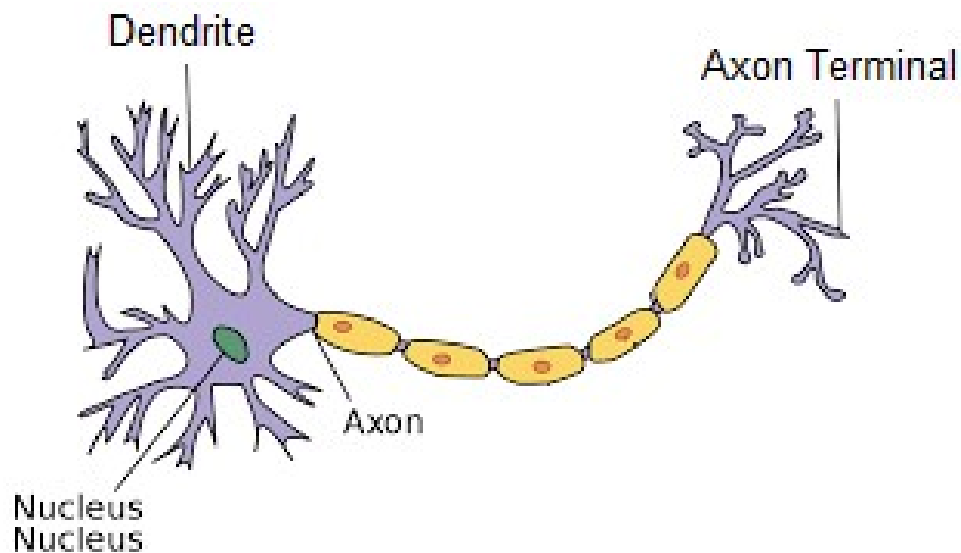
The kNN algorithm and its variants have been widely used in the literature to solve real problems. For example, the kNN classifier has been applied to feature selection (PARK; KIM, 2015) and dimensionality reduction (Ingram & Munzner, 2015).

Although KNN is a simple method, the large amount of design vectors are required in the classifiers results in a high computational cost ((HWANG; WEN, 1998)).

3.6 ARTIFICIAL NEURAL NETWORK

The average human brain has approximately 100 billion neurons and each of these neurons has from 1000 to 10000 connections with neighbor neurons. The neuron body consist of its nucleus, dendrites, axons and its terminals, as can be seen in Figure 3.26. The dendrites are responsible for receive the synapsis from other neurons and carry this information to the nucleus. The nucleus processes this synapsis and send a new synapsis to other neurons through the axon and its terminals (BERESFORD, 2000).

Figure 3.26 – Biological Neuron

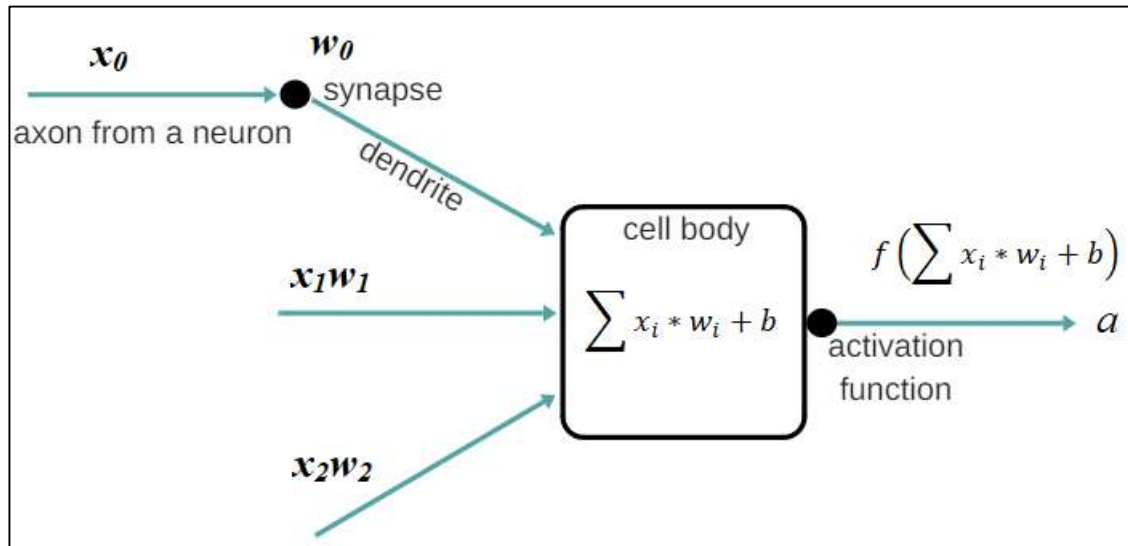


FONT: the author, 2017.

ANNs are digitalized representations of the human brain, with ability to learn from training to recognize patterns and perform other task as in classification and regression.

The mathematical representation of the neuron is seen in Figure 3.27.

Figure 3.27 – Mathematical Representation of a Neuron



FONT: the author, 2017.

The mathematical operation named activation function that represents the nucleus activity to process the synapsis is given by f in eq. 3.24.

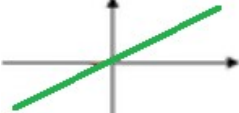
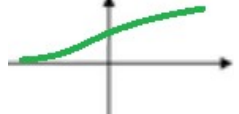

$$a = f\left(\sum x_i * w_i + b\right) \quad (3.24)$$

where x is an input column vector that represents the synapsis of a neighbor neuron i , w is the weight associated with each input in order to express the importance of this input to the neuron, b is a bias value associated to the neuron. The activation function f processes the information that come from the input and its objective is to generate a new synapse to the next neuron through the output a .

Depending on the application of the ANN the activation function can differ, in Table 3.5 is presented the most common activation functions of the ANN algorithm.

Table 3.5 – List of Activation Functions

Activation function	$f(x)$	Output Range	Curve Shape
---------------------	--------	--------------	-------------

Linear	x	$a = x$	
Sigmoid or Logistic	$\frac{1}{1 + e^{-x}}$	$0 < a < 1$	
Tanh	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$-1 < a < 1$	

FONT: the author, 2017.

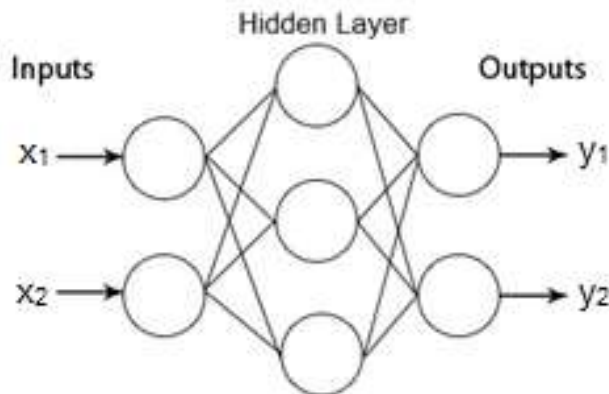
There are also some special activation functions like the softmax function (eq. 3.25). The softmax function is a generalization of the logistic function and is often used in multiclass classification applications, its output y ranges from 0 to 1 predicting the probability P of a specific class $j \in K$ from the dataset x with weights w .

$$P(y = j|x) = \frac{e^{x^T w_j}}{\sum_{k=1}^K e^{x^T w_k}} \quad (3.25)$$

There are several ANN architectures on the literature as the Feedforward network (TAHMASEBI; AMIRKABIR, 2011), Feedback network (ZAMARREN; GONZA, 2005), RNN (ROJAS, 1996) and self-organizing maps. (KOHONEN, 1990).

The Multi-Layer Perceptron (MLP), consists of layers of connected neurons that has one or more hidden layers on its architecture, Figure 3.28. The hidden layer consists of the transformation where the input data is projected into a linearly separable space, the most common transformation functions are *tanh* and *sigmoid* functions.

Figure 3.28 – MLP Structure



FONT: the author, 2017.

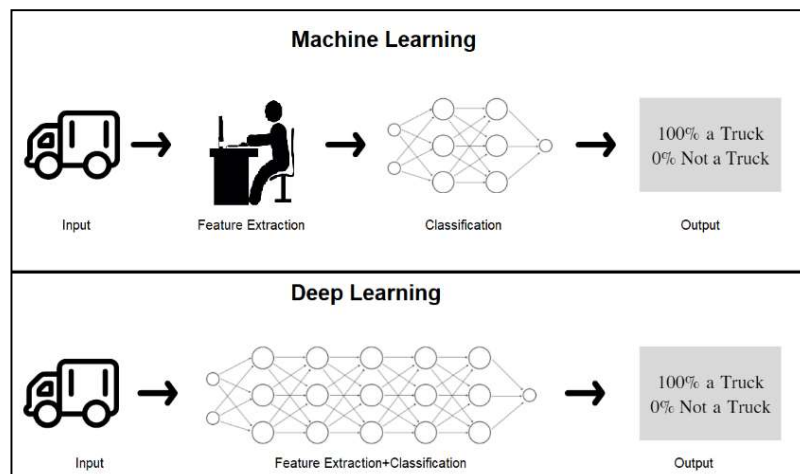
Although the ANNs have a high adaptability rate in order to identify new patterns that have not being seen and are easy to implement, when not well tuned can present a high probability of overfitting and a low generalization accuracy.

3.6.1 Deep Learning

By the time ANN were developed in the early 40's, the technology of the time limited its potentiality since store and processing capacities were both expensive and not sufficient. Given the exponential growth of the technology that allow us today have computers extremely more powerful than back in early 40's and cheaper, now it is possible to perform ANN in high dimensionality datasets (Big Data) for several real-world applications (YOO et al., 2014; AL-JARRAH et al., 2015; LANDSET et al., 2015; NAJAFABADI et al., 2015).

One successful approach when working with a high dimensionality dataset, consist on reduce the dataset size by representing the original dataset's features in a smaller dimension by using feature-extraction techniques. However, feature-extraction is a complex and time-consuming operation that is highly application-dependent impacts directly on the model's performance. DL, different from machine learning, has a unique automatic feature-extraction procedure, in which each hidden layer is responsible for training the unique set of features based on the output of the previous layer, as shown in Figure 3.29.

Figure 3.29 – Machine Learning vs. Deep Learning



FONT: the author, 2017.

DL mimics the neocortex of the brain, which is responsible by many of cognitive abilities. The brain's neocortex propagate the sensory signals through a complex hierarchy of models (LEE; MUMFORD, 2003), that performs extraction of complex data representations (features) at high levels of abstraction (NAJAFABADI et al., 2015). DL develop a layered, hierarchical architecture of learning and representing data, where higher-level (more abstract) features are defined in terms of lower-level (less abstract) features (NAJAFABADI et al., 2015).

