Activity Recognition from Accelerometer

NIKKI LAU

ABSTRACT

Activity recognition is of importance as it shares a linear relationship between energy to exert force. This enables the use of movement acceleration to reflect energy expenditure in individuals performing different activities. In this paper, the efforts to recognise different activities from a tri-axial accelerometer is evaluated and reported. Evaluation of activity recognition for this study is done through two classification models: decision tree and K-nearest neighbours, where these models are evaluated and compared. Comparison of both models are based on their performance and rate of accuracy, where the model with the higher rate of accuracy and thus better performance, is recommended as the model most suitable to recognise activities from accelerometer data.

INTRODUCTION

A tri-axial accelerometer is a device that measures vibration in three perpendicular axes simultaneously. It measures the XYZ coordinate values, where these values represent the direction and position of the device, such as left/right (X), forward/backwards (Y), and up/down (Z). There have been numerous studies that uses accelerometer to detect human movement (1). These studies have acknowledged a linear relationship between energy to exert force. This enables the use of movement acceleration to reflect energy expenditure in individuals performing different activities (2).

In this study, fifteen participants performed seven activities where data were obtained from a wearable tri-axial accelerometer mounted on their chest. The tri-axial accelerometer aimed to measure the different axes (XYZ) during each activity, with a sampling frequency of 52 Hz. The variety of activities performed by the participants consisted of working at a computer, standing, walking, going up and down stairs, etc. The data generated by the accelerometers on each participant was used to train a set of classifiers, which included decision trees and k-nearest neighbours. The performance for both classification models were evaluated and compared to determine which model were able to most accurately recognise each activity based on the accelerometer data. Through this study, we aim to answer the following questions: 1) Which is the best classifier for recognising activities; 2) Which activities are harder to recognise?

METHODOLOGY

Data Acquisition

The dataset was obtained from the UCI Machine Learning Repository website (3). A tri-axial accelerometer was mounted on each of the fifteen participants' chest while they performed activities. Data from the accelerometer includes the following attributes: acceleration along *x* axis, acceleration along *y* axis and acceleration along *z* axis. Participants were also asked to annotate the sequential order of the activities they performed and restart the accelerometer. The participant can begin performing the activity once they have pressed the 'Start' button on the device. Data were collected for a set of seven activities for each participant:

- Working at a Computer
- Standing Up, Walking and Going Up and Down Stairs
- Standing
- Walking
- Going Up and Down Stairs
- Walking and Talking with Someone
- Talking While Standing

Data preparation

The dataset comprised of fifteen CSV files – one for each participant. To analyse the full scope of the data, all fifteen CSV files were loaded into IPython via Jupyter Notebook, and concatenated. The preparation of the data consisted of assigning appropriate headers for each variable (Sequential No., X Acceleration, Y Acceleration, Z Acceleration, Activity), removing unnecessary variables, check for missing values, outliers, and erroneous data entries.

The variable, 'Sequential numbers', were removed as once the data for all fifteen participants were merged, it served no value in the dataset. Through sanity checks, it shows that the variable, 'Activity', ranged from 0-7. As there are only seven activities recorded in the data, the 'Activity' numerically labelled as '0' were all dropped, as an alternative 'Activity' numeric label could not be imputed. It is also important to note that the values for the variable, 'Activity', were left in its original state – as numeric categorical labels. Furthermore, there were no missing values for each variable. In addition, the presence of outliers were noted in 'Y Acceleration' and 'Z Acceleration' for 'Activity' 1, as depicted in figure 2. These outliers were assessed and as they were all under the same 'Activity', they were left in the dataset. The low values of 'Y acceleration' and 'Z acceleration' can be justified by the 'Activity', working at a computer (Activity 1). This indicates that these participants rarely moved forward/backward (Y acceleration) and/or up/down (Z acceleration).

Feature Selection
For this study, the three input features were 'X acceleration, Y acceleration, Z acceleration' where it will predict the target variable, activity (1-7), the participant is performing.

Data Modelling
The activity recognition algorithm is designed to recognise the accelerometer pattern corresponding to each activity. In figure 1, it is depicted that for each activity, the mean values for each axis (XYZ) differs for each activity and is seen to have a distinct pattern. The activity recognition is formulated as a classification problem where each acceleration corresponds to the activities. The two classification models chosen for this study were evaluated based on their performance:

- Decision tree
- K-Nearest Neighbours (KNN)

For both classification models, the data were divided into a training set and a test set, where 80% of the data were for training and 20% were for testing. Default parameters were also initially used to serve as a baseline to calculate the accuracy and confusion matrix. From there, various parameters were tuned, and features were rescaled, to achieve a more accurate performance. The two classification models were analysed and compared based on their performance, and the model with the better performance was recommended.

