Using Machine Learning techniques to predict the most suitable surfboard for a surfer

LIVERPOOL HOPE UNIVERSITY

School of Mathematics, Computer Science and Engineering

Department of Mathematics and Computer science

Intelligent Systems coursework

Manex Ormazabal Arregi (20211323)

03/05/2021

Abstract:

In the surfing world, one of the most discussed topics is to know what type of surfboard a surfer needs and choosing the right board for a person never has been easy, as it depends on many factors. However, thanks to Artificial Intelligence this problem can be solved, giving a surfer a more detailed and precise result of the board that he would need to use. This project aims to predict which surfboard would be the most suitable for a surfer, using Machine Learning prediction techniques and considering the features of the person and the type of waves.

1. Introduction

Today there is a great variety of different models of surfboards and for each model a large list of different sizes can be found, but the difficulty is to find the right model and size for any kind of person. It is very important to know which board size a surfer needs, especially for those who are getting started in the surfing world, since when people buy a surfboard, often they do not know which size or type they need or simply they buy the wrong one which then leads to more difficulties to surf and this could result in a big frustration for the person.

Nevertheless, choosing the surfboard is quite relative, being that it does not only depend on the surfer or sea conditions, but also depends on what style of surfing the surfer has or which board this person likes. Therefore, the topic is a bit ambiguous, and it is difficult to say what is the ideal surfboard. However, it is possible to give an idea about which type of surfboard a person should use depending on their features and sea conditions and this helps to guide the user in choosing a board that is most suitable for them.

Nowadays, there are already some tables that can calculate the volume of the surfboard that the person needs depending on their weight and surfing level. Nevertheless, since these methods only use the weight and surfing level of the surfer to calculate the volume of the board, the result of the volume obtained is very generic and is not accurate.

Besides, it gives only the result of the volume of the board, regardless of other details such as the length or model of the board, which makes it even more difficult for the person to choose

the board. These were some of the main motivations in this project to develop a more complex and accurate way to automatically predict the right surfboard type for any kind of surfer.

This project focuses on creating a *Machine Learning* (ML) program to predict 4 different characteristics of the surfboard that are the standard volume, real volume, length, and shape. For this, the surfer`s level, age, fitness, weight, and height are used as features, as well as the size of waves. Thus, the results obtained will be more precise than the current methods and it will give the user more detailed information of the board that they need. For all these features and targets, a dataset is created to feed the algorithm and, in this project,, in order to make predictions, it was opted to use Machine Learning techniques.

There are many different tools to work with ML, and in this program, three different types of algorithms have been used, which are *K-Nearest Neighbours* (KNN), *Multilayer Perceptron Regressor* (MLP), and *Random Forest Regressor* (RF). Further, with the KNN algorithm, a graph is represented to study how the accuracy changes depending on the number of neighbours that are taken to make the prediction.

In this paper, first, the research about surfing and Machine Learning will be mentioned in the methodology section, followed by the design of the dataset and the procedure of the code. Then, the results obtained with the program will be discussed, as well as the conclusion and the future work that the project will require.

2. Methodology:

In this section, the research made about surfing and Machine Learning will be discussed. First, how all the data about surfing has been collected from other sources, and a brief explanation of the research made about all those technical aspects that are necessary to know when choosing a board, will be made. After this, also a brief description of the research of Machine Learning and its different methods to predict used in this project will be discussed.

2.1. Surfing approach

As mentioned in the introduction, one of the troubles in this project has been finding the data to work with. The data needed for this project is very specific and therefore it could not be possible to find a dataset that contains all these data.

However, some online sources have been found, which contain a lot of information about surfing, an explanation of how the right board for a surfer is calculated, and what data is needed for this.

So, finally, a dataset was opted to create from scratch, and in order to write the data in a proper way and with some sense, some information about surfing has been searched from other sources of the internet and some research has been made to learn about what are the features of surfers, waves and surfboards that are needed to consider when choosing the right surfboard and to understand how these features are used to calculate the different measures of the board.

One of the targets in this project is to predict what is the generic volume of the surfboard for the person. The generic volume or standard volume of a surfboard is one of the most typical and used measures to know which surfboard a person needs, based only on his weight and surfing level. It is very important to know which volume of surfboard a person needs, since depending on the weight and level of the person it will require more or less volume (Surf Nation, 2021 and Pierson, 2018). The more the person weights, the higher the volume of the board he will need, so the board floats more.

In addition to this, depending on the surfing level of the person, the volume of the board will also vary, but in this case, the less experience the surfer has, the more volume he will need, and thus, it will be easier for the person to stand on the board, paddle and catch waves (Pierson, 2018). Surf Nation (2021) website suggests a table for the generic volume calculation, that gives the result of the standard volume of the board, based on the weight and surfing level of the person (Figure 1).

For the project, this table has been used to collect the data of different weights and surfing levels as features and the target which in this case is the generic or standard volume of the board. Later it will be explained how the "real volume" of the board is calculated, which is a more specific volume calculation for each person and the result is more precise.

Weight (Kg)	Advance +	Advance	Inter. / Adv	Intermediate	Beginner
35 & Under	15.05	15.75	16.80	21.00	25.90
40	16.40	17.20	18.80	23.60	29.20
45	18.00	18.90	20.70	26.10	32.40
50	19.00	20.50	22.50	28.50	35.50
55	19.80	21.45	23.65	30.25	37.95
60	21.00	22.80	25.20	32.40	40.80
65	22.75	24.70	27.30	35.10	44.20
70	24.50	26.60	29.40	37.80	47.60
75	26.25	28.50	31.50	40.50	51.00
80	28.00	30.40	33.60	43.20	54.40
85	29.75	32.30	35.70	45.90	57.80
90	31.50	34.20	37.80	48.60	61.20
95	33.25	36.10	39.90	51.30	64.60
100	35.00	38.00	42.00	54.00	68.00
105	36.75	39.90	44.10	56.70	71.40
110	38.50	41.80	46.20	59.40	74.80

Figure 1: Surfboard standard volume calculation table (Surf Nation, 2021)

After predicting the surfboard generic volume, the next step is to calculate the real surfboard volume for the person, which is a more specific volume calculation to obtain a more optimum volume result (Surf Nation, 2021). For this, the result of the generic volume obtained previously from the table, would be multiplied by the factors of the age and fitness of the person, shown in Figures 2 and 3. The Fitness of the surfer is the factor that indicates how many times a week the person surfs. Depending on what threshold is the age and fitness of the person, the factors that will be used to multiply will be different.

So, according to the calculation of the real volume of the surfboard, it holds:

$$V_R = V_G * F_A * F_F$$

Where the V_R is the Real volume, V_G the generic or standard volume, F_A the age factor and the F_F the fitness factor.

