REPUBLIC OF TURKEY

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technology

MACHINE LEARNING ALGORHTIMS FOR HEART RHYTHM CLASSIFICATION

Hussein Ali MOHAMMED

Master's Thesis

Supervisor

Prof. Dr. Galip CANSEVER

Istanbul, 2022

MACHINE LEARNING ALGORHTIMS FOR HEART RHYTHM CLASSIFICATION

Hussein Ali MOHAMMED

Information Technology

Master's Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled "MACHINE LEARNING ALGORHTIMS FOR HEART RHYTHM CLASSIFICATION" prepared by HUSSEIN ALI MOHAMMED and submitted on 13/06/2022 has been **accepted unanimously** for the degree of Master of Science in Information Technology.

Prof. Dr. Galip CANSEVER

Supervisor

Thesis Defense Committee Members:

Prof. Dr. Galip CANSEVER	Faculty of Engineering and Architecture, Altınbaş University	_____
Asst. Prof. Dr. Sefer KURNAZ	Faculty of Engineering and Architecture, Altınbaş University	_____
Asst. Prof. Dr. Yavuz Eren	Electrical and Electronic Engineering, Yildiz Teknik University	_____

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

Submission date of the thesis to Institute of Graduate Studies: ___/___/___

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Hussein Ali MOHAMMED

Signature

DEDICATION

First I would like to Allah Almighty for the power of mind, health, strength, guidance knowledge and skills to complete this study. This thesis is wholeheartedly dedicated to my beloved father, who have been my source of inspiration, he tells me that" every success in your life will be best gift for me". To my mother, who have been supporting me with the kind and pure love.

ABSTRACT

MACHINE LEARNING ALGORHTIMS FOR HEART RHYTHM CLASSIFICATION

Mohammed, Hussein Ali

M.Sc., Information Technology, Altınbaş University,

Supervisor: Prof. Dr. Galip CANSEVER

Date: June/2022

Pages: 58

The electrical activity of the heart is recorded via an electrocardiogram signal. Electrodes on the skin detect minor voltage changes induced by depolarization and repolarization of the heart muscle. The electric potential between two electrodes can be measured. An ECG lead is a pair of electrodes that are connected together. Each lead offers a unique perspective on a heart's electrical activity.The EEG signals have a significant potential of discovering various neurological illnesses in an early stage. We have proven why it is superior than MRI. Being affordable and efficient, it can be employed for various investigations even with a reduced budget as compared to MRI. Many Deep Learning algorithms have been implemented on EEG recordings in order to classify different neurological illnesses and brain interface applications. In this research we have done an exhaustive literature review on application of various neural networks on EEG data and how utilizing these neural networks one was able to categorize numerous neurological illnesses including Seizure, AD and Depression. The pre-processing phase considerably influences the performance of the model. In this study, we train different machine learning algorithms models on ECG datasets and compare their performance with each other. We found that at least in our experiment, both hand-crafted features and the feature extractors each have both their favourable and bad characteristics.

Keywords: EEG Signals, Machine Learning, Classification, ECG Datasets.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

ML	:	Machine Learning
NLP	:	Natural Language Processing
AI	:	Artificial Intelligence
NB	:	Naïve Bays
SVM	:	Support Victor Machine
LSI	:	Latin Semantic Index
GOM	:	Goals of Matrices

1. INTRODUCTION

Arrhythmia (abnormal heart rhythm) is one of the most frequent health disorders in the population. They can range in severity from mild palpitations to complete heart failure and death [1]. An electrocardiogram analysis is one of the approaches for diagnosing cardiac arrhythmia. For an accurate diagnosis of a cardiac problem, however, a skilled human specialist is required. Some arrhythmias, such as atrial fibrillation, need days or even weeks of continuous monitoring. Long-term cardiac monitoring signals, on the other hand, are too long to be adequately analyzed by a human, hence the analysis must be done automatically by a model. The expert simply has to look at bits of the signal indicated by the model when using automated arrhythmia classification.

The electrical activity of the heart is recorded via an electrocardiogram signal. Electrodes on the skin detect minor voltage changes induced by depolarization and repolarization of the heart muscle. The electric potential between two electrodes can be measured. An ECG lead is a pair of electrodes that are connected together. Each lead offers a unique perspective on a heart's electrical activity.

Machine learning [2] is the concept of a system learning, finding patterns, and making decisions without human involvement. It is a branch of artificial intelligence that may be classified into three groups based on the learning approach: supervised, unsupervised, and semi-supervised [3]. The presence of labels (goal values) for input data during learning distinguishes these types.

These measures allow all trained models to be compared to one another. As long as the problem is classified as a classification problem, four basic metrics, True vs. False and Positive vs. Negative output from the trained model, must be introduced initially. A part of this thesis compares several machine learning approaches on datasets, which are publicly accessible.

2. BACKGROUND AND LITERATURE REVIEW

2.1 ELECTROCARDIOGRAM

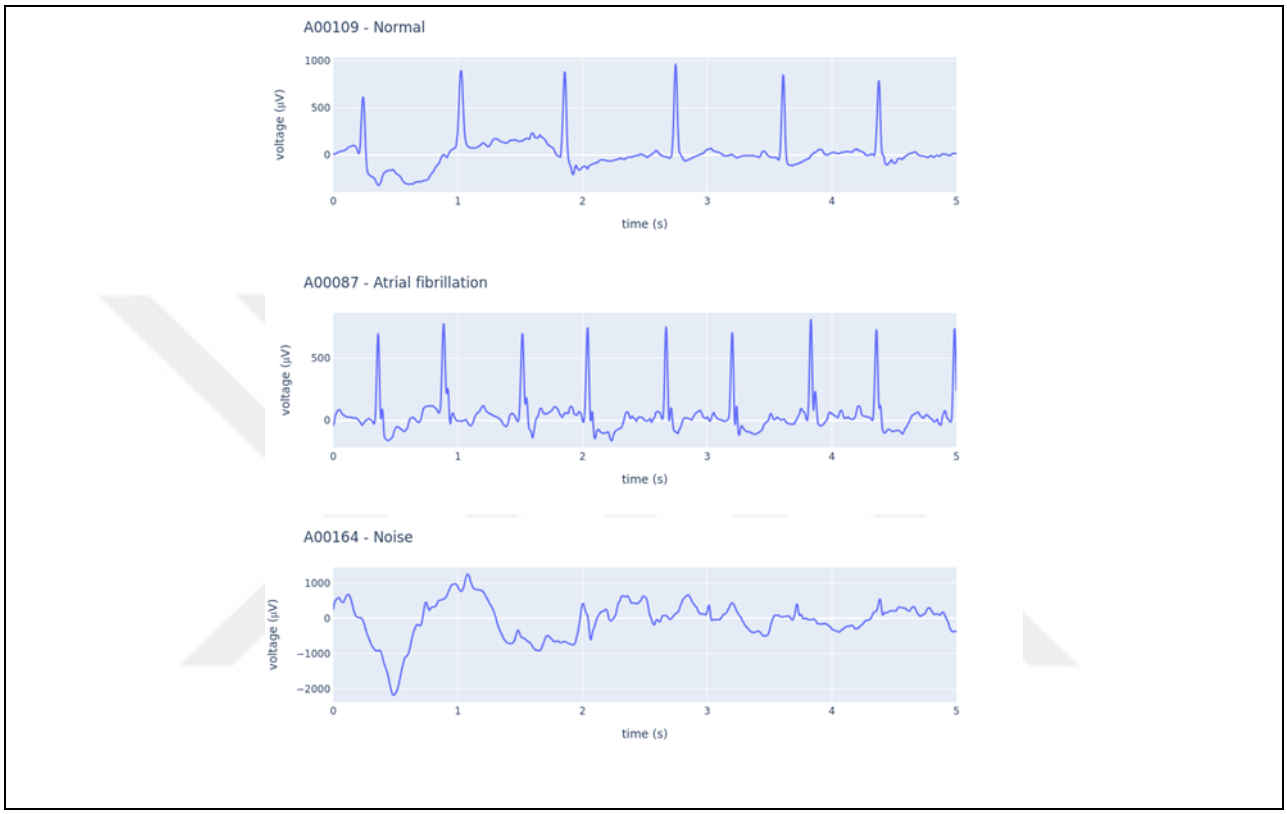


Figure 2.1: Electrocardiograms of Normal Rhythm, Atrial Fibrillation and a Noisy Signal (CINC2017 dataset [2]).

The electrical activity of the heart is recorded via an electrocardiogram signal. Electrodes on the skin detect minor voltage changes induced by depolarization and repolarization of the heart muscle. The electric potential between two electrodes can be measured. An ECG lead is a pair of electrodes that are connected together. Each lead offers a unique perspective on a heart's electrical activity. The 12-lead from 10 electrodes lead configuration is the most often employed in clinical practice. The focus of this study is on the 1-lead (two electrodes) ECG, which is commonly utilized in telemonitoring equipment.

2.1.1 Telemonitoring

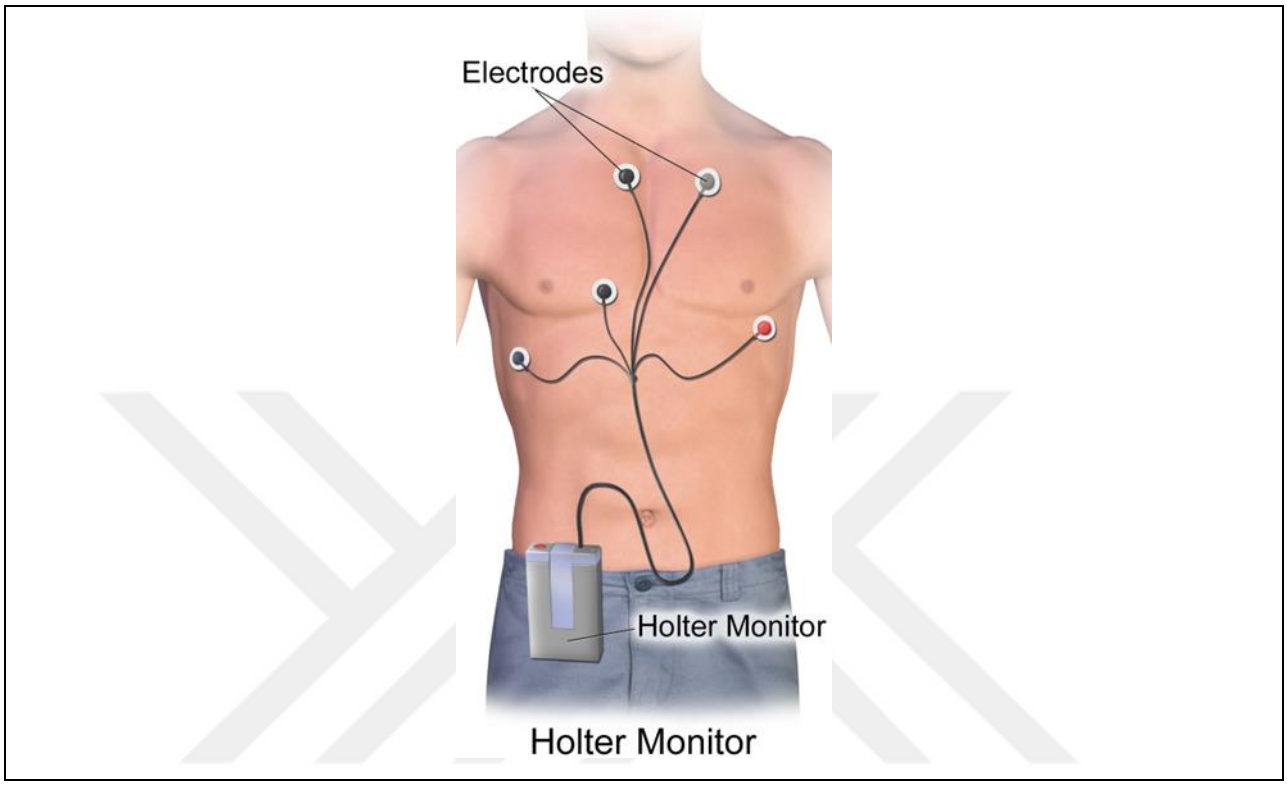


Figure 2.2: Holter Monitor, a Type of ECG Telemonitoring Device [3].

"The use of information technology to monitor patients at a distance" [4] is what telemonitoring is defined as. A telemonitoring device, in the context of ECG monitoring, is often a tiny, wearable device such as a Holter monitor (Figure 1.2). A telemonitoring device can assist cardiac patients avoid hospitalization [5].

2.1.2 Specific ECG Elements

We must detect the distinctive patterns in the ECG in order to diagnose cardiac arrhythmias. The P wave, QRS complex, and T wave are the most important components of an ECG.All these waves are shown together in Figure 1.3.

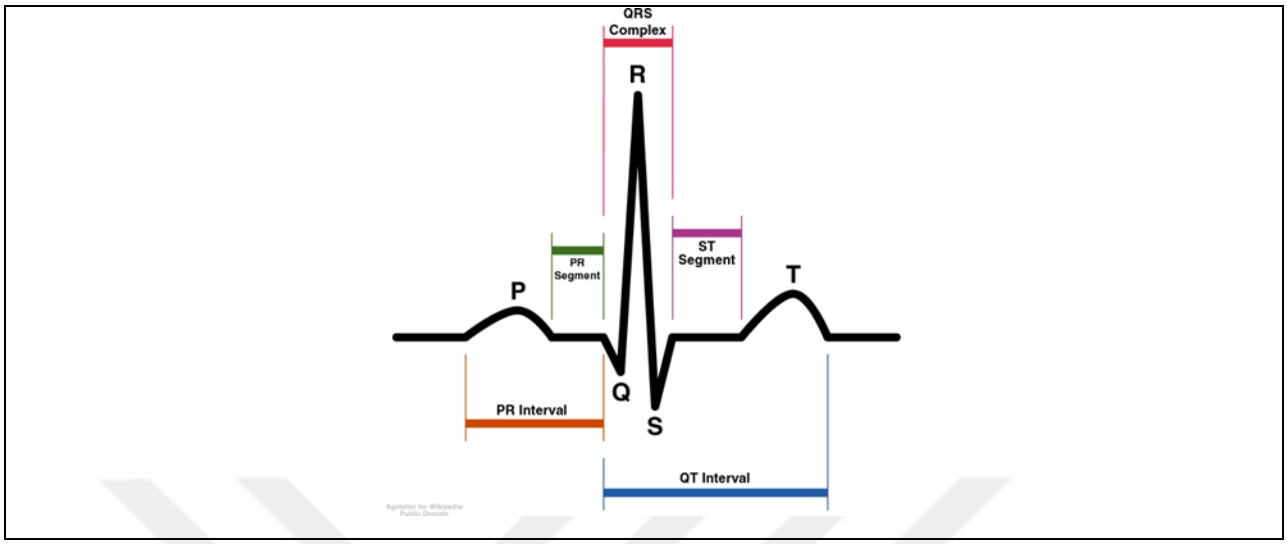


Figure 2.3: ECG of a Single Beat in Normal Sinus Rhythm [6].

