Kingdom of Saudi Arabia
Ministry of Higher Education
Qassim University
College of Computer
Information Technology Department

المملكة العربية السعودية
وزارة التعليم العالي
جامعة القصيم
كلية الحاسب
قسم تقنية المعلومات

FEATURE FUSION FOR IMAGE CLASSIFICATION

Students:

Khalid Ahmed Al-Rebdi	351115019
Waleed Saadi Al-Rashidi	351108753
Abdullah Mohammed Al-Salamah	342115873
Badr Abdulaziz Al-Ayyaf	351112730

Supervisor:

Dr. Rehan Ullah Khan

Qassim-Saudi Arabia
1439/1440 (2018/2019)

Table of Contents

Chapter Three

FEATURES AND CLASSIFIERS

CHAPTER FOUR

Methodology

List of Figures

List of Tables

C e r t i f i c a t e

It is certified that the project report has been prepared and written under my direct supervision and guidance. The project report is approved for submission for its evaluation.

Dr.Rehan Ullah Khan

D e d i c a t i o n

This Project is dedicated to our beloved parents who supported and encouraged us from the moment we were born to the moment we wrote this down.

Khalid Ahmed Al-Rebdi

Waleed Saadi Al-Rashidi

Abdullah Mohammed Al-Salamah

Badr Abdulaziz Al-Ayyaf

A c k n o w l e d g e m e n t

Firstly, we thank Allah for his graces, then are we are thankful to our supervisor Dr. Rehan Ullah Khan for his support, guidance, and continuous encouragement throughout the project.

Khalid Ahmed Al-Rebdi

Waleed Saadi Al-Rashidi

Abdullah Mohammed Al-Salamah

Badr Abdulaziz Al-Ayyaf

Abstract

Combining multiple features sets that include specific features like color, texture, and shape is a challenging task towards the improvement of the image classification process. This project covers the image-based classification area. We investigate the phenomenon of fusion of feature sets for the image classification problems. Our approach can be applied to all computer domains which are based on the features. We compare our approach with existing methods that extract features and apply it on other datasets. Our main objective is to get the optimist classification from fusing features of image classification. The objective is also to find whether a single good feature set is better than combining multiple features and whether the combination of many features increases performance.

The feature fusion performs differently with each classifier and each dataset. From the experimental results, we note that the features reactions to fusion depend on their characteristics and we see that most of these features get benefit from fusion with the correct selection of the fusion rule.

CHAPTER 1
Problem and Objectives

1.1 Introduction

Classifying an Image into known and recognizable objects is an important task in computer vision, it is a fundamental task towards image understanding and includes many applications, including object recognition, scene understanding, image tagging and recommendation [1].
Our work is to deal with the pixel-level classification which is a set of local features that includes color, texture, shape,etc. Each one of these properties plays a major part in classification. For instance, to discriminate football players from two teams, color information is crucial.

The feature combining process is known as gathering all of the important information from multiple features and their inclusion into a single one [2]. Adding more features will not always give you a better result of classification and that's why many practical cases prefer to analyze the different features separately, and later fuse together the results of the classifications.

We consider the fusion of features to get a classification which will hopefully be more reliable than using only one feature. In this Project, we are trying to analyze the relation of the classifiers with the feature set. Our objective is to find the idealist classification from fusing features of image classification.

1.2 Problem Specification and Motivation

Image classification is fundamental to many computer vision applications. The image feature set plays an important role in image classification. In this project, the objective is to investigate the fusion of features for image classification. Fusion here means to combine two feature extraction methods. Given the image, the features are extracted and they are trained by the classifier. The feature calculation is based on different features approaches. for example, color features, texture features, and others. This thesis investigates the fusion of multiple features based on classifiers.

1.3 Goals and Objectives

- To understand computer vision.
- To understand machine learning.
- To understand MATLAB.
- To understand WEKA.
- To compare different machine learning algorithms.
- To compare different feature extraction methods from images.
- To suggest how to increase the performance of image features in machine learning.
- To test feature fusion.
- To get better performance from feature fusion of image classification.
- To propose new recommendations based on experiments.

1.4 Study Scope

This project is related to feature extraction from images and then learning image classes from these features. Therefore, the project covers machine learning and computer vision. The tool used will be WEKA. The results of the project cover and can affect many applications domains where image processing and machine learning is needed.

1.5 Datasets

For this project we used three datasets:

1) Faces dataset.
2) Birds dataset.
3) Butterflies dataset.

1.6 CIFAR-10 Dataset

CIFAR-10 is a dataset that consists of about 60000 32x32 color images all in 10 classes. Each class contains 6000 images, thereby constituting a total of 50000 training images and 10000 test images. This dataset is derived from [3] and it is divided into five training batches and a test batch. Each training batch consists of 10000 images and 1000 randomly-selected images from each class. Furthermore, they contain the remaining images but in random order, however, some training batches may have more images in a class than others. In conclusion, training batches have a total of 5000 images from each class.

Figure 1.1:images of CIFAR-10 Dataset [4].

1.6.1 Faces Dataset

Frontal face dataset collected by Markus Weber at California Institute of Technology. 450 face images. 896 x 592 pixels. Jpeg format. 27 or so unique people

under with different lighting/expressions/backgrounds. ImageData.mat is a MATLAB file containing the variable SubDir_Data which is an 8 x 450 matrix. Each column of this matrix hold the coordinates of the bike within the image [5].

.

Figure 1.2: Some faces of the dataset [6].

1.6.2 Birds Dataset

This database has 600 images (100 samples each) of six different classes of birds taken from [3]. The images are color JPEG, of variable resolution. The classes (each in its own directory) are as follows:

- Egret
- Mandarin duck
- Snowy owl
- Puffin
- Toucan
- Wood duck

Figures 1.3: Samples of each class are shown in this figure [7].

1.6.3 Butterflies Dataset

This database contains 619 images of seven different classes of butterflies taken from [3]. The images are color JPEG, of variable resolution. The classes (each in its own directory) are as follows

- Admiral: 111 images
- Black Swallowtail: 42 images
- Machaon: 83 images
- Monarch 1 (wings closed): 74 images
- Monarch 2 (wings open): 84 images
- Peacock: 134 images
- Zebra: 91 images

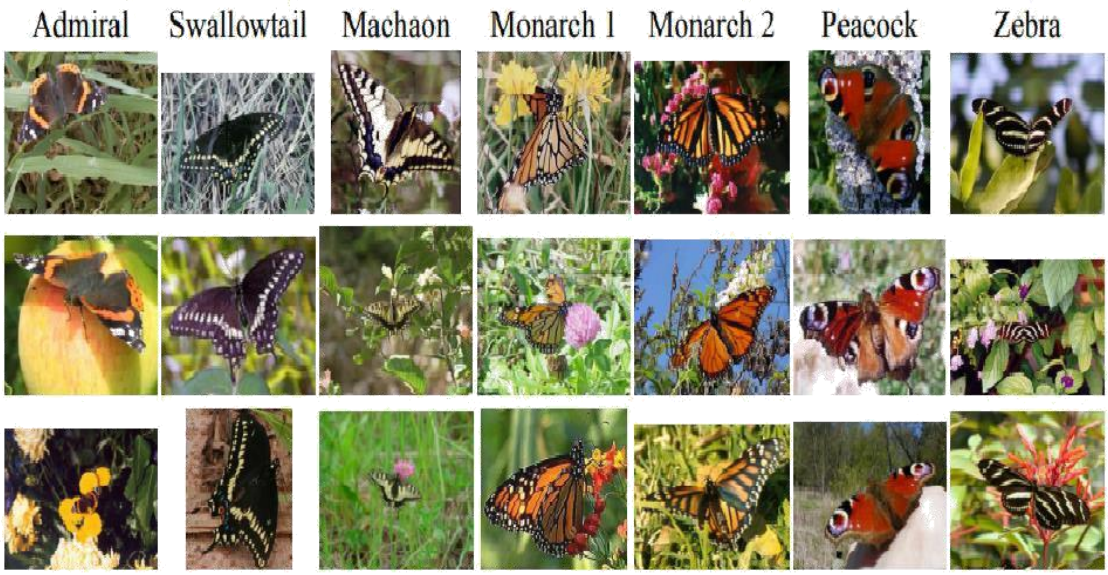


Figure 1.4: Samples of each class are shown in each column [8].

1.7 Work Break down Structure

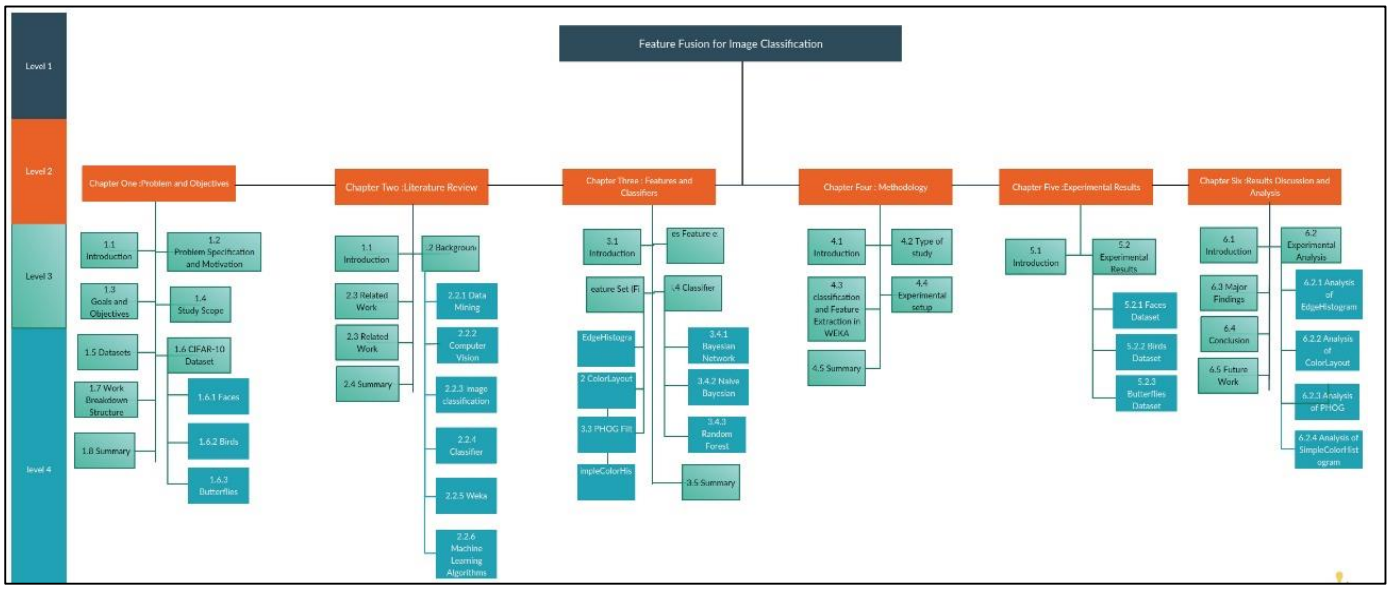


Figure 1.5: Work Breakdown Structure.

1.8 Summary

In this chapter, we mentioned our project introduction, problems, objectives, and goals. Also, we mentioned the datasets these will be used in the project. Finally, we showed the Work Breakdown Structure of the project which displayed the major parts of the project.

Chapter 2
Literature Review

2.1 Introduction

In the era of technology, information could be more valuable than gold, but it must be extracted first. To get benefit from the enormous amounts of data on images we need data mining and machine learning. In this chapter Literature Review we will give you a background concerning the image classification like computer vision and data mining and we will define topics that relate to image classification processes such as object detection and image recognition-based features and extraction features.

2.2 Background

2.2.1 Data Mining

Data mining is a technology that has the potential to improve an organization. Its tools can be used to predict future trends and their behaviors, therefore, allowing for knowledge-driven decision making. Data mining technology is the process of extracting necessary information from a huge chunk of data. To put simply, it is simply obtaining knowledge from data. The systems of data mining are classified based on their data types and models. Its model classification is based on already classified data obtained from discoveries made from predicted patterns [9].

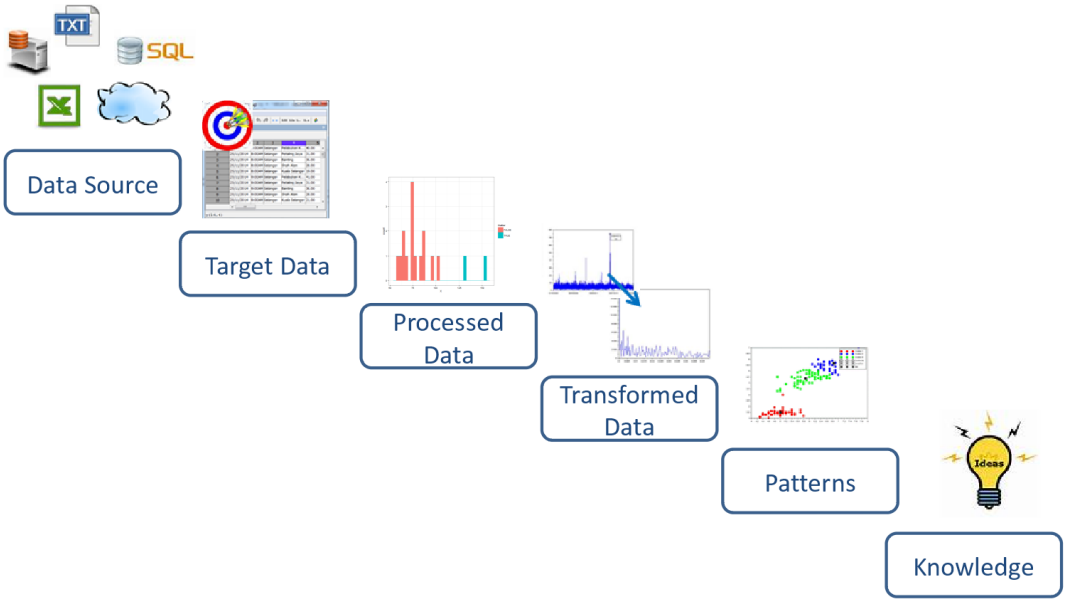


Figure 2.1 Shows Knowledge Discovery in Databases (KDD) [10].

2.2.2 Computer Vision

Computer vision is a branch of computer science that allows computers to recognize, identify and process images in a more advanced manner than the human vision. It then produces a required output. Theoretically, computer vision can be likened to incorporating human intelligence and instinct to a computer, but in practice, it is a complex task to enable computers to recognize objects as efficient as the human eye.
Computer vision shares a semblance to artificial experience. Like AI, computer vision enables the computer to not only identify physical objects but also make appropriate analysis and decision based on its rationalization. A classic example of computer vision in practice is the handwriting recognition ability of computers. Through the vision technology, it is able to rate the speed of objects in a video or even during coverage by the camera. Another example is image segmentation which allows images to be divided into multiple sets of views through the help of an algorithm partition. Scene reconstruction enables computers to create a 3D model of an inputted scene through images/videos. Noise reduction is possible through Machine Learning founded filter during the image restoration process [11].