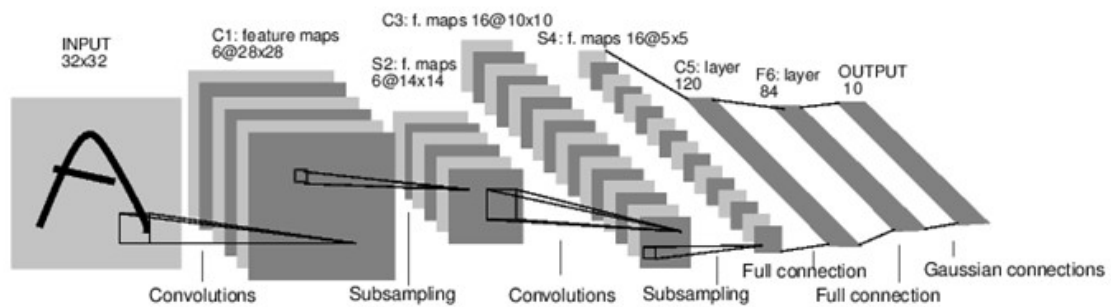
DL algorithms use a huge amount of unsupervised data to automatically extract complex representation and have the capability to generalize in non-local and global ways, generating learning patterns and relationships beyond immediate neighbors in the data (BENGIO et al., 2007).

The main DL approaches are, Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Stacked Auto-Encoders, Hierarchical Temporal Memory (HTM) and Deep Spatiotemporal Inference Network (DESTIN) (AREL et al., 2010).

3.6.2 Convolutional Neural Network

The first CNN model, named as LeNet, was developed by Lecun in 1998 (LECUN et al., 1998) in order to classify handwritten digits using The MNIST (Modified National of Standards and Technology) dataset on the training step of the LeNet, as shown in Figure 3.30. The authors concluded that the use of the CNN convolutional nature eliminates the need for hand-crafted feature extractors (LECUN et al., 1998).

Figure 3.30 – LeNet Architecture.



FONT: adapted from (LECUN et al., 1998)

The CNN's inspiration bases on the studies of the human visual cortex, in this particular cortex there is a small region with neurons that are sensible to information

of the visual range. In 1962 Hubel and Wiesel performed an experiment, as shown in Figure 3.31, where they were able to prove that when the human visual cortex is exposed to images composed by oriented edges in a specific direction there are neurons which generate strong synapses to this exposure. In this experiment they realized that these neurons form a specific structure in which each neuron generate a synapse for each determined edge orientation, and together these neurons are able to produce a visual perception (HUBEL; WIESEL, 1962).

By performing this experiment the authors manage to conclude that each neuron has a determined task in order to find a specific feature, and by combining the neurons' task all features detected by these neurons form the human visual perception (HUBEL; WIESEL, 1962).

Figure 3.31 – Hubel and Wiesel Experiment.



FONT: adapted from <https://www.youtube.com/watch?v=8VdFf3egwfg>

Based on these biological facts previously described, the CNN use filters known as kernels, that are applied on the image through convolutions in order to extract features, hence mathematically performing the biological neuron function. The application of the convolution step on images, generate new images with determined features, and then these images successively generate new inputs for the deeper layers of the CNN, where the convolutions are applied again with different kernels sizes. By this dynamic, the superficial layers of the CNN can extract simpler features as vertical and horizontal edges, and as long as the image is advancing through deeper layers, more complex features are extracted as geometric shapes, until it reaches the layers where the combinations of these features (face and objects) are extracted. In

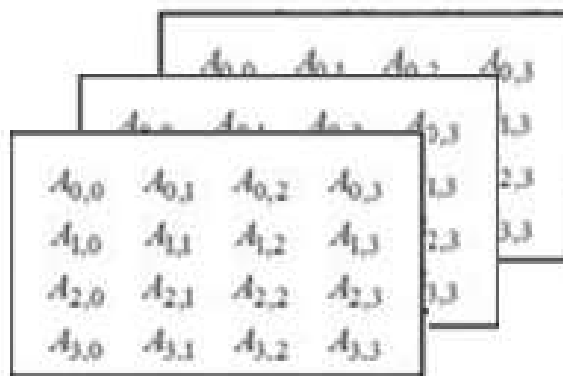
general, the deepest the layer of the CNN more abstract is the extracted feature. The analysis of all the abstracted features generate the image classification procedure.

The CNN architecture is divided in layers, and in each layer a different kind of operation is performed. The CNN layers are divided as follows.

3.6.2.1 Input Layer

In this layer is located the input data, and the size of this layer vary on the image size of the dataset in case. Usually the input layer takes an order three tensor as input with an image of M rows and N columns, and 3 channels (Red, Green and Blue color channels), as shown by Figure 3.32. The CNN can also take tensors of higher orders but the size of the input data can inflict directly on the processing time of the CNN.

Figure 3.32 – Example of a 4x4x3 Input Image.



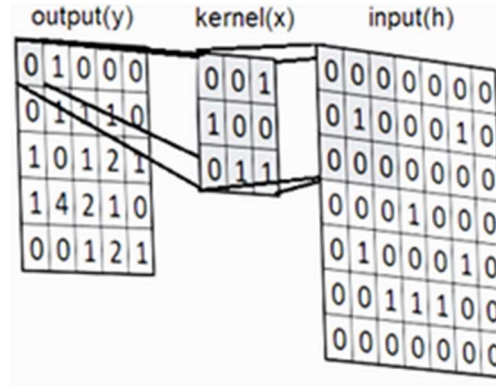
FONT: the author, 2017.

3.6.2.2 Convolution Layer

In the convolution layer of a CNN the kernels used are generally matrices with small dimensions (3x3,5x5,7x7...), as in the ANN model weights are attributed for each value of the positions of these kernels. The kernels move over the input image performing small convolutions, once the kernel has been displaced over all of the input image matrix and channels the outcome of the convolution it's the product of the small convolutions performed by the kernel over the image.

The convolution process, as shown in Figure 3.33, is the sum of the multiplications pixel-by-pixel of the kernel and the image region where the kernel is currently moving.

Figure 3.33 – 2D Image Convolution



FONT: the author, 2017.

The discrete 2D convolution is given by eq. 3.26.

$$y(m, n) = x(m, n) * h(m, n) = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i, j] \times h[m - i, n - j] \quad 3.26$$

where y is the output matrix, x is the kernel matrix and h is the input matrix. The variables m, n, i and j are the iterators used to represent each positions of the matrices' values.

More than one convolution layer can be applied in the CNN in order to extract a higher number of features in different levels of abstraction,

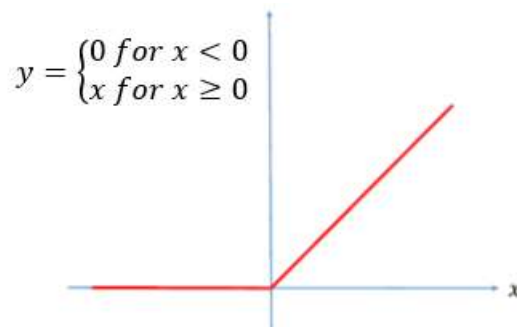
3.6.2.3 Rectified Linear Unit Layer

The Rectified Linear Unit (ReLU) layer does not change the size of the input, is applied in order to increase the non-linearity in the image, as given by eq. 3.27, since the semantic information of the input image is highly non-linear the purpose of the ReLU layer is to make the output of the convolutional layer non-linear as well (WU, 2017), as shown in Figure 3.34.

$$y_{i,j,d} = \max\{0, x_{i,j,d}^l\} \quad 3.27$$

where y and x are respective the output matrix and input matrix, the values of i, j and d are the values of the positions of the input matrix constrained between the ranges of $0 \leq i < M$, $0 \leq j < N$, and $0 \leq d < D$ (channels), and l is the layer.

Figure 3.34 – ReLU Function Graph



FONT: the author, 2017.

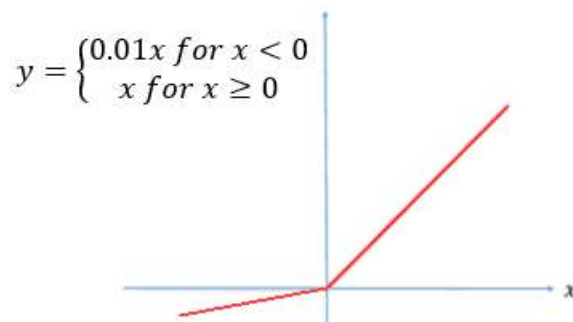
However extensively used in ANN applications the ReLU function is fragile during the training step. Depending on how the learning rate of the ANN is set, the ReLU layer could cause the weights to update in such a way that the neuron will never activate on any data input again, this phenomenon is called “dying ReLU”.

A special case of the ReLU function known as Leaky ReLU (eq. 3.28) with a quite small but significant modification can counteract the dying ReLU problem and prevent overfitting. As the ReLU function removes all the negative parts of the function the Leaky ReLU lower its magnitude and does not remove completely all the negative parts, as can be seen this behavior in Figure 3.35.

$$y_{i,j,d} = ax_{i,j,d}^l \quad 3.28$$

where a is a very small constant usually 0.01.

Figure 3.35 – Leaky ReLU Function Graph



FONT: the author, 2017

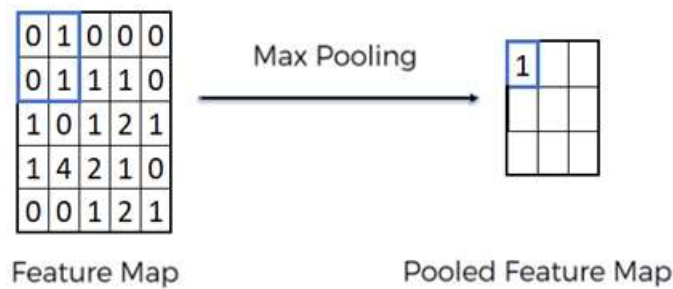
3.6.2.4 Pooling Layer

In the Pooling layer, is performed transformations on the feature map previous of the previous layer, in order to increase the spatial invariance of the CNN (SCHERER et al., 2010), which means that the CNN can distinguish features and recognize patterns regardless of the image position or texture.