The P wave represents the electrical depolarization and the re- sulting muscle contraction of both atria, the upper chambers of the heart. The QRS complex represents the depolarization of the ventri- cles, lower chambers of the heart. The horizontal ST segment follows, and the cardiac cycle ends with a broad T wave, which represents the ventricular repolarization [7].

2.1.3 RR Interval

RR interval, also called the inter-beat interval, is the time elapsed be- tween two consecutive R waves which can be directly used to calculate the average heart rate HR:

$$HR = \frac{60}{\text{RR interval in seconds}} \qquad (1.1)$$

where the RR interval is averaged across a number of successive R peaks in seconds.

The RR intervals may be used to derive a number of useful machine learning properties. The difficulty of analyzing inter-beat intervals is the focus of heart rate variability approaches.

2.2 SELECTED ARRHYTHMIA DESCRIPTIONS

To get a basic idea of the labels that are being classified in this work, a brief description of a few hearts arrhythmia follows.

2.2.1 Atrial Fibrillation

Atrial fibrillation (AF) is characterized by irregular heartbeats (RR interval duration is unstable) and the lack of a P wave preceding the QRS complex on the electrocardiogram (ECG). The most prevalent heart rhythm abnormality is atrial fibrillation (AF), which is hazardous to the patient [8]. Heart failure, stroke, coronary heart disease, and mortality are all increased by atrial fibrillation [9]. Figure 1.1 depicts an ECG of an atrial fibrillation patient. The distinctive uneven RR intervals may be seen in the image.

2.2.2 Premature Ventricular Contraction

A premature ventricular contraction (PVC) is a premature heartbeat caused by a lack of oxygen in the ventricles. The increased QRS on the ECG (Figure 1.4) and the compensatory gap following the PVC, which is generally longer than normal, make it easy to see. Even in healthy adults, single premature beats are common, but six or more PVCs per minute are deemed abnormal.[10].

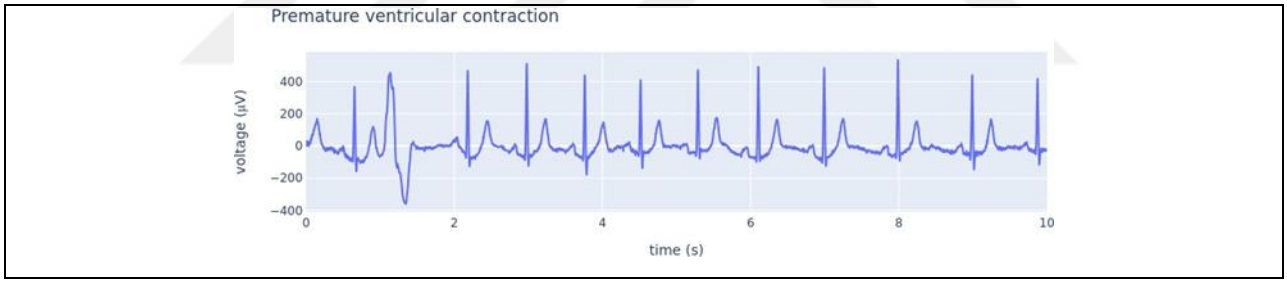


Figure 2.4: Premature Ventricular Contraction (CPSC2018/A0080) oc- Curing at 1.1 Seconds from the Start of the Recording.

2.2.3 Ventricular Tachycardia

A run of four or more premature ventricular contractions is referred to as ventricular tachycardia. Figure 1.5 depicts an example of this arrhythmia.

Ventricular tachycardia is a severe arrhythmia because it can lead to ventricular flutter and ventricular fibrillation, which necessitates rapid defibrillation due to the lack of efficient cardiac output [11].

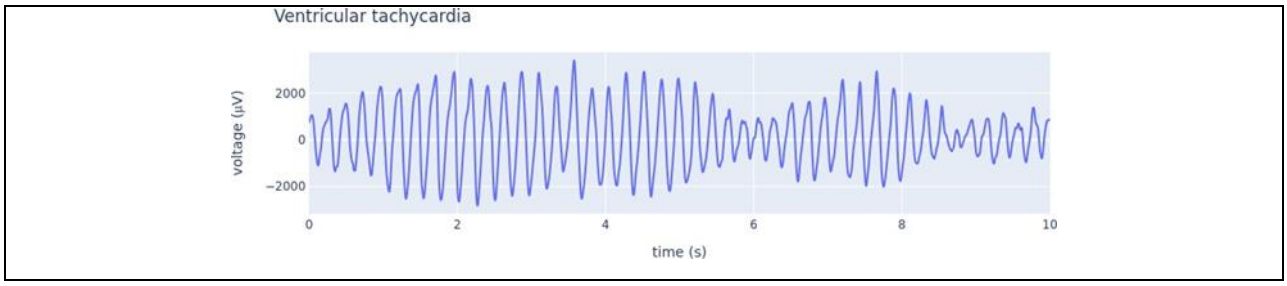


Figure 2.5: Ventricular Tachycardia / Fibrillation (CUDB/cu05) [12].

2.3 NEURAL NETWORKS

Artificial neural networks (ANNs) are inspired by the brain's organic neural connections. Artificial neurons (perceptron's) replace biological neurons in ANNs, which were initially described by Rosenblatt [53] in 1958. Multiple weighted inputs are added and transferred to an activation function like the sigmoid or threshold function in artificial neurons. The function of a neuron determines its output:

$$F(\_x) = \phi(\_x \cdot w\_), \qquad (2.1)$$

where _x is the vector of features, w_ is the vector of weights, · is a dot product operation and ϕ is the activation function. For a visual example, See Figure 2.2 for an illustration. The most frequent variant is to introduce bias, which is done by extending vector _x by one dimension and inserting 1 at the zero dimension. The vector w_ is likewise extended, so the bias is equal to the product of the initial elements in both vectors. The network consists of hierarchically ordered layers of neurons, The input and output layers are the first and final layers, respectively. There are hidden layers between the input and output layers. A hyperparameter is the number of hidden layers and the number of neurons in the hidden layers. The amount of features and predicted classes, respectively, determine the number of neurons in the input and output layers. The Multi-Layered Perceptron is a network that is made up of numerous layers of artificial neurons (perceptrons) (MLP). Neurons from the previous layer send output to the inputs of neurons in the following layer via learnt weighted connections at prediction time. The feed-forward pass is what it's called.

A random weight initialization kicks off the learning process. The network then tries to optimize Gradient Descent (GD) or its variations with regard to a loss function in order to compute weight changes for each batch of data. Goodfellow et al. [23] offer a backpropaga- tion approach for computing weight updates.

Except for the design of ANNs, the learning rate is the most important hyper-parameter. What proportion of updates is used in the GD is determined by the learning rate. A network with a high learning rate is more likely to vary about a minimum or even diverge, whereas a network with a low learning rate is more likely to converge slowly.

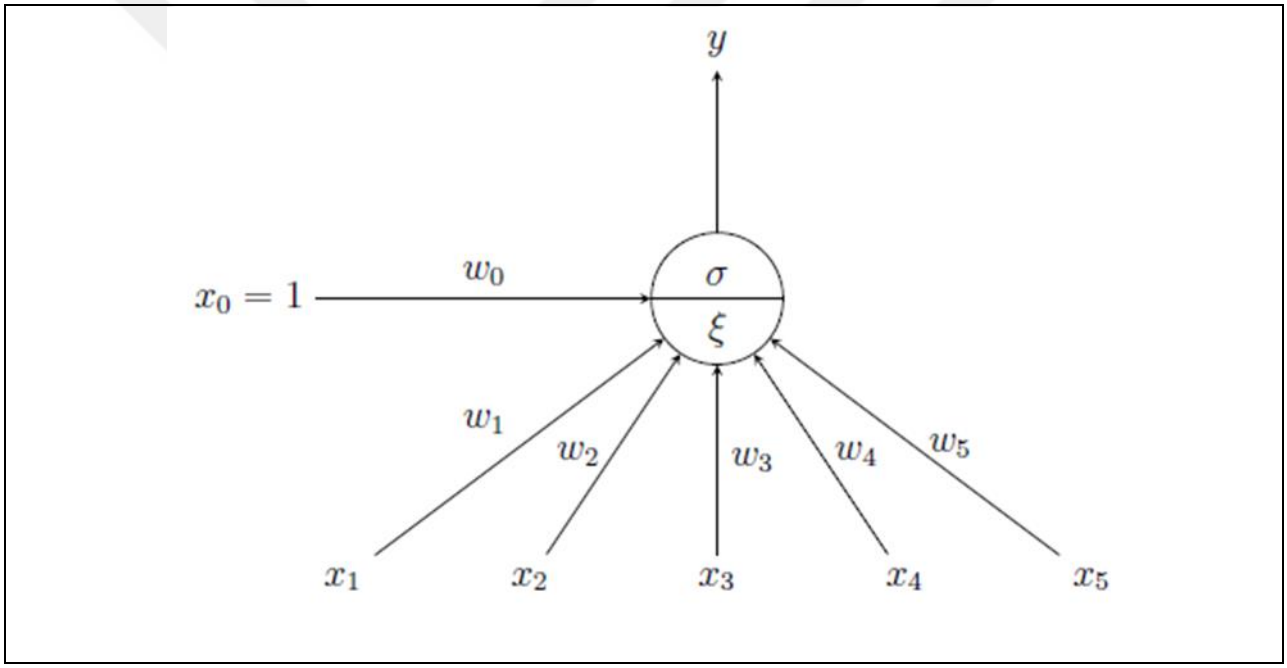


Figure 2.6: An Example Neuron Unit with Five Inputs and Bias

2.3.1 Deep Learning

The most significant moment and the beginning of Deep Learning using Convolutional Neural Networks was in 1995, when Yann LeCun and Yoshua Bengio took their steps from MLP and improved them by decreasing both the computing workload and the number of measurements. From that point forward, a publication was distributed [36] that demonstrates a more effective framework for acknowledgement by excluding the random variable.Then E. Hinton published a work in which he proposed Restricted Boltzmann Machines (RBMs) [37], which are used to filter, classify labeled and unlabeled data, and reduce data dimensionality, hence simplifying

computational tasks. Alex Krizhevsky and his colleagues utilized a convolutional neural network called AlexNet to win the ImageNet LSVRC-2012, 2010 (Large Scale Visual Recognition Competition) utilizing a train set of photos in 2012. more than one million high-resolution RGB label pictures in order to classify them into 1000 categories.

DL algorithms have improved and can now be used to solve a variety of tasks (image segmentation, detection, style transfer, analogies, and image classification). This advancement is due to the availability of large amounts of data (big data) and high-performance hardware capable of handling large computational tasks. Finally, in 2017, the study used a deep neural network and found that the error rate was 3%.

2.4 LONG SHORT-TERM MEMORY NETWORKS

Despite their widespread usage for sequential data, RNNs have certain drawbacks. RNNs, for example, are capable of remembering the past but not of keeping track of long-term dependencies. This issue arises as a result of the Vanishing Gradient and Exploding Gradient, which make RNN training challenging in a variety of ways. [31] goes into much information about gradient difficulties. Long Short-term Memory Networks, on the other hand, solve this problem (LSTMs, [32]).

LSTMs are a special kind of RNN that can learn long-term dependencies. This benefit enables the network to retain vital information for extended periods of time while forgetting less critical information. This is why, in jobs like translation, voice recognition, and even music creation, this method is now quite effective. LSTM networks vary from standard RNNs in that they feature a repeating cell structure with four inter-acting layers.

The basic LSTM cell is shown in Figure 3.10. In actuality, there are several of them in different forms. The cell's most critical components are the cell state (represented by c) and the hidden state (represented by h). The input e represents the input at a certain timestamp t. While can be seen, the condition of the cell may remain unaltered as it passes through the cell.

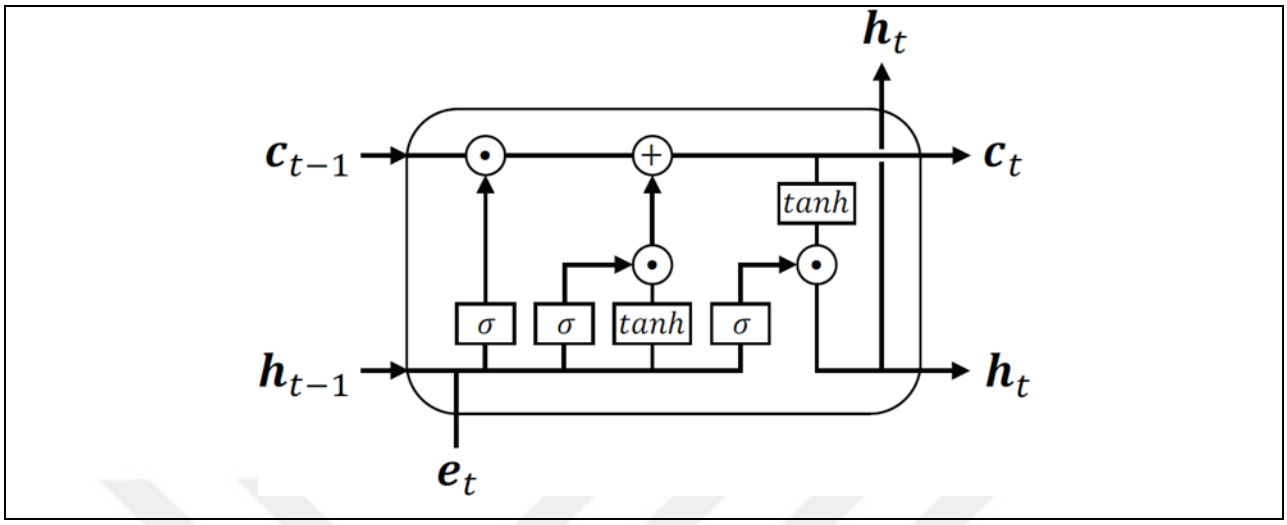


Figure 2.7: LSTM Cell of the Recurrent Neural Network [33]

Another important part of the cell are gates that manipulate with the cell memory. Gates are:

Forget gate – responsible for removing information from the cell state. It applies a sigmoid function σ for an input z and a hidden state h. Output of the sigmoid function is the vector with values from 0 to 1.