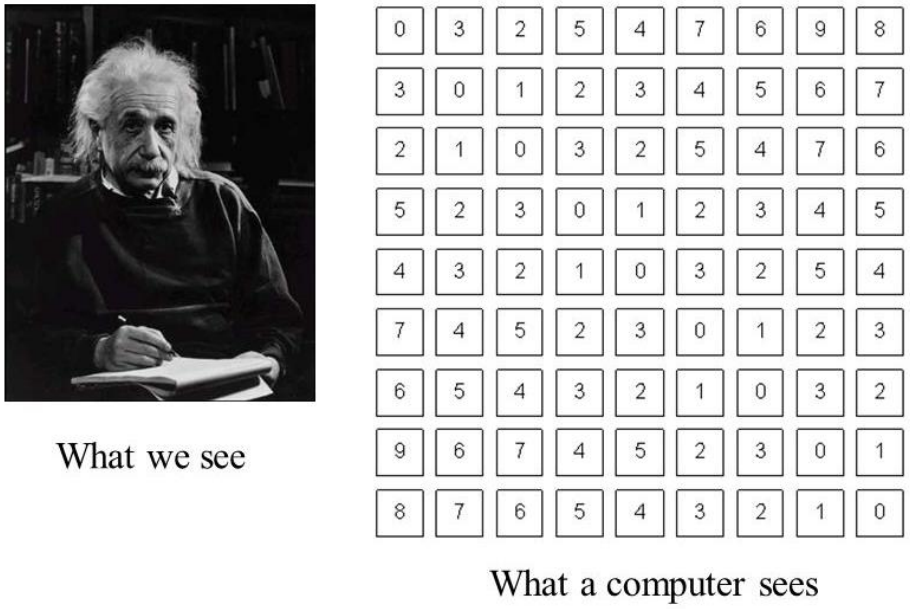


Figure 2.2 Shows how our vision differs from computers. [12]

2.2.3 Image classification

The classification process is the development of a model that assigns objects to their classes using their similarities to other objects of their types. It is accomplished using a reference which can either be the original data or the model of that data. In other words, data classification works by creating a model based on a training set and values called class labels, it then uses these labels/attributes in classifying new data. Image classification using this concept can, therefore, be defined as the process of assigning pixels to classes. Pixels here are the individual units composed of attributes within many spectral bands that will be classified.

There are two steps to the process of classification [13]:

- Model Construction – in this step, the predetermined classes are described. The model constructed is based on some rules, decision trees, mathematical formula and samples known as the training set.
- Model Usage – once a model has been constructed, it is used to classify future objects. The new sample to be classified is compared with the model attributes. If

the attribute of the new sample satisfies a predetermined accuracy percentage, it is considered to be of the new class. Test sets, however, differ from the training set to avoid overfitting.

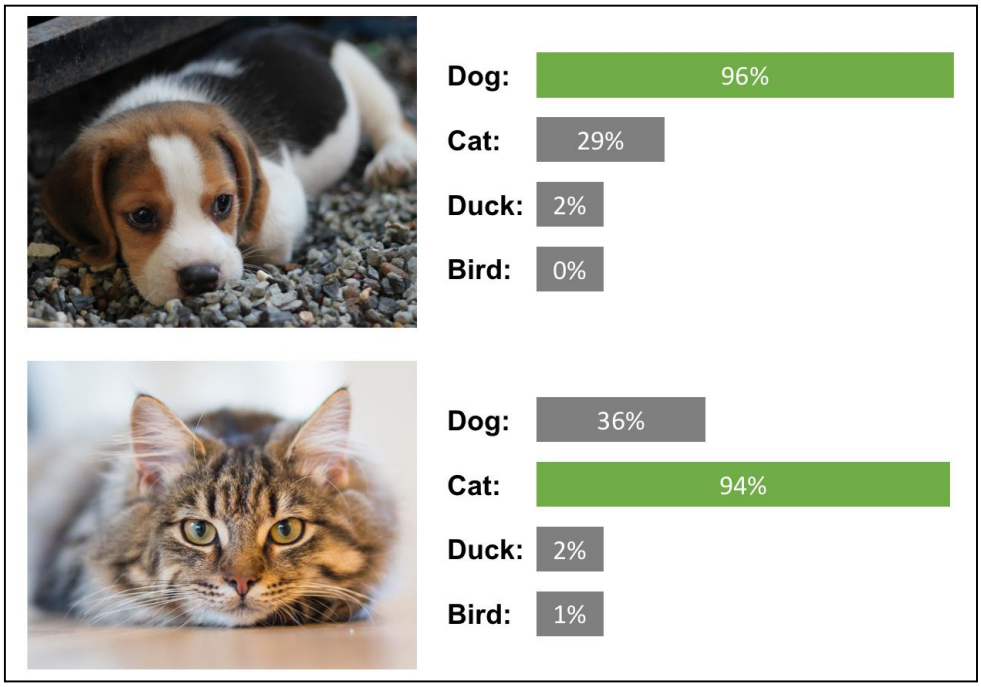


Figure 2.3 Shows how image classification indicates the correct animal. [13]

2.2.4 Classifier

classifier model tries to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes. Outcomes are labels that can be applied to a dataset.

There are two approaches to machine learning: supervised and unsupervised. In a supervised model, a training dataset is fed into the classification algorithm. That lets the model know what it is, for example, “authorized” transactions. Then the test data sample is compared with that to determine if there is a “fraudulent” transaction. This type of learning falls under “Classification”.

Unsupervised models, on the other hand, are fed a dataset that is not labeled and looks for clusters of data points. It can be used to search data for similarities, detect patterns, or identify outliers within a dataset. A typical use case would be finding similar images. Unsupervised models can also be used to find “fraudulent” transactions by looking for anomalies within a dataset. This type of learning falls under “Clustering”.

There are several classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes [14].

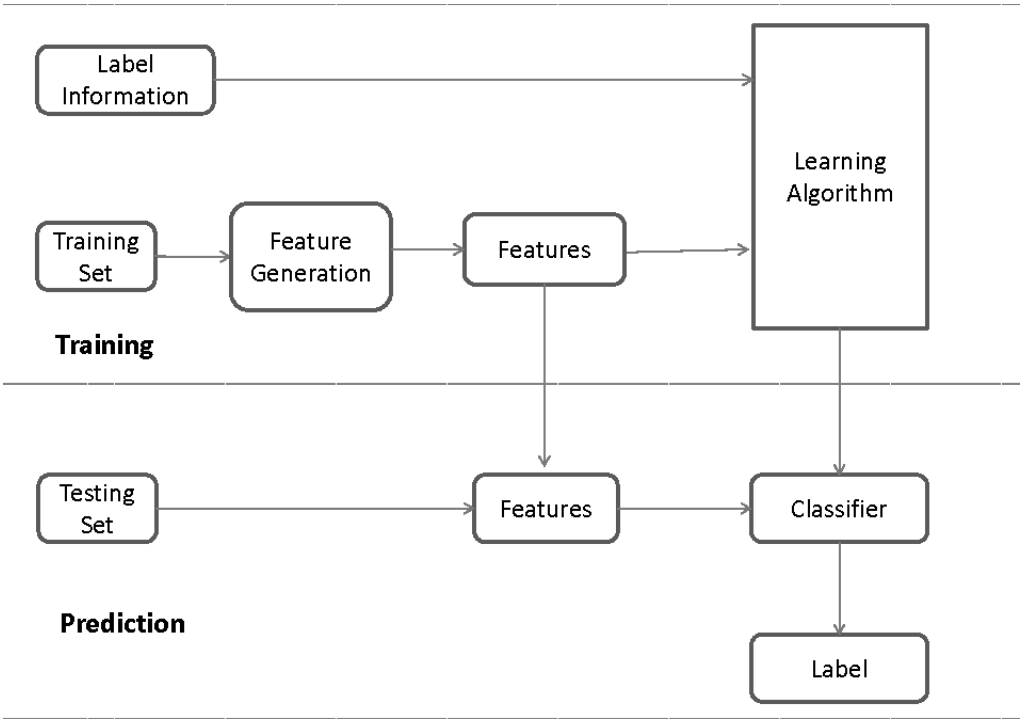


Figure 2.4: A General Process of Data Classification [15].

2.2.5 Weka

Weka is simply a collection of machine learning algorithms for the purpose of data mining. These algorithms are acquired to apply directly to datasets or called from Java code. Weka has also been known to include algorithms fit for developing new machine learning schemes [16].

It is a collection of tools for the processes of regression, clustering, association, data pre-processing, classification, and visualization.

2.2.6 MATLAB

MATLAB is a high-performance programming language created for technical computing. Through computation, visualization and programming, and a user-friendly environment, it is able to solve problems and solutions with mathematical notations [17]. Its uses are the following but not limited to; math and computation, algorithm development, modeling, simulation and prototyping, scientific and engineering graphics, software developing, and data analysis/exploration/visualization.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

2.2.7 Machine Learning Algorithms

Machine learning explores the study and structure of algorithms that can learn from and make predictions on data like algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. Machine learning is used in a wide range of computing tasks where designing and programming explicit algorithms with good performance is hard or infeasible; example applications include email filtering, detection of network intruders, and computer vision [18].

2.2.7.1 Bayesian Network

Bayesian networks are a type of Probabilistic Graphical Model that can be used to build models from data and/or expert opinion [19].
They can be used for a wide range of tasks including prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction and decision making under uncertainty. Figure 1 below shows these capabilities in terms of the four major analytics disciplines, Descriptive analytics, Diagnostic analytics, Predictive analytics, and Prescriptive analytics.

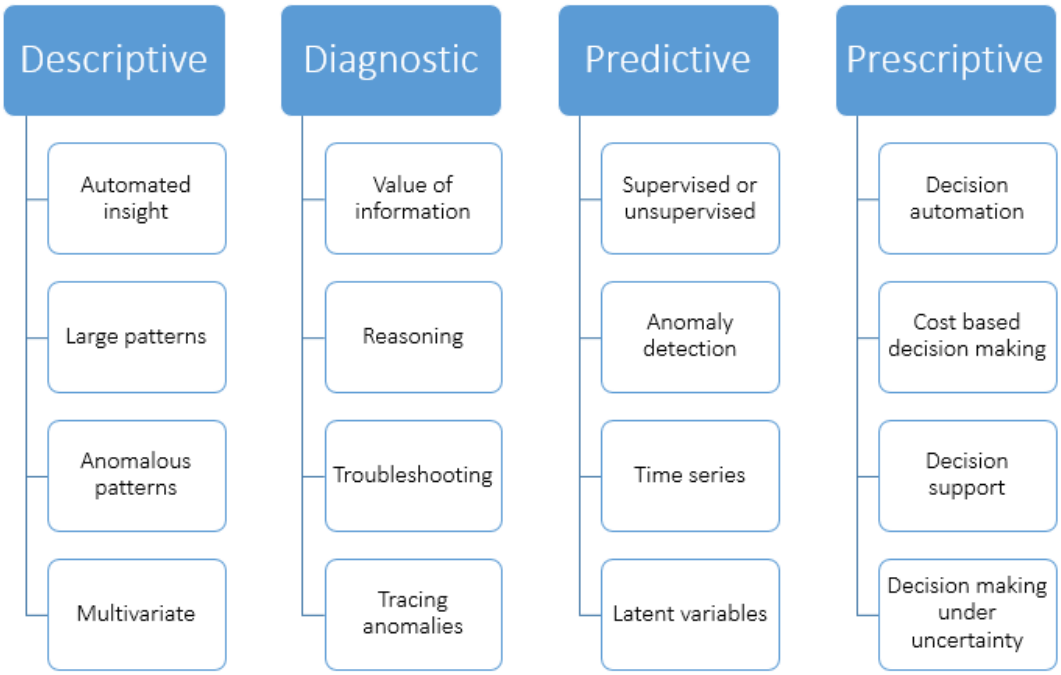


Figure 2.5 - Descriptive, diagnostic, predictive & prescriptive analytics with Bayesian networks

They are also commonly referred to as Bayes nets, Belief networks and sometimes Causal networks [20].

2.2.7.2 Neural Networks

Neural networks (NN), is a mathematical model or computational model based on biological neural networks. It contains a unified group of artificial neurons and processes information using a connectionist approach to computation as shown in Figure 2.2. In most cases, the NN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase.

NN are programmed or "trained" to store, recognize, and associatively retrieve patterns or database entries; to solve combinatorial optimization problems; to filter noise from measurement data; to control ill-defined problems; in summary, to estimate sampled functions when we do not know the form of the functions." It is precisely these two abilities (pattern recognition and function estimation) which make NN so prevalent a utility in data mining [21].

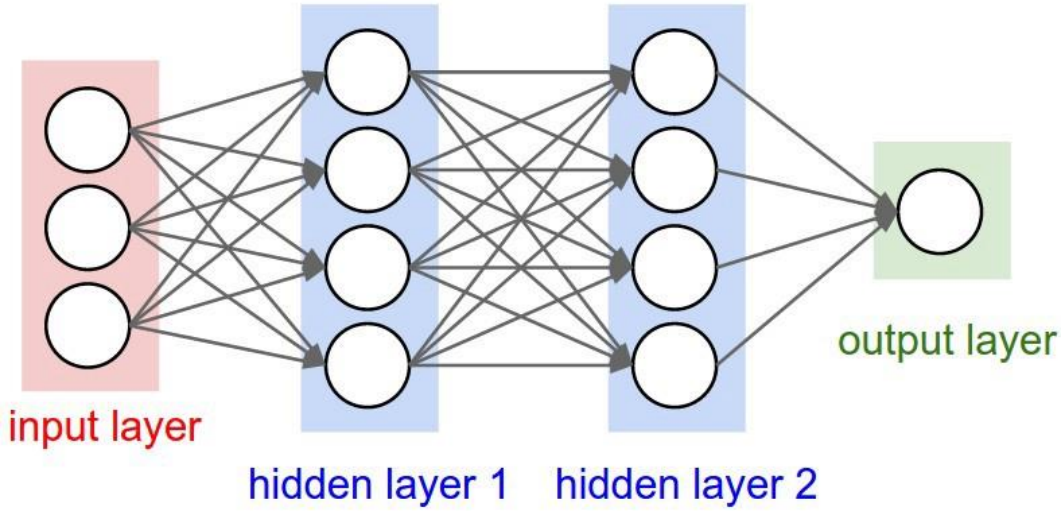


Figure 2.6 Shows how Neural networks works [22].

2.3 Related Work

2.3.1 Feature Extraction for images

In general, images have the following features – color, texture, shape, edge, shadows, temporal details,etc. The features that were most promising were color, texture, and edge [23]. The reasons are as follows:

1. Color: Egeria occurs in 2 colors – pink (rusty rose) and black. Hence the picture elements can be compared to these spectra.
2. Texture: Texture is defined as a neighborhood feature [24] – as a region or a block. The variation of each pixel with respect to its neighboring pixels defines texture. In our case, Egeria occurs in open water or in water at the shoreline. Hence the textural details of similar regions can be compared with a texture template.
3. Edge: Edge is simply a large change in frequency. This is particularly important here, as the distinction between the dark Egeria and the lighter water bodies or land can be considered as an edge.