Age	Factor
0 - 30	Add 0% (0.00)
31 - 50	Add 8% (1.08)
51 - 60	Add 20% (1.20)
61+	Add 30% (1.30)

Example: Age 33. 30.40 x 1.08 (8%) = 32.83 ltrs

Figure 2: Age factor calculation table (Surf Nation, 2021)

Fitness	Factor
Excellent - Surfing 4 times per week or other training 4 times per week	Add 0% (0.00)
Good - Surfing 3 times per week or other training 3 times per week	Add 5% (1.05)
Average - Surfing 2 times per week or other training 2 times per week	Add 10% (1.10)
Poor - Surfing 1 times per week or other training 1 times per week	Add 20% (1.20)

Example: Fitness is Average - 32.83 x 1.10 (10%) = 36.11

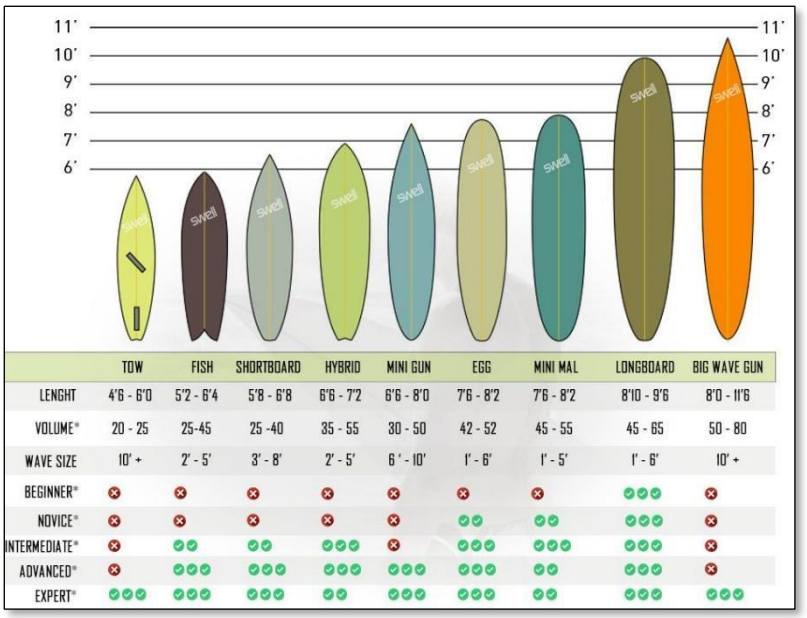
Figure 3: Fitness factor calculation table (Surf Nation, 2021)

Apart from the volume of the board, there are also other 3 more features to consider when choosing the right surfboard, which are the length and shape of the board, and the type of waves. Each of these factors depends on the other, and they need to go together.

The surfboard length is calculated based on the surfer´s height, the volume of the board that he needs, and the waves condition. Depending on the height of the person, he would need to use a longer or shorter board, and although it is not very important to consider this factor, normally as a reference it is said that for an intermediate level surfer the right surfboard length needs to be one palm higher than his height (Surfertoday, 2021). Also, the volume of the board can determine whether the length of the board is longer or shorter, so after calculating the real volume that the person needs, that result can be used as one feature to calculate the length of the board.

The third factor is the type of waves, and this also conditions the calculation of the surfboard length, as depending on the size of waves a longer or shorter board will be needed, although this also is dependent on the level of the surfer (Figure 4). Normally, for small size waves, a longboard is used, which has more volume to float, and hence, easier to catch the

small waves, and easier to control. However, for high-size waves, it is more suitable to use a shortboard if the surfer has more experience, being that shortboards are faster, easier for maneuvers, and does not sink with bigger waves (Surfscience, 2021).



	TOW	FISH	SHORTBOARD	HYBRID	MINI GUN	EGG	MINI MAL	LONGBOARD	BIG WAVE GUN
LENGHT	4'6 - 6'0	5'2 - 6'4	5'8 - 6'8	6'6 - 7'2	6'6 - 8'0	7'6 - 8'2	7'6 - 8'2	8'10 - 9'6	8'0 - 11'6
VOLUME*	20 - 25	25-45	25 -40	35 - 55	30 - 50	42 - 52	45 - 55	45 - 65	50 - 80
WAVE SIZE	10' +	2' - 5'	3' - 8'	2' - 5'	6 ' - 10'	1' - 6'	1' - 5'	1' - 6'	10' +
BEGINNER*	✗	✗	✗	✗	✗	✗	✗	✓✓✓	✗
NOVICE*	✗	✗	✗	✗	✗	✓✓	✓✓	✓✓✓	✗
INTERMEDIATE*	✗	✓✓	✓✓	✓✓✓	✗	✓✓✓	✓✓✓	✓✓✓	✗
ADVANCED*	✗	✓✓✓	✓✓✓	✓✓✓	✓✓✓	✓✓✓	✓✓	✓✓✓	✗
EXPERT*	✓✓✓	✓✓✓	✓✓✓	✓✓	✓✓✓	✓✓✓	✓✓	✓✓✓	✓✓✓

Figure 4: Table of different surfboard types (Surfertoday, 2021)

The last feature of the surfboards is the shape, and although it may be thought that it is just an aesthetic feature, it is very important to know which shape the person needs. Choosing the shape of the surfboard will depend on the volume the person needs, which is calculated based on the weight and the surfing level, the length of the board that he needs, and the type of wave that he is going to surf. Besides, it is important to know that the type of shape of the board and the type of waves will also determine the style of surfing that is required for the surfer, which is shown in Figure 5.

There are many different types of shapes, as shown in Figure 4, and each model can be used in some types of waves but not in others (Surfscience, 2021). So, for example, a longboard has a rounded nose, large and thick body, which has a big volume and therefore, more buoyancy and stability for smaller waves or for beginners, who need more ease to catch a wave or stand up. However, a shortboard has a pointed shape nose, short and thin body, and a very lightweight, which makes the board very fast, easier to control in medium or bigger waves, but

much less stability and buoyancy, so it is not valid for small waves and beginners, as it will sink (Surfertoday, 2021).

So, in conclusion, depending on both the conditions of the surfer and waves, a good approximate result of the volume, length, and shape of the board can be obtained.
Figure 5 gives a guide to show what shape of surfboard is used based on the size of waves and the style of surfing. The horizontal axis represents the size of waves, whereas the vertical axis represents the surfing style that each type of surfboard is suitable for. I.e., the closer the board is from the "pitching" point, the more suitable it is to be used for more extreme surfing and fast maneuvers such as "aerials", "tube rides", "snaps", etc. However, the closer the board is from the "rolling" point, the calmer surfing will be and the maneuvers will be slower.

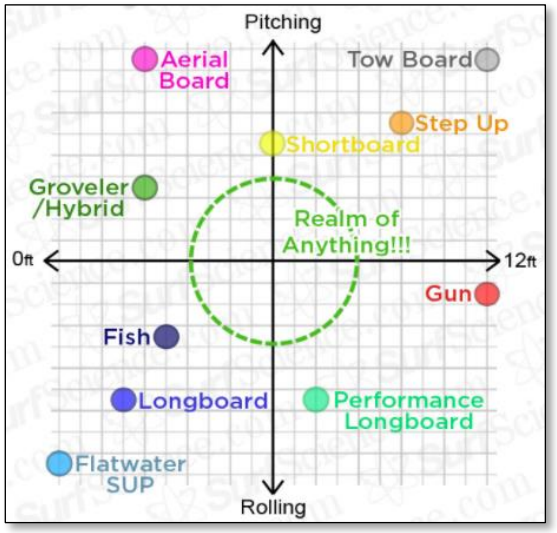


Figure 5: Surfboard models for different waves (SurfScience, 2021)

2.2. Machine Learning approach

Machine Learning (ML) is a branch of the broad field of *Artificial Intelligence* (AI), which using algorithmic methods and based on experiences, can improve performances, or make accurate predictions (Mohri et al., 2018 and Alpaydin, 2020). Inside this field, there are three different learning approaches, which differ on the type of data that the learner receives and the method of training and testing the data they use (Mohri et al., 2018 and Mirjalili et al., 2019). These three learning methods are supervised, unsupervised, and the reinforcement learning.