In the max pooling operation, as shown in

Figure 3.36, the maximum value of the feature map that is inside the pooling kernel goes to the pooled feature map. Although some researches use subsampling as a pooling approach, (SCHERER et al., 2010) shown that a max pooling operation is vastly superior for capturing invariances in image-like data, compare to a subsampling operation.

Figure 3.36 – Example of Max Pooling Operation



FONT: the author, 2017.

The max pooling operation is given by

$$y_{j+1} = \max(x_j^{n \times n} u(n, n)) \quad 3.29$$

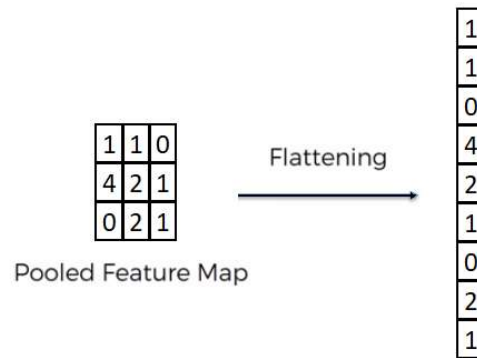
where, x_j represents the feature map and y_{j+1} the pooled feature map, the index j represents the layer, n represents the pooling kernel size and $u(n, n)$ a window function that is applied on the input patch.

3.6.2.5 Flattening Layer

The flattening layer is a very simple step, consists on the vectorization of the previous layer, which is usually a pooled feature map, as shown in Figure 3.37. This step is performed as a preparation for input to the Fully Connected (FC) Layer, is

usually added to the CNN when there are several pooling layers in the CNN architecture.

Figure 3.37 – Example of Flattening



FONT: the author, 2017.

3.6.2.6 Fully Connected Layer

The Fully Connected (FC) layer is commonly used at the end of the CNN, as well in other approaches used two or more consecutive FC layers. The FC layer is a Fully Connected ANN that perform the combination of the extracted features from the previous layers into more attributes in order to increase the prediction accuracy. Is where the classification/regression process occurs, when working with classification one important step to configure the FC layer is needed one output per class. The output of the FC layer is the probabilities referring to the predictions of each class.

The FC can benefit from pre-trained CNN models by having the ability to re-interpret existing classifications nets (HARICH, 2016).

3.6.2.7 Special Operations

In order to prevent overfitting and enhance classification accuracy there are some special operations that can be performed in a CNN, as for example the batch normalization and dropout operations.

The batch normalization operation, normalizes the data in each training mini-batch in order to avoid internal covariant shift prevenient from normalizing layers inputs, batch normalization allows the use much higher learning rates and to be less careful about initialization (IOFFE; SZEGEDY, 2015).

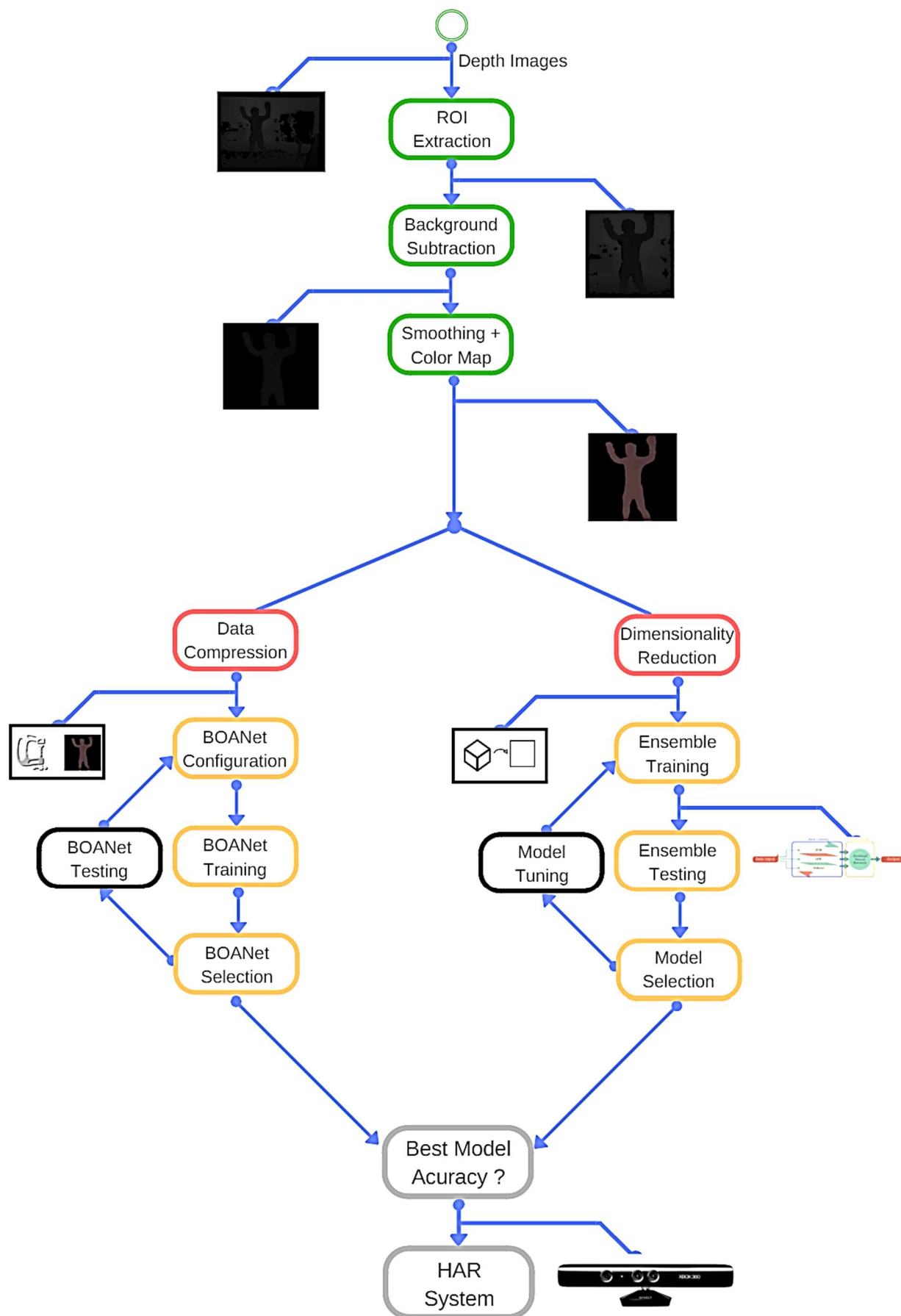
The dropout operation, applied in the FC layers, as quoted by (SRIVASTAVA et al., 2014) is a simple operation that the key idea is to randomly drop neurons (along with their connections) from the neural network during training. This technique can be understood as training multiple models on different portions of the data and averaging the model's predictions to reduce variance (ALMOUSLI, 2014). By performing dropout in the FC layer, it prevents neurons to co-adapt too much reducing overfitting and giving major improvements over other regularization methods.

4 METHODOLOGY

This chapter will focus on the methodology applied on solving the problem in case. As can be seen in Figure 4.1 an illustrative flowchart giving a brief description of the methodology.

Each topic will be described in this chapter with the respective tools and software, as follows.

Figure 4.1 - Methodology Flowchart



4.1 IMAGE PROCESSING

As the dataset was already introduced in Chapter 2, this chapter starts with the image processing step, which is an important step that deals with noise correction and visualization.

The image processing operation was performed with the OpenCV library.

The depth images that come in the dataset for convenience of distribution were all in the “.ppm” format, the first step of the image processing was to convert all the images from “.ppm” to “.jpg”. Figure 4.2 shows the raw image that come in the dataset.

Figure 4.2 - Depth Image from the Kinect Sensor



FONT: the author, 2017.

All raw depth images were in the size of 640x480 pixels, the focus of this problem is to recognize human activity, hence not all the elements of the picture are important for the analysis. For all the images, only the area where the human activity is projected will be important and the rest will be considered as noise.

To extract the relevant features of the image a Region of Interest (ROI) extraction will be performed by a simple crop of the original depth images reducing the original size to a 320x320 pixels frame, as shown in Figure 4.3 a.

Figure 4.3 - a) Original Depth Image ROI Left, b) Background Subtraction Result Image ROI Right.



FONT: the author, 2017.

Even after the ROI extraction, there are still some elements that are not relevant to the analysis, the background is considered as noise as well. To counteract this issue, a background subtraction was used to isolate only the human pose by performing a bitwise-AND mask operation with the mask generated by manual analysis of the image values and the cropped image generated by the previous ROI extraction step. This operation compares the values bitwise and remove the values that are not in the mask range, as result the image in Figure 4.3 b.

To enhance the quality of the visualization and hopefully improve the efficiency on the image detection a smoothing operation with a median filter was used. A median filter run through the whole image and replace each picture with the median of its neighborhood, result can be seen in Figure 4.4 a. In order to replace the dark shades of the grey image, which can sometimes be difficult to identify nuances and small shade variation a different colormap was applied in the image, as shown in Figure 4.4 b, the difference of before and after the colormap change.

Figure 4.4 - a) Smoothed Image Left, b) Smoothed Image + Colomar Map Change Right.



FONT: the author, 2017.

As result of the whole image processing operation, instead of having a noisy large depth image from the original dataset, the analysis can be performed with smaller non-noisy images with contain only relevant aspects. By reducing the size of the images, the computational effort demanded for the next steps will be reduced, hence fastening the whole machine learning process.

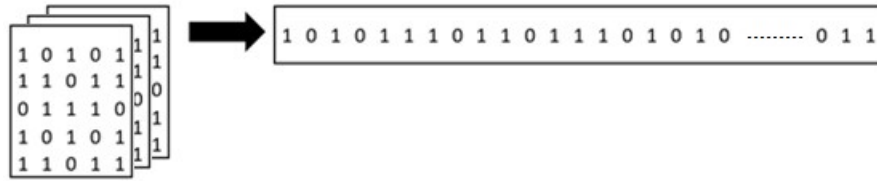
4.2 DIMENSIONALITY REDUCTION

As discussed in Chapter 3, dimensionality reduction techniques are very important in order to work with large datasets that demands a lot of computational effort and to extract a relevant reduced feature space that describes precisely the original feature space.

For both dimensionality reduction and classifications steps the scikit-learn package for python were used for the experiments.

When working with image classification, the image generally comes in a size of $M \times N \times K$, where M and N represents the image resolution and K the image dimension which is 1 for a grayscale image and 3 for an RGB image. The images are then transformed in a vector of size $1 \times (M \times N \times K)$, ending with a dataset of $R \times (M \times N \times K)$ where R is the number of samples in the dataset, in Figure 4.5 is presented a simple example of Image vectorization with a black and white 6x6x3 matrix.