After the features have been identified, a step-wise procedure extracts the features and combines them using rules that would detect the maximum coverage of Egeria in the image. Two sets of images were provided. The first set is the set of 30 images. The second set is the corresponding coverage of Egeria, which was manually detected and

was provided for experimental verification. In the individual image, each pixel corresponds to a particular intensity value. In order to correctly identify Egeria in the images, we divide the image into blocks of size nXn. We define a block as a “block” of pixels – say 10X10 or 8X8 (both were used in the experiments). These sizes were considered after experimenting with various other sizes. If the size is too small, then texture features cannot be described. If it is too large, small patches of Egeria cannot be detected. Block size of 10X10 or 8X8 was an appropriate size both in terms of processing time and accuracy. The training image is divided into blocks and the domain expert makes the parameter adjustments and sets the thresholds only once [23].

The general procedure, which involves all the automatic feature extraction tasks, is called IClass. For texture features, there are templates from the training image with representative properties for that feature. The following are the methods that were tried on this training image [23].

2.3.1.1 Color Feature Extraction:

The techniques used for color extraction include the average color in Gray scale, average color in RGB, and Average color in YCBCR format where the Y is the luminance and CB and CR are the chrominance components [25]. After evaluating these methods using Precision and Recall, we were able to conclude that YCBCR performs better than the other two techniques of color extraction.

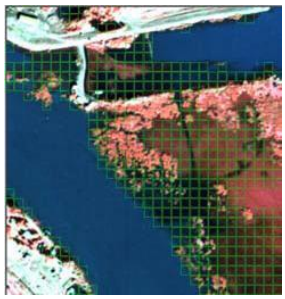


Figure 2.7: Shows color extraction [23].

The procedure used gives a region matrix output with 30x30 (for 10x10 block, and a 37x37 for an 8x8) block size. Each of these blocks is coded with ‘1’ for color match or ‘0’ for unmatched color.

2.3.1.2 Texture Features Extraction:

The texture extraction was conducted using textures with pure Egeria, Egeria with the land, and Egeria with land and water. Histograms were also used with and without bins [26], it was then normalized with bins and then in Discrete Cosine Transform [25]. The histogram method was used by comparing the template with each image block. The mean squared difference between the individual peaks is then taken and calculated. Any block with a relatively small difference is considered a match with the template and extracted as part of that texture. A small difference between the template and the tested block gauged with a threshold is marked YES for the block and vice versa. The different techniques for texture extraction were also compared as for color and the Histogram with bins method was evaluated to be the most accurate of them and hence adopted.

Figure 2.8: Shows Texture extraction [23].

Finally, the three textures were combined to acquire the final texture region matrix. The histogram method was found to be time-consuming given the extensive calculations it required. One of the aims of this work is to create an interactive system that allows users to define and extract features. The DCT method [27, 28] as discovered is the simplest and fastest method of all techniques. It avoids the constraints of histograms by eliminating the need to obtain the peak values and laborious computations. The DC coefficient is considered representative of the block and when compared with other DC values, it gives a similarity measure. Its importance is also explained in the next section.

2.3.1.3 Edge Feature extraction:

Edge features are particularly important for some of the darker images. Fortunately, the training image was of normal quality and hence we did not use the edge feature. However, we do use it for some of the darker images in the set for testing. The Canny edge detection [29] method with default threshold (0) was used. Edge feature alone has very little efficiency; hence we need to combine it with a stronger feature, like the color. It is combined with the color feature to describe the boundaries and inner regions of Egeria [23].

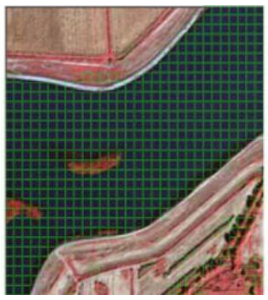
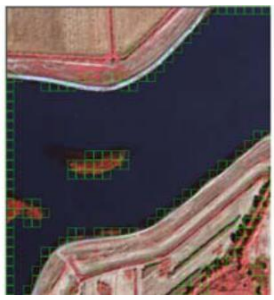


Figure 2.9: Color extracted image – The water body is also covered



Edge extracted image – Egeria alone is extracted [23].

2.3.1.4 Combining Features

The extracted features combined give the final image shown to the left. Each block is similar in its feature. In other words, following the extraction, each block has a sequence of ‘1s’ and/or ‘0s’ which stands for YES or NO respectively qualifying the various features represented. The combined features can be likened to the creation of rules. One of these rules is to combine three features in the order color AND edge OR texture.

Color	Textures	Edge	Class
1	0	1	1
0	0	0	0
1	1	0	1
1	0	0	2 (Uncertain)

Table 2.1 The outcome of combining (Color-Texture-Edge).

The rules, however, vary with domain and features used. For example, if a feature is considered accurate for a domain, it is labeled as YES (1) as a class (check the table 2.1 above). Conversely, if the IClass is uncertain, the class label is 2 (NO). This shows areas that could either be Egeria or not. This rule is applied to the training phase as done in the testing phase. However, for 3 features, the table below shows a distinct set of rules that should be followed. In this case, the first and third rules demand that color and texture/edge determine if Egeria is present in that block. The second rule, however, commands a 1 for none of the features implie an absent Egeria. The fourth rule states that color itself is not sufficient in ascertaining the presence of Egeria. To combat these constraints, an interactive system is used to resolve uncertain, misclassified or missed regions. It accomplishes this by first of all alerting the experts to their existent then ascribing them classes. The training image in this work has actual coverage and therefore can be assigned class labels from the cover. Once the Egeria regions have been resolved and well-covered, the formed dataset with featured class labels and columns or rows is created [23].

2.3.1.5 Evaluation of Features:

All the extracted features were obtained for this particular section. The extracted features from the training image were used on the testing image. In other words, a system had been built and trained to accommodate other images using the same technique. The effectiveness of these features can be evaluated using a criterion. The second set of images from the Egeria was used to compare and validate other images. The validation images are of the same size as the extracted images and therefore can be compared with it in terms of blocks [23].

Two-class problems have four possible outcomes of prediction [30]. They include; True Positives (TP), True Negatives (TN), False Negatives (FN) and the False Positives (FP). The true positives are the correct extracted regions, true negatives are the incorrect and not retrieved regions. False positives are the regions that are incorrect but have been extracted, while the false negatives were supposed to be extracted but have been missed for one reason or another.

• Precision: Defined as the fraction of the retrieved information, which is relevant.

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

• Recall: Defined as the fraction of the relevant retrieved information versus all relevant information.

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

Method	Precision	Recall
Color:		
Gray scale	0.7530	0.6349
RGB	0.7668	0.8600
YCbCr	0.7306	0.8927
Texture:		
Histo bins	0.6880	0.6732
Norm Histo - Bins	0.5851	0.6547
DCT	0.7465	0.5006
Edge:	0.3220	0.4675
Edge and Color:	0.5724	0.4707

Table 2.2 The outcome of each method in precision and recall.

In the above table, the methods used (in bold face) are better than their counterparts. Precision and Recall are used below for performance evaluation [23].

Image Name	Calculation			
	Certain	Uncertain	Precision	Recall
lvi2	406	0	0.8188	0.9010
bb1	338	191	0.9595	0.6614
di1	301	133	0.7181	0.7461
hr1	248	342	0.2956	0.9309
ft1	847	29	0.3778	0.9795
lps1	227	426	0.2032	0.9056
ls1	281	306	0.3180	0.6807
lvi1	150	505	0.2863	0.8909
orh1	209	466	0.1901	0.7571
qi1	235	217	0.6099	0.8471
vc1	213	76	0.8931	0.6383
wi1	305	539	0.5393	0.8186
ds1_7-02	236	107	0.1847	0.7562
7ms_7-02	467	94	0.1727	0.9785
ft1_7-02	471	164	0.5834	0.8381
ft2_7-02	494	82	0.4994	0.7247
ft3_7-02	231	115	0.2707	0.4820
lvi1_7-02	233	168	0.3600	0.4755
ri1_7-02	140	71	0.4494	0.6555
vc1_7-02	191	391	0.1111	0.8192
wdc1_7-02	524	279	0.1297	0.9944
wi1_7-02	436	112	0.5487	0.6450
bb1-7-02	459	78	0.1637	0.4214
bb2-7-02	397	151	0.2699	0.3736
ls1-7-02	344	141	0.2789	0.8558
ps1-7-02	506	48	0.0893	0.8609
sl1-7-02	737	192	0.2967	0.7161
sl2-7-02	218	87	0.5106	0.6695

Table 2.3: The certainty of image classification in precision and recall.

2.4 Summary

In this chapter, we learned what image classification is all about, its correlated topics such as data mining and computer vision. We introduced the two programs that are used in this project which are WEKA and MATLAB. We also defined machine learning algorithms, for instance, Bayesian networks and Neural networks. Finally, we have given an idea about image classification extraction how it works using features like color and texture and edge and how to combine these features to get the best precision and recall of the classification.

Chapter 3
Features and Classifiers

3.1 Introduction

There is no benefit for classification without features that extract information from images. The features can describe shape information in a specific pattern so that the pattern classification task becomes easy through a proper procedure. The application of the algorithm on features determines the classification process. In this chapter, we will show the features that will be used in this project, their functions and how they work. We will also show the classifiers those will be applied to those features.

3.2 Images Feature extraction

Feature extraction is the process of transforming the raw pixel values from an image to more meaningful and useful information that can be used in other techniques, such as point matching or machine learning.
First, a set of these points are found because you are dealing with discrete pixels, there is always some distortion. Next, a set of measurements based on the surrounding pixels is calculated. The concept is that we can uniquely identify small patches in an image. Then we match all the measurements on one image with all the measurements on the comparison image. Once each point has found its best match, we analyze the set of matches for correspondence. The theory is that if the correspondence is coherent, you have a similar image. Feature plays a very important role in image processing. Before you obtain the advantages, pre-processing techniques are applied to the image such as binary encoding, threshold, resizing, normalization, etc. On the image from which samples were taken.

Feature extraction techniques are then applied to obtain features that will be useful in image classification and Identify them. Feature extraction techniques are useful in many image processing applications, for example. Character recognition. Because features determine the behavior of the image, it displays its location in terms of the storage capacity captured, and the competence of the classification is also evident in the consumption of time. What is called a feature? "Feature or side of something." Therefore, there must be a set of values for a given instance that is different from that of isotopes.

In the area of images, features may be crude pixels for easy problems like recognizing numbers for known Monist data sets but in natural images, the use of simple image pixels is not descriptive enough. Instead, there are two basic types of steam to follow. One of them is to use the methods of extracting an engineered attribute (e.g., SIFT, VLAD, HOG, GIST, LBP) and another stream is to know the features that characterize the given context (e.g. sparse coding, automated coding, restricted Boltzmann machines, PCA, ICA, means)[31].

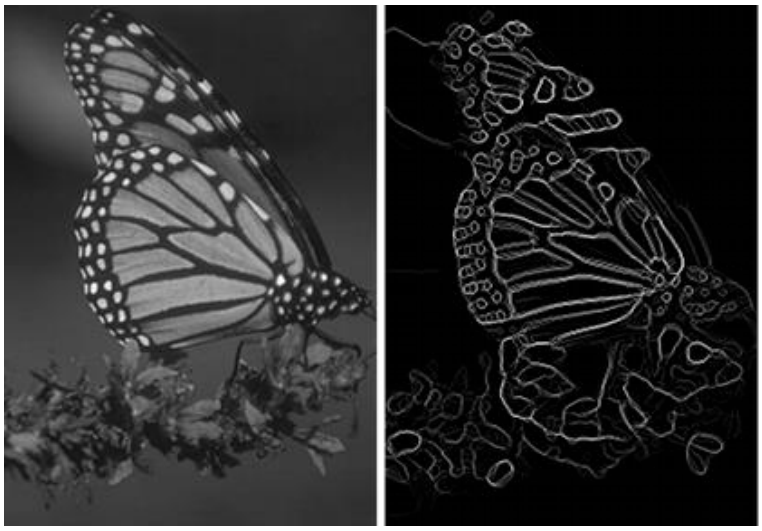


Figure 3.1: shows Edge Detection of a butterfly image [32].

3.3 Feature Set (Filters)

3.3.1 EdgeHistogram Filter:

An edge histogram in the image space represents the frequency and the directionality of the brightness changes in the image. It is a unique feature for images, which cannot be duplicated by a color histogram or the homogeneous texture features [33].

3.3.2 ColorLayout Filter:

is designed to capture the spatial distribution of color in an image. The extraction process consists of two parts; a color selection representing the grid and a separate cosine conversion with quantization. Color is the basic quality of visual content, so it is possible to use colors to describe and represent an image. This specification suggests different ways to obtain these descriptors, and a specific color-naming tool is ColorLayout, which allows describing the color relationship between the sequences or the set of images. ColorLayout captures spatial mapping of representative colors on an overlapping grid on an area or image. The representation depends on the CT transactions. This is a very small descriptor for high efficiency in browsing and quick search applications. It can be applied to still images as well as to video clips.

3.3.3 PHOG Filter:

Color images provide stronger discriminating information than grayscale images and color-based image search can be very effective for face, object, scene, and texture image classification. Some desirable properties of the descriptors defined in different color spaces include relative stability over changes in photographic conditions such as varying illumination. Global color features such as the color histogram and local invariant features provide varying degrees of success against image variations such as rotation, viewpoint and lighting changes, clutter, and occlusions. The shape and local features also provide important cues for

content-based image classification and retrieval. Local object shape and the spatial layout of the shape within an image can be described.

3.3.4 SimpleColorHistogram Filter:

is the representation of the distribution of colors in an image. For digital images, a color histogram represents the number of pixels that have colors in each of a fixed list of color ranges that span the image's color space, the set of all possible colors. Used in photo processing, color schemes can be created for any type of color space, although this term is usually used for 3D spaces like RGB or HSV. For monochrome images, the density chart can be used instead. For multi-spectral images, where each pixel is represented by a random number of measurements (for example, outside the three measurements in RGB), the color graph is N-dimensional, with N being the number of measurements taken. Each measurement has its own wavelength range from the optical spectrum, and some may be outside the visible spectrum. If the set of possible color values is small enough, each of these colors can be placed on a scale by itself; the graph is just the number of pixels that each color contains. Often, space is divided into an appropriate number of ranges, often arranged as a normal grid, each of which contains many similar color values. The color graph can also be represented and displayed as a smooth function across the color space that approximates the number of pixels. Like other graphing patterns, a color scheme is a statistic that can be considered as a continuous distribution of color values.

3.4 Classifier

The classifier model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes [14].

3.4.1 Bayesian Network:

Bayesian networks are a type of Probabilistic Graphical Model that can be used to build models from data and/or expert opinion [19].
They can be used for a wide range of tasks including prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction and decision making under uncertainty.

3.4.2 Naive Bayesian:

Naive Bayesian technique is based on the Bayesian theory and it is easy to build models that often outperform other methods of classification despite their simplicity. The Naive Bayesian algorithm predicts the measurement of patterns and relationships. Naive Bayes uses the concept of conditional probability and defines conditional probability is the probability of an event using prior knowledge.

3.4.3 Random Forest:

An Algorithm is used for classification tasks and gives a wonderful result of the classification even if the information is not tuned and because of its flexibility is one of the most commonly used algorithms [34].
Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Also, it can be used for both classification and regression problems, which form the majority of current machine learning systems.