Supervised learning is the method that has been used in this project. In this learning method, all the data given to the algorithm is labelled, and also, the targets are given so that the algorithm can identify the target associated with each feature (Mirjalili et al., 2019 and Zhu et al., 2009). Some data is kept for training the algorithm, also known as "training data", and some other data is used to test the algorithm, which is known as "testing data".

The training data comprises correctly classified features with their corresponding targets, and the algorithm is trained to recognize the features with its associated targets and that it can match them correctly. Thus, the aim of this is that by giving a new feature it can know how to classify it according to its corresponding target (Mirjalili et al., 2019). Depending on the type of task, in supervised learning it can be distinguished two, which are Classification and Regression (Mohri et al., 2018 and Zhu et al., 2009).

In classification problems, the output variables, also called targets, are of categorical type, i.e., the data can be classified into different classes. One of the most used examples is image recognition, where depending on the features that the algorithm receives, it can recognize an image or also, assign a category to an image (Mohri et al., 2018). If the algorithm does not predict the correct image a "punishment" will be applied to in order that it learns.

Supervised learning is quite common in classification problems since the aim is usually that the machine can learn the pre-determined classification of features and targets, and the classification can be suitable for any problem provided that the classification is useful and easy to determine (Zhang, 2010). In this project, for example, the classification is used to predict the surfboard shape.

However, in regression problems, the targets are real values, and continuous. The aim is to predict the corresponding value for each feature, and in this case, the penalty is given depending on how far the predicted value is from the desired value (Mohri et al., 2018). For example, in this project one example of a regression task would be, to predict the surfboard volume. There are many values the volume can have and depending on the prediction accuracy of the algorithm, the predicted result will be closer or further away from the target value.

There are many types of algorithms in supervised learning and in this case, the three most common types that have been used in this project will be mentioned, which are *K-Nearest Neighbours* (KNN), *Multilayer Perceptron* (MLP), and *Random Forest* (RF).

K-Nearest Neighbours, or KNN, is considered one of the most simple and easiest types of algorithms in supervised learning to implement, used for classification and regression (Rajaguru et al., 2017 and Khalid et al., 2013). The principle of this algorithm is to classify a new unknown data based on the type of labels of its nearest neighbours in the feature space (Figure 6). It has two phases, one is the training phase, where the training samples and class labels are stored, and the other is the classification phase, where a query or an unlabelled data point is classified by assigning a label, which is "the most recurrent among the K training samples nearest to that query point" (Rajaguru et al., 2017, p.31).

So, the function of this algorithm is basically that given a new unknown data point, the algorithm has to assign it a type of label, according to how many training samples or instances are taken, which are defined by the K value, and the type of labels that have its closest data instances around it. Looking at Figure 6, for example, the type of label of the unknown data point would be a red triangle.

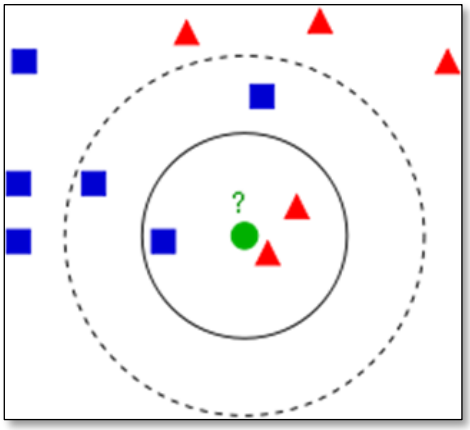


Figure 6: KNN Feature space (Rajaguru et al., 2017)

The next algorithm also used in this project is *Multilayer Perceptron*, also known as MLP, and is the most used feed forward *Artificial Neural Network* (ANN) model, which uses the back-propagation training algorithm (Ramchoun et al., 2016 and Mohanty, 2019).

An MLP algorithm, as shown in the Figure 7, consists of three or more layers, which are the input layers, hidden layers, and output layers (Kulala et al., 2017). Basically, the performance of this algorithm is that inputs feed on features, and there are some coefficients called “weights" that are between the inputs and hidden layers, which are added to the input values (features) and after they are stored in the hidden layers. After applying some functions in the hidden layers, the result is verified in the output layers and if it is not the desired prediction, the previous steps are repeated adjusting the weights parameters, until a correct result is obtained, i.e., an accurate prediction is reached.

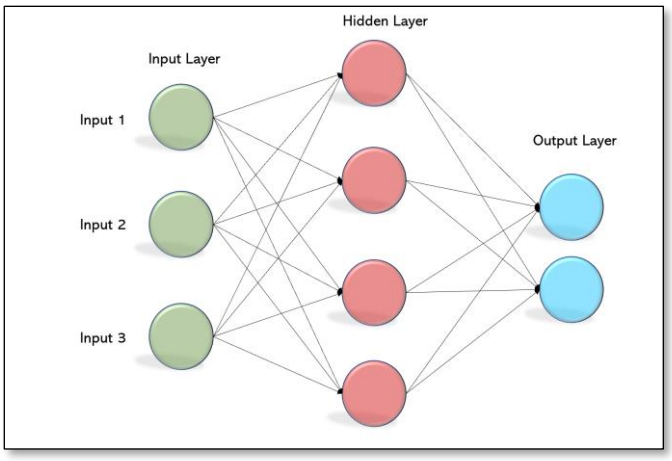


Figure 7: Neural network schematic representation (Mohanty, 2019)

Random Forest or RF is another type of machine learning algorithm mainly used for classification, but also used in regression, and it is based on the decision tree method, although it does not work the same. The problem in decision trees is that at the end the tree tends to memorize the solutions rather than generalize the learning. This is also known as overfitting. The solution to avoid this is to create many trees and have them work together.

The RF algorithm consists of multiple decision trees for classification, but unlike the decision tree method, utilizes feature randomness to create an uncorrelated forest of decision trees (IBM Cloud Education, 2020). The idea is that input data or features are passed by different decision points of the trees until reaching the predicted result in each output, which will be counted as votes (Figure 8). Thus, each decision tree will have several votes, and the classification tree that contains the most votes, which will be selected by the forest, will be the final result of the prediction of the algorithm (Livingston, 2005).

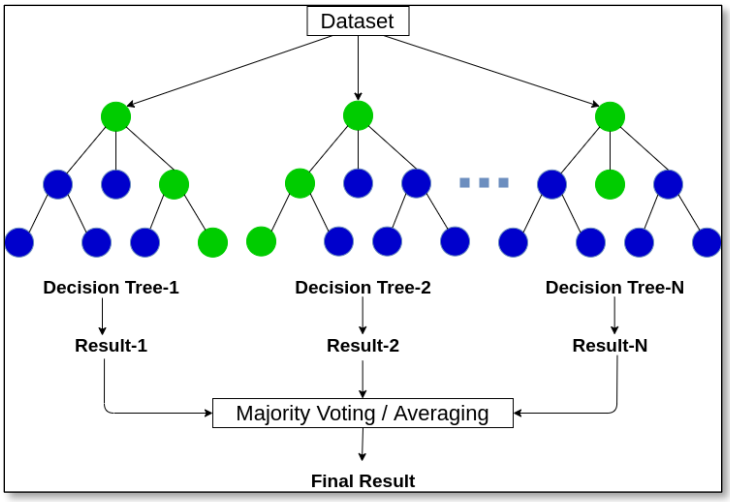


Figure 8: Random Forest schematic representation (Sharma, 2020)

3. Design and procedure:

This section will talk about all the design that has been made for the project and the code that has been written to develop the machine learning program. First, how the dataset has been created will be discussed, and then, how the code of the program has been done will be explained step by step.