Figure 4.5 - Image Vectorization Example

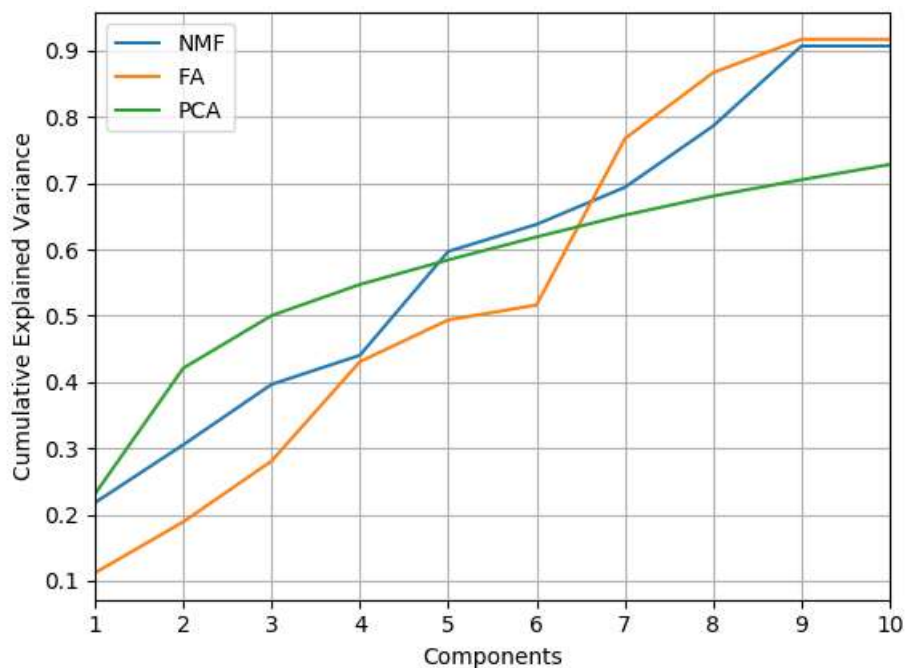


FONT: the author, 2017.

Dimensionality reduction techniques in image classification outcomes a reduced feature space that does not produce an image anymore but image features, and based on these set of features is where the classification will be performed.

From previous test, by analyzing the values of the cumulative explained variance as can be seen in Figure 4.6. which explains the amount of how much variance of the dataset is explained by each one of the components, from each one of the three dimensionality reduction techniques PCA, FA and NMF. It was able to identify that with only 10 components explains more than 70% the variance. So, for each of the three techniques the number of components were set to 10 in order to compare their performance against each other in terms of the same number of components

Figure 4.6 – Cumulative Explained Variance



FONT: the author, 2017.

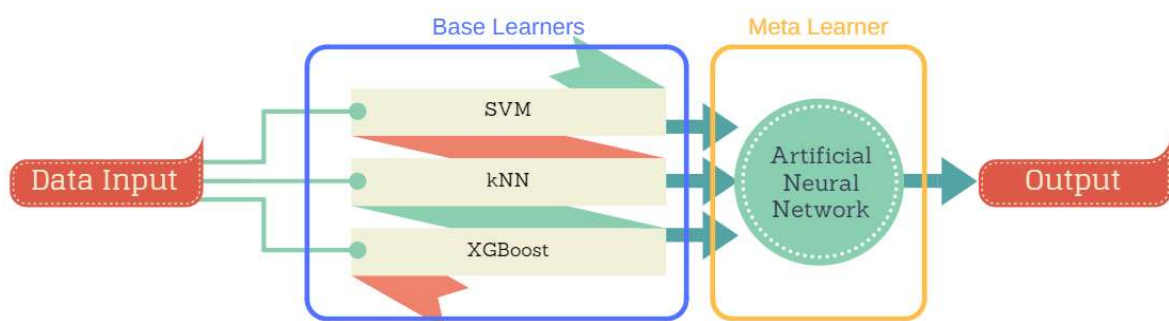
The original size of the dataset without any kind of pre-processing or dimensionality reduction is approximately 34.8 Gigabytes (GB) or 37,447,190,832 *bytes* and after the 2 first steps of the project (Image Processing and Dimensionality Reduction) the new dataset has only 5.35 Megabytes (MB) or 5,611,520 *bytes*.

The feature extraction process was performed individually by subject, it means that the features were extracted from a single subject performing a single action instead a whole group of subjects performing one action. The features extracted from each person performing an action are clearer to identify and has a more reliable representation, since each subject has different silhouette and by using the whole group it can generate inaccurate features of the action, hence this approach seems more trustworthy in order to maintain the diversity in the dataset and prevent overfitting in the classifier.

4.3 ENSEMBLE LEARNING CLASSIFICATION

As can be seen in Figure 4.7 the stacking implementation used in this project, where for the base learners is used SVM, kNN and XGBoost and an ANN was adopted as the meta learner that combines the three base learners, the stacking approach will be combined with a 5-fold cross-validation (CV).

Figure 4.7 – Proposed Stacking Approach



FONT: the author, 2017.

The parametrization of the base learners XGBoost, SVM and KNN is an important step to improve generalization accuracy and prevent overfitting, several tests have to be performed in order to achieve satisfactory results. The process that consist on finding the best parameters of a classifier model is called tuning, an efficient tuning can make a difference since the base learners are very sensitive to its parameters.

The parameters tuned for the ensemble in the base and meta learners are presented in Table 4.1.

Table 4.1 – Base and Meta Learners Parameters List

BASE LEARNERS						META LEARNER	
SVM Algorithm		XGBoost		KNN		ANN	
Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
Kernel Function	Radial-Basis	Maximum Depth	15	K	3	Neurons	100
Gamma(γ)	Automatic	Learning Rate	0.1				
Decomposition Function	one-vs-all	Number of Estimators	400				

FONT: the author, 2017.

4.3.1 K-fold Cross-Validation

CV is a widely used model validation technique, as cited in BROWNE (2000), CV was originally employed to evaluate the predictive validity of linear regression equations used to forecast a performance criterion from scores on a battery of tests (MOSIER, 1951).

To understand the K-fold CV, first it must be explained the concept of holdout, which is the simplest kind of cv. Holdout consists on splitting the dataset in two subsets, the training set and the test set, usually the division takes 70% of data for the training set and the remaining for the test set. However simple the holdout method is considered a pessimistic estimator for only a portion of the data is given for training (KOHAVI, 1995).

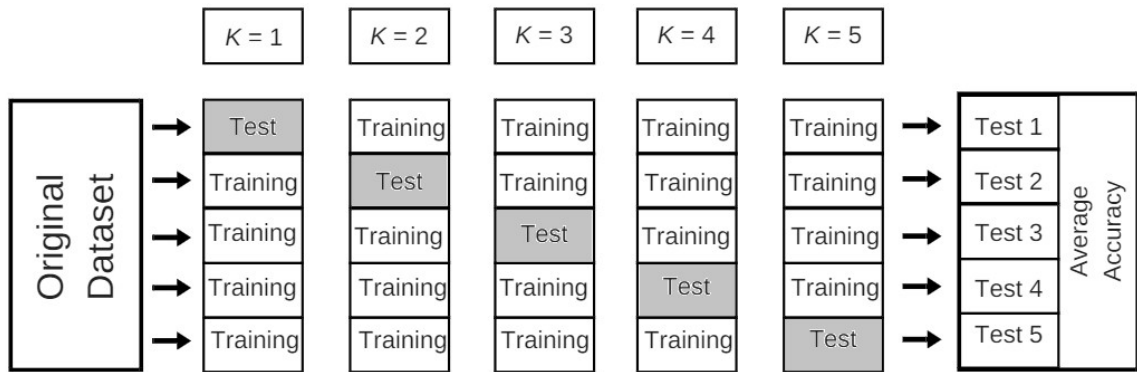
The general idea of the CV consists on dividing the original dataset in K equal sub-sets, so the holdout method is repeated K times, Figure 4.8.

A general idea of the CV as described by (GEISSER, 1975), CV in brief involves the mean of K holdout estimators from different sub-sets of an original dataset. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once, the disadvantage of this method is that the training algorithm has to be rerun from scratch k times, the CV error (eq. 4.1) is the average the K test errors.

$$CV(\lambda) = \frac{1}{K} \sum_{i=1}^K E_i \quad (4.1)$$

where λ is the parameters to the $K-1$ parts, and E is the error of each predicted K .

Figure 4.8 - CV Algorithm



FONT: the author, 2017.

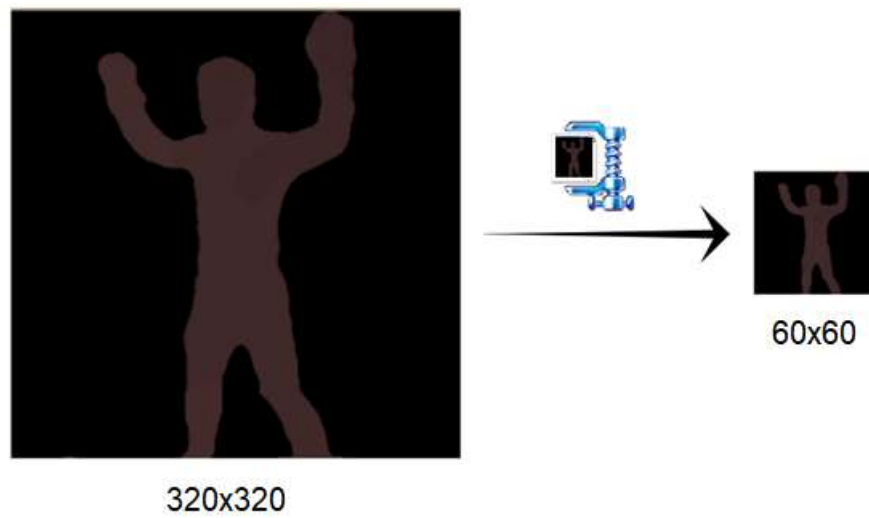
4.4 DEEP LEARNING CLASSIFICATION

As already mentioned previously in Chapter 3, the use of convolutional layers in the CNN eliminates the need of feature extraction techniques, and the CNN architecture complexity (deepness) will depend on the non-linearity of the input data.