3.5 Summary

In this chapter, we have specified the features and classifiers those which will be used in this project. The four features are chosen because they vary from each other and they are commonly used. The same reasons can be said to the three classifiers.

Chapter 4
Methodology

4.1 Introduction

There many ways to fuse two features of image classification. In this chapter, we will show our methodology of feature fusion for image classification. we'll show it step by step using screenshots.

4.2 Type of study

This project's type of study is experimental where the researcher controls one variable, and control/randomizes whatever is left of the variables. It has a control group, the subjects have been randomly allocated between the groups, and the researcher just tests one impact at any given time. It is additionally critical to know what variables you want to be tested and measured.

4.3 classification and Feature Extraction in WEKA

The Weka GUI Chooser lets you choose one of the Explorer, Experimenter, Knowledge-Flow, workbench and the Simple CLI (command line interface).
This GUI lets you load datasets and run classification algorithms. It also provides other features, like data filtering, clustering, association rule extraction, and visualization.

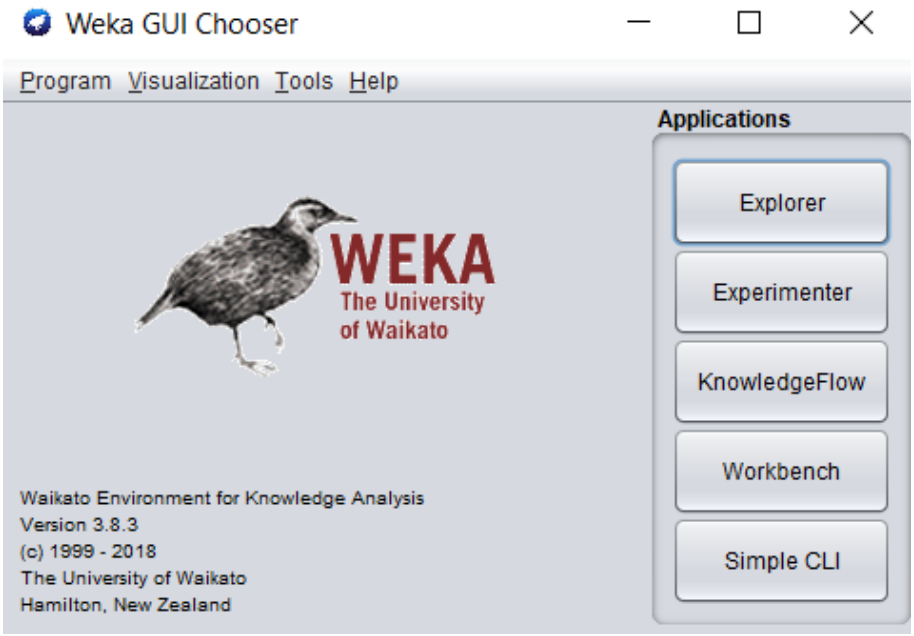


Figure 4.1: WEKA GUI Chooser

A good approach to begin exploring features and selecting them in WEKA is in the WEKA Explorer.

1. Open the Weka GUI Chooser.
2. Click the “Explorer” button to launch the Explorer.
3. Open the butterfly_vs_owl.arff to load dataset.
4. Click the “Select Attributes” tab to access the feature selection methods.

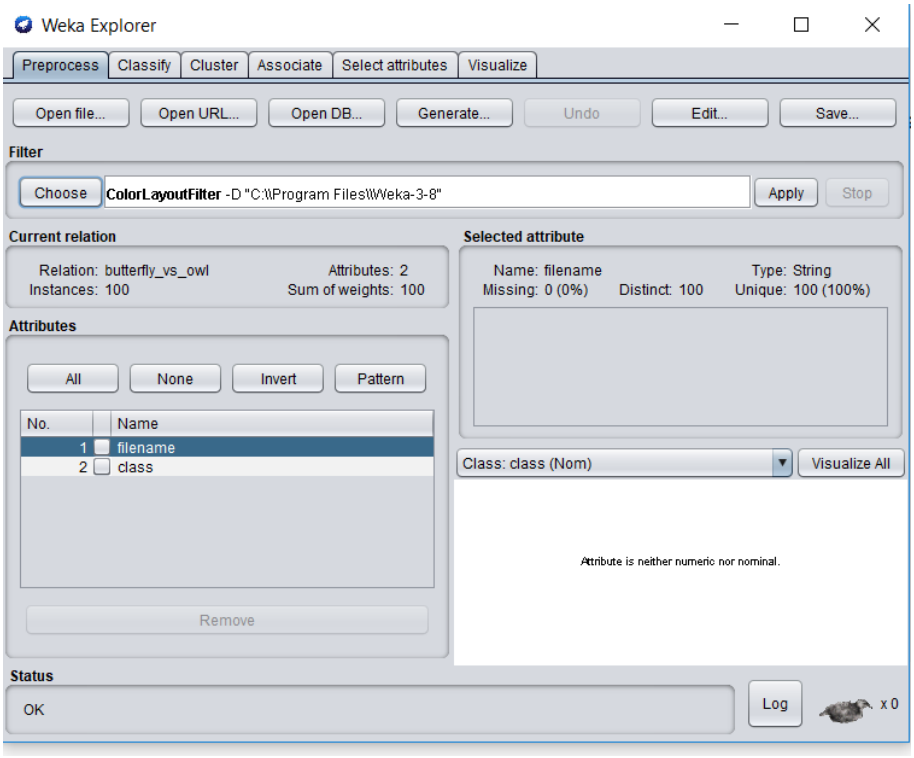


Figure 4.2: WEKA explorer

Generally, for non-image problems, the feature selection is divided into two parts:

·Attribute Evaluator

·Search Method.

Each section has multiple techniques from which to choose.

The attribute evaluator is the technique by which each attribute in your dataset (also called a column or feature) is evaluated in the context of the output variable (e.g. the class). The search method is the technique by which to try or navigate different combinations of attributes in the dataset in order to arrive on a short list of chosen features.

Some Attribute Evaluator techniques require the use of specific Search Methods. For example, the Correlation-Attribute-Eval technique used in the next section can only be used with a Ranker Search Method, that evaluates each attribute and lists the results in rank order. When selecting different Attribute Evaluators, the interface may ask you to change the Search Method to something compatible with the chosen technique.

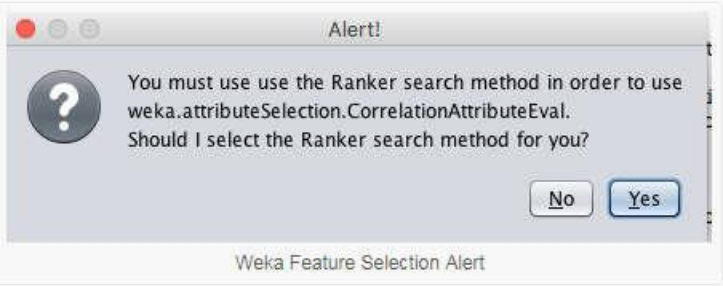


Figure 4.3

Both the Attribute Evaluator and Search Method techniques can be configured. Once chosen, click on the name of the technique to get access to its configuration details.

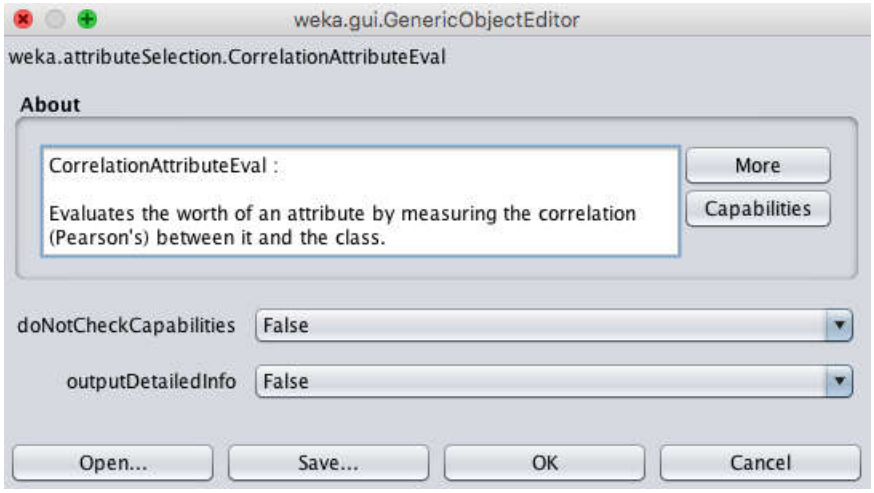


Figure 4.4: WEKA Feature Selection Configuration

Click the "More" button to get more documentation on the feature selection technique and configuration parameters. Hover your mouse cursor over a configuration parameter to get a tooltip containing more details.

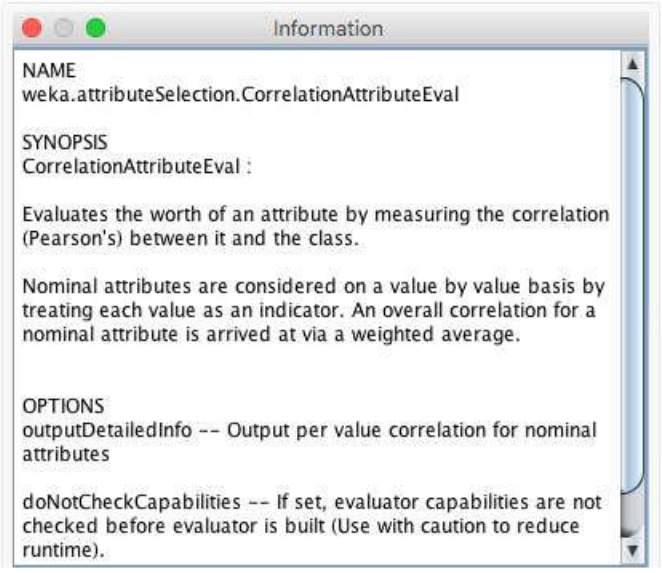


Figure 4.5: WEKA Feature Selection More Information

Now that we know how to access feature selection techniques in WEKA, let's take a look at how to use some popular methods on our chosen standard dataset.

4.4 Experimental setup

In System Design Procedure it will work in WEKA

Step1: Run WEKA then from GUI menu, select Tools→ package manager

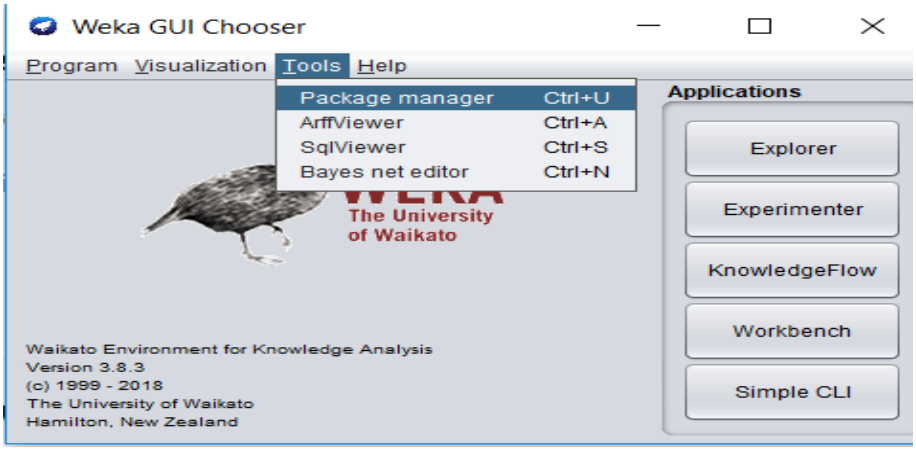


Figure 4.6: Choosing the package manager.

When the Package Manager appears, find the package “ImageFilters” in the list and click Install.

when Restart WEKA will find it installed.

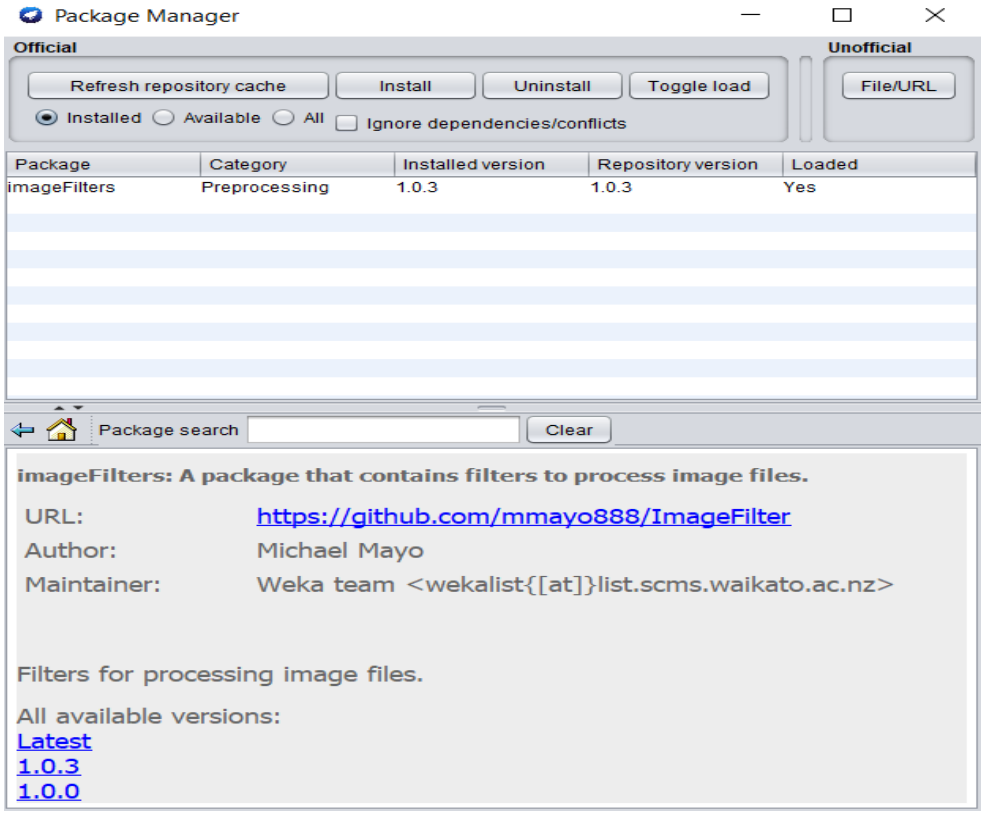


Figure 4.7: image filter installed

Step2: Test Options

The experimenter configures the test options for you with sensible defaults. The experiment is configured to use Cross-Validation with 10 folds. It is a “Classification” type problem and each algorithm + dataset combination is run 10 times (iteration control). it will make the folder of the dataset (birds, face or butterflies) for example select

image from two types of birds and then make another folder and put the image here, go to WEKA open explorer then open file and select folder which great then open show see figure 4.8.