Input gate – responsible for the addition of information to the cell state.

Output gate – responsible for what we are going to output from the cell. This decision is based on the current cell state.

LSTM networks also have some drawbacks. They take longer to train and require more memory during training. The downside is the requirement for many training data; otherwise, there is a possibility of overfitting. Those problems arise mainly because LSTM networks have more parameters comparing with other neural network architectures.

2.4.1 Convolutional Neural Network

Fukushima [19], who called the CNN artificial neurons as neocog- nitrons, was the first to present them, and Krizhevsky et al. [36] popularized them with their AlexNet CNN design. CNNs are made up of convolutional, pooling, and ReLU layers that automatically extract important information and are usually followed by a fully connected layer for classification. VGG [58],

GoogleNet [61], and ResNet [24] are some of the most often utilized CNN designs. Many deep learning algorithms for semantic segmentation are built on top of these frameworks.

2.4.1.1 Convolutional Layer

Let the first layer's input be a 32-bit picture with three channels: red, green, and blue (see Figure 2.4a). Depending on stride (distance between sub-regions) and padding, we apply a single convolutional filter of size 3 x 3 x 3 by calculating the dot product with all feasible sub-regions (number of pixels placed around the image). As seen in Figure 2.4a, the outcome is a feature map.

Figure 2.4b shows the convolutional layer, which is made up of separate filters that generate a stack of feature maps. Parameter sharing refers to the sharing of weights across neurons in the same feature maps. The number of free parameters is reduced through parameter sharing, which takes use of the fact that the same feature might be helpful in several places.

Local connection is another crucial aspect in CNNs. In contrast to ANNs, where all neurons are completely linked, not all neurons are coupled to all inputs. Each neuron in a CNN links to a limited subset of input data; as a result, each neuron only sees a small portion of the picture rather than the whole image. This method keeps the image's spatial information.

2.4.1.2 Pooling layer

The pooling layer is responsible for non-linear downsampling. For example, max-pooling [51] takes the maximum value from each parti- tioned non-overlapping sub-region, see Figure 3.5.

2.4.1.3 ReLU layer

ReLU is the x max (0, x) activation function, see Figure 2.6. ReLU preserves non-linear properties and its derivatives needed for GD are easily computed. It was first used by Glorot et al. [22].

2.4.1.4 ResNet

One of the most significant advancements in CNN design was ResNet [24]. Figure 2.7 shows how it creates residual blocks with skip connections. There are two pieces to the skip connection: The first section has two stacks of convolutional, ReLU, and pooling layers, whereas the second section bypasses these levels and adds the original input to the outcome of these layers. The goal is to keep

inputs intact as they pass through the network2; as a result, a large number of layers may be stacked on top of each other. Figure 2.8 shows the comparison of VGG and ResNet.

2.4.1.5 Transfer learning

As backbone feature extractors, many of the following algorithms leverage the aforementioned CNN pre-trained models. Models can distinguish various low-level patterns by training on strong datasets like ILSVRC. By retraining (also known as fine-tuning) a network, models with their gained information may be moved to a new domain. Because training big networks from scratch takes substantially longer than retraining them, the notion of retraining is widely employed. Advanced approaches include freezing and unfreezing layers in a network to enable just certain layers to be trained.

2.4.1.6 Fully convolutional network

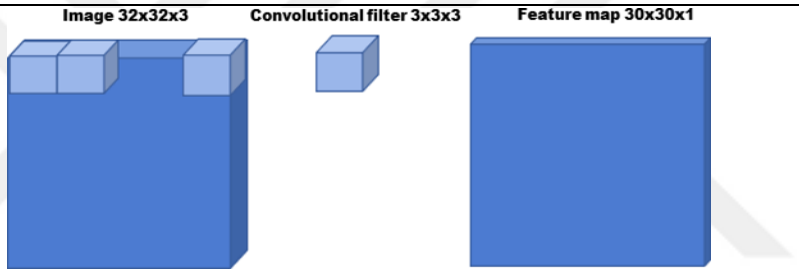
Shelhamer et al. [56] advocated replacing fully connected classification layers with convolutional layers with 1 1 filters in previously trained classification models such as AlexNet [36] and ResNet [24]. These layers minimize the number of feature maps (dimensionality reduction) while still providing pixel-by-pixel forecasts. However, the projections were lower than expected. The amount of shrinkage is determined on the stride and padding utilized. Figure 2.4b shows this impact with zero paddings and default stride.

The authors started by using linear upsampling to solve the issue. Another technique is to use skip layers: This method takes the last convoluted output, upsamples it to the output size of the second last pooling layer, and pixel-wise averages them. The result, as well as the third and final pooling layer, are treated in the same way. Skip layers is based on the premise that upsampling may be taught. Figures 2.9 and 2.10 show that the results are more fine-grained than simple linear upsampling.

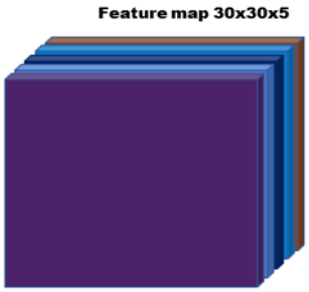
Shelhamer et al. [56] presented the Fully Convolutional Network (FCN) network, which obtained state-of-the-art results in PASCAL VOC and is considered a milestone in semantic segmentation. FCNs do, however, have several limitations: They don't take into account global background information, and inference takes a long time.

2.4.1.7 U-Net

U-Net provides a u-shaped architecture in Figure 2.12 by combining FCN skip layers with encoder-decoder upsampling. Encoders and decoders convert data from one domain to another. As an encoder, most image recognition CNNs may be employed. The network extracts features and builds a latent space representation in the encoder component. Figure 3.3 shows how the decoder conducts fractionally-strided convolution before concatenating output with the symmetric layer from the encoder. The size of the feature maps is increased via fractionally-strided convolution. Local pattern information is pre-served via feature maps transferred from the encoder to the decoder component. The ISBI challenge 2015 for medical imaging was the first time U-Net was used. After then, U-Net was employed in fields other than medicine.

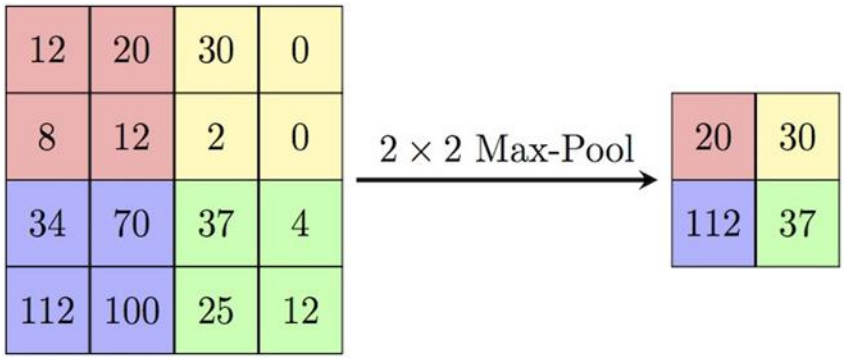


(a)When a single convolutional filter of size 3x3x3 (middle) is applied to the original image (left), the result is a feature map of size

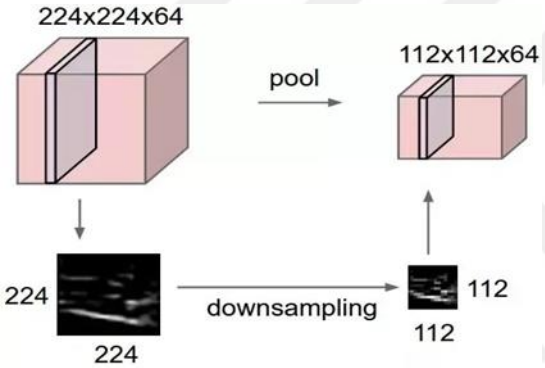


(b)The result of five convolutional filters applied to the original im- age is and image of size 30 × 30 with five feature channels

Figure 2.8: The Effect of the Convolutional Layer on the size of the data



(a)Max pooling from an original image with size 2 2, resulting in a down- sampled output



(b)The effect of pooling on an image

Figure 2.9: Max Pooling

3. MACHINE LEARNING ALGORITHMS

Machine learning [20] is the concept of a system learning, finding patterns, and making decisions without human involvement. It is a branch of artificial intelligence that may be classified into three groups based on the learning approach: supervised, unsupervised, and semi-supervised [21]. The presence of labels (goal values) for input data during learning distinguishes these types.

Several measures are used to assess the performance of machine learning models:

a) Confusion matrix.

b) Accuracy.

c) F1-measure.

d) Precision.

e) Recall

These measures allow all trained models to be compared to one another. As long as the problem is classified as a classification problem, four basic metrics, True vs. False and Positive vs. Negative output from the trained model, must be introduced initially. As an example, consider a binary issue with positive (1) and negative (0) classes. A true positive (TP) model output is one that properly predicts the positive class, whereas a true negative (TN) output is one that correctly predicts the negative class. A false positive (FP) is a model output that predicts the positive class erroneously, whereas a false negative (FN) is a model output that predicts the negative class wrongly [22].

The confusion matrix is the most often used measure, and it requires at least one input.

The first used metric, confusion matrix, takes as input at least two arrays, the predicted values for the data, and actual (original) values. As an output, it shows how many of predicted values by model belongs to categories TP, TN, FP or FN [23]. An example of a confusion matrix with two classes is shown in figure 3.1:

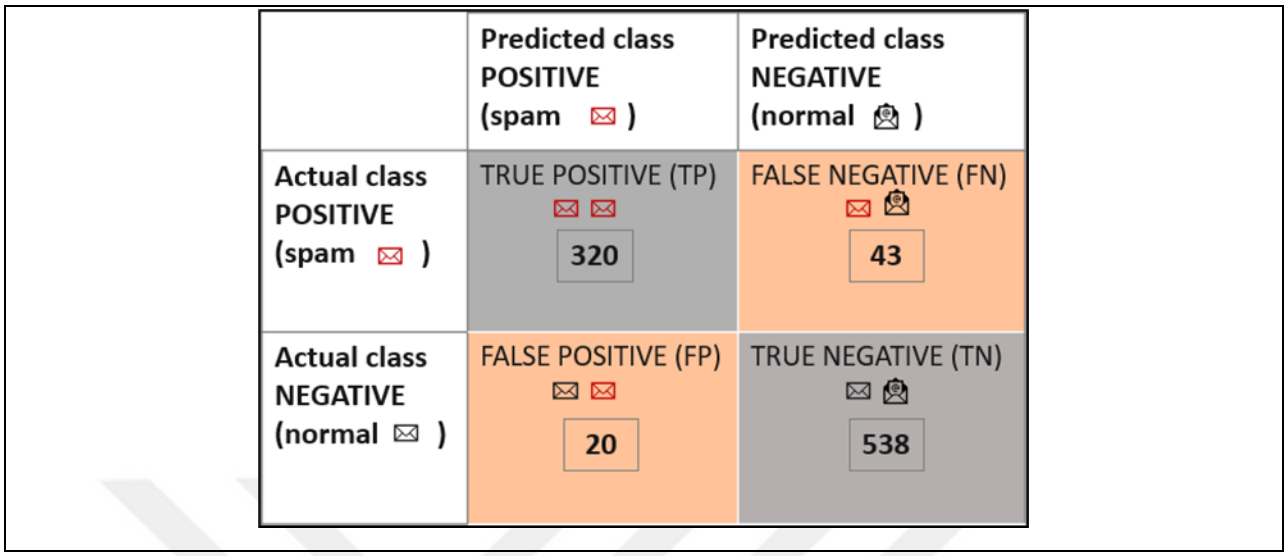


Figure 3.1: An Example of The Confusion Matrix with Two Classes [24].

The accuracy measure, which shows the perfect match between anticipated and original data, is the second most commonly utilized statistic. The accuracy may be described as: if the predicted value of the i-th sample is yi, the matching original value is yi, and the number of samples is n.

$$\text{Accuracy}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} 1(\hat{y}_i = y_i)$$

Precision, recall, and F-measure are some of the other measures that are employed. Precision refers to the model's ability to properly categorize only positive classifications. The recall measures the model's ability to correctly categorize all positive classifications. The F-measure is the harmonic mean of the accuracy and recall. Furthermore, if = 1, the value is referred to as the F1-measure, which indicates that both recall and accuracy are equally essential [24]. Equations demonstrate how to compute these values in binary classification.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_\beta = (1 + \beta^2) \frac{\text{precision} \times \text{recall}}{\beta^2 \text{precision} + \text{recall}}$$

In a multiclassification job, precision, recall, and F-measure can be applied to each class separately. The multiclass and multilabel classification part of reference [24] explains the complete equations with the appropriate notation.

3.1 SUPERVISED MACHINE LEARNING METHODS

learning from exemplars, also known as supervised machine learning, gives learning on a training data set with accurate replies (targets). the method generalizes such that it may reply appropriately to any input. as a result, all algorithms in the supervised category must first have accurate objectives (labels) for training data, which might be difficult to come by. we could store samples of every potential piece of input data into a large look-up database and eliminate the requirement for machine learning entirely. this isn't the situation in reality [25].

the thesis's purpose, however, is to compare the models' performance. the python package scikit-learn that was chosen provides a large range of algorithms, and it is impossible to discuss them all in the scope of this thesis. instead, the focus was on choosing at least the most important approaches from a different perspective, such as linear models and support vector machines.