3.1. The Dataset

The dataset has been created in an Excel program (Figure 9), so that later from the python program the data can be imported and thus can be used to work with it. The range of values of the data, which is also the number of rows, used in this dataset is 80 and there are 12 columns for features and targets. In total, 6 features and 4 targets have been gathered, although two of the targets (surfboard length and real volume) will also be used as features, which will be shown later.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Surfing_Level	Surfing_Level_N	Surfer_Age	Surfer_Fitness	Surfer_Weight	Surfer_Height	Wave_Size	Surfboard_Generic_Volume	Surfboard_Real_Volume	Surfboard_Length	Surfboard_Shape	Surfboard_Shape_N
2	Beginner	1	7	3	35	130	1	25.9	27.195	5	Softboard	1
3	Novice	2	10	2	35	134	3	21	23.1	5.2	Malibu	3
4	Intermediate	3	13	4	35	138	5	16.8	16.8	5.6	Hybrid/Evolutive	4
5	Advanced	4	11	1	35	147	9	15.75	18.9	5.8	Tow In	7
6	Expert	5	9	2	35	131	8	15.05	16.555	5.1	Shortboard	5
7	Beginner	1	14	2	40	155	6	29.2	32.12	6.1	NOT SUITABLE	0
8	Novice	2	8	4	40	125	4	23.6	23.6	6	Malibu	3
9	Intermediate	3	12	3	40	152	4	18.8	19.74	6.2	Hybrid/Evolutive	4
10	Advanced	4	14	4	40	134	10	17.2	17.2	6.1	Tow In	7
11	Expert	5	15	1	40	144	3	16.4	19.68	6.2	Malibu	3
12	Beginner	1	16	3	45	143	2	32.4	34.02	6.5	Softboard	1
13	Novice	2	15	2	45	142	7	26.1	28.71	6.4	NOT SUITABLE	0
14	Intermediate	3	19	3	45	166	6	20.7	21.735	6	Shortboard	5
15	Advanced	4	42	2	45	150	1	18.9	22.45	5.8	Fish	6
16	Expert	5	19	4	45	169	11	18	18	5.9	Tow In	7
17	Beginner	1	13	1	50	145	3	35.5	42.6	6.5	NOT SUITABLE	0
18	Novice	2	35	2	50	164	6	28.5	33.86	6.5	NOT SUITABLE	0
19	Intermediate	3	18	3	50	172	6	22.5	23.625	6.4	Shortboard	5
20	Advanced	4	16	2	50	143	3	20.5	22.55	6.2	Hybrid/Evolutive	4
21	Expert	5	32	4	50	154	8	19	20.52	6	Tow In	7
22	Beginner	1	14	3	55	148	2	37.95	39.84	8.5	Softboard	1
23	Novice	2	31	2	55	154	1	30.25	35.93	7.4	Longboard	2
24	Intermediate	3	20	4	55	156	1	23.65	23.65	6.3	Longboard	2
25	Advanced	4	17	1	55	139	6	21.45	25.74	5.4	Shortboard	5
26	Expert	5	16	4	55	143	4	19.8	19.8	5.2	Shortboard	5

Figure 9: The created dataset

The first feature is the surfing level of the surfer, and there are two columns for it. The first column represents the names of different levels used in surfing and the only purpose of this column is to show the name of each respective number, but they will not be used in the program. The other column represents the numbers that go from 1 to 4 in a repetitive way, and correspond to each name of the level, which will be used to work with in the program.

The next two features are the age and the fitness of the surfer, and they are only used for calculating the real volume of the board, as mentioned in the previous section. Depending on the value of age and fitness, the corresponding coefficients of these features are multiplied by the respective value of the "generic volume", which is in the same row, to obtain the result of the real volume.

The rest of the features, which are the weight and height of the person and the wave size, are probably the most important data to consider when calculating the surfboard type. The weight is in kilograms and the values go from 35 to 110, while the height is represented in centimetres with the lowest number of 130 and the highest number is 208. The wave size is shown in feet, starting from 1 foot, the smallest wave, to 12 feet, big waves.

Although most of the features have the numbers in a random order, two of the features (level and weight) have the values following a specific order. This is because these two features are used for predicting the surfboard generic volume, which uses the same system to calculate

as the table shown previously (Figure 9), and according to this table, it is needed to have the values of the level and weight features in order.

The next 5 columns represent the targets and are identified by a different colour. The first target is the generic volume of the board, which is represented in litres and, as previously explained, it is calculated based on the features of the level and weight of the surfer. Next to this column is the target of the real volume (also the values in litres), which is a calculation of the previously obtained volume, multiplied by the factors of the age and fitness of the surfer.

The next column is the length of the board, whose values are in feet, and it will be predicted based on the surfer's height and level, the real volume obtained previously, and the size of the wave.

Finally, the last two columns are used for the target of different surfboard shapes. The first column contains the names of different shapes of boards, whereas the second column represents the numbers assigned to each name of the shape, in order that these numbers can be used in the program to make predictions. Unlike with the feature of the level, where the column of names is only used for showing the name in the dataset, in this case, a dictionary is created in the program which after predicting the number that represents the shape, it shows the corresponding name of the shape, of the predicted number.

3.2. The code

In this project, all the code of the program has been made using the programming language Python. This language is very popular nowadays, being that it is very easy to learn and to use, for its readable code, but, at the same time, it is a high-level language, which practically can be used for any kind of applications, such as Artificial intelligence, Data science, Big data, web development and much more (Gowrishankar, 2018). Also, another advantage is that Python offers a lot of libraries of all kinds, and it is considered a multi-paradigm language, because of its facilities for object-oriented programming, imperative and functional (Challenger-Pérez, 2014).

The code of the program has been developed using the *Integrated Development Environment* (IDE) *PyCharm*. This IDE is one of the most popular code editors, and it has been developed by *JetBrains*. It offers many features, such as code analysis, graphical debugger, and an integrated unit tester, which makes itself a very comprehensive coding environment (Kaur Arora, 2020).