After the image processing and the ROI extraction steps, the images of 320×320 *pixels* of resolution are still unfeasible to process with CNN with a machine with limited resources. A simple and efficient measure that can counteract this issue, as can be seen in Figure 4.9, is to compress each image original size (320×320 *pixels*) into a smaller size (60×60 *pixels*) be processed in a CNN.

The compression is performed by using the `resize` function in python, the process consists on reducing the original image size. As the original image consist on a very sparse image, there is no danger of losing too much information.

Figure 4.9 – Image Compression



FONT: the author, 2017.

After the image compression it is possible to perform the training step with the CNN.

The proposed CNN architecture is called BOANet, its architecture can be seen in APPENDIX A with a detailed description of its layers and configurations.

5 EXPERIMENTAL RESULTS

The experiments were performed on the following system, a Windows 10 – 64bit OS, CPU core i7-4700HQ 2.4 GHz, RAM 8 GB and on the Python 3.5.2.

The scoring parameter adopted for measuring the performance of the SVM algorithm is the accuracy of the confusion matrix. The confusion matrix is an easy and intuitive way to view a classifier performance in supervised learning, as can be seen in Figure 5.1 an example of a confusion matrix for binary classification.

The confusion matrix generates a set of 4 variables, that helps on understanding the prediction performance compared with the original labels(targets), given a binary classification problem with two classes named as positive P and negative N:

- True Positive (TP): correct positive prediction.
- False Positive (FP): incorrect positive prediction.
- True Negative (TN): correct negative prediction.
- False Negative (FN): incorrect negative prediction.

Figure 5.1 - Example of Confusion Matrix

		Predicted	
		P	N
Observed	P	TP	FN
	N	FP	TN

FONT: the author, 2017.

The accuracy of a confusion matrix (eq. 5.1) consists on the number of all correct predictions divided by the total number of the dataset, ranging from 0, which represents the worse accuracy, to 1 which represents the best accuracy.

$$Accuracy = \frac{TP + TN}{P + N} \quad (5.1)$$

The experiments were performed with the MHAD with 10 subjects performing 10 actions, in total 11920 observations were used for the tests.

The accuracy results from the tests compared with previous results shown in the literature are shown in Table 5.1.

Table 5.1 – Results

Values from the Two Proposed Approaches		
Approach	Techniques	Accuracy
(i) approach	PCA+SVM	18.42%
	PCA+KNN	22.38%
	PCA+XGBoost	20.13%
	PCA+Stacking	96.01%
	FA+SVM	13.79%
	FA+KNN	17.68%
	FA+XGBoost	12.78%
	FA+Stacking	99.93%
	NMF+SVM	28.04%
	NMF+KNN	27.64%
	NMF+XGBoost	25.09%
	NMF+Stacking	91.26%
(ii) approach	BOANet	99.05%
Reference Values from the Literature		
Approach	Techniques	Accuracy
(BRUN et al., 2014)	HACK system	97.70%
(ZHANG; PARKER, 2016)	CoDe4D + Adaptive MCOH + SVM	92.40%
(CHEN, CHEN et al., 2015)	SVM	92.39%
(OFLI et al., 2013)	Kernel-SVM	91.24%
(CHEEMA et al., 2014)	KNN	77.73%
(BRUN et al., 2015)	String Edit Distance (HARED)	87.10%
(FOGGIA et al., 2014)	Deep Belief Network	85.80%
(ZHU et al., 2016)	Recurrent Neural Network	100.00%
DU et al., 2015)	Hierarchical Bidirectional Recurrent Neural Network	100.00%
(SHAFAEI; LITTLE, 2016)	CNN	98.10%

FONT: the author, 2017.

Given the results presented in Table 5.1, it is possible to see that both methods proposed in this project achieved remarkable results.

Regarding the ensemble approach, the techniques that were applied in the ensemble present very low classification accuracy when used alone, however in an ensemble form all three combinations of ensemble with FA, NMF and SVM achieved

accuracy higher than 90%. The best ensemble approach was the one that used FA as dimensionality reduction technique, losing for only two of the reference approaches from the literature that used CNN in order to perform HAR.

The proposed CNN architecture BOANet, even though achieved high classification accuracy of 99.05% winning from most of the reference values of the literature, it still loses from the proposed stacking with FA approach and from the works of (ZHU et al., 2016) and DU et al., 2015).

In order to get a better view of both approaches performance, Table 5.2 shows the values of accuracy per class for each class.

Table 5.2 – Results of accuracy per class

Actions	Accuracy (%)	
	FA + Stacking	BOANet
Sit down	100.00%	100.00%
Jumping in place	100.00%	100.00%
Jumping jacks	100.00%	99.18%
Bending - hands up all the way down	100.00%	100.00%
Punching (boxing)	99.83%	98.31%
Waving - two hands	99.85%	99.26%
Waving - one hand (right)	100.00%	98.37%
Clapping hands	100.00%	97.71%
Throwing a ball	99.54%	97.69%
Sit down then stand up	100.00%	100.00%
Total	99.93%	99.05%

FONT: the author, 2017.

Both approaches present amazing results in the actions of sit down, sit down and then stand up, Bending and jumping in place. And also, both algorithm present its lower performance on the action of Throwing a ball.

6 FINAL CONSIDERATIONS

In this Chapter is presented the conclusion of the theses and also the details for the future works, regarding suggestions for future researches and improvement of the results.

6.1 CONCLUSION

In this work two approaches of machine learning were compared against each other and with reference values from the literature in a HAR problem with depth images from the MHAD.

One approach consisted on an ensemble form with stacking combining SVM, kNN, XGBoost and ANN. In order to decrease the computational cost of processing a large dataset of images, three different dimensionality reduction techniques FA, PCA and NMF were compared and combined with the stacking ensemble.

The other approach consisted on building an architecture of CNN specifically for this application known as BOANet. In order to be able to process the large dataset without using hand crafted features, data compression was used to decrease the image size and speed the process.

Both approaches performed very well, achieving remarkable results of classification accuracy that were higher than most of the reference values from the literature 99.05% for the BOANet and 99.93% for the Stacking with FA, losing only by two methods from the works of (ZHU et al., 2016 and DU et al., 2015) that with the use of DL techniques were able to reach 100% of accuracy.

The Stacking with FA approach, from the proposed method was the one with the second highest accuracy, and even outperformed some DL techniques,

6.2 FUTURE WORKS

For future references, in order to improve classification accuracy to 100%, for the ensemble learning approach, a study with more dimensionality reduction techniques and machine learning techniques must be done, in order to find out which combinations of techniques present better results for the classes that presented the lowest accuracy rates.

For the DL approach, a study of the CNN architecture with a proper configuration must be taken in account in order to build a network that can outperform BOANet.

REFERENCES

- ABDI, H.; WILLIAMS, L. J. Principal component analysis. **Wiley Interdisciplinary Reviews: Computational Statistics**, v. 2, n. 4, p. 433–459, 2010.
- ABNEY, S.; SCHAPIRE, R.; SINGER, Y. **Boosting applied to tagging and PP attachment**. Florham Park, NJ, 1999.
- AIZENBERG, I.; AIZENBERG, N. N.; VANDEWALLE, J. P. L. **Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications**. Kluwer Academic Publishers, 2000.
- AL-JARRAH, O. Y.; YOO, P. D.; MUHAIDAT, S.; KARAGIANNIDIS, G. K.; TAHA, K. Efficient machine learning for big data: A review. **Big Data Research**, v. 2, n. 3, p. 87–93, 2015. Elsevier.
- ALER, R.; GALVÁN, I. M.; RUIZ-ARIAS, J. A.; GUEYMARD, C. A. Improving the separation of direct and diffuse solar radiation components using machine learning by gradient boosting. **Solar Energy**, v. 150, p. 558–569, 2017. Elsevier Ltd.
- ALMOUSLI, H. **Recognition of Facial Expressions with Autoencoders and Convolutional-Nets**, 2014. Universite de Montreal.
- AREL, I.; ROSE, D. C.; KARNOWSKI, T. P. Deep machine learning—a new frontier in artificial intelligence research. **IEEE Computational Intelligence Magazine**, v. 5, n. 4, p. 13–18, 2010.
- ATHEY, S.; IMBENS, G. W. Machine learning methods for estimating heterogeneous causal effects. **stat**, v. 1050, n. 5, 2015.
- BAUER, E.; KOHAVI, R.; CHAN, P.; STOLFO, S.; WOLPERT, D. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. **Machine Learning**, v. 36, n. August, p. 105–139, 1999.
- BELLMAN, R. E. **Adaptive control processes - A guided tour**. Princeton, New Jersey, U.S.A.: Princeton University Press, 1961.
- BENALI, H. ; BUVAT, I. ; FROUIN, F. ; BAZIN, J. P. ; PAOLA, R. DI. A statistical model for the determination of the optimal metric in factor analysis of medical image sequences (FAMIS). **Physics in Medicine & Biology**, v. 38, n. 8, p. 1065–1080, 1993.
- BENGIO, Y.; LECUN, Y.; OTHERS. Scaling learning algorithms towards AI. **Large-scale kernel machines**, v. 34, n. 5, p. 1–41, 2007.
- BERESFORD, R. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. , v. 22, p. 717–727, 2000.
- BOARETTO, M. A. R.; BUSSATO, R. C.; COELHO, L. S. Abordagens de aprendizado de máquina aplicados na classificação da ocorrência de incêndios. **XLIX Simpósio Brasileiro de Pesquisa Operacional (XLIX SBPO)**. p.1–12, 2017.
- BOARETTO, M. A. R.; SEGUNDO, E. H. V.; MARIANI, V. C.; COELHO, L. S. Support vector machine approaches with features selection to detect cognitive states from brain image. **Brain Function Assessment in Learning**. p. 211–213, 2017. Patras-Greece.