3.1.1 Logistic Regression

A machine learning classification approach called logistic regression is used to estimate the likelihood of a categorical dependent variable. Linear regression may be transformed into logistic regression by using the sigmoid function [26]. The sigmoid function is represented by equation (3.1), and the linear regression function with vector w = (3.2) is represented by equation (3.2). (w0, ..., wp). The coefficients of the model are represented by the vector w. The bias or intercept is represented by w0, while the weight of the features is represented by (w1,..., wp) (x1, ..., xp). The sigmoid function returns a number between 0 and 1. Equation (3.3) may therefore be used to illustrate logistic regression, together with equation (3.3) for mapping (3.4). If the output of the sigmoid function is greater than 0.5, logistic regression will predict class 1. Class 0 will be expected otherwise [27].

$$p = \frac{1}{1 + e^{-z}} \tag{3.1}$$

$$z(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p \tag{3.2}$$

$$p(w, x) = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + \ldots + w_p x_p)}} \tag{3.3}$$

$$\hat{y}(w, x) = \begin{cases} 1 & \text{if } p(w, x) \geq 0.5, \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

Logistic regression is one of the best models for anomaly detection categorization. Logistic regression is a dependent variable-based approach. It is now one of the most popular and commonly utilized algorithms. Although the term regression appears in the name, this is a classification algorithm. The Python package scikit-learn implements one of the potential implementations under the class Logistic Regression [28]. When creating an instance, the class has a number of arguments to choose from. Optional regularization 1 is supported by the penalty parameter, which minimizes the cost function seen in equation (3.5). the equation is solved as a minimization of the default regularization 12. Elastic-Net is the last regularization that is supported which is combination of 11 and 12.The class LogisticRegression allows you to utilize alternative solvers through the solver parameter. Liblinear, newton-cg, lbfgs, sag, and saga are some of the ones that have been implemented. By setting multiclass to'multinomial,' the solvers may also be utilized for multi classification problems. The Broyden–Fletcher–Goldfarb–Shanno [29] optimization algorithm from the family of quasi-Newton techniques [30] with limited memory is known by the abbreviation lbfgs. Due to speed difficulties, it is only advised for small datasets.In the actual world, data is seldom separable by linear feature combinations. It is feasible to convert features into polynomial combinations in order to improve model performance by better fitting the data. With the help of a concrete example of two features, feature translation into polynomial can be easily described.

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1) \tag{3.5}$$

$X = [x_1, x_2]$ and degree 2. The transformed features into polynomial are $X = [x_1, x_2, x_1x_2, x2, x2]$.

3.1.2 Support Vector Classifier

The supervised machine learning approach utilized for classification is the support vector classifier (SVC). It's one of the Support vectors machines (SVMs) that the scikit-learn package has developed.

Vapnik created SVMs in 1992, and they quickly gained popularity due to their superior classification performance on datasets of reasonable size. They don't function well on really big datasets as long as they entail data matrix inversion, which is a computationally costly operation. The technique creates a hyper-plane, or a series of hyper-planes, that divides the dataset into classes. The hyper-plane is essentially a line that establishes a decision boundary in two dimensions and two classes. The margin is the distance between the closest point and the decision border.

SVMs are a class of algorithms that may be used for classification, regression, and outlier detection. The memory efficiency of these approaches is caused by a subset of training points in the decision function known as support vectors. Different kernel functions for the decision function are supported by SVMs. It's also possible to utilize a custom kernel function.

The scikit-learn library's class SVC [31] is a concrete implementation of SVMs. The technique will choose a new decision boundary, for example, a straight line or a polynomial one of the features, using the kernel function, which is changeable by parameter with the name kernel of the class SVC. Provided kernel functions [32] by SVMs in scikit-learn:

a) Linear.

b) Polynomial.

where degree of polynomial function is specified by parameter with the name degree of class SVC.

The gamma parameter of class SVC must be supplied for the Radial Basis Function. Gamma may be described as the kernel's spread (decision region). When the gamma is higher, the data points must be closer together in order to be influenced. Scikit-Sigmoid is a library that employs the sigmoid function as a kernel, commonly known as Hyperbolic Tangent Kernel [33].

The SVC approach, like the other supervised methods, uses an array X to hold the training samples and an array Y to contain the labels as inputs to the fit method. The array y in our situation just holds the values of the two classes.

$$gamma = \frac{1}{features\_count * X.var()} \quad (3.6)$$

$$gamma = \frac{1}{features\_count} \quad (3.7)$$

3.1.3 Decision trees, Ensemble Methods and Random Forest Classifier

Decision trees (DTs): For classification and regression, DTs are a non-parametric supervised learning approach. Learning basic decision rules derived from data attributes is used to develop a model that predicts the value of a target variable [23]. DTs are straightforward to comprehend, interpret, and depict. When predicting data with a decision tree, the cost is proportional to the quantity of data records required to train the tree. The substantial variance in the trees, which might be the result of slight alterations in the data, is one of the most major downsides of DTs. Use one of the ensemble approaches that will be discussed later as a solution. Overfitting is another issue that may be addressed by increasing the tree's maximum depth.

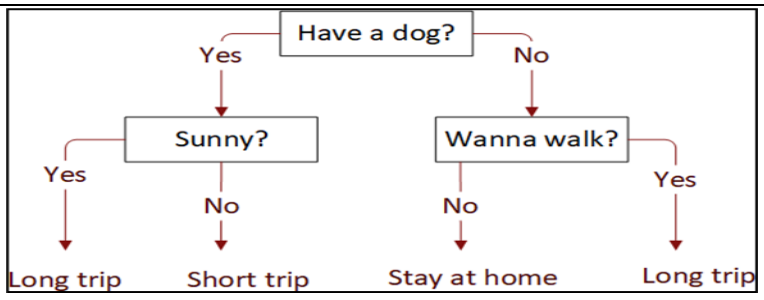


Figure 3.2: An Example of the Decision Tree.

Ensemble Methods: Ensemble methods combine the predictions of various machine learning algorithms to increase the robustness of a single prediction [23].

The majority of the literature divides ensemble techniques into two groups: boosting and bagging. Boosting is a method of creating a powerful learner by combining numerous extremely weak learners (estimators). Bagging is a simpler ensemble approach that combines many forecasts and selects the average of them [24].

The Random forest method is one of the randomized decision tree-based bagging techniques included in the scikit-learn learn toolkit. The Random forest method is a set of perturb-and-combine algorithms [33] tailored to trees. A varied group of classifiers is formed by injecting randomness into the classifier. The average forecast is created by combining the ensemble of individual predictions.

The Random forest classifier, on the other hand, is a concrete supervised implementation of the bagging techniques utilized in this thesis for training.

The Random Forest Classifier [34] class in the Python package scikit-learn may be used to implement the technique. The number of decision trees that will be built in an ensemble from the training set may be specified using the constructor option named n estimators. A sparse or dense array X storing the training samples and an array Y carrying the labels must also be included in the classifier. The optimal split of each node is discovered during the building of each tree, either from all input features or a random selection of size max features from the constructor's argument. The rationale for this is to reduce the forest estimators' variance. Parameter specifies the minimum number of samples necessary to divide an internal node with the name min samples split. The internal node [35] is recursively partitioned by a decision tree until the maximum permitted depth is achieved. Allow Q to represent the data at node m. A feature j and a threshold tm make up each candidate split = (j, tm).

3.1.4 Naive Bayes - Multinomial Naive Bayes classifier

The Naive Bayes techniques are a set of supervised probability-based approaches. They assume, in particular, that given the value of the class variable, every pair of characteristics is conditionally independent. The chance of acquiring the string of feature values (equation (3.8) is equal to the product of multiplying all of the individual probabilities (equation (3.9).

$$P(X_j^1 = a_1, X_j^2 = a_2, \ldots, X_j^n = a_n \mid C_i) \qquad (3.8)$$

$$P(X_j^1 = a_1 \mid C_i) \times \ldots \times P(X_j^n = a_n \mid C_i) = \prod_k P(X_j^k = a_k \mid C_i) \quad (3.9)$$

This assumption implies that the traits are unrelated to one another, which is not the case. This simplification, on the other hand, improves computing performance, and the algorithm produces results that are comparable to classification techniques in a given area. As a result, the Naive Bayes classifier's classification criterion is to choose class Ci for which the computation from equation (3.10) is the highest.

$$P(C_i) \prod_k P(X_j^k = a_k \mid C_i) \qquad (3.10)$$

Because conditional feature distribution may be computed individually as a one-dimensional distribution, the Naive Bayes classifier is quicker than other, more advanced approaches. When the number of dimensions grows, considerably more data is required, which is where this strategy comes in handy. The different naive Bayes' classifiers differ in their distribution of P (Xk = ak | Ci) according to assumptions they make [23]. Class Multinomial [36] is a concrete implementation for multinomial distributed data from the Python scikit-learn module. Because it is often used in text categorization [23], it was chosen.

3.2 UNSUPERVISED MACHINE LEARNING METHODS

It might be challenging to get labels for data in many cases. It may, for example, entail someone manually labeling the data. Unsupervised machine learning algorithms are an excellent choice in this scenario since they don't require any labels. Due to the lack of proper outputs, the algorithm must detect certain commonalities between input data points. The categorization might also be based on the detection of similarity between data items [24].

Unsupervised techniques, unlike supervised methods, cannot learn based on an external error criterion (the difference between objectives and accurate outputs). They are based on the fact that an aberrant instance generally appears as a remote outlier point from other examples.

So, the inputs that are closer together are recognized as similar (they are clustered together to the same class). In contrast, the inputs that are far apart are not recognized as similar (they are put into different classes if possible) [24].

3.2.1 K-means

One of the clustering algorithms, K-means [37], separates input data X into k disjoint clusters C with equal variance, minimizing the inertia, or within-cluster sum of squares, which is defined by equation (3.11).

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2) \qquad (3.11)$$

The mean (also known as the centroid) of the samples in the cluster is j, and xi is X. The means aren't input values, although they are from the same space [5].

The package scikit-learn provides a concrete implementation of the technique as a class K-Means [38]. It is possible to modify the number of clusters k using the n-clusters option. There are three steps in the K-means algorithm. The centroids of all clusters must first be initialized. The simplest method is to choose them randomaly , However, the scikit-learn library's K-means library provides a smarter approach to initialize them. The core idea is to arrange centroids' values to be far apart, resulting in demonstrably superior outcomes [39]. The second step involves assigning all data

points from X to the cluster's closest centroid. All centroids are updated to the mean value of all data points from X to each preceding centroid to reach a minimum result for equation (3.11). The difference between the new and old centroids values is calculated. The loop between the second and third stages is repeated until the difference between the new and old centroids values is less than the threshold.

The distance of each data point from X to centroids is often computed as Euclidean distance, but there are other alternatives also [24].

The method will always converge in general, however it may converge to a local minimum rather than the global minimum. It is very dependent on the centroids' starting values. The n-init option in the K-Means class allows you to change the number of times the algorithm runs with various centroid seeds. The drawback is that it may take a long time for the algorithm to complete. The method K-means from the scikit-learn library may be executed in parallel, making it quicker but at the expense of memory.

3.2.2 Gaussian Mixture

A Gaussian mixture model [23] is a probabilistic model in which all data points are derived from a combination of a finite number of Gaussian distributions with unknown parameters. The Gaussian distribution, often known as the Normal distribution, is a symmetric probability distribution around the mean. Data that are distant from the mean are less common than data that are close to the mean. An example of a probability density function is shown in figure 3.3, where indicates the mean.

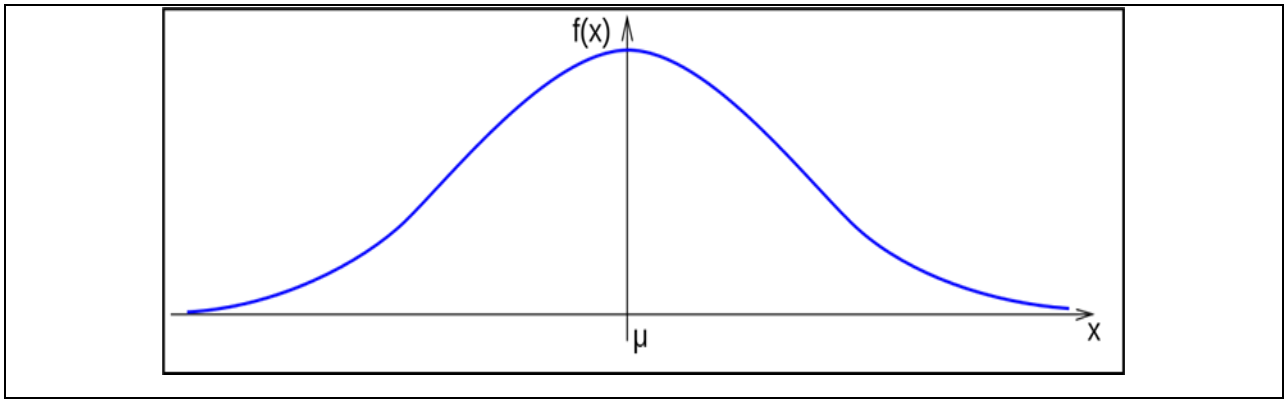


Figure 3.3: An Example of a Probability Density Function of GAUSSIAN Distribution.

The expectation-maximization (EM) approach is used to fit the Gaussian mixture object built from a class Gaussian Mixture [37] given by the scikit-learn toolkit. The EM method uses an iterative technique to solve the challenge of determining which points came from which hidden component. The EM method assumes random components in the first step, either randomly centered on data points or normally distributed about the origin. The number of components may be changed using the n components parameter of the class Gaussian Mixture. For each point, the chance of being created by each component is calculated. The parameters are tweaked in the second stage to increase the chance of the data given those assignments. The loop between these phases continues, and convergence to a local optimum is always ensured [38]. The class Gaussian Mixture allows you to change the number of times the algorithm runs with various centers of components (means) using the n init parameter.

3.3 SEMI-SUPERVISED MACHINE LEARNING METHODS

Semi-Supervised techniques were developed in response to two major drawbacks of both supervised and unsupervised approaches. The high expense of supervised learning on large amounts of data, as well as the restricted scope of applicability of unsupervised machine learning approaches. Semi-supervised algorithms only require a limited quantity of labeled data, with the remainder remaining unlabeled. So, in most cases, we utilize clustering (an unsupervised approach) to cluster a large portion of the data, and then label the rest using a supervised method [39].

3.4 GENERAL PROPERTIES OF ARTIFICIAL NEURAL NETWORKS

3.4.1 Nonlinear I/O Mapping

Analysis of high-dimensional data is becoming more crucial as sensor resolution and computer memory capacity improve. Examples include photos, voice signals, and multi-sensor knowledge. The examination of these signals shows them in a sample space as a whole or in part. A 16x16 sub-image for a single data point is an example of this. A point cloud in a 256-dimensional space is a picture collection. In most multi-sensor data sets, there is a high level of reliance between the sensors or between the sensor components. This is unquestionably correct for surrounding image pixels. However, no physical experiment can be expected to generate hundreds of degrees of

freedom of significant significance. As a result, multi-dimensional data sets will be represented by lower-dimensional representations. Various explanations might be useful in these presentations. This may be used to identify the data structure or query, to compare data points (for example, to locate the most similar in a database), or to get missing data values.

3.4.2 Ability to Generalize

The ability of neural networks to generalize is one of its key advantages. To put it another way, a qualified network will recognize data that it has never seen before. Real-world developers often only have access to a small part of all possible neural network patterns. The data set should be divided into three sections in order to obtain greater generalization: Getting ready for neural network testing During the preparation process, this mistake is reduced.