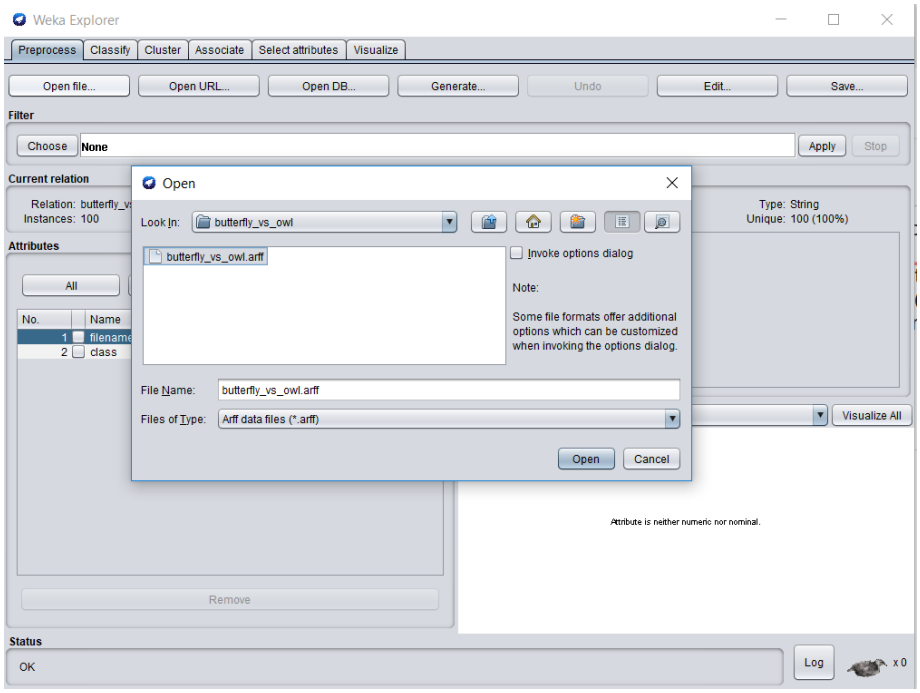


Figure 4.8: How to open a file and to select a folder of the dataset

Step3: extract features, go to filter then instance then image filters there are so many types of image filters. Select any one from this and then copy the path of the folder will be created before and paste an image direct press ok and then press apply to mean extract features so many types of this, figure 4.9.

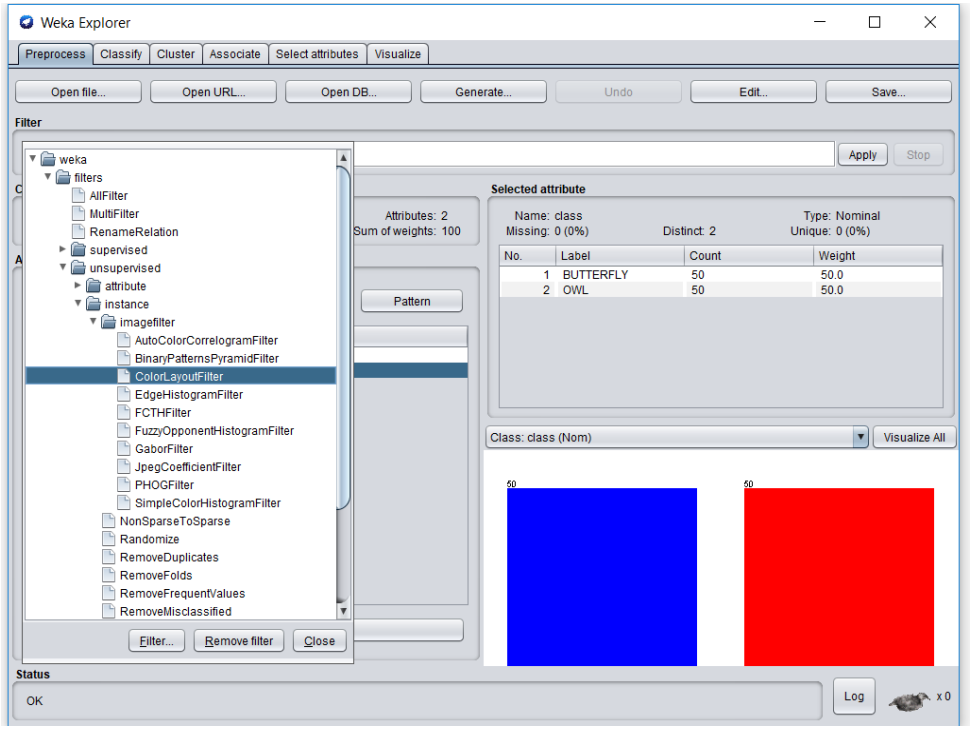


Figure 4.9: Type of image filters

Also, you can choose one of the algorithms to run our dataset Click the “Choose” button in the “classify” section. Click “ZeroR” under the “rules” selection then click the big “Start”

After running the ZeroR algorithm, you can note the results in the “Classifier output” section.

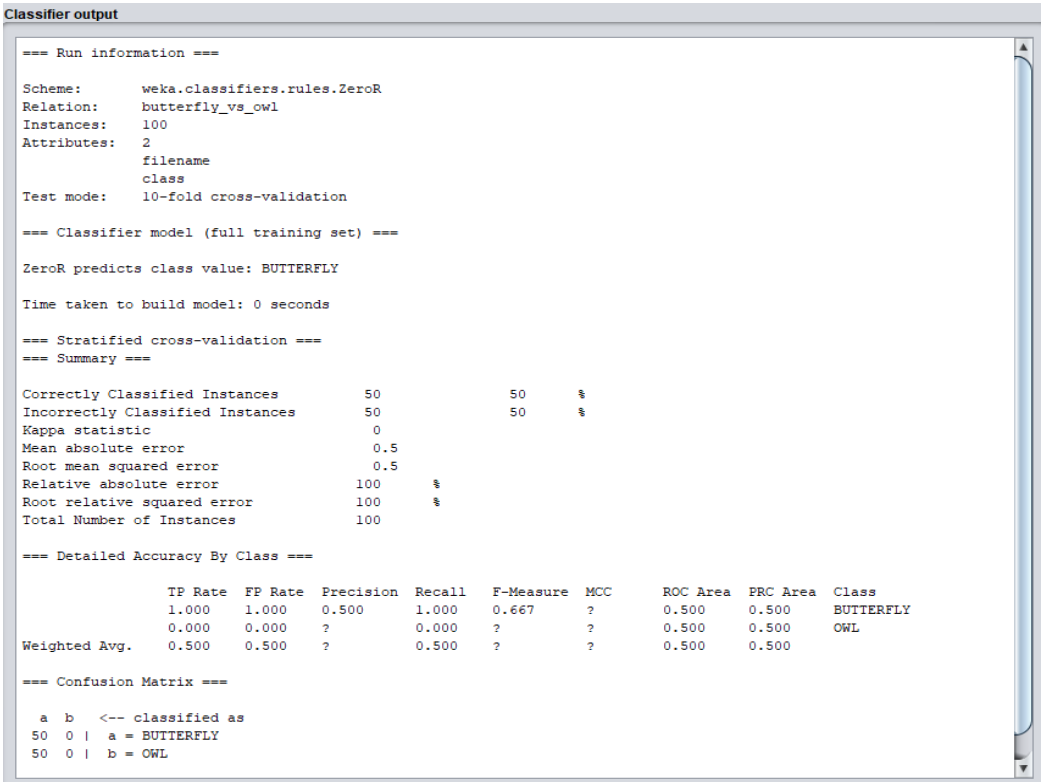


Figure 4.10: classifier output

4.5 Summary

In this chapter, we have summarized multiple steps and tools to to-do the experiments over the classifiers and the images. The purpose of this chapter was to present concepts related to classification and load datasets and run machine learning algorithm in Weka. Now, we have the skills that are required to design and run experiments using any algorithms given by Weka on datasets. We can select and confidently report results that we attain. In chapter 5, we will have experiments with various combinations of classifiers and features.

CHAPTER 5
Experimental Results

5.1 Introduction

In this final chapter, we display the results of experiments performed on the three datasets. We also show the analysis of the experiments. The results are divided into three parts. Each part is for each dataset.

In the beginning, we show the results of all features without fusion displaying only the precision, recall and F-measure for each feature with all three classifiers Bayesian Network, Naïve Bayesian and Random Forest. Then, we represent the fusion of features with a chart for each feature in every dataset that chart shows how the feature behaves with and without fusion. It also illustrations how the feature performs with different classifiers.

5.2 Experimental Results

5.2.1 Faces Dataset

Feature	Classifier	Precision	Recall	F-Measure
EdgeHistogram	BayesianNET	0.730	0.733	0.733
	Naive Bayes	0.707	0.698	0.701
	RandomForest	0.790	0.776	0.760
ColorLayout	BayesianNET	0.745	0.736	0.738
	Naive Bayes	0.756	0.727	0.730
	RandomForest	0.790	0.791	0.786
PHOG	BayesianNET	0.704	0.698	0.700
	Naive Bayes	0.716	0.709	0.711
	RandomForest	0.837	0.813	0.800
SimpleColorHistogram	BayesianNET	0.722	0.713	0.716
	Naive Bayes	0.699	0.569	0.551
	RandomForest	0.786	0.780	0.769

Figure 5.1: Results of Faces dataset

5.2.1.1 EdgeHistogram Filter:

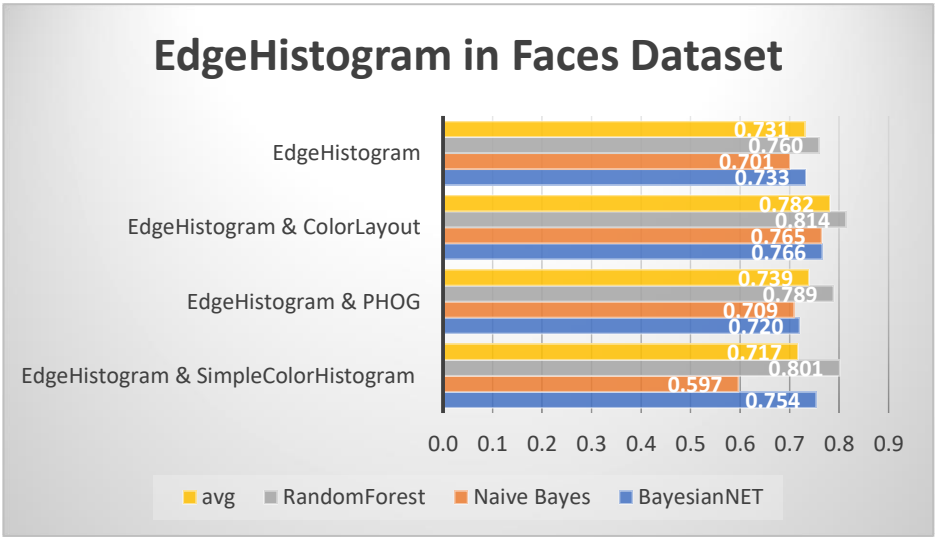


Figure 5.2: Shows EdgeHistogram results in the Faces dataset

Figure 5.2 shows the F-measure of the EdgeHistogram filter in the Faces dataset. Starting from the top row, the figure shows the F-measure of EdgeHistogram without Fusion, then the next rows show the EdgeHistogram feature fused with the ColorLayout, PHOG, and SimpleColorHistogram features.

The F-measure of EdgeHistogram in the Bayesian network classifier was 0.733, but with the Naïve Bayesian classifier, it was 0.701, while with the Random Forest classifier, it was 0.760. The total average of the three classifiers was 0.731.

Figure 5.2 also shows the F-measure of EdgeHistogram fused with the ColorLayout filter. From the figure, it can be seen that the Bayesian network gave a result of 0.766, but the result of the Naïve Bayesian was 0.765, while Random Forests shows a value of 0.814. The total average of the results was 0.782.

In the fusion of EdgeHistogram with PHOG, the F-measure of The Bayesian network has a value of 0.720, but the Naïve Bayesian has a value of 0.709 and 0.789 for Random Forests. All three classifiers have an average value of 0.739.

Besides that, The EdgeHistogram fusion with the SimpleColorHistogram was 0.754 in the Bayesian network classifier. While the F-measure of the Naïve Bayesian classifier was 0.597. On the other hand, Random Forest gets the F-measure of 0.801 which gives us 0.717 as the total average of the three classifiers.

5.2.1.2 ColorLayout Filter:

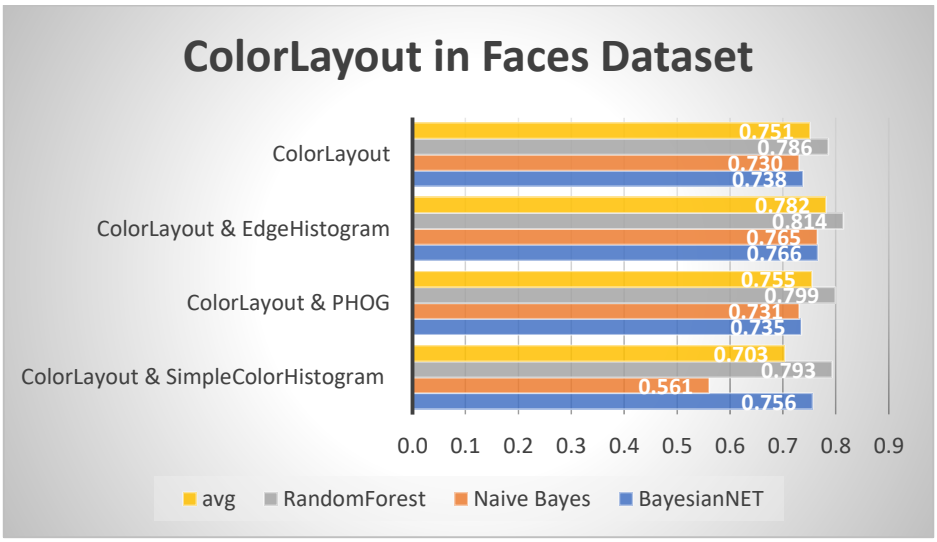


Figure 5.3: Shows ColorLayout results in the Faces dataset

Figure 5.3 clearly illustrates the F-measure of the ColorLayout filter in the Faces dataset. As shown in the figure, the F-measure of ColorLayout without Fusion is seen on the first row, and this was followed by a display of the ColorLayout feature fused with the EdgeHistogram, PHOG, and SimpleColorHistogram features as seen in the subsequent rows.

The result gotten showed that the F-measure of the ColorLayout in the Bayesian network classifier was 0.738, but with the Naïve Bayesian classifier, it was 0.730, while with the Random Forest classifier, it was 0.786. All three classifiers have a total average of 0.751.

Figure 5.3 also shows the F-measure of ColorLayout fused with the EdgeHistogram filter. From the figure, it can be seen that the Bayesian network gave a result of 0.766, but the result of the Naïve Bayesian has a value of 0.765,

while Random Forests shows a value of 0.814. An average of the three results gave a value of 0.782.

Further, in the fusion of the ColorLayout with PHOG, the F-measure of The Bayesian network gave a resulting value of 0.735, 0.731 for the Naïve Bayesian, and 0.799 for Random Forests. The total average value for the three classifiers was 0.755.