- BOUZALMAT, A.; KHARROUBI, J.; ZARGHILI, A. Comparative study of PCA, ICA, LDA using SVM classifier. **Journal of Emerging Technologies in Web Intelligence**, v. 6, n. 1, p. 64–68, 2014.
- BREIMAN, L. Bagging predictors. **Machine Learning**, v. 24, n. 2, p. 123–140, 1996.
- BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001.
- BROWNE, M. W. Cross-Validation Methods. **Journal of mathematical psychology**, v. 44, n. 1, p. 108–132, 2000.
- BRUN, L.; FOGGIA, P.; SAGGESE, A.; VENTO, M. Recognition of Human Actions using Edit Distance on Aclet Strings. **Proceedings of the 10th International Conference on Computer Vision Theory and Applications**. p.97–103, 2015.
- BRUN, L.; PERCANNELLA, A.; SAGGESE, M.; VENTO, M. HAcK: A system for the recognition of human actions by kernels of visual strings. **Proc. IEEE International Conference on Advanced Video and Signal Based Surveillance**. p.142–147, 2014.
- BURGES, C. J. A Tutorial on Support Vector Machines for Pattern Recognition. **Data Mining and Knowledge Discovery**, v. 2, n. 2, p. 121–167, 1998.
- BURRASCANO, P.; FIORI, S.; MONGIARDO, M. A Review of Artificial Neural Networks Applications in Microwave Computer-Aided Design (Invited Article). , p. 158–174, 1998.
- CAMPS-VALLS, G.; BRUZZONE, L.; ROJO-ÁLVAREZ, J. L.; MELGANI, F. Robust support vector regression for biophysical variable estimation from remotely sensed images. **IEEE Geoscience and Remote Sensing Letters**, v. 3, n. 3, p. 339–343, 2006.
- CHATTOPADHYAY, P.; ROY, A.; SURAL, S.; MUKHOPADHYAY, J. Pose depth Volume extraction from RGB-D streams for frontal gait recognition. **Journal of Visual Communication and Image Representation**, v. 25, n. 1, p. 53–63, 2014. Elsevier Inc..
- CHAUDHRY, R.; OFLI, F.; KURILLO, G.; BAJCS, R.; VIDAL, R. Bio-inspired dynamic 3D discriminative skeletal features for human action recognition. **Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops**. p.471–478, 2013.
- CHEEMA, M. S.; EWEIWI, A.; BAUCKHAGE, C. Human activity recognition by separating style and content. **Pattern Recognition Letters**, v. 50, p. 130–138, 2014. Elsevier B.V.
- CHEN, C.; JAFARI, R.; KEHTARNAVAZ, N. Utd-mad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. **Proc. IEEE International Conference on Image Processing**. 2015.
- CHEN, C.; JAFARI, R.; KEHTARNAVAZ, N. Improving human action recognition using fusion of depth camera and inertial sensors. **IEEE Transactions on Human-Machine Systems**, v. 45, n. 1, p. 51–61, 2015.
- CHEN, T.; GUESTRIN, C. XGBoost: A Scalable Tree Boosting System. **Proceedings**

of the **22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. p.785–794, 2016. New York, NY, USA: ACM.

CHEN, W.-S.; PAN, B.; FANG, B.; LI, M.; TANG, J. Incremental Nonnegative Matrix Factorization for Face Recognition. **Mathematical Problems in Engineering**, v. 2008, p. 1–17, 2008.

CHOI, J. Y.; KIM, D. H.; PLATANIOTIS, K. N.; RO, Y. M. Classifier ensemble generation and selection with multiple feature representations for classification applications in computer-aided detection and diagnosis on mammography. **Expert Systems with Applications**, v. 46, p. 106–121, 2016. Elsevier Ltd.

CHOUDHURY, S. DAS; TJAHHADI, T. Silhouette-based gait recognition using Procrustes shape analysis and elliptic Fourier descriptors. **Pattern Recognition**, v. 45, n. 9, p. 3414–3426, 2012. Elsevier.

CORTI, A.; GIANCOLA, S.; MAINETTI, G.; SALA, R. A metrological characterization of the Kinect V2 time-of-flight camera. **Robotics and Autonomous Systems**, v. 75, p. 584–594, 2016. Elsevier B.V.

COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, p. 21–27, 1967.

CRUZ, L.; LUCIO, D.; VELHO, L. Kinect and RGBD images: Challenges and applications. **Proceedings: 25th SIBGRAPI - Conference on Graphics, Patterns and Images Tutorials, SIBGRAPI-T 2012**, p. 36–49, 2012.

CUTLER, D. R.; EDWARDS, T. C.; BEARD, K. H.; et al. Random Forests for Classification in Ecology. **Ecology**, v. 88, n. 11, p. 2783–2792, 2007.

DARBY, J.; SÁNCHEZ, M. B.; BUTLER, P. B.; LORAM, I. D. An evaluation of 3D head pose estimation using the Microsoft Kinect v2. **Gait & Posture**, v. 48, p. 83–88, 2016.

DECHTER, R. **Learning while searching in constraint-satisfaction problems**. University of California, Computer Science Department, Cognitive Systems Laboratory, 1986.

DECOSTER, J.; HALL, G. P. Overview of Factor Analysis. **In Practice**, v. 37, n. 2, p. 141, 1998.

DEO, R. C. Machine learning in medicine. **Circulation**, v. 132, n. 20, p. 1920–1930, 2015. Am Heart Assoc.

DING, I. J.; CHANG, C. W. An eigenspace-based method with a user adaptation scheme for human gesture recognition by using Kinect 3D data. **Applied Mathematical Modelling**, v. 39, n. 19, p. 5769–5777, 2015. Elsevier Inc.

DOLATABADI, E.; TAATI, B.; MIHAILIDIS, A. Concurrent validity of the Microsoft Kinect for Windows v2 for measuring spatiotemporal gait parameters. **Medical Engineering & Physics**, v. 0, p. 1–7, 2016. Elsevier Ltd.

DU, Y.; WANG, W.; WANG, L. Hierarchical recurrent neural network for skeleton based action recognition. **Proc. IEEE Conference on Computer Vision and Pattern Recognition**. p.1110–1118, 2015.

DUTTA, S.; PAL, S. K.; SEN, R. On-machine tool prediction of flank wear from machined surface images using texture analyses and support vector regression. **Precision Engineering**, v. 43, p. 34–42, 2016. Elsevier Inc.

EDITOR IJSMI. Tutorial: Factor analysis revisited – An overview with the help of SPSS, SAS and R packages. **International Journal of Statistics and Medical Informatics**, v. 3, n. 1, p. 1–14, 2017.

ELLIS, C.; MASSOD, S. Z.; TAPPEN, M. F.; LAVIOLA JR, J. J.; SUKTHANKAR, R. Exploring the trade-o between accuracy and observational latency in action recognition. **International Journal of Computer Vision**, v. 101, n. 3, p. 420–436, 2013.

FABRIGAR, L. R.; WEGENER, D. T.; MACCALLUM, R. C.; STRAHAN, E. J. Evaluating the use of exploratory factor analysis in psychological research. **Psychological methods**, v. 4, n. 3, p. 272, 1999. American Psychological Association.

FILKO, D.; CUPEC, R.; NYARKO, E. K. Detection, Reconstruction and Segmentation of Chronic Wounds Using Kinect v2 Sensor. **Procedia Computer Science**, v. 90, n. July, p. 151–156, 2016. The Author(s).

FOGGIA, P.; SAGGESE, A.; STRISCIUGLIO, N.; VENTO, M. Exploiting the Deep Learning Paradigm for Recognizing Human Actions. **11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)**. p.93–98, 2014.

FRIEDMAN, J. H. Greedy Function Approximation: A Gradient Boosting Machine. **Annals of Statistics**, v. 29, n. 5, p. 1189–1232, 2001.

GAN, G. Application of data clustering and machine learning in variable annuity valuation. **Insurance: Mathematics and Economics**, v. 53, n. 3, p. 795–801, 2013. Elsevier.

GARDNER, M. .; DORLING, S. . Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. **Atmospheric Environment**, v. 32, n. 14–15, p. 2627–2636, 1998.

GEISSER, S. The predictive sample reuse method with applications. **Journal of the American Statistical Association**, v. 70, n. 350, p. 320–328, 1975.

GHAEDI, M.; RAHIMI, M. REZA; GHAEDI, A. M.; et al. Application of least squares support vector regression and linear multiple regression for modeling removal of methyl orange onto tin oxide nanoparticles loaded on activated carbon and activated carbon prepared from Pistacia atlantica wood. **Journal of Colloid and Interface Science**, v. 461, p. 425–434, 2016. Elsevier Inc.

GHOSH, R.; PURKAYASTHA, P. Forecasting Profitability in Equity Trades Using Random Forest , Support Vector Machine and Xgboost. **10th International Conference on Recent Trends in Engineering Science and Management**. p.473–486, 2017. Guntur Dist, Andhra Pradesh, India.

GONÇALVES, A. R. Máquina de Vetores Suporte. 2009.

GOSWAMI, G.; BHARADWAJ, S.; VATSA, M.; SINGH, R. On RGB-D face recognition

using Kinect. **2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)**, p. 1–6, 2013.

GUO, Y.; LIU, Y.; OERLEMANS, A.; et al. Deep learning for visual understanding: A review. **Neurocomputing**, v. 187, p. 27–48, 2016.

HALD, A. On the History of Maximum Likelihood in Relation to Inverse Probability and Least Squares. **Statistical Science**, v. 14, n. 2, p. 214–222, 1999. Institute of Mathematical Statistics.

HARICH, N. **Fully Convolutional Networks for Semantic Segmentation from RGB-D images**, 2016. Hochschule der Medien.

HAYAT, M.; BENNAMOUN, M.; EL-SALLAM, A. A. An RGB-D based image set classification for robust face recognition from Kinect data. **Neurocomputing**, v. 171, p. 889–900, 2016. Elsevier.

HOLLOWAY, E.; MARKS, R. High Dimensional Human Guided Machine Learning. n. 2, 2016.

HOTELLING, H. Analysis of a complex of statistical variables into principal components. **J. Educ. Psych.**, v. 24, 1933.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. **The Journal of physiology**, v. 160, n. 1, p. 106–154, 1962.

HUGHES, G. F. On the mean accuracy of statistical pattern recognizers. **IEEE Transactions on Information Theory**, v. 14, n. 1, p. 55–63, 1968.

HWANG, W.; WEN, K.-W. Fast kNN classification algorithm based on partial distance. **Electronics Letters**, v. 34, n. 21, p. 2062–2063, 1998.

IBAÑEZ, R.; SORIA, Á.; TEYSEYRE, A.; RODRÍGUEZ, G.; CAMPO, M. Approximate string matching: A lightweight approach to recognize gestures with Kinect. **Pattern Recognition**, v. 62, p. 73–86, 2016. Elsevier.

IJJINA, E. P.; MOHAN, C. K. Human action recognition based on mocap information using convolution neural networks. **Proc. International Conference on Machine Learning and Applications**. p.159–164, 2014.

IJJINA, E. P.; MOHAN, C. K. Classification of human actions using pose-based features and stacked auto encoder. **Pattern Recognition Letters**, v. 83, p. 268–277, 2016.