The validation kit is used to evaluate neural network output in patterns that haven't been trained. A check series for a neural network's overall efficiency. In the event of the smallest testing error, training will be suspended. The net generalizes the most in this stage. If you don't stop learning, you'll end up with extra training and worse net efficiency across the board, even if the error in the training data is still decreased. Finally, the network must be tested on the third dataset, the study collection, after the learning process is completed. Each training phase, ANNS conducts a testing procedure.

3.4.3 Tolerance to Faults (Graceful Degradation)

When other components (or one or more internal failures) fail, fault tolerance is a characteristic that allows the system to continue to function properly. In contrast to an artificial system, the fall in service efficiency does not correspond to the degree of the loss. Failure tolerance is particularly important in high-efficiency or life-critical situations. Graceful degradation is a term used to describe how a machine component breaks down.

If a device in a defect-tolerant design does not totally operate, the system will run at a lower level. When a program does not function properly. When the output is partially unsuccessful or the response time is prolonged, the notion is most often used to describe computer systems that tend to operate more or less totally. This implies that the complete machine is not blocked due to hardware or software failures. A automobile is built to keep running even if a tyre is punctured or

structural stability is maintained in the event of a disruption caused by fatigue, corrosion, manufacturing flaws, or collisions. Another example is the automobile vehicle.

Failure tolerance should be achieved in the preparation of unexpected scenarios and the design of a system to correct them inside a given system, as well as in the overall effort to stabilize the system so that it may be merged into a flawless one. Nonetheless, replication of any kind is best done when the consequences of an insufficient device, or the cost of making it strong enough, are significant. If a system failure is too severe in any circumstance, the application must be able to recover it in a protected state. The regeneration is similar to an inversion, but if humans are present, it may be a human activity.

3.4.4 An Analogy from Biology

Analogy is a cognitive process through which a single topic – analogy or context – is turned into a particular purpose or linguistic language. Unlike deduction, induction, and kidnapping, a less tightly defined inference or reasoning from one premise to another occurs when at least one hypothesis is universal. In the context of biology, the word analogy may also refer to the link between the source and the aim, which is sometimes but not always the same. The atomic theory of Rutherford connects the molecule to the solar system. Decision-making, discussion, interpretation, generality, memory, imagination, invention, foresight, empathy, description, conceptualization, and communication are all significant aspects of problem-solving. An analogic vocabulary, although not metonymous, incorporates examples, comparatives, and metaphors such as alligories and parables. The semantic core of comparison is comparison. This isn't a methonym, and it doesn't refer to any specific places, objects, or people. As if, words like, etc., etc., as well as the same word, rely on the recipient's corresponding interpretation of the letter. Analogy is important not just in everyday language and common sense, but also in science, philosophy, law, and humanism. The concepts of comparison, benchmarking, correspondence, the unity of mathematics and morphology, homomorphism, and iconicity are all closely related to analogy. The principle of analogy in semantic linguistics may be analogous to conceptual metaphor. All empirical statements and experiments whose outcomes are communicated to non-examined things are therefore founded on analogy.

3.4.5 Artificial Neural Networks Inputs Elements

This layer receives data, signals, processes, and measurements from the outside world. Within the bounds of the activation function, these inputs are generally standardized. The values of the template or sequence. This standardization improves the accuracy of the network's mathematical processes.

3.4.6 Weights

Numbers of "weights" or "parameters" are assigned to the relationship of neurons in artificial neural networks. As fresh information is supplied into the neural network, both weights change. The neural network understands.

An artificial neural network's weights are a rough representation of the numerous organic neuronal activities. Myelination, on the other hand, is crucial. Myelination. Myelination. Weights in artificial neural networks may be either positive or negative. Weight: the number of post-synaptic terminal neurotransmitter receptors, as well as the development and integration of vesicular and pre-synaptic terminal neurotransmitters, is equal to the increased combination of neuron dendrites, dendrite synapses, and the number of post-synaptic terminal neurotransmitter receptors. Weight:

Synaptic weights release anticipatory neurotransmitters, and positive weights are equivalent to pre-synaptic terminals (i.e. glutamates). The possibility of activation is increased by the receiving cell.

The negative weight is similar to the synapse of a neurotransmitter (i.e. GABA). There is a negative weight. The receiving unit would not fire if an intervention was likely.

Myelination: As myelination occurs, the disparity between movement and axon reduces. Because the membrane is not myelinated, its stress potentials are much lower than those of the cell body. An comparable pair of greenhouse pants. The garden pants leak when the axon does not myelinize, and the pants are put to an end by a decreasing water pressure (which is important for water control, the possible effect).

3.4.7 Function That is Additive

When a and b are compressed, an additive function in a positive integer is an arithmetic f(n) function that is the sum of its functions. Despite the fact that the positive integral components of f(a)+f(b) are not co prime, f(a)+f(b) offers a totally additive function. The function F / f(b) is complete additive. In this context, the comparison of fully multiplied functions to fully additive functions is common. F(1)=0 if f is a completely additive feature. An additive, on the other hand, is a completely additive ingredient.

3.4.8 Activation Feature

The threshold or transition functionality is also known as the activation function. The functions that activated neuron stimulation have been converted into output signals. Activation functions for neural networks are accessible in a variety of forms. Identity, phase, component linear function, and sigmoid function are examples of certain function forms.

3.4.9 Learning That is Supervised

Training sets are provided for guided teaching. This law covers a wide range of examples of a network's proper behavior. The expected results are derived once the inputs are supplied as a guided learning lesson. Parameters will be gently adjusted with the error signal during this investigation; parameters will be slowly updated with the error signal.

The learning rule covers a wide range of scenarios (established trainings) as well as optimal network behavior.

x1,d1; x2,d2; xn,dn; xn,dn; xn,dn; xn,dn; xn,dn; xn,dn; xn (3.7)

The network input is xn in this example, and the target input is dn. The output is determined by the input. The research rule is used to alter network biases and weights in order to improve the reliability of network outputs.

We engage in supervised learning so that when the input is submitted, we can offer the computer the appropriate answer (d). The disparity between the actual response and the predicted answer is used to externally adjust the network parameter. This error may be utilized to alter the weight, for

example, in the analysis of input patterns or instances when the answer to the error is known. The learning mode requires the training collection, as well as numerous input and output models.

3.4.10 Learning Without Supervision

Unexpected studying also makes use of self-organized analysis. Objective output is impossible to achieve in unrestrained learning. Weights and biases are solely impacted by network data in this situation. For pattern rearrangement, unattended research classification is utilized. Because unattended learning does not know what response is required, particular mistake information cannot be utilized to adjust network behavior. There is no such evidence to rectify inaccurate replies, thus research must be conducted based on experiences with omitted or unfamiliar responses to outcomes.

Unchecked learning algorithms rely on duplicate row data that lacks a mark for class or club membership. To define its parameters, the network must first identify any existing structures, objects, rules, and so forth. Because the teacher is not necessary, but must establish objectives, studying without attendance implies studying without the instructor. The importance of receiving input on neural networks cannot be overstated. Feedback is referred to be incremental learning, and it is critical for unrestricted learning.

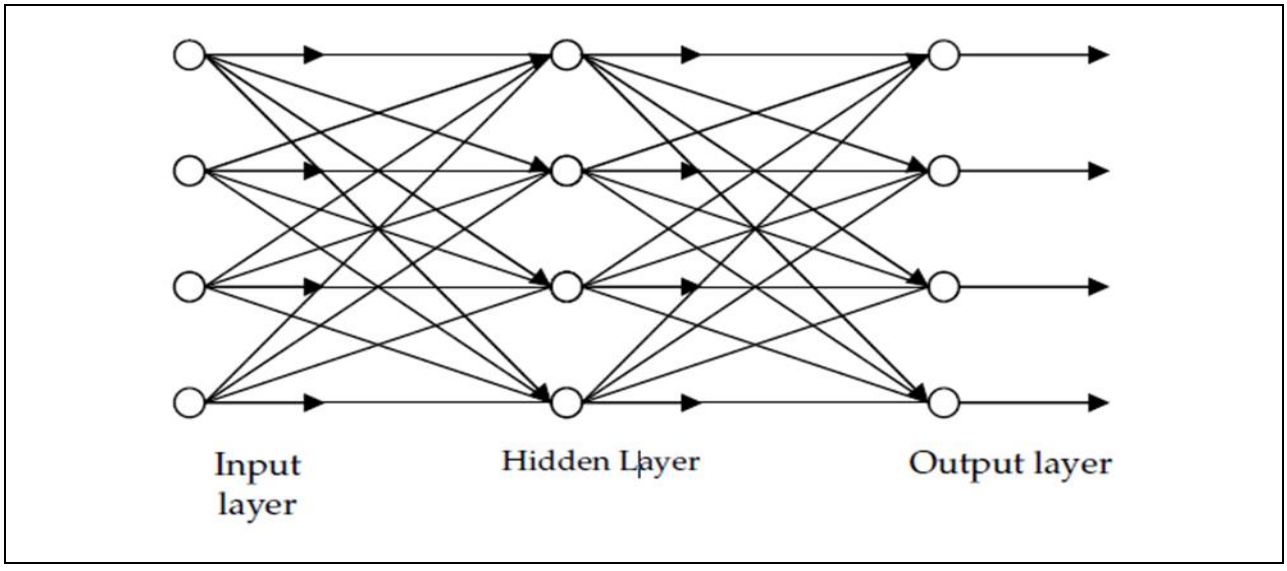


Figure 3.4: Multilayer Feed Forward Network [2].

Artificial neural networks according to learning algorithms When a network for a specific program is configured, it is ready to be created. The initial weights are automatically chosen at the beginning and beginning of instruction. There are two approaches; regulated and unattended.

3.5 SUPERVISED INSTRUCTION

The inputs and outputs for supervised instruction are provided. The results are compared to the information sent through the network. Errors in network weight change are produced by the system. As weight is altered on a regular basis, this cycle continues. As the weight of connections is raised during networking instruction, the same data collection is processed many times. The collecting of test data is referred to as a "test session." It is often difficult for a network to comprehend. Because the data used to create the intended outcome does not include any unique information, this might happen. If data is insufficient to enable thorough analysis, networks will not converge. In an ideal world, enough data would be accessible to save part of the data as a backup. Data will be stored by many nodes over several tiered networks. The data would be held in the monitored testing until the system was certified to track the network, at which point it would be decided whether the system just saves the data.

If the network cannot handle the issue easily, the designer must consider inputs and outputs, number of layers, layer numeric components, layer-based networking, transition, and training features and starting weights. A crucial component of the designer's imagination is governed by the rules. Several rules (algorithms) are utilized throughout the training session to allow for the appropriate weight change input. The most prevalent approach is back-propagation. This isn't just a mechanism to prevent network overtraining; it's an intentional decision. The broad statistical patterns of the findings are the artificial neural network's initial set-up. Then you learn about certain data points that are generally incorrect.

The weights will be frozen on request after the machine has been properly trained and no more testing is necessary. This network will subsequently be converted to hardware for additional devices. During development, many gadgets do not lock in, but instead continue to learn.

3.5.1 Adaptive or Unsupervised Training

The other kind of instruction is unrestricted (learning). The network can use the inputs, but the outputs aren't required. The gadget must then specify its own functions in order to arrange the input data. Sovereignty or adaptability are other terms for this. External influences have no impact on the weighting of such networks. Rather, their success is continuously monitored. The networks look for regularity or patterns in the input signal and adapt them to the network's functioning. The network also tries to figure out how to organise itself without knowing whether it is correct. The topology and analytical rules of the network make use of such features. The cooperation amongst the computing unit clusters will be stressed by an uncontrolled learning algorithm. Clusters will work together in such a strategy. The overall cluster functioning may be enhanced when an external input triggers a cluster node. By increasing the external input to the cluster nodes, this might potentially disrupt the whole network. Concurrence between processing components might also help with comprehension. Individual groups' reactions to unusual opportunities may be boosted through competitive cluster preparation. As a result, such classes would be connected and yield a different response. When the training competition is a success, the weight of the winning processing item is usually altered. Unattended education is not well-known, and further research is needed.

3.5.2 Laws of Learning (Algorithms)

There are a variety of additional frequent uses for learning rules. The most popular and oldest version of 'Hebb's rules' is the varying majority. Hebb's Law: This was first mentioned by Donald Hebb in the Reliability Company. The weight will be raised if the neuron comes from another nerve and all nerves are extremely strong (the same sign). Hopfield Law: Increase the weight of communication across the analysis rate, regardless of whether the outcomes are active or inactive inputs. The Delta Rule: the goal is to adapt the intensity of the input connections to the gap between the anticipated result and the actual output on a continual basis (delta).

The Delta Law is similar to the Delta Law in that the Transfer derivative function is always utilized to adjust the delta error by utilizing linking weights. However, a relative extra constant linked with the learning rate follows the final weight, which is dependent on the shift factor. Kohonen's Law: This aids in obtaining or changing the weight of items. The participants and their neighbors will

be distracted and passionate, and the winner is intended to be the best item. Only the winning group's weight may be adjusted, and it can only be changed by the winner and his or her neighbors.For the determination of classifier efficiency, sensitivity, septicity and accuracy are favored statistics. Susceptibility is the rate of estimation, particularly for the sake of healthy people and precision. for patients with epileptic diseases. Fair.-Equality. The statistics are calibrated with (3.8) and (3.9) and (3.10) respectively.

$$Sensitivity = \frac{TP}{TP+FN} \quad (3.8)$$

$$Specificity = \frac{TN}{TN+FP} \quad (3.9)$$

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (3.10)$$

3.6 TRAINING OF ARTIFICIAL NEURAL NETWORKS

The neural biological systems in which animal brains are constructed are known as artificial neural networks or connecting mechanisms. These systems, for example, generally learn functions without requiring complicated task programming (a step towards efficiency improvements). You can learn to detect photographs with cats by looking at flames that are manually marked with 'cat' or 'no-cat,' but you won't be able to distinguish the cats in the other pictures if you use the tests. Alternatively, students may assemble their own set of specialized characteristics from the learning materials they have on hand. An artificial neural array is coupled to the unit or node in animal brains to form the ANN (similar to biological neurons). Artificial neurons will be signaled by each contact (similar to a synapse) between them. The artificial neuron that receives the signal will begin to signal the related artificial neurons. For a typical ANN implementation, the signal is a real value in the interaction between artificial neurons, and the output is defined by the nonlinear input sum function of each artificial neuron. Artificial neurons and interactions, for the most part, measure according to experience. The weight changes the signal power of the link. An artificial neuron threshold will be identified only if the aggregate signal surpasses this threshold. Artificial neurons are generally structured in layers. The inputs will be transformed into various formats by several levels. After passing through the layers, signals are sent from the first layer to the final layer (output), perhaps many times.The ANN approach was created to solve issues like those involving the human brain. The focus Shiftedn throughout time to balancing those analytical

talents that contribute to biodiversity. Computer vision, voice recognition, machine translation, social network filters, video games and playboards, and medical diagnostics using ANNs are among the activities carried out. This section shows how to use BP and GA algorithms to increase the efficiency of an ANN.