In addition, The ColorLayout fusion with the SimpleColorHistogram was 0.756 in the Bayesian network classifier, whereas the F-measure of the Naïve Bayesian classifier was 0.561. Contrarily, the F-measure of the Random Forest was 0.793 which gives 0.703 as the total average of the three classifiers.

5.2.1.3 PHOG Filter:

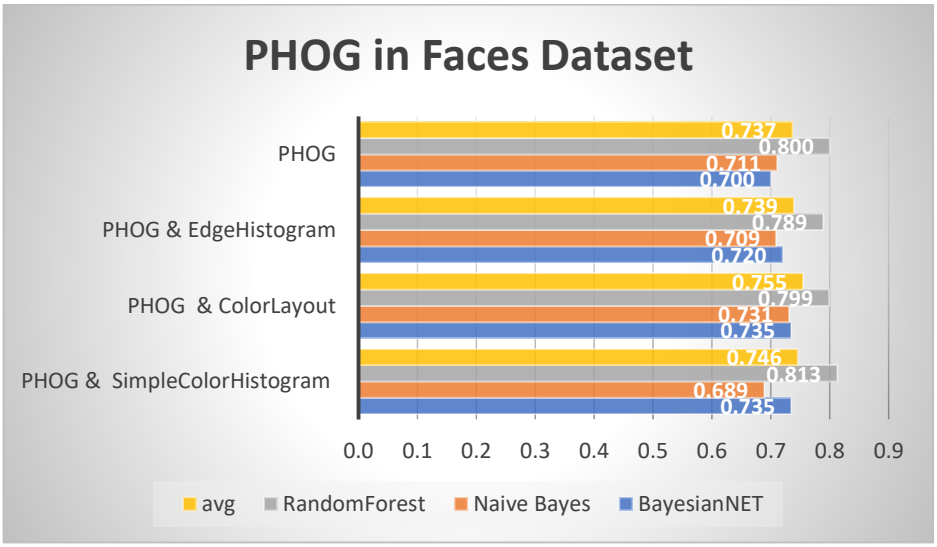


Figure 5.4: Shows PHOG results in the Faces dataset

The F-measure of the PHOG filter in the Faces dataset is shown in figure 5.4. From the first row, we can see the F-measure of PHOG without Fusion. The rows coming after show the PHOG feature fused with the EdgeHistogram, ColorLayout, and SimpleColorHistogram features.

Furthermore, 0.700 was gotten as the F-measure of PHOG in the Bayesian network classifier, but with the Naïve Bayesian classifier, the result was 0.711,

while with the Random Forest classifier, it was 0.800. Averagely, the value for the three classifiers was 0.737.

Also, displayed in figure 5.4 was the F-measure of PHOG fused with EdgeHistogram filter. The Bayesian network result as seen in figure 5.4 has a value of 0.720, 0.709 for the Naïve Bayesian, while 0.789 for the Random Forests. On average, the value of the three results was 0.739.

In the fusion of PHOG with ColorLayout, the F-measure of The Bayesian network gave a value of 0.735, 0.731 for the Naïve Bayesian, and 0.799 for the Random Forests. An average of the three classifiers gave a value of 0.755.

Besides that, The PHOG fusion with SimpleColorHistogram was 0.735 in the Bayesian network classifier, while the F-measure of the Naïve Bayesian classifier was 0.689. On the other hand, Random Forest gets the F-measure value of 0.813 which gives the total average value of the three classifiers as 0.746.

5.2.1.4 Simple Color Histogram Filter:

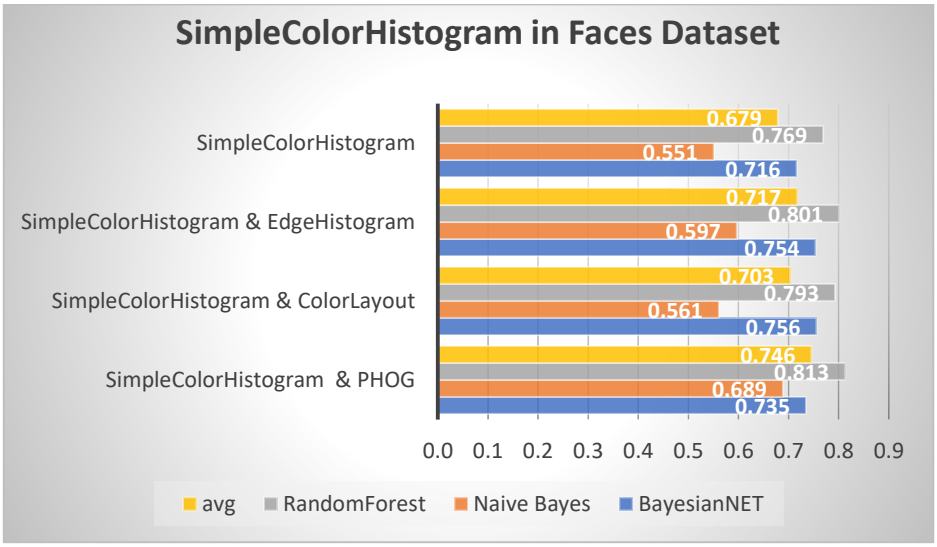


Figure 5.5: Shows SimpleColorHistogram results in the Faces dataset

Figure 5.5 shows the F-measure of the SimpleColorHistogram filter in the Faces dataset. Starting from the top row, the figure shows the F-measure of SimpleColorHistogram without Fusion. The next rows show the

SimpleColorHistogram feature fused with the EdgeHistogram, ColorLayout and PHOG features.

The F-measure of SimpleColorHistogram in the Bayesian network classifier was 0.716, but 0.551 with the Naïve Bayesian classifier, while 0.769 with the Random Forest classifier. The total average for the three classifiers was 0.679.

Figure 5.5 also shows the F-measure of SimpleColorHistogram fused with the EdgeHistogram filter. From the figure, it can be seen that the Bayesian network gave a result of 0.754, 0.597 for the Naïve Bayesian, while the Random Forests shows a result of 0.801. All three has a total average value of 0.717.

In the fusion of the SimpleColorHistogram with ColorLayout, the F-measure of The Bayesian network gets 0.756, but the Naïve Bayesian has a value of 0.561 and 0.793 for Random Forests. The three classifiers have an average value of 0.703.

More to the point, the SimpleColorHistogram fusion with PHOG gave a value of 0.735 in the Bayesian network classifier, while the F-measure of the Naïve Bayesian classifier was 0.689. Contrariwise, the Random Forest gave the F-measure value of 0.813 which gives the total average value of the three classifiers as 0.746.

5.2.2 Birds Dataset

Feature	Classifier	Precision	Recall	F-Measure
EdgeHistogram	BayesianNET	0.412	0.418	0.408
	Naive Bayes	0.437	0.453	0.429
	RandomForest	0.498	0.498	0.494
ColorLayout	BayesianNET	0.465	0.458	0.461
	Naive Bayes	0.518	0.515	0.515
	RandomForest	0.587	0.58	0.582
PHOG	BayesianNET	0.404	0.405	0.385
	Naive Bayes	0.402	0.405	0.389
	RandomForest	0.479	0.478	0.472
SimpleColorHistogram	BayesianNET	0.488	0.467	0.461
	Naive Bayes	0.352	0.34	0.311
	RandomForest	0.519	0.525	0.52

Figure 5.6: Results of Birds Dataset

5.2.2.1 EdgeHistogram Filter:

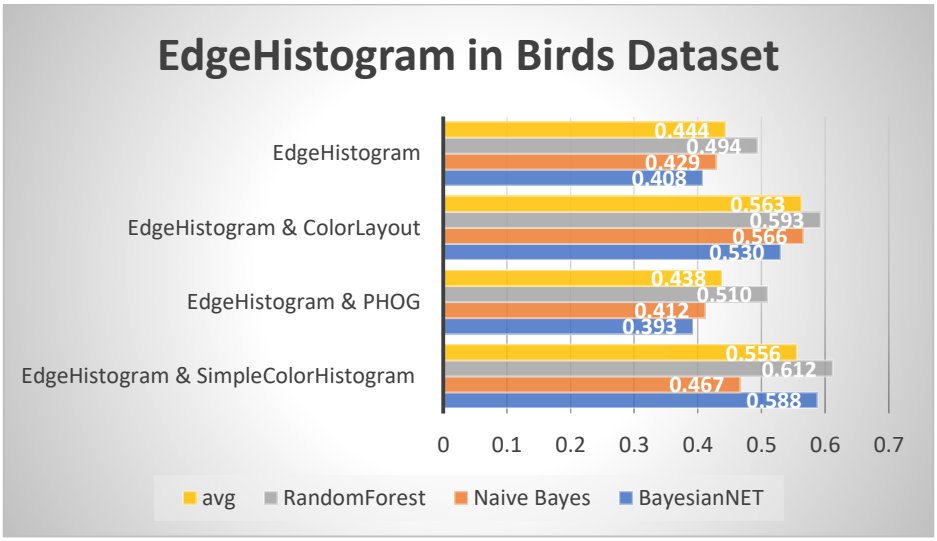


Figure 5.7: Shows EdgeHistogram results in the Birds dataset.

As illustrated by figure 5.7 is the F-measure of the Edge Histogram filter in the Birds dataset. From the uppermost row, the figure points out the F-measure of EdgeHistogram without Fusion. The rows afterward point out the EdgeHistogram feature fused with the ColorLayout, PHOG, and SimpleColorHistogram features.

In the Bayesian network classifier, the F-measure of EdgeHistogram was 0.408, but with the Naïve Bayesian classifier, it was 0.429, whereas it was 0.494 with the Random Forest classifier. The three classifiers, however, have a total average value of 0.444.

Fused with ColorLayout filter, figure 5.7 exemplifies the F-measure of EdgeHistogram. From the figure, we can clearly see that the Bayesian network gave a result of 0.530. In contrast, the Naïve Bayesian gave a result of 0.566, while the Random Forests gave a result of 0.593. The total average for the three results was 0.563.

In the fusion of EdgeHistogram with PHOG, the F-measure of The Bayesian network gave a value of 0.393, 0.412 for the Naïve Bayesian, and 0.510 for the Random Forests with an average of 0.438 for the three classifiers.

Above and beyond, The EdgeHistogram fusion with SimpleColorHistogram was 0.588 in the Bayesian network classifier. While the F-measure of the Naïve Bayesian classifier was 0.467. Conversely, the Random Forest gets the F-measure of 0.612. The total average of the three classifiers as 0.556.

5.2.2.2 ColorLayout Filter:

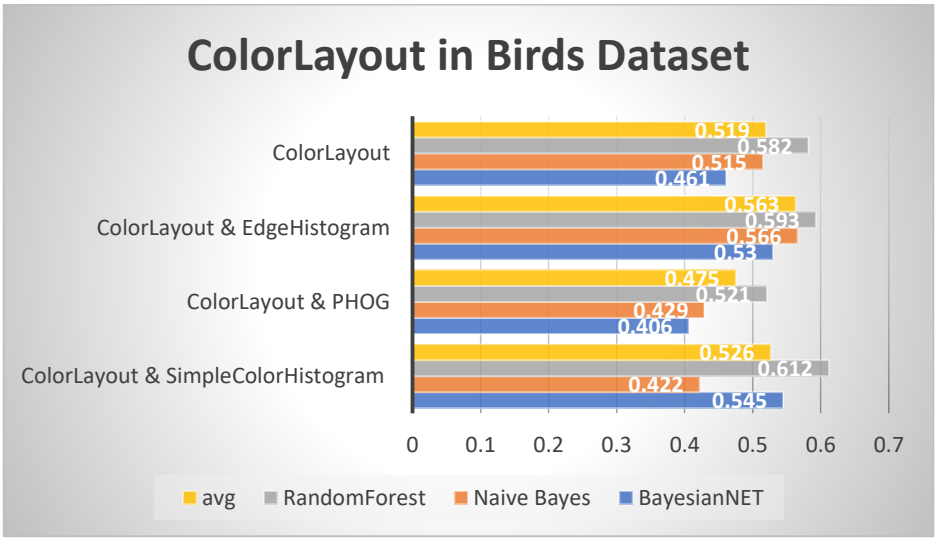


Figure 5.8: Shows ColorLayout results in the Birds dataset

The F-measure of the ColorLayout filter in the Birds dataset is clearly demonstrated in figure 5.8. Commencing with the topmost row, the figure highlights the F-measure of ColorLayout without Fusion. The rows after show the ColorLayout feature fused with the EdgeHistogram, PHOG, and SimpleColorHistogram features.

The F-measure of ColorLayout in the Bayesian network classifier was 0.461, but with the Naïve Bayesian classifier, it was 0.515, while 0.582 with the Random Forest classifier. Total average for the three classifiers was 0.519.

Also pointed out in figure 5.8 is the F-measure of ColorLayout fused with EdgeHistogram filter. The Bayesian network, as illustrated in the figure, gave a result of 0.530, 0.566 with the Naïve Bayesian, while the Random Forests gave a value of 0.593 giving the total average as 0.563.

Additionally, the F-measure of The Bayesian network gave a value of 0.406 in the fusion of ColorLayout with PHOG, 0.429 with the Naïve Bayesian, and 0.521 with the Random Forests. The average value for the three classifiers was 0.475.

On top of that, The ColorLayout fusion with SimpleColorHistogram was 0.545 in the Bayesian network classifier. While it was 0.422 with the F-measure of the Naïve Bayesian classifier. Oppositely, the Random Forest gets the F-measure of 0.612 which gives the total average of three classifiers as 0.526.

5.2.2.3 PHOG Filter:

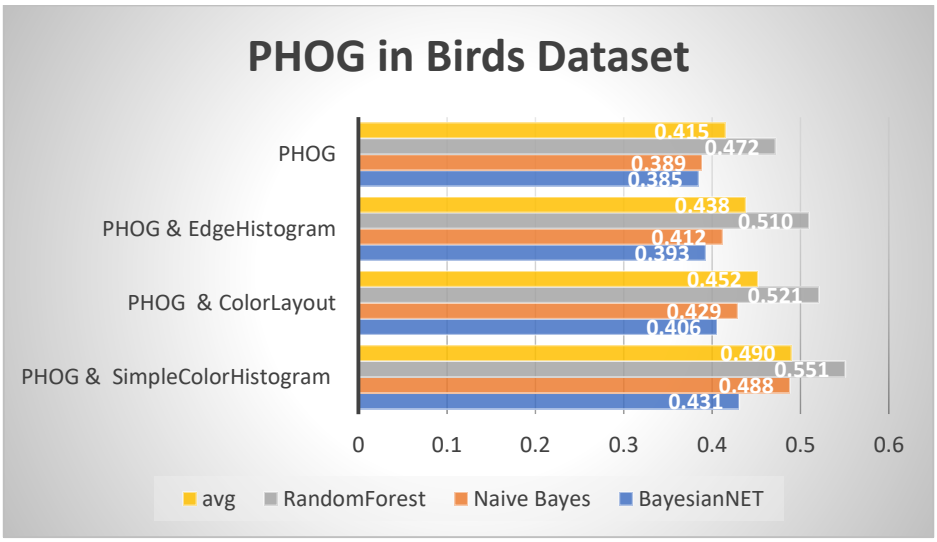


Figure 5.9: Shows PHOG results in the Birds dataset

Elucidated in figure 5.9 is the F-measure of the PHOG filter in the Birds dataset. As seen in the figure, the top row demonstrates the F-measure of PHOG without Fusion. The other rows indicate the PHOG feature fused with the EdgeHistogram, ColorLayout, and SimpleColorHistogram features.