IOFFE, S.; SZEGEDY, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. **Proceedings of the 32nd International conference on Machine Learning**. v. 37, p.81–87, 2015. Lille, France: JMLR: W&CP.

IZMIRLIAN, G. Application of the Random Forest Classification Algorithm to a SELDI-TPF Proteomics Study in the Setting of a Cancer Prevention Trial. **National Cancer Institute**, v. 6, 2004.

JOSÉ, D.; RIBEIRO, S. Daniel José Silva Ribeiro Support Vector Machines na

Previsão do Comportamento de uma ETAR. , 2012.

KAISER, H. F. The varimax criterion for analytic rotation in factor analysis. **Psychometrika**, v. 23, n. 3, p. 187–200, 1958.

KALOGIROU, S. A. Applications of artificial neural-networks for energy systems. **Applied Energy**, v. 67, n. 1–2, p. 17–35, 2000.

KANG, S.; KANG, P.; KO, T.; et al. An efficient and effective ensemble of support vector machines for anti-diabetic drug failure prediction. **Expert Systems with Applications**, v. 42, n. 9, p. 4265–4273, 2015. Elsevier Ltd.

KASTANIOTIS, D.; THEODORAKOPOULOS, I.; THEOHARATOS, C.; ECONOMOU, G.; FOTOPOULOS, S. A framework for gait-based recognition using Kinect. **Pattern Recognition Letters**, v. 68, p. 327–335, 2015. Elsevier Ltd.

KEARNS, M. J.; VALIANT, L. G. Learning Boolean formulae or finite automata is as hard as factoring. Harvard University. **Center for Research in Computing Technology, Aiken Computation Laboratory**, 1988.

KEARNS, M.; VALIANT, L. Cryptographic Limitations on Learning Boolean-Formulas and Finite Automata. **Journal Of The Acm**, v. 41, n. 1, p. 67–95, 1994.

KECK, T. FastBDT: A speed-optimized and cache-friendly implementation of stochastic gradient-boosted decision trees for multivariate classification. , p. 1–16, 2016.

KHOSLA, N. Dimensionality reduction using factor analysis. **Components**, 2004.

KLINE, P. **An Easy Guide to Factor Analysis**. illustrate ed. Psychology Press, 1994.

KLUMP, J.; HUBER, R.; ROBERTSON, J.; COX, S. J. D.; WOODCOCK, R. Linking descriptive geology and quantitative machine learning through an ontology of lithological concepts. **AGU Fall Meeting Abstracts**. 2014.

KOHAVI, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection 2 Methods for Accuracy Estimation. **Proceedings of IJCAI'95**, p. 1137–1145, 1995.

KOHONEN, T. The self-organizing map. 1990.

KORUP, O.; STOLLE, A. Landslide prediction from machine learning. **Geology Today**, v. 30, n. 1, p. 26–33, 2014. Wiley Online Library.

KRIZHEVSKY, A.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. **NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems**, p. 1097-, 2012.

LANDSET, S.; KHOSHGOFTAAR, T. M.; RICHTER, A. N.; HASANIN, T. A survey of open source tools for machine learning with big data in the Hadoop ecosystem. **Journal of Big Data**, v. 2, n. 1, p. 24, 2015. Springer International Publishing.

LECUN, Y.; BENGIO, Y.; HINTON, G.; et al. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, 2015.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-Based Learning Applied to Document Recognition. , 1998.

LEE, D. D.; SEUNG, H. S. Learning the parts of objects by non-negative matrix factorization. **Nature**, v. 401, n. 6755, p. 788–91, 1999.

LEE, D.; SEUNG, H. Algorithms for non-negative matrix factorization. **Advances in neural information processing systems**, , n. 1, p. 556–562, 2001.

LEE, T. S.; MUMFORD, D. Hierarchical Bayesian inference in the visual cortex. **Journal of the Optical Society of America**, v. 20, n. 7, p. 1434–1448, 2003.

LI, B. Y. L.; MIAN, A. S.; LIU, W.; KRISHNA, A. Using Kinect for face recognition under varying poses, expressions, illumination and disguise. **Proceedings of IEEE Workshop on Applications of Computer Vision**, p. 186–192, 2013.

LI, J.; TAO, D. On preserving original variables in Bayesian PCA with application to image analysis. **IEEE Transactions on Image Processing**, v. 21, n. 12, p. 4830–4843, 2012. IEEE.

LIMA, C. A. D. M. **Comitê de Máquinas: Uma Abordagem Unificada Empregando Máquinas de Vetores-Suporte**, 2004. Universidade Estadual de Campinas.

LINNA, M.; KANNALA, J.; RAHTU, E. Real-time Human Pose Estimation from Video with Convolutional Neural Networks. , p. 16, 2016.

LOCHTER, J. V.; ZANETTI, R. F.; RELLER, D.; ALMEIDA, T. A. Short text opinion detection using ensemble of classifiers and semantic indexing. **Expert Systems with Applications**, v. 62, p. 243–249, 2016. Elsevier Ltd.

LORENA, A. C.; CARVALHO, A. C. P. L. F. DE. Uma Introdução às Support Vector Machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43–67, 2007.

MALINOWSKI, E. R. Theory of error for target factor analysis with applications to mass spectrometry and nuclear magnetic resonance spectrometry. **Analytica Chimica Acta**, v. 103, n. 4, p. 339–354, 1978. Elsevier.

MCAFEE, A.; BRYNJOLFSSON, E.; DAVENPORT, T. H.; PATIL, D.; BARTON, D. Big data: the management revolution. **Harvard business review**, v. 90, n. 10, p. 61–67, 2012.

MCCULLOCH, W.; PITTS, W. originally published in: **Bulletin of Mathematical Biophysics**, Vol. 5, 1943, p. 115-133. , v. 5, p. 115–133, 1943.

MEJDOUB, M.; AMAR, C. BEN. Classification improvement of local feature vectors over the KNN algorithm. **Multimedia Tools and Applications**, v. 64, n. 1, p. 197–218, 2013.

MILJKOVIC, D.; ALEKSOVSKI, D.; PODPECAN, V.; et al. Machine Learning and Data Mining Methods for Managing Parkinsons Disease. **Machine Learning for Health Informatics**. p.209–220, 2016. Springer.

MITCHELL, T. M. **Machine Learning**. 1997.

MO, Q.; DRAPER, B. A. Semi-Nonnegative Matrix Factorization for Motion

Segmentation with Missing Data. , p. 402–415, 2012.

MØRUP, M.; HANSEN, L. K.; HERRMANN, C. S.; PARNAS, J.; ARNFRED, S. M. Parallel Factor Analysis as an exploratory tool for wavelet transformed event-related EEG. **NeuroImage**, v. 29, n. 3, p. 938–947, 2006.

MOSIER, C. I. I. Problems and Designs of Cross-Validation 1. **Educational and Psychological Measurement**, v. 11, n. 1, p. 5–11, 1951.

MUSTAFA, M.; TAIB, M.; MURAT, Z.; SULAIMAN, N. Comparison between KNN and ANN classification in brain balancing application via spectrogram image. **Journal of Computer Science & ...**, v. 2, n. 4, p. 17–22, 2012.

NAJAFABADI, M. M.; VILLANUSTRE, F.; KHOSHGOFTAAR, T. M.; et al. Deep learning applications and challenges in big data analytics. **Journal of Big Data**, v. 2, n. 1, p. 1, 2015.

NEGIN, F.; OZDEMIR, F.; AKGUL, C. B.; YUSKEL, K. A.; ERÇİL, A. A decision forest based feature selection framework for action recognition from RGB-depth cameras. **Image Analysis and Recognition**. p.648–657, 2013. Springer.

OFLI, F.; CHAUDHRY, R.; KURILLO, G.; VIDAL, R.; BAJCS, R. Berkeley MHAD: A Comprehensive Multimodal Human Action Database. In **Proceedings of the IEEE Workshop on Applications on Computer Vision (WACV)**, 2013.

OKUN, O. G. Non-negative Matrix Factorization and Classifiers: Experimental Study. **the 4th IASTED International Conference on Visualization, Imaging, and Image Processing (VIIP'04)**, , n. May, p. 550–555, 2004.

PAATERO, P.; TAPPER, U. Positive matrix factorization: a nonnegative factor model with optimal utilization of error estimates of data values. **Environmetrics**, v. 5, n. 2, p. 111–126, 1994.

PADILLA, P.; LOPEZ, M.; GORRIZ, J. M.; et al. NMF-SVM based CAD tool applied to functional brain images for the diagnosis of Alzheimer's disease. **IEEE Transactions on Medical Imaging**, v. 31, n. 2, p. 207–216, 2012.

PAISLEY, J.; CARIN, L. Nonparametric Factor Analysis with Beta Process Priors. **Proceedings of the 26th International Conference on Machine Learning**, p. 1–8, 2009. Disponível em: <<http://portal.acm.org/citation.cfm?doid=1553374.1553474>>. .

PATIL, U.; MUDENGUDI, U. Image fusion using hierarchical PCA. image Information Processing (ICIIP), 2011 International Conference on. **Anais...** . p.1–6, 2011.

PEARSON, K. On lines and planes of closest fit to systems of points in space. **Philosophical Magazine**, v. 2, n. 6, p. 559–572, 1901.

PRADHAN, A. SUPPORT VECTOR MACHINE-A Survey. **International Journal of Emerging Technology and Advanced Engineering**, v. 2, n. 8, p. 82–85, 2012.

QUINTANILHA, I. M. Algoritmos para fatoração de matrizes não-negativas com aplicação em transcrição de instrumentos percussivos. , v. 1, p. 78, 2016.

RAMTEKE, R. J.; Y, K. M. Automatic Medical Image Classification and Abnormality

Detection Using K- Nearest Neighbour. **International Journal of Advanced Computer Research**, v. 2, n. 4, p. 190–196, 2012.

REBENTROST, P.; MOHSENI, M.; LLOYD, S. Quantum support vector machine for big data classification. **Physical review letters**, v. 113, n. 13, p. 130503, 2014. APS.

REN, Y.; ZHANG, L.; SUGANTHAN, P. N. Ensemble classification and regression: Recent developments, applications and future directions. **IEEE Computational Intelligence Magazine**, v. 11, n. 1, p. 41–53, 2016.

RICHARDSON, M. Principal component analysis. **Signal Processing**, 2009.

ROJAS, R. The Hopfield Model. 1996.

ROSENBLATT. The perceptron A perceiving and recognizing automaton. 1957.