First, all the libraries that will be needed throughout the program are imported. These include "*pandas*", which is used to import the dataset into the program, "*numpy*" to work with arrays, "*matplotlib*", which is used to plot the accuracy results of KNN, and "scikit-learn", which is the package that contains all the modules to work with machine learning. From this last package, we import the KNN, MLP, Random forest, and training algorithms that will be used to make the predictions. To make it easier, some imported module names have been abbreviated, such as Pandas as "*pd*", Numpy as "*np*" and Matplotlib as "*plt*" (Figure 10).

```
## START THE PROGRAM:

# STEP 1:
# Import all the libraries that we will use:
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split  # Training algorithm
from sklearn.neighbors import KNeighborsRegressor  # KNN regressor
from sklearn.neural_network import MLPRegressor  # MLP Regressor
from sklearn.ensemble import RandomForestClassifier  # Random forest classifier
```

Figure 10: Step 1

To read all the features and targets of the dataset, which has been created in an Excel file, the function "*read_excel*" from the Pandas module is used and the imported data is stored in a variable called "data". Then, all the values as "nan" are replaced by zeros, using the "*data.replace*" function and followed by the file address, to avoid any errors when working with the data. Also, a data frame is created with all the imported data, just to show, although will not be used in the program (Figure 11).

```
# STEP 2:
# Read the Excel file in order to import the data that we will use:

data = pd.read_excel(r'C:\Users\elo\Desktop\ROBOTICS BENG (LHU)\PROJECTS (Courseworks)\ECCCORE1H-ROBCORE1H\
AI Surfing program\Python program_AI coursework\Surfing_Dataset.xlsx')

data = data.replace(np.nan, "0")  # Replace any values as "nan" by zeros

df = pd.DataFrame(data)  # Create a dataframe with the imported data
print(df)
```

Figure 11: Step 2

The next step is to create a dictionary that will be used later when calculating the surfboard shape, in order to show as results, the names of the surfboard shapes and not the numbers assigned to those names that are used only to make the predictions (Figure 12).

```
# STEP 3:
#Create a dictionary for the data of surfboard shapes:

surfboardShape_dict = dict(
    zip(data.Surfboard_Shape_N.unique(), data.Surfboard_Shape.unique()))

print(surfboardShape_dict)

print(data['Surfboard_Generic_Volume'].value_counts())
```

Figure 12: Step 3

The first approach of the project is the prediction of the surfboard standard or generic volume. First, the features and targets are defined and saved as variables "*X*" and "*y*". In this case, we use two features, which are the surfing level and the weight of the person, and the target is the generic volume. After, the defined features and targets are trained by the algorithm "*train_test_split*", which divides the data into two subsets.

The first subset is the training data (*X_train*, *y_train*), which is used to fit the machine learning model and the second subset is testing data (*X_test*, *y_test*), which is used to evaluate the fit machine learner. At last, to show how many features are kept for training the algorithm, the function "*X_train.describe*" is used and printed (Figure 13).

```
# STEP 4:
### APPROACH 1: PREDICT THE SURFBOARD STANDARD VOLUME.

print("||| PREDICT SURFBOARD VOLUME (using K-NEIGHBORS REGRESSOR) |||")

# Define variables for Features (X) and Targets (y):
X = data[['Surfing_Level_N', 'Surfer_Weight']]  # Features
y = data['Surfboard_Generic_Volume']  # Target

# Train the data:
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=42)

print(X_train.describe())  # Show how many features we keep to train the algorithm
```

Figure 13: Step 4

After training the algorithm, the prediction of the surfboard volume is made, using the KNN regressor algorithm. In this case, 6 neighbours were opted to use to make the prediction, since thanks to the accuracy graph, that will be shown in the next step, we know that the highest prediction is reached by using 6 neighbours. After assigning the number of neighbours, the training features are fitted with their corresponding targets using the function "*knn.fit*", and then the result of the accuracy is printed (Figure 5).

The next step is to introduce the surfer`s level and weight when we run the program, so that based on those features the machine can predict the surfboard standard volume that would need this person. For this, the "input" function is used for both features, and then the prediction is made using "*knn.predict*" and inside of this specifying the two features. Finally, the result of the prediction is printed, as shown in the Figure 14.

```
# STEP 5:
# Make the prediction with KNN using 6 neighbors and print the accuracy:
print("Prediction of Surfboard Volume using KNN")
knn = KNeighborsRegressor(n_neighbors=6)  # Specify how many neihbors to take
knn.fit(X_train, y_train)  # Fit the training features to the targets
predAccuracy = knn.score(X_test,y_test)  # Prediction accuracy of the algorithm
print("The prediction accuracy of the machine using KNN algorithm was: ",predAccuracy)

# Predict the result of the entered values of surfer level and weight:
surferLevel = input("Enter the surfer Level [1 to 5] : ")  # Type the surfer level
surferWeight = input("Enter surfer Weight (kg) : ")  # Type the surfer weight
surfboard_std_volume = knn.predict([[surferLevel, surferWeight]])  # Prediction
print("*** Based on the entered surfer Level and Weight, the suitable surfboard standard volume for this"
      " surfer is: ",surfboard_std_volume, "Litres ***")
```

Figure 14: Step 5

To see how the prediction accuracy of the KNN algorithm varies by changing the number of K, which is the number of neighbours that are considered to make the prediction, and to compare the training and testing accuracies, it is added a graph on which both accuracy results are plotted varying the number of neighbours.

First, an array with numbers from 1 to 9 is defined, which contains the numbers of neighbours that will be used, so in total will be a range of 8 numbers. Also, two empty arrays are created, so that the results of the predictions are stored on those arrays. Using a For loop, it will run the array of numbers previously crated one by one, and calculating the prediction with each number, while it is storing the results in the empty arrays (Figure 15).

```
# STEP 6:
# Make the prediction by using different values of K (number of neighbors) and plot the accuracy results:
neighbors = np.arange(1, 9)  # Define how many neighbors we will take (in this case 8. From 1 to 9).
train_accuracy = np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))

for i, k in enumerate(neighbors):
    knn = KNeighborsRegressor(n_neighbors=k)
    knn.fit(X_train, y_train)  # Fit the Features with their respective targets
    train_accuracy[i] = knn.score(X_train, y_train)  # Obtain the train accuracy
    test_accuracy[i] = knn.score(X_test, y_test)  # Obtain the test accuracy
```

Figure 15: Step 6

After making all the predictions and storing the results of the accuracy of the training and testing predictions with KNN, some parameters from the “matplotlib” library are used to plot the results in a graph.

First, a title of the graph is added, and then, to plot the both line curves, the arguments of both accuracy results are written, and also a label for each of them is assigned. Finally, a legend is added to the graph, and the names of the axis *X* and *Y* are defined (Figure 16).
Later in the results section will be shown the graph that we obtain when running the program.

```
# STEP 7:
# Show and compare the accuracy of Training and Testing accuracy in a graph:
plt.title('Neighbors variation with KNN')  # Title of the graph
plt.plot(neighbors, train_accuracy, label='Training accuracy')  # Represent the line of Training accuracy values
plt.plot(neighbors, test_accuracy,label='Testing accuracy')  # Represent the line of Testing accuracy values
plt.legend()  # Show the legend
plt.xlabel('Number of Neighbors')  # Name of the X axis
plt.ylabel('Accuracy')  # Name of the Y axis
plt.show()
```

Figure 16: Step 7

In the second approach of the program, the aim is to calculate surfboard real volume for the surfer, based on the age and fitness of that person (Figure 8).