4. EVALUATION

4.1 INTRODUCTION

In general, the whole process of data science as shown in figure 4.1 , or in this case, a machine learning experiment follows an all-purpose template, which can be outlined in the following seven steps:

a) Problem Definition

b) Dataset Collection.

c) Dataset Pre-Processing

d) Feature Extraction

e) Model Training

f) Evaluation

g) Classification

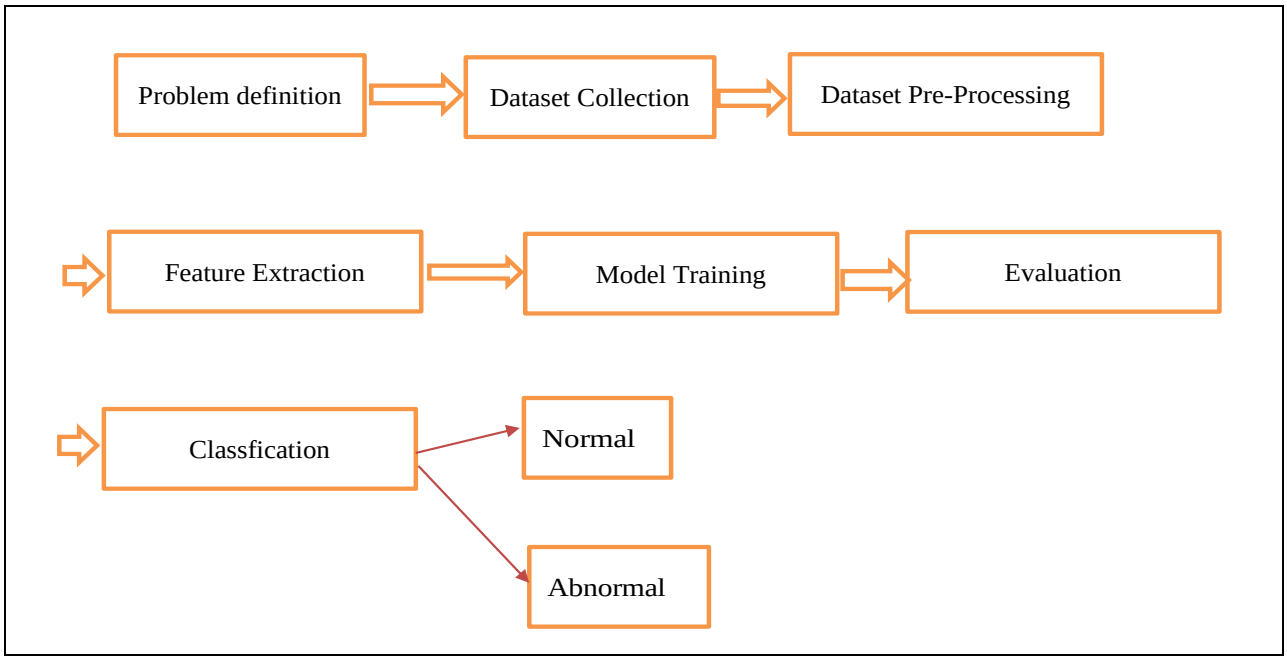


Figure 4.1: Research Methodology.

The first two steps have already been addressed in the previous chapters. Therefore this chapter will describe methodologies used for the remaining five. An important note to make is that although these steps are generally realized in a similar order to the one given above, it is common that individual steps are intertwined to some degree.

4.2 DATASET

For The China Physiological Signal Challenge (CPSC) 2018 [42], a collection of 12-lead ECGs was made available. There are 9831 recordings from 9458 distinct patients in all. Sixteen of them are accessible as a training set, with 6877 being one of them. At 500 Hz, the recordings are sampled. The recording lasts anything from six to sixty seconds. There are nine different classes in all. We do not utilize the 477 multi-label recordings (the patient had more than one cardiac ailment); some of our models are not suitable for multi-label categorization. Because it is identical to a single-lead ECG, we only used lead I. Moreover, the winners of the challenge achieved only slightly worse performance when using a single lead for classification [43].

Table 4.1: CPSC2018 Labels[42]

Label	Condition	Count
Normal	Sinus Rhythm	918
AF	Atrial Fibrillation	1098
I-AVB	Atrioventricular Block	704
LBBB	Left Bundle Branch Block	207
RBBB	Right Bundle Branch Block	1695
PAC	Premature Atrial Contraction	556
PVC	Premature Ventricular Contraction	672
STD	ST Segment Depression	825
STE	ST Segment Elevation	202
Total		6877

4.3 DATA PRE-PROCESING

4.3.1 Resampling

By resampling - modifying the sampling rate of the signal – we may lower the dimensionality of the signal without sacrificing too much of its quality. Due to the fact that the majority of cardiac information is contained in lower frequencies around 50 Hz, we can still observe the ECG clearly even when utilizing a relatively low sampling rate.

If we look at dataset we can see the second recording, which has been resampled to 100 Hz, keeps the quality of the signal, and the distinct segments of the ECG are plainly distinguishable from one another. Using a sample rate of 10 Hz is insufficient; portions of the recording will always be lost.

We did not use any resampling in the feature extraction process. Some features, particularly those in the frequency domain, benefit from a higher sampling rate than others. It is preferable to have a sample rate of 250 Hz or higher. It is permissible to use a frequency of 100 hertz for temporal domain analysis [16]. As a result, we used a polyphase filter to resample all data for neural network-based approaches to a frequency of 100 Hz. We used the resample poly function from the SciPy Python library [17] with the default settings to sample polygons.

4.3.2 Band-Pass Filtering

Because the majority of the signal's significant information is contained inside a restricted frequency spectrum, we may suppress the frequencies that are outside of the band in the signal. A digital band-pass filter may be used to do this, which eliminates both the extremely low (0–0.5 Hz) and the very high (>40 Hz) frequency noise.

For datasets that provide no information regarding filtering, we preprocess the signals by applying a band-pass filter on the signals in the frequency range of 0–40 Hz. We perform zero-phase filtering using a fifth-order Butterworth finite impulse response filter to get the desired result. For the purpose of filter design, we make use of the SciPy Python module.

4.4 FEATURE ENGINEERING

R peak identification is accomplished with the usage of the Python physiological signal module Neu- roKit2. It is compatible with a wide range of R peak detection methods. We picked the NeuroKit custom approach because it can extract R peaks even from noisy inputs, which is what we were looking for. According to the classic Pan–Tompkins method [19], there are no true R peaks detected in the noisy data shown in the figure.

Human heart rate variability (HRV) is a term used to describe the fluctuation in the time between heartbeats (RR intervals). Heart rate variability characteristics may be divided into three categories: time domain characteristics, frequency domain characteristics, and non-linear characteristics [22]. It is our intention to make use of the Neurokit2 package, which is self-described as "the most complete program for calculating HRV indices" [23]. In NeuroKit2 version 0.0.40, the library can calculate 19 time-domain features, 11 frequency-domain features, and nine nonlinear features, in addition to many other capabilities. All HRV techniques use as inputs the locations of R peaks, which are the same for all.

4.5 TRAINING AND TESTING

All of the datasets were divided into train and test partitions, which were the same for all of the models. The models are trained on the train partition, and the final assessment is performed on the test partition, which the model has never seen before, in order to assess the model's generalization capacity. We segregate a hold-out validation set from the training set for neural network-based models to evaluate learning performance during training.

For the train and test partitions, respectively, 90–10 percent of data is partitioned. The ultimate split of train, validation, and test partitions is 80–10–10 percent when employing the validation partition. The same train-test split is used to train and assess all of the models. The train-validation-test partitions are separated using Group Shuffle Split from scikit-learn by the telemonitoring device ID because the private datasets include a high number of recordings from the same patients. As a result, the same patient's record should not appear in both the training and test partitions.

The k-nearest neighbors (k-NN) technique is one of the most straightforward supervised learning algorithms available. The essential concept is that fresh instances belong to the same class as their closest neighbors. The most significant parameter is k, which indicates how many neighbors have an impact on the categorization. It is also critical to consider the statistic that was employed when comparing the samples. A number of regularly used metrics are available, including but not limited to the Makowski distance, correlation, cosine similarity, and dynamic time warping [50].In terms of implementation, we utilized the KNeighbors TimeSeries-Classifier with its default dtw measure from the tslearn Python package, which is available on the tslearn website.

4.5.1 Time Dilation In Real Time

When it comes to identifying arrhythmias from ECG data, we're dealing with a time series classification challenge. Time series classification is a well-established study subject, and there are several techniques available to attempt to tackle the challenge [51]. However, the majority of the methods are assessed on tiny datasets (hundreds of records) derived from The UCR (University of California, Riverside) Time Series Archive [52] [52]. Dynamic time warping (DTW) is widely regarded as the gold standard for determining the similarity of time series. In [51], it was shown that a 1-NN classifier with DTW distance is a very effective time-series classification approach.

4.5.2 Cross-Validation

Cross-validation is an alternate method of assessing a model's generalization performance. The K-fold cross-validation method divides a dataset into K partitions (folds), then trains the model K times, each time training on the first fold and assessing on the last. The validation score may be calculated by averaging the K scores.

4.6 METRICS

On the same data, we must assess the performance of several models. A confusion matrix contains two rows and two columns for a binary classification issue with two classes, such as Pos and Neg. True positives are labeled as Pos and categorized as Pos, true negatives are labeled as Neg and classified as Neg, false positives are labeled as Neg and false negatives are labeled as Pos and classified as Neg.

Table 4.2: A Binary Classification.

True Positives (TP)	True Negatives(TN)
False Positives (FP)	False Negatives (FN)

A confusion matrix for a multi-class problem has shape $N \times N$, where N is the number of classes. A single score instead of a full matrix is preferable for comparing the classifiers' performance. We calculate the F_1 score for all classifiers and use it for a rough comparison.

4.6.1 F_1 Score

The F_1 score is the harmonic mean of precision (positive predictive value) and recall (sensitivity) [68]:

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \tag{4.1}$$

where

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{ and recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4.2}$$

F_1 score is a simple way to compare classifiers. It favours classifiers with similar precision and recall; this is not always what we want when our task is classifying heart pathologies.

We use F_1 score for simplicity, but some types of arrhythmias are more dangerous than the others. For example, ventricular tachycar- dia is life-threatening as it can lead to deadly ventricular fibrillation. Therefore, a metric that takes into account the severity of different classes could prove useful in future work.

4.7 RESULTS

This section provides a summary of the results and demonstrates some of the weaknesses in the presented approach and suggests possible improvements. We used precision, Recall, F-Measure and accuracy to evaluate each algorithm.

the overall results were as below:

In term of Precision, the analysis results show that SVM and logistic regresion has the heights (Figure 4.2):

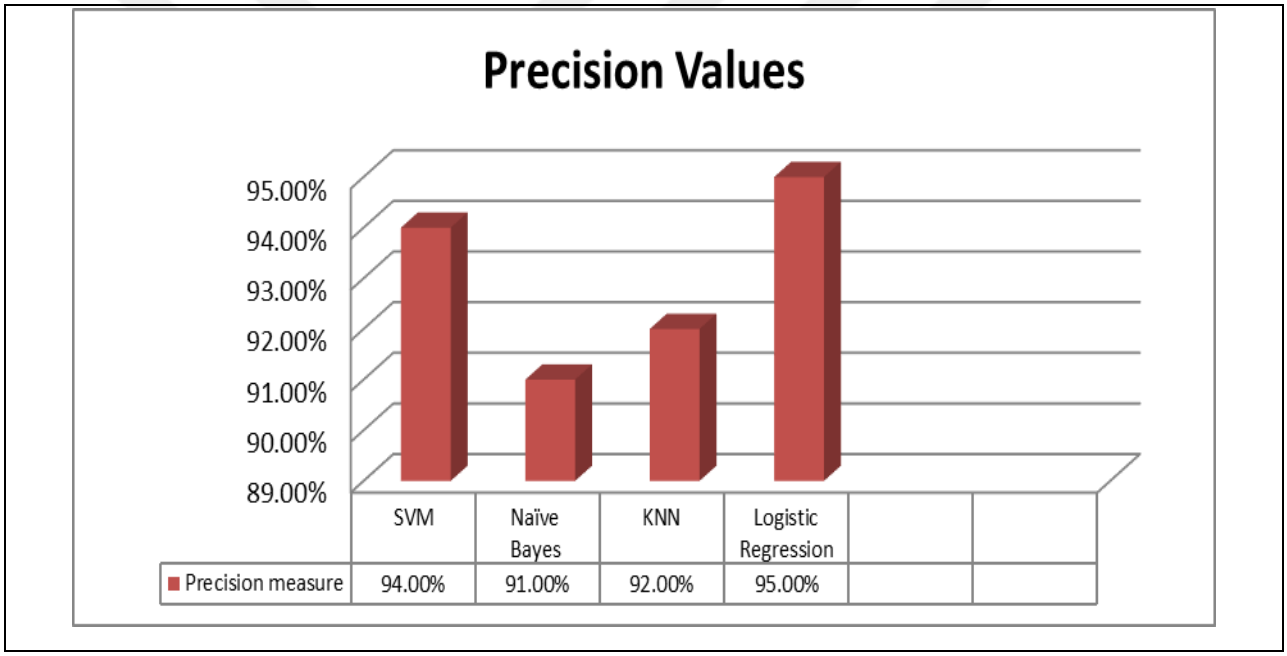


Figure 4.2: Precision Values of All the Classifiers.

In this figure chart, we show the precision Metric for our algorthims, as we can see the SVM and logistic regression came with best results for our model ,the SVM algorthims came with 94% and logistic regression 95%.