Activity	X Acceleration	Y Acceleration	Z Acceleration
1	1977.689653	2376.558532	1966.415593
2	1969.489431	2371.051965	1940.448703
3	1996.272755	2378.303095	1965.729391
4	1976.819111	2386.292905	1978.708646
5	2000.554449	2385.493844	1997.001573
6	2027.107076	2374.075277	1952.189366
7	1997.845983	2388.535898	1973.053026

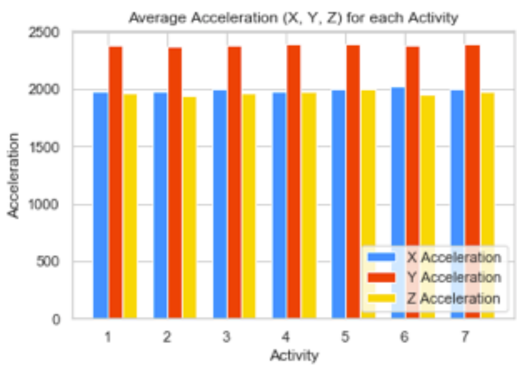

Figure 1: Average acceleration (XYZ) for each activity (1-7) (left). Histogram depicting average acceleration (XYZ) for each activity (1-7) (right)

RESULTS
Data analysis
From the boxplots which illustrates the summary statistics for each acceleration axis with each activity, it

can be seen that Y and Z acceleration have relatively short boxplots which depicts that many participants share the similar distribution of acceleration values. This however, is not depicted in X acceleration as the boxplot appears comparatively longer, which depicts that there is a variation in the acceleration measured between participants. Furthermore, it illustrates that X and Z acceleration share similar medians across all activities, are similar, while Y acceleration have a relatively higher median across all activities. In addition, there are outliers present across all acceleration axes for each activity. However, most outliers are present in activity 1 (computer) and activity 7 (talking while standing). It can be justified that as more participants performed activity 1 and 7, there would be a wider variety in the acceleration values, leading to outliers (figure 2).

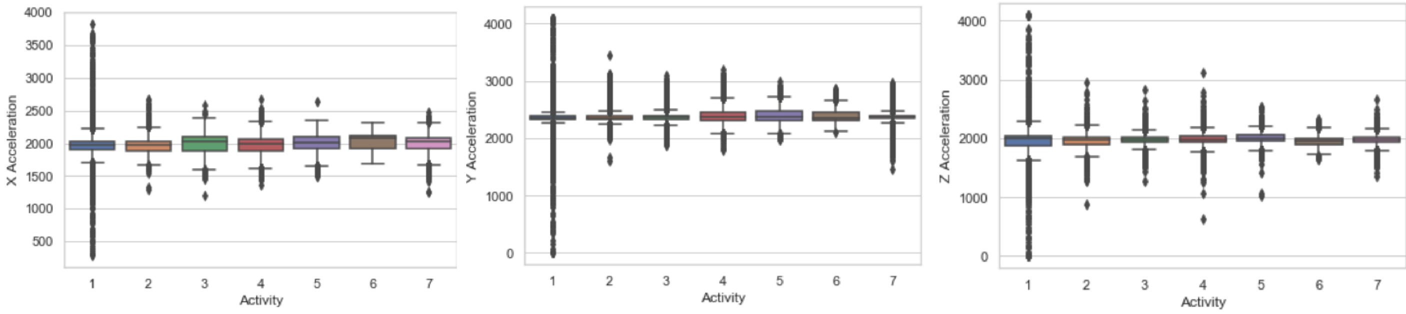

Figure 2: Boxplots visualising summary statistics for each acceleration (XYZ) with each activity (1-7).

The activity recognition is formulated as a classification problem where each acceleration corresponds to the activities. The XYZ acceleration data for each activity should be approximately the same size to ensure proper model training without bias (4). As depicted in figure 3, it shows that participants mainly performed Activity 1 (working at a computer) and Activity 7 (talking while standing), where these two activities share approximately the same size. While, there is a significant imbalance between Activity 2-6. As there was an imbalance in the distribution of the classes (figure 3), features were scaled so that they could be uniformly evaluated (4). Furthermore, parameters were tuned after initialising default parameters, in hopes to achieve a

Activity	
1	608667
2	47878
3	216737
4	357064
5	51498
6	47770
7	593563

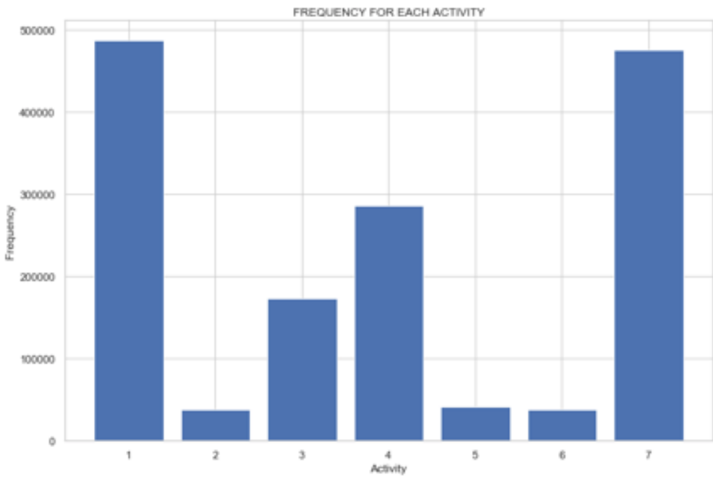

Figure 3: Frequency XYZ acceleration for each activity

Decision Tree

Due to the imbalance in the distribution of the classes, features were scaled using the MinMax scaler. This scaling method uses the minimum and maximum values from the features (5). For the initial decision tree model, default parameters were used to predict the accuracy and performance of the model. Such parameters include (6):