First, the age and the fitness values of the surfer are necessary to specify, and these will be introduced when running the program. For this, for each value the "input" function is used, and both are converted to integers. These values are stored in the variables named "*surferAge*" and "*surferFitness*" (Figure 17).

```
# STEP 8:
### APPROACH 2: CALCULATE THE REAL SURFBOARD VOLUME.
# Calculate the Real Volume, multiplying the obtained surfboard volume by the coefficients of surfer Age and Fitness.

print("||| Calculate de Real Surfboard Volume for the surfer |||")

surferAge = int(input("Enter the surfer Age : "))
surferFitness = int(input("Enter surfer Fitness [1 to 4] : "))
```

Figure 17: Step 8

After the age and fitness of the surfer are entered, we proceed to calculate the real volume of the surfboard. The first step is to multiply the previously obtained standard volume by the factor of age. Depending on what threshold is the age of the person, the standard volume will be multiplied by a different factor. So, for example, if the surfer`s age is 25, then the factor to use will be 1, which does not change anything, but, if the age of the person is between 51 and 60, the standard volume will be multiplied by the value of 1.20, and so on. For this, "*if*" functions are used to write each threshold and the multiplications inside each of them. Finally, the result obtained by multiplying the age factor is printed (Figure 18).

```python
# STEP 9:
# Multiply the corresponding Age coefficients:
if surferAge <= 30:                      # First threshold: Age below 30

    resultAge = surfboard_std_volume * 1

elif 31 <= surferAge <= 50:              # Second threshold: Age between 31 and 50

    resultAge = surfboard_std_volume * 1.08

elif 51 <= surferAge <= 60:              # Third threshold: Age between 51 and 60

    resultAge = surfboard_std_volume * 1.20

elif surferAge >= 61:                    # Fourth threshold: Age above 61

    resultAge = surfboard_std_volume * 1.30

print("The result of volume multiplied by the Age coefficient is : ",resultAge)
```

Figure 18: Step 9

The same procedure will be made with the fitness factors, but in this case, the factor will be multiplied by the result obtained from the previous age calculation. Depending on what is the fitness of the person, the factor that will be used will be different. Here, the "*if*" functions are also used to indicate each threshold. Finally, the result that we get after this calculation, which is the surfboard real volume, is printed (Figure 19).

```python
# STEP 10:
# Multiply by the corresponding Fitness coefficients:

if surferFitness == 1:
    surfboard_real_volume = resultAge * 1.20

elif surferFitness == 2:
    surfboard_real_volume = resultAge * 1.10

elif surferFitness == 3:
    surfboard_real_volume = resultAge * 1.05

elif surferFitness == 4:
    surfboard_real_volume = resultAge * 1

print("*** The real surfboard volume for this surfer is : ",surfboard_real_volume, "Litres ***")
```

Figure 19: Step 10

In the third approach, the prediction of the surfboard length is made using MLP Regressor. For this, 4 features are used, which are the surfer`s height, level, and weight, and the surfboard real volume previously calculated, and they are stored in the variable "*X1*". As a target, the "*surfboard length*" data from the dataset is used and stored in the variable "*y1*" (Figure 20).

```
# STEP 11:
### APPROACH 3: PREDICT THE SURFBOARD LENGTH.

print("||| PREDICT THE SURFBOARD LENGTH (using MLP REGRESSOR) |||")

# Define the features (X1) and targets (y1):
X1 = data[['Surfer_Height', 'Surfing_Level_N', 'Surfer_Weight','Surfboard_Real_Volume']]
y1 = data['Surfboard_Length']
```

Figure 20: Step 11

The next step is to train the algorithm and to make predictions with the features and targets of the dataset. These will be put inside a "*While*" loop, in order that the machine keeps learning and predicting until a desired prediction accuracy percentage is reached, which in this case is set an accuracy value of 0.80 (80%). For learning, the "*train_test_split*" function is used, which has been explained in a previous section, and for the predictions, the "*MLPRegressor*" module is used, which is imported from the Scikit-learn package, and the "*mlr.fit*" to fit the training features with the respective training targets (Figure 21).

```
# STEP 12:
while True:

    X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1)  # Train the algorithm

    #  1.2 Prediction:

    mlr = MLPRegressor(solver='lbfgs',alpha=1e-5,hidden_layer_sizes=(3, 3),max_iter=400,random_state=3)
    mlr.fit(X1_train, y1_train)
    print("Prediction accuracy : ", mlr.score(X1_train, y1_train))    # Show the accuracy score reached
    if mlr.score(X1_train, y1_train) > 0.80:  # Stop predicting when the accuracy is of 80%
        break
```

Figure 21: Step 12

After the algorithm is trained and the desired prediction accuracy is reached, the result of the surfboard length is predicted. For this, some previous features are reused, such as the surfer`s weight, level, and the board real volume, and a new feature is added, which is the height of the surfer. Based on those features, the algorithm makes the prediction, and the result is printed and stored in the variable "*surfboard_length*" (Figure 22).

```
# STEP 13:
# Make the prediction with Multiple-layer regressor (MLP):

# We will use the Weight, Level and Volume that we introduced in the previous part, and add the surfer`s height.

surferHeight = input("Enter the surfer Height (cm) : ")
surfboard_length = mlr.predict([[surferHeight, surferLevel, surferWeight, surfboard_real_volume]]) # Features
print("*** Based on the surfer's Level, Weight, Height and the real surfboard volume, "
      "the most suitable surfboard Length for this surfer is: ",surfboard_length, "Feet (ft) *** ")
```

Figure 22: Step 13

In the last approach, the aim is to predict what is the suitable surfboard shape for the person using the Random Forest classifier.

First, the features and the targets are defined, which are saved in two variables named "X2" and "y2", and then, the algorithm is trained. For features, some previous features are again used, which are the surfboard real volume, surfboard length, and the level of the surfer, and it is also added a new feature, which is the wave size. For training the algorithm, it is used again the "*train_test_split*" function (Figure 23).

```
# STEP 14:
### APPROACH 4: PREDICT THE SURFBOARD SHAPE.
# We use as features the surfboard volume and length previously obtained, and also surfer Level and the size of wave.

print("||| PREDICT THE SURFBOARD SHAPE (using RANDOM FOREST CLASSIFIER) |||")
X2 = data[['Surfboard_Real_Volume', 'Surfboard_Length', 'Surfing_Level_N', 'Wave_Size']] # Features
y2 = data['Surfboard_Shape_N']                                                           # Targets

# Train the data:
X2_train, X2_test, y2_train, y2_test = train_test_split(X2,y2,test_size=0.2,random_state=42)
```

Figure 23: Step 14

Using the Random Forest classifier, which is abbreviated as "*rfc*", the training features are fitted with the training targets, the prediction is made, and the accuracy score of the prediction is printed. After that, we use the "input" function, to enter the wave size when running the program, and then, the surfboard shape is predicted using the surfboard real volume, length, surfer`s level, and the wave size. It is important to know that the predicted result that we get, is the number associated to the name of the surfboard shape, and does not show the name of the board, which is the purpose of this approach (Figure 24).

```
# STEP 15:
# Make the prediction with Random Forest Classifier
rfc = RandomForestClassifier()
rfc.fit(X2_train, y2_train)
print("Prediction accuracy : ",rfc.score(X2_test, y2_test))  #  print the prediction accuracy

## Introduce the value of Surfer level and Weight to predict the surfboard volume:
waveSize = input("Enter the wave size [1 to 12 (ft)] : ")
surfboard_shape = rfc.predict([[surfboard_real_volume, surfboard_length, surferLevel, waveSize]])
```