In term of Recall, the analysis results show that KNN and logistic regresion classifier has the heights value as (Figure 4.3):

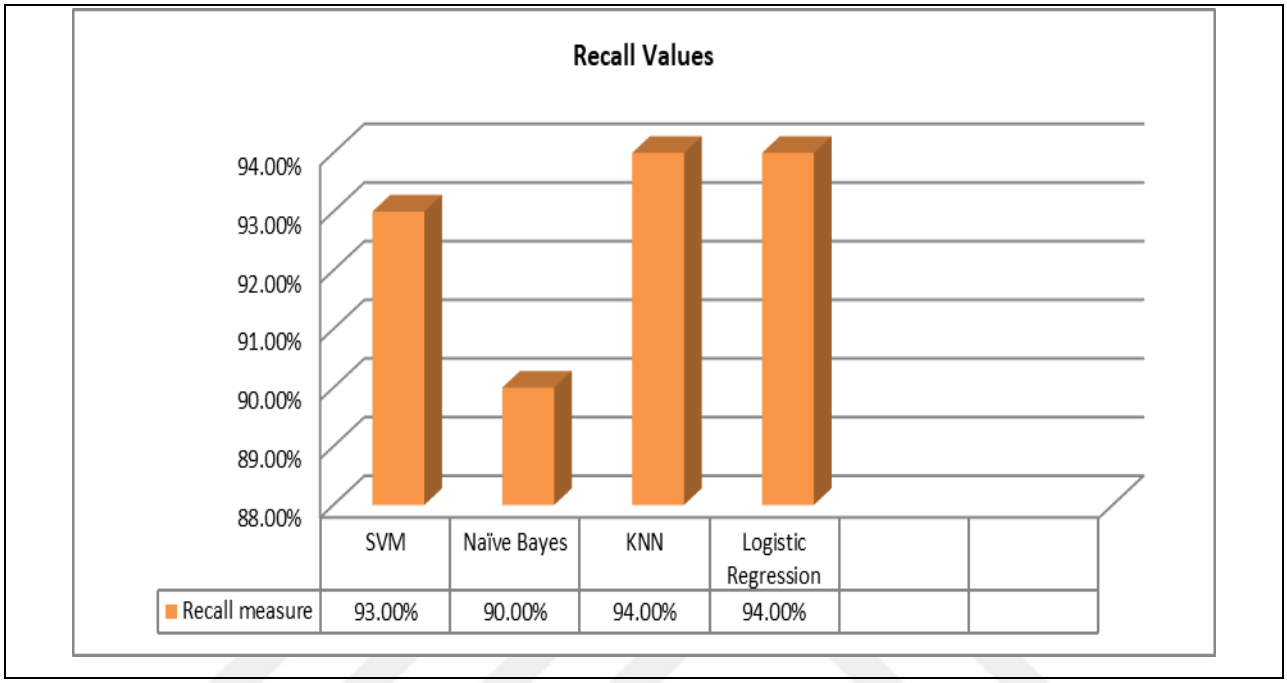


Figure 4.3: Recall Values of All the Classifiers.

In this figure chart, we show the Recall Metric for our algorthims, as we can see again the SVM and logistic regression came with best results for our model ,the SVM algorthims came with 93% and logistic regression 94%.

In term of F-Measure, the analysis results show that logistic regresion has the heights value , while the KNN and SVM have the same results (Figure 4.4):

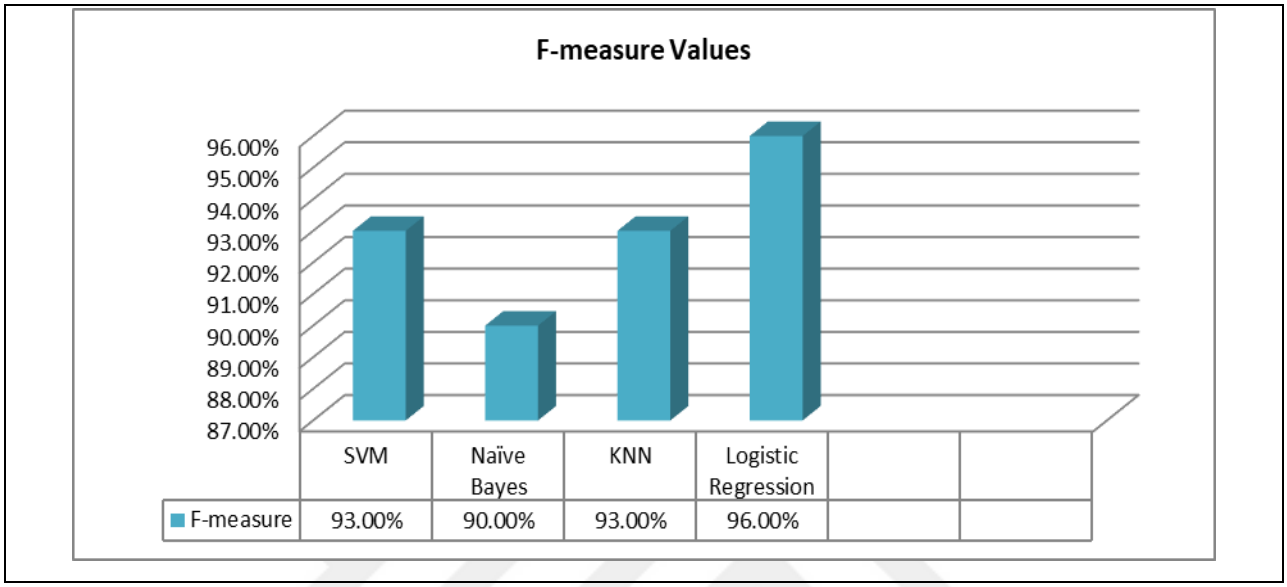


Figure 4.4: F-Measure Values of All the Classifiers.

In this figure chart, we show the F-Measure Metric for our algorthims, as we can see the logistic regression came with best results for our model ,the algorthims logistic regression came with 96%.

In term of Accuracy, the analysis results show that logistic regression classifier has the heights value (Figure 4.4):

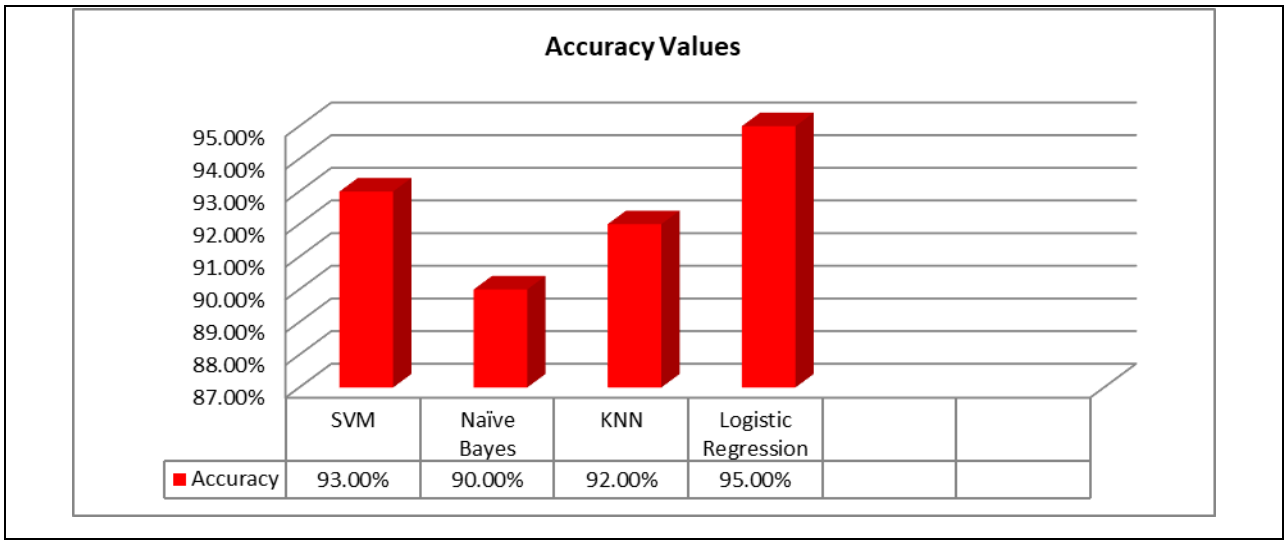


Figure 4.5: Accuracy Values of All the Classifiers.

The results obtained through the analysis show that logistic regression classifier is the best classifier as the values of all of the comparison factors were heights then the value of the other algorithms.

both the hand-engineered features technique and the direct signal method are equivalent in their results. Automatic feature extractors, on the other hand, may perform better and deliver faster inference times when the dataset has more distinct classes or is larger in size.

Analysis results of all the classifiers shown in Table 5.1.

4.8 SUMMARY OF THE RESULTS

Table 4.3: Summary of the experiment results

Classifier	**Accuracy**	**Precision**	**Recall**	**F1 score**
Logistic Regression	**(b) 95%**	**(c) 0.95**	**(d) 0.94**	**(e) 0.96**
Naive Bayes	(f) 90%	(g) 0.91	(h) 0.90	(i) 0.90
K-Nearest Neighbors	(j) 92%	(k) 92	(l) 94	n) 93
Support Vector Machines	(n) 94%	(o) 94	(p) 93	(q) 94

As we can observe, logistic regression achieved the best results in all of the metrics. Interestingly enough, Naive Bayes was able to achieve almost as good precision as K-Nearest Neighbors, but less impressive recall. If we compare the precision of Logistic Regression versus Support Vector Machines , we can observe how the "wisdom of the crowd" was able to outperform a single Naive Bayes. Naive Bayes, SVM and KNN had almost identical values of AUC metric. Naive Bayes performed the worst during this experiment.

5. CONCLUSION

Machine learning is the concept of a system learning, finding patterns, and making decisions without human involvement. It is a branch of artificial intelligence that may be classified into three groups based on the learning approach: supervised, unsupervised, and semi-supervised. The presence of labels (goal values) for input data during learning distinguishes these types.

These measures allow all trained models to be compared to one another. As long as the problem is classified as a classification problem, four basic metrics, True vs. False and Positive vs. Negative output from the trained model, must be introduced initially.

The EEG signals have a significant potential of discovering various neurological illnesses in an early stage. We have proven why it is superior than MRI. Being affordable and efficient, it can be employed for various investigations even with a reduced budget as compared to MRI. Many Deep Learning algorithms have been implemented on EEG recordings in order to classify different neurological illnesses and brain interface applications. In this research we have done an exhaustive literature review on application of various neural networks on EEG data and how utilizing these neural networks one was able to categorize numerous neurological illnesses including Seizure, AD and Depression. The preprocessing phase considerably influences the performance of the model. In this study, we train seven distinct models on three different ECG datasets and compare their performance with each other. We found that at least in our experiment, both hand-crafted features and the neural network feature extractors each have both their favorable and bad characteristics. The next stage is the integration of the model to the ECG processing system operating on the telemonitoring server.

REFERENCES

[1]. LILLY, Leonard S. Chapter 11. Mechanisms of Cardiac Arrhyth- mias. In: Pathophysiology of heart disease: a collaborative project of medical students and faculty. Wolters Kluwer, 2016, p. 268. ISBN 1605477230.

[2]. CLIFFORD, Gari; LIU, Chengyu; MOODY, Benjamin; LEHMAN, Li-wei; SILVA, Ikaro; LI, Qiao; JOHNSON, Alistair; MARK, Roger. AF Classification from a Short Single Lead ECG Recording: the Physionet Computing in Cardiology Challenge 2017. In: 2017 Computing in Cardiology Conference (CinC). Computing in Cardi-ology, 2017. Available from DOI: 10.22489/cinc.2017.065-469.

[3]. BLAUS, Bruce. An illustration depicting a Holter monitor. 2017. Available also from: https://commons.wikimedia.org/wiki/ File:Holter_Monitor.png.

[4]. MEYSTRE, Stephane. The Current State of Telemonitoring: A Comment on the Literature. Telemedicine and e-Health. 2005, vol. 11, no. 1, pp. 63–69. Available from DOI: 10.1089/tmj.2005.11.63.

[5]. CORDISCO, Marie Elena; BENIAMINOVITZ, Ainat; HAMMOND, Kim; MANCINI, Donna. Use of telemonitoring to decrease the rate of hospitalization in patients with severe congestive heart failure. The American Journal of Cardiology. 1999, vol. 84, no. 7, pp.860–862. Available from DOI: 10.1016/s0002-9149(99)00452-x.

[6]. ATKIELSKI, Anthony. ECG of a heart in normal sinus rhythm. 2007. Available also from: https://commons.wikimedia.org/wiki/ File:SinusRhythmLabels.svg.

[7]. DUBIN, Dale. Chapter 1: Basic Principles. In: Rapid Interpretation of EKG's, Sixth Edition. Tampa, Fla: Cover Pub. Co, 2000, pp. 14–27. ISBN 0912912065.

[8]. LIP, Gregory Y. H. et al. Atrial fibrillation. Nature Reviews Disease Primers. 2016, vol. 2, no. 1, pp. 16016. ISSN 2056-676X. Available from DOI: 10.1038/nrdp.2016.16.

[9]. ODUTAYO, Ayodele; WONG, Christopher X; HSIAO, Allan J; HOPEWELL, Sally; ALTMAN, Douglas G; EMDIN, Connor A. Atrial fibrillation and risks of cardiovascular disease, renal dis- ease, and death: systematic review and meta-analysis. BMJ. 2016,pp. i4482. Available from DOI: 10.1136/bmj.i4482.

[10]. DUBIN, Dale. Chapter 5: Rhythm, Part I. Premature Ventricular Contraction (PVC). In: Rapid Interpretation of EKG's, Sixth Edition. Tampa, Fla: Cover Pub. Co, 2000, p. 135. ISBN 0912912065.

[11]. DUBIN, Dale. Chapter 5: Rhythm, Part I. Runs of Ventricular Tachycardia. In: Rapid Interpretation of EKG's, Sixth Edition. Tampa, Fla: Cover Pub. Co, 2000, p. 156. ISBN 0912912065.

[12]. NOLLE, Floyd M; BOWSER, Richard W. Creighton University Ven- tricular Tachyarrhythmia Database. physionet.org, 1992. Available from DOI: 10.13026/C2X59M.

[13]. OPPENHEIM, Alan. Discrete-time signal processing. Upper Saddle River: Pearson, 2010. ISBN 0131988425.

[14]. OPPENHEIM, Alan. Chapter 8: The Discrete Fourier Transform. In: Discrete-time signal processing. Upper Saddle River: Pearson, 2010, pp. 541–600. ISBN 0131988425.

[15]. OPPENHEIM, Alan. Chapter 7: Filter Design Techniques. In: Discrete-time signal processing. Upper Saddle River: Pearson, 2010, pp. 439–511. ISBN 0131988425.