ROST, B.; RADIVOJAC, P.; BROMBERG, Y. Protein function in precision medicine: deep understanding with machine learning. **FEBS letters**, v. 590, n. 15, p. 2327–2341, 2016. Wiley Online Library.

RUMMEL, R. J. **Applied Factor Analysis**. Evanston: Northwestern University Press, 1988.

SADHASIVAM, S. K.; KEERTHIVASAN, M. B.; MUTTAN, S. Implementation of max principle with PCA in image fusion for surveillance and navigation application. **ELCVIA Electronic Letters on Computer Vision and Image Analysis**, v. 10, n. 1, 2011.

SAMUEL, A. L. Some Studies in Machine Learning Using the Game of Checkers. **IBM Journal of Research and Development**, v. 3, n. 3, p. 210–229, 1959.

SCHAPIRE, R. E. The Strength of Weak Learnability. **Mach. Learn.**, v. 5, n. 2, p. 197–227, 1990. Hingham, MA, USA: Kluwer Academic Publishers.

SCHAPIRE, R. E. A brief introduction to boosting. **IJCAI International Joint Conference on Artificial Intelligence**, v. 2, n. 5, p. 1401–1406, 1999.

SCHERER, D.; ANDREAS, M.; BEHNKE, S. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. September, p. 92–101, 2010.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. **Neural networks**, v. 61, p. 85–117, 2015. Elsevier.

SEGUNDO, M. P.; SARKAR, S.; GOLDFOG, D.; SILVA, L.; BELLON, O. Continuous 3D face authentication using rgb-d cameras. **IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops**, p. 64–69, 2013.

SHAF AEI, A.; LITTLE, J. J. Real-time human motion capture with multiple depth cameras. **Proceedings - 2016 13th Conference on Computer and Robot Vision, CRV 2016**. p.24–31, 2016.

SILL, J.; TAKACS, G.; MACKEY, L.; LIN, D. Feature-Weighted Linear Stacking. **CoRR**, v. abs/0911.0, n. November 2009, p. 17, 2009.

SPEARMAN, C. General Intelligence, Objectively Determined and Measured. **The American Journal of Psychology**, v. 15, n. 2, p. 201–292, 1904.

SPEARMAN, C. **The Abilities Of Man**. 1st ed. Macmillan And Company., Limited, 1927.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. **Journal of Machine Learning Research**, v. 15, p. 1929–1958, 2014.

TAHMASEBI, P.; AMIRKABIR, A. H. Application of a Modular Feedforward Neural Network for Grade Estimation Application of a Modular Feedforward Neural Network for Grade Estimation. , , n. April 2016, 2011.

TAM, K. Y.; KIANG, M. Y. Managerial Applications of Neural Networks: The Case of Bank Failure Predictions. **Management Science**, v. 38, n. 7, p. 926–947, 1992.

TAN, S. An effective refinement strategy for KNN text classifier. **Expert Systems with Applications**, v. 30, n. 2, p. 290–298, 2006.

TANNOUS, H.; ISTRATE, D.; HO BA THO, M. C.; DAO, T. T. Serious game and functional rehabilitation for the lower limbs. **European Research in Telemedicine / La Recherche Européenne en Télémédecine**, v. 5, n. 2, p. 65–69, 2016. Elsevier Masson SAS.

THURSTONE, L. L. **The Vectors Of Mind Multiple Factor Analysis For The Isolation Of Primary Traits**. 1st ed. Chicago: The University Of Chicago Press, 1935.

TRAN, C.; TRIVEDI, M. M. 3-D posture and gesture recognition for interactivity in smart spaces. **IEEE Transactions on Industrial Informatics**, v. 8, n. 1, p. 178–187, 2012.

URRACA, R.; MARTINEZ-DE-PISON, E.; SANZ-GARCIA, A.; ANTONANZAS, J.; ANTONANZAS-TORRES, F. Estimation methods for global solar radiation: Case study evaluation of five different approaches in central Spain. **Renewable and Sustainable Energy Reviews**, , n. November, 2016.

ÜZÜMCÜ, M.; FRANGI, A. F.; SONKA, M.; REIBER, J. H. C.; LELIEVELDT, B. P. F. ICA vs. PCA active appearance models: Application to cardiac MR segmentation. **International Conference on Medical Image Computing and Computer-Assisted Intervention**. p.451–458, 2003.

VAPNIK, V.; CORTES, C. Support-vector Networks. , v. 7, 1995.

WANG, J.; LIU, Z.; WU, Y.; YUAN, J. Mining actionlet ensemble for action recognition with depth cameras. **Proc. IEEE Conference on Computer Vision and Pattern Recognition**. p.1290–1297, 2012.

WANG, Y.-X. Nonnegative Matrix Factorization: A Comprehensive Review. **IEEE Transactions on Knowledge and Data Engineering**, v. 25, n. 6, p. 1–18, 2013.

WANG, Z.; SONG, X.; WANG, S.; et al. Filling Kinect depth holes via position-guided matrix completion. **Neurocomputing**, p. 1–5, 2016. Elsevier.

WILFREDO, J.; VILLANUEVA, P. **Comitê de Máquinas em Predição de Séries**, 2006. Universidade Estadual de Campinas.

WITTEN, I. H.; FRANK, E.; HALL, M. A.; PAL, C. J. **Data Mining Practical Machine**

Learning Tools and Techniques. 4th ed. Cambridge, MA, USA: Elsevier, 2017.

WOLD, S.; ESBENSEN, K.; GELADI, P. Principal component analysis. **Chemometrics and Intelligent Laboratory Systems**, v. 2, n. 1–3, p. 37–52, 1987.

WOLPERT, D. Stacked Generalization. **Neural networks**, v. 5, n. 2, p. 241–259, 1992.

WU, J. Introduction to Convolutional Neural Networks. , p. 1–31, 2017.

WUEST, T.; IRGENS, C.; THOBEN, K.-D. An approach to monitoring quality in manufacturing using supervised machine learning on product state data. **Journal of Intelligent Manufacturing**, v. 25, n. 5, p. 1167–1180, 2014. Springer.

XIA, L.; CHEN, C. C.; AGGARWAL, J. K. View invariant human action recognition using histograms of 3D joints,. **Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops**. p.20–27, 2012.

XIA, Y.; LIU, C.; LI, Y.; LIU, N. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. **Expert Systems with Applications**, v. 78, p. 225–241, 2017. Elsevier Ltd.

XU, J.; TANG, L.; LI, T. System situation ticket identification using SVMs ensemble. **Expert Systems with Applications**, v. 60, p. 130–140, 2016. Elsevier Ltd.

XU, L.; DONG, J. Y.; CAI, C. B.; CHEN, Z. Multi-focus image fusing based on non-negative matrix factorization. **Proceedings 14th International Conference on Mechatronics and Machine Vision in Practice, M2VIP2007**, p. 108–111, 2007.

YAMAMOTO, K.; TSUBOKURA, M.; IKEDA, J.; ONISHI, K.; BALERIOLA, S. Effect of posture on the aerodynamic characteristics during take-off in ski jumping. **Journal of Biomechanics**, v. 49, n. 15, p. 3688–3696, 2016. Elsevier.

YOO, C.; RAMIREZ, L.; LIUZZI, J. Big data analysis using modern statistical and machine learning methods in medicine. **International neurourology journal**, v. 18, n. 2, p. 50, 2014. Korean Continence Society.

ZAMARREN, M.; GONZA, P. A. Prediction of hourly energy consumption in buildings based on a feedback artificial neural network. , v. 37, p. 595–601, 2005.

ZHANG, H.; PARKER, L. E. CoDe4D: Color-depth local spatio-temporal features for human activity recognition from RGB-D videos. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 26, n. 3, p. 541–555, 2016.

ZHANG, L.; ZHAN, C. Machine Learning in Rock Facies Classification: An Application of XGBoost. **International Geophysical Conference**. p.1371–1374, 2017. Qingdao, China.

ZHANG, Y.; HAGHANI, A. A gradient boosting method to improve travel time prediction. **Transportation Research Part C: Emerging Technologies**, v. 58, p. 308–324, 2015. Elsevier Ltd.

ZHAO, ..; LIU, Z.; CHENG, H. RGB-depth feature for 3D human activity recognition. **Communications**, v. 10, n. 7, p. 93–103, 2013.

ZHOU, B.; KHOSLA, A.; LAPEDRIZA, A.; OLIVA, A.; TORRALBA, A. Learning Deep

Features for Discriminative Localization. **arXiv:1512.04150 [cs]**, p. 2921–2929, 2015.

ZHOU, C.; WANG, L.; ZHANG, Q.; WEI, X. Face recognition based on PCA image reconstruction and LDA. **Optik-International Journal for Light and Electron Optics**, v. 124, n. 22, p. 5599–5603, 2013. Elsevier.

ZHU, W.; LAN, C.; XING, J.; et al. Co-occurrence Feature Learning for Skeleton based Action Recognition using Regularized Deep LSTM Networks. **Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)**. p.3697–3703, 2016.

ZHU, X. **Semi-Supervised Learning with Graphs** Doctoral Thesis, 2005. Carnegie Mellon University.

APPENDIX A

Layer	Parameters					
conv1	Kernels (8)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv2	Kernels (16)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
MaxPool1	Pool Size (2x2)	Strides (2x2)				
conv3	Kernels (32)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
MaxPool2	Pool Size (2x2)	Strides (2x2)				
conv4	Kernels (16)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv5	Kernels (128)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv6	Kernels (32)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
MaxPool3	Pool Size (2x2)	Strides (2x2)				
conv7	Kernels (32)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv8	Kernels (256)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv9	Kernels (64)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
MaxPool4	Pool Size (2x2)	Strides (2x2)				
conv10	Kernels (64)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv11	Kernels (512)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
conv12	Kernels (128)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)
MaxPool5	Pool Size (2x2)	Strides (2x2)				
conv13	Kernels (64)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalization (True)	Activation (LeakyReLU)

conv1 4	Kernels (128)	Kernel Size (3x3)	Padding (same)	Input Shape (60x60x1)	Batch Normalizaton (True)	Activation (LeakyReLU)
Flatten 1						
FC1	Output Size (256)	Batch Normalizaton (True)	Activation (ReLU)	Dropout (0.5)		
FC2	Output Size (128)	Batch Normalizaton (True)	Activation (ReLU)	Dropout (0)		
FC3	Output Size (64)	Batch Normalizaton (True)	Activation (ReLU)	Dropout (0)		
FC4	Output Size (10)	Batch Normalizaton (False)	Activation (Softmax)			