Figure 24: Step 15

In the previous step, the surfboard shape has been predicted, but it does not show the name of the surfboard shape, but only gives the number which has been written in the dataset for making the prediction. To obtain the name of the surfboard that corresponds to the predicted number, the dictionary created at the start of the program is used. For this, we access the dictionary with the variable, "*surfboard_shape_number*", which contains the predicted number, and this extracts the name that corresponds to the predicted number. Finally, the name of the surfboard shape is printed (Figure 25).

```
# STEP 16:
# Access the dictionary created previously, and extract the corresponding name of the surfboard:
surfboard_shape_name = surfboardShape_dict[surfboard_shape_number[0]]
print("*** Based on the surfer's Level, surfboard volume, surfboard length and the size of the wave, "
      "the surfboard Shape for this surfer is: ",surfboard_shape_name, " ***")
```

Figure 25: Step 16

The final step is to summarize and collect all the predicted results, and to print them in a proper way so that for the reader it is easier to visualize. Also, a message has been written that in case that the predicted name is “NOT SUITABLE”, which means that the waves are too big for the surfer, it shows the message that says the waves are too big for the level of the surfer (Figure 26).

```
# STEP 17:
# Print the final result of the ideal Surfboard for the surfer:

print("Based on all made predictions and calculations, the ideal surfboard for this surfer is: ")
print("-Surfboard Volume: ", surfboard_real_volume, " Litres (L)")
print("-Surfboard Length: ", surfboard_length, " Feet (ft)")
print("-Surfboard Shape or Model: ", surfboard_shape_name)

if surfboard_shape_name == "NOT SUITABLE":
    print("The size of wave is too big for the level of this person. This person can not surf such a wave")
```

Figure 26: Step 17

4. Results:

In this section the results obtained when running the Machine learning program are discussed. First the graph of the variation of accuracy with KNN obtained is explained and then, the results of the prediction of the surfboard type are mentioned.

a. The graph:

In Figure 17, a graph which plots the variation of the training and testing accuracy results when running the program, is shown. As it can be seen in the graph, the results obtained show that the training accuracy curve is above the testing accuracy curve, so this means that the machine gets more accuracy when training than testing and hence, it can be deduced that the algorithm predicts easier when training than testing. In both line curves, it starts higher with one neighbour, then starts to go down until one point and after goes up until it stabilizes. The lowest accuracy of the training curve is around 0.65 using 5 neighbours, and the highest accuracy is at almost 0.8 using 7 neighbours.

Nevertheless, the lowest accuracy in the testing curve is at 0.4 with 3 neighbours, and the highest point is at 0.7 with 6 neighbours. Therefore, knowing that the highest prediction is reached with 6 neighbours, this number of neighbours has been choosed to predict the surfboard standard volume using KNN.

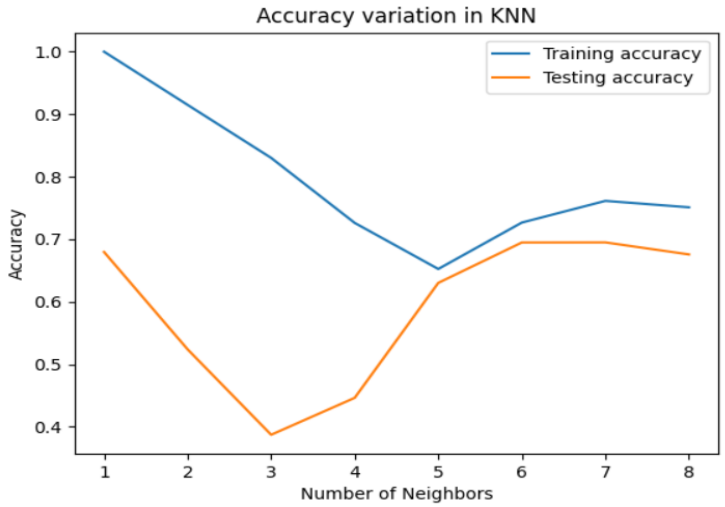


Figure 27: KNN prediction accuracy Graph

b. The dataset:

One of the first ideas in this project was to use real-time data of the sea conditions of different locations and thus, simply select where the person would like to surf and predict the surfboard type based on the real conditions of that place. However, the difficulty to implement this and the work that would suppose, it was quite far from the main purpose of this project, which was to implement Machine Learning to solve a specific task, in this case, to predict the type of surfboard for a surfer, using different examples of features and targets of the people and waves.

So, finally it was decided to create a dataset from scratch using features of the surfers and waves collected from many sources about surfing. However, this has not been easy, as each piece of data has been written carefully, following the rules of surfing learned from other sources and writing each feature with the corresponding target. Nevertheless, finally the created dataset has been valid to make the predictions of surfboards, as the prediction results obtained have a sense with the features and targets of the dataset.

c. The prediction:

In the Figures 28, 29 and 30, a demonstration of a prediction example is made. First, as shown in Figure 28, the features highlighted in yellow are used to feed them into the Machine Learning, so that after, whether the prediction results match with the correct targets can be checked. So, when running the program, the machine will ask to introduce the features, which in this case, the highlighted ones will be used.

Surfing_Level	Surfing_Level_N	Surfer_Age	Surfer_Fitness	Surfer_Weight	Surfer_Height	Wave_Size
Beginner	1	7	3	35	130	1
Novice	2	10	2	35	134	3
Intermediate	3	13	4	35	138	5

Figure 28: Features for the prediction test

After the program finishes executing and predicting with the introduced data, the final result of the respective surfboard characteristics is printed. In the Figure 29, the results obtained with the prediction using the selected features are shown. So, based on the introduced features of the person and the size of wave, the suitable surfboard would have a volume of *22.95 Litres*, with a length of *5.77 Feet*, and a *Malibu* shape. Besides, in this test the machine had an accuracy of *69%* with KNN to predict the standard surfboard volume, *86%* with MLP regressor to predict the length of the board and *50%* with Random Forest to predict the shape.

```
*** Based on the surfer's Level, surfboard volume, surfboard length and the size of the wave, the surfboard Shape for this surfer is:  Malibu  ***
Based on all made predictions and calculations, the ideal surfboard for this surfer is:
-Surfboard Volume:  [22.94416667]  Litres (L)
-Surfboard Length:  [5.77642239]  Feet (ft)
-Surfboard Shape or Model:  Malibu
```

Figure 29: Results of the prediction test

Comparing the obtained results with the corresponding targets of the introduced features, which are highlighted in yellow in the Figure 30, there is almost no difference between them, and the results obtained are as they should be, which means that the machine predicted incredibly well.

Surfboard_Real_Volume	Surfboard_Length	Surfboard_Shape
27.195	5	Softboard
23.1	5.2	Malibu
16.8	5.6	Hybrid/Evolutive

Figure 30: Desired targets for the prediction test

Although the program predicts the surfboard type incredibly well, sometimes the accuracy of some prediction algorithms is not as accurate as expected, and the values that are obtained from those predictions are not very close to the desired target. This is because the number of features and targets used in the dataset is not maybe very large, and hence, the accuracy of the algorithm is not very high, as it has got less data to train and learn about it. This happens only with regression algorithms, which in this project are used the K-Nearest neighbours and the Multilayer perceptron when predicting the surfboard volume and length. So, for a future work, it would have to extend the dataset to add more values for each feature and targets, so that the machine has more accurate prediction and hence, the prediction results are more precise.