The F-measure of PHOG in the Bayesian network classifier was 0.385, but with the Naïve Bayesian classifier, it was 0.389, while 0.472 with the Random Forest classifier. The total average for the three classifiers was 0.415.

Figure 5.9 also exemplifies the F-measure of PHOG fused with the EdgeHistogram filter. As highlighted in the figure, the Bayesian network gave a

result of 0.393, 0.412 for the Naïve Bayesian has the result, and 0.510 for Random Forests. The total average of the results was 0.438.

In the fusion of PHOG with ColorLayout, the F-measure of The Bayesian network has a value of 0.406. In contrast, the Naïve Bayesian has a value of 0.429, while the value for the Random Forests was 0.521 with an average of 0.452 for the three classifiers.

Also, The PHOG fusion with SimpleColorHistogram was 0.431 in the Bayesian network classifier. While the F-measure of the Naïve Bayesian classifier was 0.488. Then again, the Random Forest gave the F-measure of 0.551 which gives the total average of three classifiers as 0.490.

5.2.2.4 Simple Color Histogram Filter:

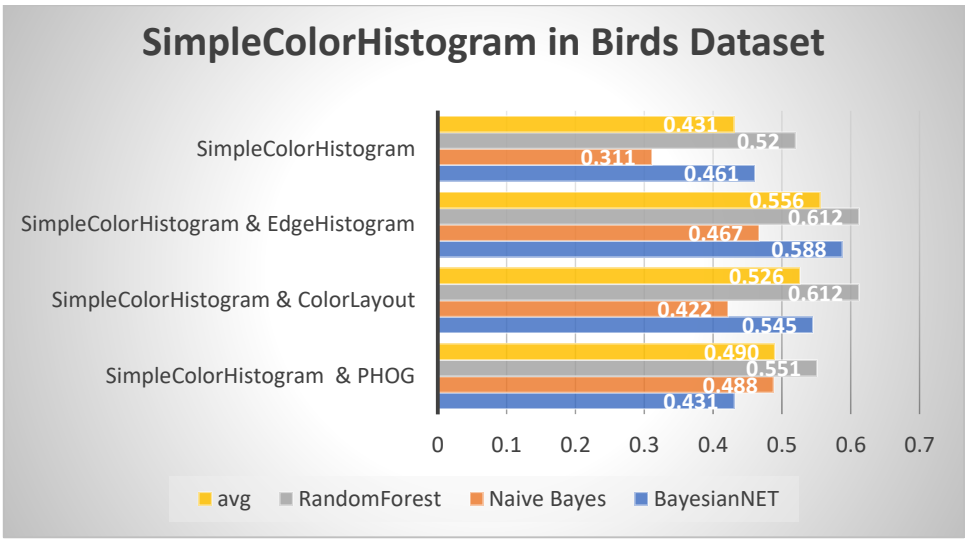


Figure 5.10: Shows SimpleColorHistogram results in the Birds dataset

Highlighted in figure 5.10 is the F-measure of the Simple Color Histogram filter in the Birds dataset. The uppermost row shows the F-measure of Simple Color Histogram without Fusion, and this was followed by the other rows which show the Simple Color Histogram feature fused with the EdgeHistogram, ColorLayout, and PHOG features.

The F-measure of SimpleColorHistogram in the Bayesian network classifier was 0.461, but with the Naïve Bayesian classifier, it was 0.311, while with the

Random Forest classifier, it was 0.520. The total average of the three classifiers was 0.431.

Also emphasized in figure 5.10 is the F-measure of SimpleColorHistogram fused with EdgeHistogram filter. As emphasized by the figure, the Bayesian network gave a result of 0.588, but the Naïve Bayesian has the result 0.467, while Random Forests shows a value of 0.612. The total average of all three results was 0.556.

In the fusion of SimpleColorHistogram with ColorLayout, the F-measure of The Bayesian network was 0.545, but the Naïve Bayesian has a value of 0.422 and 0.612 for Random Forests. The total average for the three classifiers was 0.526.

What is more, The SimpleColorHistogram fusion with PHOG was 0.431 in the Bayesian network classifier while the F-measure of the Naïve Bayesian classifier was 0.488. Contrary to this, the Random Forest gave the F-measure value of 0.551 which gives the total average of the three classifiers as 0.490.

5.2.3 Butterflies Dataset

Feature	Classifier	Precision	Recall	F-Measure
EdgeHistogram	BayesianNET	0.387	0.406	0.392
	Naive Bayes	0.492	0.444	0.438
	RandomForest	0.534	0.501	0.497
ColorLayout	BayesianNET	0.41	0.415	0.412
	Naive Bayes	0.464	0.469	0.458
	RandomForest	0.581	0.565	0.555
PHOG	BayesianNET	0.475	0.475	0.468
	Naive Bayes	0.458	0.449	0.442
	RandomForest	0.516	0.512	0.495
SimpleColorHistogram	BayesianNET	0.421	0.446	0.422
	Naive Bayes	0.271	0.248	0.23
	RandomForest	0.555	0.573	0.543

Figure 5.11: Results of Butterflies Dataset

5.2.3.1 EdgeHistogram Filter:

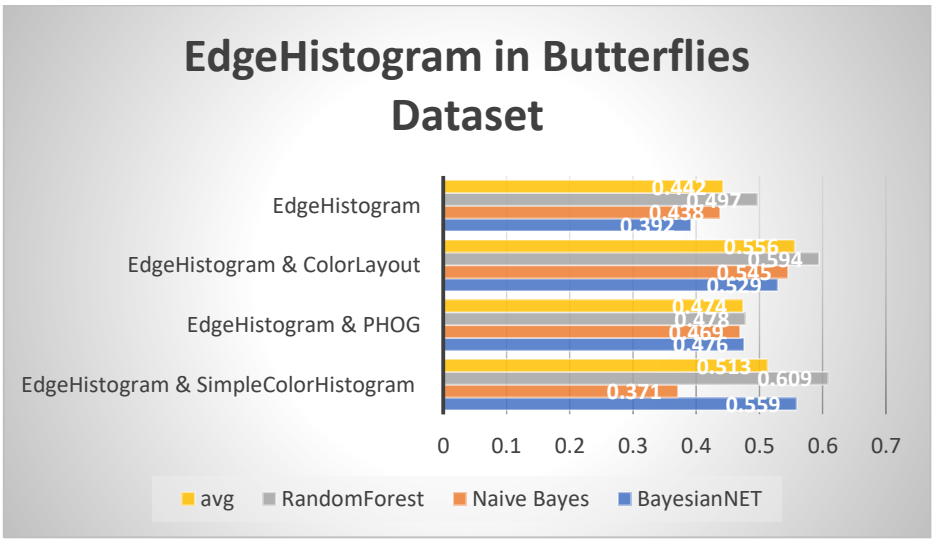


Figure 5.12: Shows Edge Histogram results in the Butterflies dataset.

Diagram 5.12 indicates the F-measure of the Edge Histogram filter in the Butterflies dataset. beginning with the first row, the diagram shows the F-measure of EdgeHistogram without Fusion. The succeeding rows are the EdgeHistogram feature fused with the ColorLayout, PHOG, and SimpleColorHistogram features.

The F-measure of EdgeHistogram in the Bayesian network classifier was 0.392, but was 0.438 with the Na ïve Bayesian classifier, while with the Random Forest classifier, it was 0.497. The total average value for the three classifiers was 0.442.

Additionally, figure 5.12 points out the F-measure of EdgeHistogram fused with ColorLayout filter. Also from the figure, we can see that the Bayesian network gave a result of 0.529, 0.545 for the Na ïve Bayesian, while 0.594 with Random Forests. The total average value for all three results was 0.556.

In the fusion of EdgeHistogram with PHOG, the F-measure of The Bayesian network gets 0.476, 0.469 for the Na ïve Bayesian, and 0.478 for Random Forests. The average value for the three classifiers was 0.474.

Further, The EdgeHistogram fusion with SimpleColorHistogram was 0.559 in the Bayesian network classifier, while the F-measure of the Na ïve Bayesian

classifier was 0.371. Conversely, the Random Forest gets the F-measure value of 0.609 which gives the total average value of three classifiers as 0.513.

5.2.3.2 ColorLayout Filter:

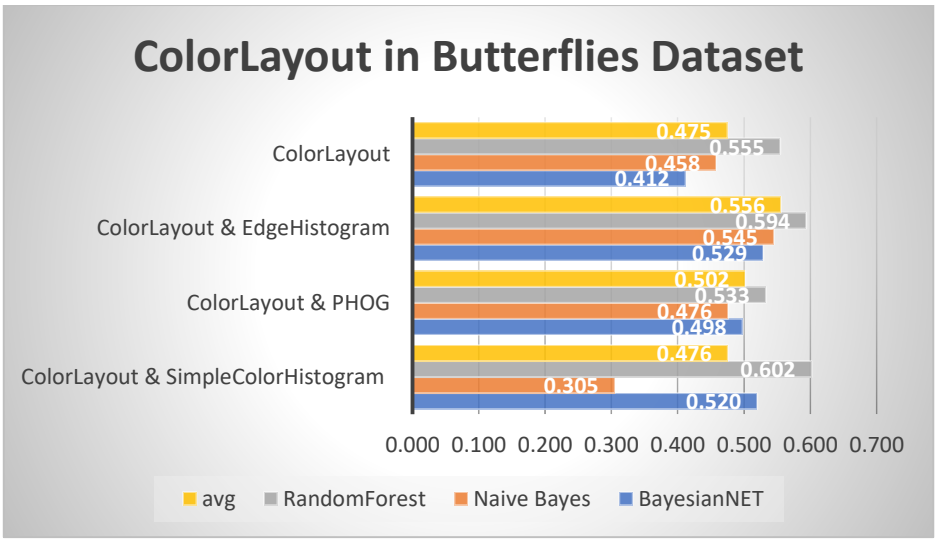


Figure 5.13: Shows ColorLayout results in the Butterflies dataset

Chart 5.13 shows the F-measure of the ColorLayout filter in the Butterflies dataset. The first row shows the F-measure of ColorLayout without Fusion. The rows afterward are the ColorLayout feature fused with the EdgeHistogram, PHOG, and SimpleColorHistogram features.

The F-measure of ColorLayout in the Bayesian network classifier was 0.412, but with the Naïve Bayesian classifier, it was 0.458, while 0.555 with the Random Forest classifier. The total average value for the three classifiers was 0.475.

Chart 5.13 also indicates the F-measure of ColorLayout fused with EdgeHistogram filter. As seen in the chart, the Bayesian network gave a result of 0.529, 0.545 for the Naïve Bayesian, and 0.594 for Random Forests. The total average for all three results was 0.556.

In the fusion of ColorLayout with PHOG, the F-measure of The Bayesian network gets 0.498, 0.476 for the Naïve Bayesian, and 0.533 for Random Forests. The average value for the three classifiers was 0.502.

As well, The ColorLayout fusion with SimpleColorHistogram was 0.520 in the Bayesian network classifier, while the F-measure of the Naïve Bayesian classifier was 0.305. Conversely, the Random Forest gets the F-measure of 0.602 which gives the total average value of 0.476 for three classifiers.

5.2.3.3 PHOG Filter:

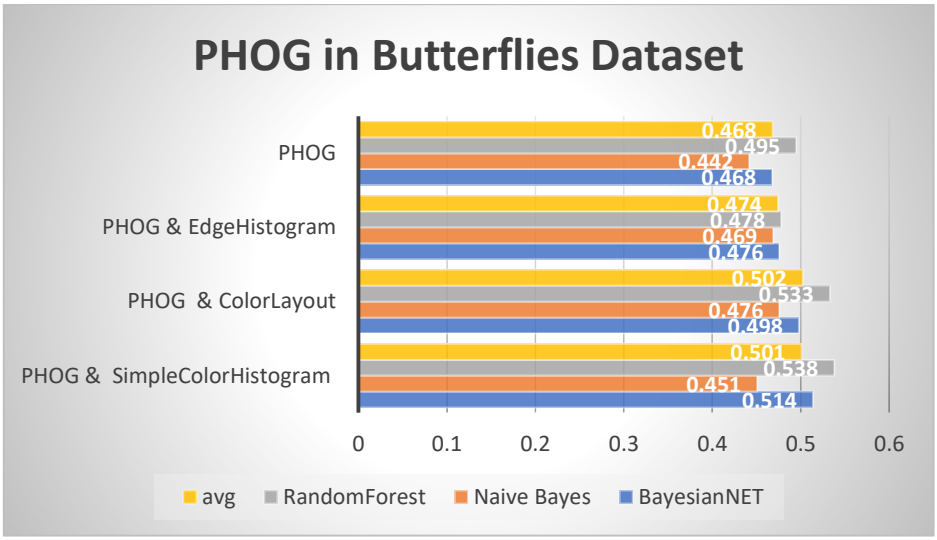


Figure 5.14: Shows PHOG results in the Butterflies dataset

Show in diagram 5.14 is the F-measure of the PHOG filter in the Butterflies dataset. Explaining from the topmost row, the diagram shows the F-measure of PHOG without Fusion. The subsequent rows are the PHOG feature fused with the EdgeHistogram, ColorLayout, and SimpleColorHistogram features.

The F-measure of PHOG in the Bayesian network classifier was 0.468, but with the Naïve Bayesian classifier, it was 0.442, while 0.495 with the Random Forest classifier. The total average value for the three classifiers was 0.468.

Diagram 5.14 also highlights the F-measure of PHOG fused with the EdgeHistogram filter. From the figure, it can be seen that the Bayesian network gave a result of 0.476, 0.469 for the Naïve Bayesian, and 0.478 for Random Forests. The total average value of all three results was 0.474.

In the fusion of PHOG with ColorLayout, the F-measure of The Bayesian network gets 0.498, 0.476 for the Naïve Bayesian, and 0.533 for Random Forests. The average value for the three classifiers was 0.502.
In addition to that, The PHOG fusion with SimpleColorHistogram was 0.514 in the Bayesian network classifier, while the F-measure of the Naïve Bayesian classifier was 0.451. Oppositely, a value of 0.538 was obtained for the Random Forest which gives the total average value of three classifiers as 0.501.