[16]. KWON, Ohhwan; JEONG, Jinwoo; KIM, Hyung Bin; KWON, In Ho; PARK, Song Yi; KIM, Ji Eun; CHOI, Yuri. Electrocardiogram Sampling Frequency Range Acceptable for Heart Rate Variability Analysis. Healthcare Informatics Research. 2018, vol. 24, no. 3, pp.198. Available from DOI: 10.4258/hir.2018.24.3.198.

[17]. scipy.signal.resample_poly — SciPy v1.5.0 Reference Guide. 2020. Available also from: https://docs.scipy.org/doc/scipy-1.5. 0/reference/generated/scipy.signal.resample_poly.html.

[18]. GAO, Hongxiang; LIU, Chengyu; WANG, Xingyao; ZHAO, Lina; SHEN, Qin; NG, E. Y. K.; LI, Jianqing. An Open-Access ECG Database for Algorithm Evaluation of QRS Detection and Heart Rate Estimation. Journal of Medical Imaging and Health Informatics.2019, vol. 9, no. 9, pp. 1853–1858. Available from DOI: 10.1166/ jmihi.2019.2800.

[19]. PAN, Jiapu; TOMPKINS, Willis J. A Real-Time QRS Detection Algorithm. IEEE Transactions on Biomedical Engineering. 1985, vol. BME-32, no. 3, pp. 230–236. Available from DOI: 10.1109/tbme.1985.325532.

[20]. PLESINGER, Filip; ANDRLA, Petr; VISCOR, Ivo; HALAMEK, Josef; BULKOVA, Veronika; JURAK, Pavel. Shape Analysis of Consecutive Beats May Help in the Automated Detection of Atrial Fibrillation. In: 2018 Computing in Cardiology Conference (CinC).Computing in Cardiology, 2018. Available from DOI: 10.22489/ cinc.2018.036.

[21]. DUBIN, Dale. Chapter 6: Rhythm, Part II. 1°AV Block. In: Rapid Interpretation of EKG's, Sixth Edition. Tampa, Fla: Cover Pub. Co, 2000, p. 178. ISBN 0912912065.

[22]. ACHARYA, U. Rajendra; JOSEPH, K. Paul; KANNATHAL, N.; LIM, Choo Min; SURI, Jasjit S. Heart rate variability: a review.Medical & Biological Engineering & Computing. 2006, vol. 44, no. 12, pp. 1031–1051. Available from DOI: 10.1007/s11517-006-0119-0.

[23]. Heart Rate Variability (HRV) — NeuroKit 0.0.39 documentation. 2020. Available also from: https://neurokit2.readthedocs . io/en/latest/examples/hrv.html.

[24]. VIRTANEN, Pauli et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020, vol. 17, pp. 261–272. Available from DOI: https://doi.org/10.1038/s41592-019-0686-2.

[25]. MAKOWSKI, Dominique; PHAM, Tam; LAU, Zen J.; BRAM- MER, Jan C.; LESPINASSE, François; PHAM, Hung; SCHÖLZEL, Christopher; S H CHEN, Annabel. NeuroKit2: A Python Toolboxfor Neurophysiological Signal Processing. Zenodo, 2020. Available from DOI: 10.5281/ZENODO.3597887.

[26]. PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python.Journal of Machine Learning Research. 2011, vol. 12, pp. 2825–2830.

[27]. PASZKE, Adam et al. PyTorch: An Imperative Style, High Perfor- mance Deep Learning Library. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; D'ALCHÉ-BUC, F.; FOX, E.; GARNETT,R. (eds.). Advances in Neural Information Processing Systems 32. Curran Associates, Inc., 2019, pp. 8024–8035.

[28]. BAI, Junjie; LU, Fang; ZHANG, Ke, et al. ONNX: Open Neural Network Exchange [https://github.com/onnx/onnx]. GitHub, 2019.

[29]. CORPORATION, Microsoft. ONNXMLTools. 2020. Available also from: https://github.com/onnx/onnxmltools.

[30]. REBACK, Jeff et al. pandas-dev/pandas: Pandas 1.0.5. Zenodo, 2020. Available from DOI: 10.5281/ZENODO.3898987.

[31]. COSTA-LUIS, Casper O. da. tqdm: A Fast, Extensible Progress Meter for Python and CLI. Journal of Open Source Software. 2019, vol. 4, no. 37, pp. 1277. Available from DOI: 10 . 21105 / joss .01277.

[32]. WALT, Stéfan van der; COLBERT, S Chris; VAROQUAUX, Gaël. The NumPy Array: A Structure for Efficient Numerical Compu- tation. Computing in Science & Engineering. 2011, vol. 13, no. 2, pp.22–30. Available from DOI: 10.1109/mcse.2011.37.

[33]. TAVENARD, Romain et al. Tslearn, A Machine Learning Toolkit for Time Series Data. Journal of Machine Learning Research. 2020, vol. 21, no. 118, pp. 1–6. Available also from: http://jmlr.org/papers/v21/20-091.html.

[34]. CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD Inter- national Conference on Knowledge Discovery and Data Mining. San Francisco, California, USA: ACM, 2016, pp. 785–794. KDD '16.ISBN 978-1-4503-4232-2. Available from DOI: 10.1145/2939672. 2939785.

[35]. INC., Plotly Technologies. Collaborative data science. Montreal, QC: Plotly Technologies Inc., 2015. Available also from: https://plot.ly.

[36]. WAGNER, Patrick; STRODTHOFF, Nils; BOUSSELJOT, Ralf-Dieter; KREISELER, Dieter; LUNZE, Fatima I.; SAMEK, Wojciech; SCHA- EFFTER, Tobias. PTB-XL, a large

publicly available electrocardio- graphy dataset. Scientific Data. 2020, vol. 7, no. 1. Available fromDOI: 10.1038/s41597-020-0495-6.

[37]. TEIJEIRO, Tomas; GARCIA, Constantino A.; CASTRO, Daniel; FÉLIX, Paulo. Arrhythmia Classification from the Abductive In- terpretation of Short Single-Lead ECG Records. In: 2017 Comput- ing in Cardiology Conference (CinC). Computing in Cardiology,2017. Available from DOI: 10.22489/cinc.2017.166-054.

[38]. ZABIHI, Morteza; RAD, Ali Bahrami; KATSAGGELOS, Agge- los K.; KIRANYAZ, Serkan; NARKILAHTI, Susanna; GABBOUJ, Moncef. Detection of Atrial Fibrillation in ECG Hand-held De- vices Using a Random Forest Classifier. In: 2017 Computing in Cardiology Conference (CinC). Computing in Cardiology, 2017.Available from DOI: 10.22489/cinc.2017.069-336.

[39]. MAHAJAN, Ruhi; KAMALESWARAN, Rishikesan; HOWE, John Andrew; AKBILGIC, Oguz. Cardiac Rhythm Classification from a Short Single Lead ECG Recording via Random Forest. In: 2017Computing in Cardiology Conference (CinC). Computing in Cardi- ology, 2017. Available from DOI: 10.22489/cinc.2017.179-403.

[40]. DATTA, Shreyasi et al. Identifying Normal, AF and other Abnor- mal ECG Rhythms using a Cascaded Binary Classifier. In: 2017 Computing in Cardiology Conference (CinC). Computing in Cardi-ology, 2017. Available from DOI: 10.22489/cinc.2017.173-154.

[41]. HONG, Shenda; WU, Meng; ZHOU, Yuxi; WANG, Qingyun; SHANG, Junyuan; LI, Hongyan; XIE, Junqing. ENCASE: an EN- semble ClASsifiEr for ECG Classification Using Expert Features and Deep Neural Networks. In: 2017 Computing in Cardiology Con- ference (CinC). Computing in Cardiology, 2017. Available fromDOI: 10.22489/cinc.2017.178-245.

[42]. LIU, Feifei et al. An Open Access Database for Evaluating the Algorithms of Electrocardiogram Rhythm and Morphology Abnormality Detection. Journal of Medical Imaging and Health Infor- matics. 2018, vol. 8, no. 7, pp. 1368–1373. Available from DOI:10.1166/jmihi.2018.2442.

[43]. CHEN, Tsai-Min; HUANG, Chih-Han; SHIH, Edward S.C.; HU, Yu-Feng; HWANG, Ming-Jing. Detection and Classification of Cardiac Arrhythmias by a Challenge-Best Deep Learning Neural Network Model. iScience. 2020, vol. 23, no. 3, pp. 100886. Availablefrom DOI: 10.1016/j.isci.2020.100886.

[44]. MOODY, George B; MARK, Roger G. MIT-BIH Arrhythmia Database. physionet.org, 1992. Available from DOI: 10.13026/C2F305.

[45]. MIT-BIH Arrhythmia Database - Google Scholar. 2020. Available also from: https :// scholar . google . com / scholar ? q = MIT - BIH+Arrhythmia+Database.

[46]. BOUSSELJOT, Ralf-Dieter; KREISELER, D; SCHNABEL, A. The PTB Diagnostic ECG Database. physionet.org, 2004. Available from DOI: 10.13026/C28C71.

[47]. STRODTHOFF, Nils; WAGNER, Patrick; SCHAEFFTER, Tobias; SAMEK, Wojciech. Deep Learning for ECG Analysis: Benchmarks and Insights from PTB-XL. 2020. Available from arXiv: 2004.13701[cs.LG].

[48]. GÉRON, Aurélien. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intel- ligent Systems. O'Reilly Media, 2019. ISBN 1492032646. Availablealso from: https://www.xarg.org/ref/a/1492032646/.

[49]. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. MIT Press, 2016. http://www.deeplearningbook. org.

[50]. CUNNINGHAM, Padraig; DELANY, Sarah Jane. k-Nearest Neigh- bour Classifiers: 2nd Edition (with Python examples). 2020. Available from arXiv: 2004.04523 [cs.LG].

[51]. BAGNALL, Anthony; LINES, Jason; BOSTROM, Aaron; LARGE, James; KEOGH, Eamonn. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery. 2016, vol. 31, no.3, pp. 606–660. Available from DOI: 10.1007/s10618-016-0483-9.

[52]. DAU, Hoang Anh; BAGNALL, Anthony; KAMGAR, Kaveh; YEH, Chin-Chia Michael; ZHU, Yan; GHARGHABI, Shaghayegh; RATANAMAHATANA, Chotirat Ann; KEOGH, Eamonn. The UCR time series archive. IEEE/CAA Journal of Automatica Sinica.2019, vol. 6, no. 6, pp. 1293–1305. Available from DOI: 10.1109/ jas.2019.1911747.

[53]. GÉRON, Aurélien. Chapter 4: Training Models. Logistic Regres- sion. In: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Sys- tems. O'Reilly Media, 2019, pp. 142–150. ISBN 1492032646. Avail-able also from: https://www.xarg.org/ref/a/1492032646/.

[54]. GÉRON, Aurélien. Chapter 7: Ensemble Learning and Random Forests. Bagging and Pasting, Random Forests. In: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts,Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019, pp. 192–199. ISBN 1492032646. Available also from: https://www.xarg.org/ref/a/1492032646/.

[55]. GÉRON, Aurélien. Chapter 7: Ensemble Learning and Random Forests. Boosting. In: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelli-gent Systems. O'Reilly Media, 2019, pp. 203–208. ISBN 1492032646. Available also from: https://www.xarg.org/ref/a/1492032646/.

[56]. CHRISLB. Diagram of a multi-layer feedforward artificial neural net- work. 2012. Available also from: https://commons.wikimedia. org / wiki / File : MultiLayerNeuralNetworkBigger _ english .png.

[57]. Course materials and notes for Stanford class CS231n: Convolutional Neural Networks for Visual Recognition. 2020. Available also from: https://cs231n.github.io/convolutional-networks/.

[58]. HUANG, Jingshan; CHEN, Binqiang; YAO, Bin; HE, Wangpeng. ECG Arrhythmia Classification Using STFT-Based Spectrogram and Convolutional Neural Network. IEEE Access. 2019, vol. 7,pp. 92871–92880. Available from DOI: 10.1109/access.2019. 2928017.

[59]. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).IEEE, 2016. Available from DOI: 10.1109/cvpr.2016.90.

[60]. GÉRON, Aurélien. Chapter 15: Processing Sequences Using RNNs and CNNs. Recurrent Neurons and Layers. In: Hands-On Ma- chine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts,Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019, pp. 497–501. ISBN 1492032646. Available also from: https://www.xarg.org/ref/a/1492032646/.

[61]. GÉRON, Aurélien. Chapter 15: Processing Sequences Using RNNs and CNNs. Handling Long Sequences. In: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019,pp. 511–520. ISBN 1492032646. Available also from: https:// www.xarg.org/ref/a/1492032646/.

[62]. CHO, Kyunghyun; MERRIËNBOER, Bart van; GULCEHRE, Caglar; BAHDANAU, Dzmitry; BOUGARES, Fethi; SCHWENK, Hol- ger; BENGIO, Yoshua. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natu- ral Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1724–1734. Available from DOI: 10.3115/v1/D14-1179.

[63]. BLAD, John Erling. Gated Recurrent Unit, fully gated version. 2018. Available also from: https://commons.wikimedia.org/wiki/ File:Gated_Recurrent_Unit,_base_type.svg.

[64]. KINGMA, Diederik P.; BA, Jimmy. Adam: A Method for Stochastic Optimization. 2014. Available from arXiv: 1412.6980 [cs.LG].

[65]. CARUANA, Rich; LAWRENCE, Steve; GILES, Lee. Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping. In: Proceedings of the 13th International Conference on Neural Information Processing Systems. Denver, CO: MIT Press, 2000, pp. 381–387. NIPS'00.

[66]. LAARHOVEN, Twan van. L2 Regularization versus Batch and Weight Normalization. 2017. Available from arXiv: 1706.05350 [cs.LG].

[67]. PASCANU, Razvan; MIKOLOV, Tomas; BENGIO, Yoshua. On the Difficulty of Training Recurrent Neural Networks. In: Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28. Atlanta, GA, USA: JMLR.org, 2013, III–1310–III–1318. ICML'13.

[68]. GÉRON, Aurélien. Chapter 3: Classification. Performance Mea- sures. In: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Sys- tems. O'Reilly Media, 2019, pp. 88–93. ISBN 1492032646. Availablealso from: https://www.xarg.org/ref/a/1492032646/.

[69]. PEREZ ALDAY, Erick Andres; GU, Annie; SHAH, Amit; LIU, Chengyu; SHARMA, Ashish; SEYEDI, Salman; BAHRAMI RAD, Ali; REYNA, Matthew; CLIFFORD, Gari. Classification of 12-lead ECGs: the PhysioNet - Computing in Cardiology Challenge 2020. Phy-sioNet, 2020. Available from DOI: 10.13026/F4AB-0814.