5. Conclusion:

This project started out with the goal of creating an AI program to predict the type of surfboard that a surfer would need based on the conditions of the person and the sea conditions. During the project, some issues were found related to the dataset and the prediction accuracy, which some of them could be fixed straight away and other will require future work to improve the program.

Overall, the study and development of the AI program has been a success and it has achieved the main aim of this project, implementing a Machine learning technique to predict different features of the suitable surfboard for a person, using different types of algorithms, and creating a dataset from scratch.

6. Future work:

Future work for this project would involve the improvement of the dataset by adding new more data and extending the range of the values of already used features and targets. Also, the implementation of real-time data of sea and weather conditions of different locations would be discussed.

The range of values of each feature and target used in the created dataset is quite limited and therefore the prediction result that gives the program is not very accurate, being that the Machine Learning algorithm has less data for learning and make the prediction. As a consequence of this, the algorithm can be confused when learning which feature belongs to its target and the result of the prediction will not be very accurate. Future work for this would be to extend the range of values of all features and targets by adding more numbers. Thus, the algorithm would have much more data to learn about and the result that would be obtained would be more precise.

Another improvement that could be made in this project would be to add some other new features and targets to make the program much richer and have more detailed results. This new data would be the weather condition of a specific place and also more sea conditions apart from the size of waves, such as the temperature of the water, the tide height, the currents of the water, or if the sea is rough or calm. With this data, we could predict some new targets that would be, for example, which wetsuit would need a surfer depending on the water temperature of the water or the weather, or if it is too dangerous to surf if the sea is rough. Furthermore, information of different locations where people do surf could be added, in order to have more specific data of sea and weather conditions of different locations.

Finally, it would be a big advantage to use real-time data of sea and weather conditions of any surfing locations. This was one of the potential ideas to implement in the program, as this would bring further functionalities to the program.

The idea would be that the user after typing his data, chooses the location where this person would like to surf, and immediately the program would get all the information about the weather and sea conditions of that place in real time or a forecast. Thus, the program would

import this data, and based on the information that it gets it would predict which type of surfboard the person would need depending on the type of waves of that place, if the sea conditions are too dangerous for surfing or which wetsuit should wear depending on the temperature. Additionally, a map could also be added to the program where the user could simply select the location where this person would like to go surfing and so the program would be much more practical and intuitive for the user.

References:

Surf Nation. n.d. *How to choose the right surfboard: Ultimate Guide*. [online] Available at: <https://www.surfnation.com.au/pages/how-to-choose-the-right-surfboard#what-is-the-right-surfboard-height> [Accessed 12 January 2021].

Pierson, D., 2018. *Understanding Surfboard Volume*. [online] Surfline. Available at: <https://www.surfline.com/surf-news/understanding-surfboard-volume/35441#:~:text=We%20all%20know%20our%20standard,W%20x%20H%20%3D%20V> [Accessed 19 January 2021].

Surfertoday. n.d. Surfboard Size Chart. [online] Available at: <https://www.surfertoday.com/board-size-chart/surf> [Accessed 4 February 2021].

Kroleski, G., n.d. Choosing A Surfboard Type Based On the Waves. [online] Surfscience.com. Available at: <http://www.surfscience.com/topics/surfing-tips/tribal-knowledge/choosing-a-surfboard-type-based-on-the-waves> [Accessed 6 February 2021].

Mohri, M., Rostamizadeh, A. and Talwalkar, A., 2018. *Foundations of machine learning, second edition*. 2nd ed. MIT Press, pp.1-6.

ALPAYDIN, E., 2020. *INTRODUCTION TO MACHINE LEARNING*. [Place of publication not identified]: MIT Press, pp.1-6.

Mirjalili, S., Faris, H. and Aljarah, I., 2019. *Evolutionary Machine Learning Techniques*. Springer Nature, pp.1-4.

Zhu, X. and Goldberg, A., 2009. *Introduction to Semi-supervised Learning*. Morgan & Claypool Publishers, pp.3-5.

Zhang, Y., 2010. *New Advances in Machine Learning*. BoD – Books on Demand, pp.1-3, 20-23, 31-33.

Rajaguru, H. and Kumar Prabhakar, S., 2017. *KNN Classifier and K-Means Clustering for Robust Classification of Epilepsy from EEG Signals. A Detailed Analysis*. Anchor Academic Publishing, pp.31-33.

Khalid, A., Najadat, H., Ismail, H. and Mohammed Khair, S., 2013. Stock Price Prediction Using K-Nearest Neighbor (kNN) Algorithm. *ResearchGate*, [online] pp.33, 35. Available at: <https://www.researchgate.net/publication/262456253_Stock_Price_Prediction_Using_K-Nearest_Neighbor_kNN_Algorithm> [Accessed 9 February 2021].

Ramchoun, H., Amine, M., Idrissi, J., Ghanou, Y. and Ettaouil, M., 2016. Multilayer Perceptron: Architecture Optimization and Training. *International Journal of Interactive Multimedia and Artificial Intelligence*, 4(1), pp.26-27.

Kulala, H. and Rani, K., 2017. Advancements in Multi-Layer Perceptron Training to Improve Classification Accuracy. *International Journal on Recent and Innovation Trends in Computing and Communication*, [online] 5, pp.352-356. Available at: <https://www.researchgate.net/publication/317597671_Advancements_in_Multi-Layer_Perceptron_Training_to_Improve_Classification_Accuracy> [Accessed 12 February 2021].

Mohanty, A., 2019. Multi-layer Perceptron (MLP) Models on Real World Banking Data. *Becoming Human: Artificial Intelligence Magazine*, [online] p.1. Available at: <https://becominghuman.ai/multi-layer-perceptron-mlp-models-on-real-world-banking-data-f6dd3d7e998f> [Accessed 26 February 2021].

Livingston, F., 2005. Implementation of Breiman's Random Forest Machine Learning Algorithm. Machine Learning Journal Paper, [online] pp.1. Available at: <https://www.researchgate.net/publication/242751368_Implementation_of_Breiman's_Random_Forest_Machine_Learning_Algorithm> [Accessed 9 February 2021]

IBM Cloud Education. 2020. Random Forest. [online]. IBM. Available at: <https://www.ibm.com/cloud/learn/random-forest> [Accessed 10 February 2021]

SHARMA, A., 2020. Decision Tree vs. Random Forest – Which Algorithm Should you Use?. *Analytics Vidhya*, [online] p.1. Available at: <https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/> [Accessed 26 February 2021].

Challenger-Pérez, I., Díaz-Ricardo, Y., and Becerra-García, R., (2014), "El lenguaje de programación Python." Ciencias Holguín, Vol. XX, núm.2, pp.2-11. Available at: <https://www.redalyc.org/articulo.oa?id=1815/181531232001> [Accessed: 13 de February 2021].

Gowrishankar, S. and Veena, A., 2018. *Introduction to Python programming*. CRC Press, pp.7-12, 16.

Lutz, M., 2001. *Programming Python*. 2nd ed. O'Reilly Media, pp.1-5, 13.

Kaur Arora, S., 2020. What is PyCharm? Features, Advantages & Disadvantages. [online]. Hackr.io. Available at: <https://hackr.io/blog/what-is-pycharm> [Accessed 16 February]