5.2.3.4 Simple Color Histogram Filter:

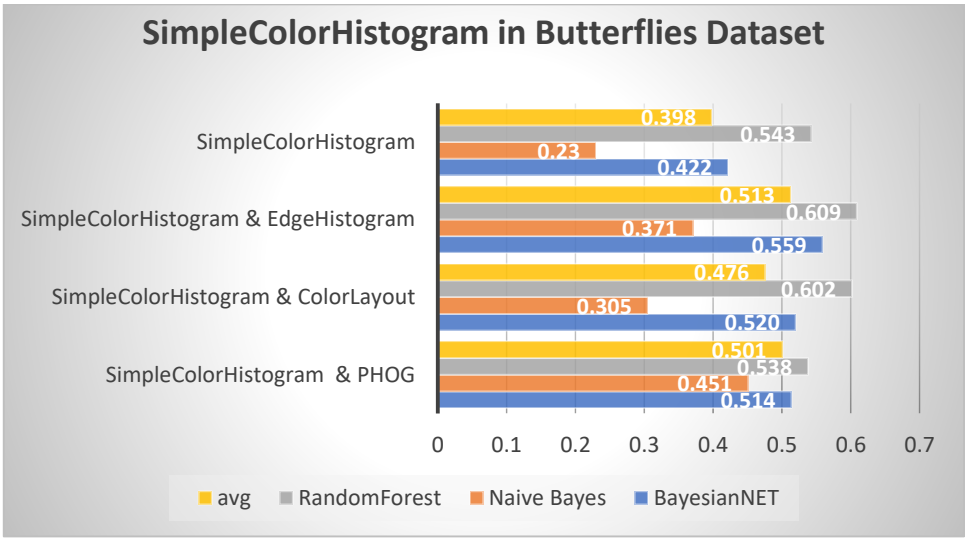


Figure 5.15: Shows SimpleColorHistogram results in the Butterflies dataset.

The F-measure of the SimpleColorHistogram filter in the Butterflies dataset is clearly illustrated in figure 5.15. Looking top row, we can see that the figure shows the F-measure of SimpleColorHistogram without Fusion. Then, the succeeding rows are the SimpleColorHistogram feature fused with the EdgeHistogram, ColorLayout, and PHOG features.
The F-measure of SimpleColorHistogram in the Bayesian network classifier was 0.422, but with the Naïve Bayesian classifier, it was 0.230, while 0.543 with the Random Forest classifier. The total average value for the three classifiers was 0.398.
Figure 5.15 also illustrated the F-measure of SimpleColorHistogram fused with the EdgeHistogram filter. As seen in the figure, the Bayesian network gave a

result of 0.559, 0.371 for the Naïve Bayesian, while 0.609 for Random Forests. The total average value of all three results was 0.513.

In the fusion of SimpleColorHistogram with ColorLayout, the F-measure of The Bayesian network gets 0.520, 0.305 for the Naïve Bayesian, and 0.602 for Random Forests. The total average for the three classifiers was 0.476.

Furthermore, The SimpleColorHistogram fusion with PHOG was 0.514 in the Bayesian network classifier, while the F-measure of the Naïve Bayesian classifier was 0.451. On the other hand, Random Forest gets the F-measure of 0.538 which gives the total average of three classifiers as 0.501.

Chapter 6
Results Discussion and Conclusion

6.1 introduction

Results are beneficial, but it will be much more valuable when they are analyzed. In this final chapter, we will discuss the results of experiments applied to the four features.

Our analysis will be based on the features. We'll compare the features performance with each other and with the three classifiers in every dataset.

We will conclude with our opinion regarding feature fusion whichever it is useful or not for image classification.

6.2 Experimental Analysis

6.2.1 Analysis of EdgeHistogram:

It seems that the EdgeHistogram usually gets a very good boost from fusion especially while applying the Random Forest classifier. However, it gives you a minor boost or decreases the performance while being fused with PHOG filter so the PHOG and EdgeHistogram are a bad combination. The EdgeHistogram is very effective when it is combined with ColorLayout that it gives you a better result than EdgeHistogram alone every time. For instance, a 35% increase in the F-measure in the Butterflies dataset using Bayesian Network.

The fusion of EdgeHistogram and SimpleColorHistogram looks like it depends on the classifier because it can give you surprisingly the optimist results like 42.6% increase in the Butterflies dataset using Bayesian Network and it can give the worst results as well such as 15% decrease in the same dataset using Naive Bayesian. With the random forest and the Bayesian Network classifiers, the EdgeHistogram gets its best fusion with SimpleColorHistogram two out of three datasets but with the Naive Bayesian this combination decreases performance and give the worst results two out of three datasets.

Therefore, with the right classifier and the correct fusion, the EdgeHistogram should get a great increase that shows how effective is the feature fusion.

6.2.2 Analysis of ColorLayout:

Unlike the EdgeHistogram the ColorLayout feature usually gets only a slight boost from the feature fusion and that may be due to its impressive performance on its own. Its best combination is with the EdgeHistogram always increasing the performance. On the other hand, The ColorLayout feature gives unstable results when it is fused with PHOG such as 20.8% Increase in the Butterflies dataset applied by Bayesian Network and 11.9% decrease in the Birds dataset applied by the same classifier and that might be due to the nature of PHOG. ColorLayout and SimpleColorHistogram seem like a decent combination when the right classifier is chosen. For instance, when we used the Bayesian Network classifier the SimpleColorHistogram increased the F-measure of ColorLayout by 26.2% in the Butterflies dataset. However, when these two features are applied by Naive Bayesian the SimpleColorHistogram always reduces the F-measure of ColorLayout significantly such as 23.1% decrease in the Faces dataset.
Overall, ColorLayout is an efficient feature that does good on its own but with the feature fusion, it can get a considerable boost.

6.2.3 Analysis of PHOG:

The PHOG seems like it gives different performance based on the datasets more than the features those who are fused with it. The performance is varying with each dataset. For example, when we fused the PHOG with the SimpleColorHistogram applied by Naive Bayesian we got a 24.4% boost in the Birds dataset and a 3.09% decrease in the Faces dataset and 2.03% increase the Butterflies dataset. The PHOG is the only feature that gives better results in the Butterflies dataset more than its results in the Birds dataset.
The PHOG doesn’t regularly gets a great increase from fusion but sometimes it scores good results like the 10.3% increase when it was fused with ColorLayout in the Birds dataset using Random Forest. The PHOG best combination is with the SimpleColorHistogram. The performance of the two-features combined is

very decent considering that the PHOG usually gets the least results between the four features. The Bayesian Network Always gets a boost from fusion in all three datasets so it is not useless to fuse features with PHOG.
In conclusion, the PHOG is an unstable and unpredictable feature that probably counts on the nature of data used more than the classifier.

6.2.4 Analysis of SimpleColorHistogram:

It looks like SimpleColorHistogram works badly with Naive Bayesian that it only got the F-measure 0.23 in the Butterflies dataset. Giving that poor result it will surely get greater outcome when it is fused with any feature. However, the SimpleColorHistogram is an efficient feature that it scores outstanding results with other classifiers such as 0.543 in the Butterflies dataset.
The fusion of features with SimpleColorHistogram seems very effective because the results of fusion are more likely to outperform the results without fusion such as the fusion with EdgeHistogram 32.4% increase in the Butterflies Dataset using the Bayesian network Algorithm.
The best combination of SimpleColorHistogram is confidently the EdgeHistogram this feature gives SimpleColorHistogram the best performance in two datasets the Birds and the butterflies whereas the PHOG leads the Faces dataset. However, we say that the EdgeHistogram is better for SimpleColorHistogram than PHOG because it improves the F-Measure every time while the PHOG did get a 6.5% reduction in the Birds dataset when the fusion was applied by Bayesian Network Classifier.
The ColorLayout feature is similar to EdgeHistogram that it always boosts the performance such as the 17% boost in the Birds dataset using Random Forest.
The greatest increase in this entire research was obtained by the SimpleColorHistogram and PHOG. The later feature has boosted the first by over than 96% while applying Naive Bayesian in the Butterflies dataset. However, this number can be understood because of the very poor performance by the unfused SimpleColorHistogram classified by Naive Bayesian.
In general, the SimpleColorHistogram got the most benefit from feature fusion with only two reductions out of 27 results.

6.3 Major Findings

- If properly adjusted (fusion rule selection), the fusion of features should increase classification performance.
- Fusion performance depends on the selection of classifiers used for fusion.
- Fusion performance is also affected by the rule selected for fusion.
- Fusion may reduce the performance if the fusion rule is not selected and investigated.
- Some features depend on the nature of images data more than the classifier or the feature that are fused with them.
- The disadvantage of fusion is the increased amount of processing time.

6.4 Conclusion

Finally, it seems that feature fusion is very helpful in the image classification area that it increases the accuracy of classification, but the fusion rule must be properly selected.

Some classifiers do not work efficiently with some features and some features are bad with some images data so the right combination should be carefully selected in order to improve the image classification.

It is better to test a dataset that is hard to be classified than an easy one such as Faces dataset because the differences between the outcomes are barely noticeable.

Overall, the feature fusion is recommended to be used, but you need to know the nature of features and which feature combination can benefit you the most.

6.5 Future work

We have only used the fusion of two features. However, it will be interesting to see how the feature fusion works with many features.

It will be very interesting to see how the feature fusion would perform with many features together like 8 features. With the use of many features, the performance might decrease due to conflicts between features. It also would be exciting to experiment with the fusion of classifiers over features and then, we can compare the results to this project.
Another future goal is the use of deep learning and applies the fusion of feature on big data.

REFERENCES

[1] Lingxi Xie, Qi Tian ,Bo Zhang. (2016). IEEE Transactions on Circuits and Systems for Video Technology. Simple Techniques Make Sense: Feature Pooling and Normalization for Image Classification. 26 (1), 1251 – 1264

[2] Guyon, I. and Elisseeff, A. (2003) An Introduction to Variable and Feature Selection. The Journal of Machine Learning Research, 3, 1157-1182.

[3] Song, D.J. and Tao, D.C. (2010) Biologically Inspired Feature Manifold for Scene Classification, IEEE Transaction on Image Processing, 19, 174-184.

[4] Alex Krizhevsky. (2017). the CIFAR-10 and CIFAR-100 datasets.Available: http://www.cs.utoronto.ca/~kriz/index.html. Last accessed 20th April 2018.

[5] Computational Vision Group. "http://www.vision.caltech.edu/html-files/archive.html" Archive. Computational Vision, March 17, 2005. Web. 21 October 2018

[6] Svetlana Lazebnik, Cordelia Schmid, Jean Ponce. "http://www-cvr.ai.uiuc.edu/ponce_grp/data/#texture" Datasets for Computer Vision Research. The Ponce Group, October 2005, Web. 20 October 2018

[7] Anon, (2018). [image] Available at: https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcTpfbaaT_XrIfvlTH0cNf2ocyuAXq2dK7NkRueGp6OA4HbzW4SM-gdT [Accessed 16 Nov. 2018].

[8] Anon, (2018). [image] Available at: https://ai2-s2-public.s3.amazonaws.com/figures/2017-08-08/6b59cea2f573ecae7f14adf47b984f1095a8c341/7-Figure5-1.png [Accessed 16 Nov. 2018].

[9] A Comparative Study of Classification Techniques in Data Mining Algorithms. (2017). International Journal of Modern Trends in Engineering & Research, 4(7), pp.58-63.

[10] Scilab.io. (2018). [image] Available at: https://scilab.io/typical-processes-of-data-mining/ [Accessed Oct. 2018].

[11] Anon, (2018). [ebook] Available at: http://people.scs.carleton.ca/~c_shu/Courses/comp4900d/notes/lect1_intro_2up.pdf [Accessed 10 Nov. 2018].

[12] Anon, (2018). [image] Available at: https://medium.com/@eternalzer0dayx/demystifying-convolutional-neural-networks-ca17bdc75559 [Accessed 10 Nov. 2018].

[13] Anon, (2018). [image] Available at: https://www.oreilly.com/library/view/deep-learning-for/9781788295628/d9eef68b-4586-472c-bd5c-a244471a277f.xhtml [Accessed 10 Nov. 2018].

[14] Medium. (2018). Machine Learning: Classification Models – Fuzz – Medium. [online] Available at: https://medium.com/fuzz/machine-learning-classification-models-3040f71e2529 [Accessed 10 Nov. 2018].

[15] Anon, (2018). [image] Available at: https://www.semanticscholar.org/paper/Feature-Selection-for-Classification%3A-A-Review-Tang-Alelyani/310ea531640728702fce6c743c1dd680a23d2ef4 [Accessed 10 Nov. 2018].

[16] Anon, (2018). [ebook] Available at: https://www.cs.auckland.ac.nz/courses/compsci367s1c/tutorials/IntroductionToWeka.pdf [Accessed 10 Nov. 2018].

[17] Cimss.ssec.wisc.edu. (2018). What is Matlab. [online] Available at: https://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm [Accessed 4 Nov. 2018].

[18] Anon, (2018). [ebook] Available at: http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-12e794df-d22f-45ed-aed6-7033891bab07/c/015-20_Boyko_Sokil.pdf [Accessed 10 Nov. 2018].

[19] introduction, B. (2018). Introduction to Bayesian networks. [online] Bayesserver.com. Available at: https://www.bayesserver.com/docs/introduction/bayesian-networks [Accessed 10 Nov. 2018].

[20] Anon, (2018). [image] Available at: https://www.bayesserver.com/docs/introduction/bayesian-networks [Accessed 15 Nov. 2018].

[21] Anon, (2018). [ebook] Available at: http://jatit.org/volumes/research-papers/Vol5No1/1Vol5No6.pdf [Accessed 10 Nov. 2018].

[22] Anon, (2018). [image] Available at: https://www.slideshare.net/Tricode/deep-learning-stm-6 [Accessed 10 Nov. 2018].

[23] G. Foschi, Patricia & Kolippakkam, Deepak & Liu, Huan & Mandvikar, Amit. (2002). Feature Extraction for Image Mining.. 103-109.

[24] Y. Rui, T. Huang and S. Chang. Image retrieval: current techniques, promising *directions and open issues.* Journal of Visual Communication and Image Representation,
10(4): 39-62, April 1999.
Based Visual Query. SPIE Symposium on Visual Communications and Signal Processing, May 1995.

[25] R. Gonzalez and R. Woods. Digital Image Processing, Addison-Wesley publications Co,March 1992.

[26] A. Jain, A. Vailaya. *Image Retrieval using Color and Shape*. Pattern Recognition, 29(8):

1233-1244, August 1996.

[27] S. F. Chang and J.R. Smith. Extracting Multi-Dimensional Signal Features for Content-
Based Visual Query. SPIE Symposium on Visual Communications and Signal Processing, May 1995

[28] Sharlee Climer, Sanjiv K. Bhatia. Image Database indexing using JPEG coefficients. The journal of the Pattern Recognition Society, Pattern Recognition 35 (2002) 2479-2488

[29] MATLAB Image Processing Toolbox User's guide, Version 3, 1993-2001 by Mathworks Inc., www.mathworks.com
W. Ma, Y. Deng and B. S. Manjunath. Tools for texture/color based search of images.

[30] I. Witten and E. Frank. Data Mining: Practical Machine Learning tools and techniques
with Java Implementations, Morgan Kaufmann publications, 2000.

[31] Anon, (2019). [ebook] Available at: http://ai.stanford.edu/~quocle/LeKarpenkoNgiamNg.pdf [Accessed 15 Mar. 2019].

[32] Anon, (2018). [image] Available at: http://cs.brown.edu/courses/cs143/2011/results/proj2/mmschnei/pics/butterfly.png [Accessed 16 Nov. 2018].

[33] Anon, (2019). [ebook] Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.182.5856&rep=rep1&type=pdf [Accessed 7 Apr. 2019].

[34] NiklasDonges. (2018).the Random Forest Algorithm.Available: https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd. Last accessed 12th Nov 2018.