- Criterion: gini
- Max depth = None

- Min_samples_split = 2
- Min_samples_leaf = 1
- Max_features = None
- Max_leaf_nodes = None

From the default parameters, the model achieved 65.93% accuracy (table 3). To attempt to achieve a better performance from the model, the parameters were tuned to the following:

- Criterion: entropy
- Max_depth = None
- Min_samples_split = 4
- Min_samples_leaf = 4
- Max_features = auto
- Max_leaf_nodes = None

Due to the tuning of the parameters, the model achieved 70.82% accuracy (table 3), which is an increase from the default model. In addition, a confusion matrix (table 1) is provided to show the predictions for each activity, as well as the prediction errors. For this model, the main activities that yielded high prediction accuracies are computer, standing, walking and talking while standing. While, standing walking up down stairs, up down stairs and walking while talking, yielded relatively low prediction accuracies, in comparison. The values outside the range of the highlighted values in table 1 shows the prediction errors ie. For the activity computer, 109213 out of 384636 were correctly predicted, 1167 were incorrectly identified as standing, walking up down stairs, 1795 were incorrectly identified as standing, 4148 were incorrectly identified as walking, 245 were incorrectly identified as going up down stairs, 116 were incorrectly identified as walking while talking, and 5267 were incorrectly identified as talking while standing.

The parameters were adjusted based higher yield of accuracy. As the parameters were increased in increments of 5 for both criterions 'gini' and 'entropy', the accuracy dropped to 50%. This shows that the most ideal parameters were below 5. However, as the parameters increased in increments of 1 from 1-4, 4 yielded the highest accuracy.

	Computer	Standing, walking up down stairs	Standing	Walking	Up down stairs	Walking while talking	Talking while standing
Computer	**109213**	1167	1795	4148	245	116	5267
Standing, walking up down stairs	3780	**1853**	668	1858	72	44	1304
Standing	2915	506	**20590**	8824	1100	673	8606
Walking	6213	989	8306	**46077**	1116	561	8196
Up down stairs	1003	98	2508	3621	**1317**	193	1550
Walking while talking	514	74	1497	1857	359	**2051**	3232
Talking while standing	6277	796	8202	9300	690	2003	**91294**

Table 1: Representative Confusion Matrix for Decision Tree Classification Modelling

K-Nearest Neighbour

The imbalance in the distribution of the classes required features to be scaled. Features for this model were scaled using the MinMax scaler, and default parameters were used to evaluate the performance. Default parameters are as follows (7):

- Number of neighbours = 5
- Weights = uniform
- p = 2

Based on these parameters, the model achieved 72.46% accuracy (table 3). In order to achieve an even better rate of accuracy, features were rescaled and the parameters were tuned. The features were rescaled using standard scaling, which rescales the features such that they are approximately standard normally distributed (6). The parameters were also tuned as per below:

- N_neighbours = 40
- Weights = uniform
- Metric = minkowski
- P = 1

From these changes, the model was able to yield 75.43% accuracy (table 3), which is an increase from the default parameters. From the confusion matrix, depicted in table 2, it can be established that the activities that yielded high accuracies are computer, standing, walking and talking while standing. While, standing walking up down stairs, up down stairs and walking while talking, yielded relatively low predicted accuracies, in comparison. For the activity computer, 111285 out of 384636 were correctly predicted, 561 were incorrectly identified as standing, walking up down stairs, 1107 were incorrectly identified as standing, 3634 were incorrectly identified as walking, 80 were incorrectly identified as going up down stairs, 46 were incorrectly identified as walking while talking, and 5238 were incorrectly identified as talking while standing.

The parameters were adjusted based higher yield of accuracy. With our model, k values in increments of 10 were tested, which showed an increase in the rate of accuracy. When the k value was 30, 40 and 50, the accuracy was 75.36%, 75.43% and 75.43%, respectively. As the accuracy remained the same when the k value is 50, it was decided that a k value of 40 is ideal for the model.

Activity	Computer	Standing, walking up down stairs	Standing	Walking	Up down stairs	Walking while talking	Talking while standing
Computer	**111285**	561	1107	3634	80	46	5238
Standing, walking up down stairs	3815	**1658**	473	2065	30	18	1520
Standing	2287	124	**20217**	9859	648	419	9660
Walking	5166	144	3334	**53595**	203	186	8830
Up down stairs	895	12	1930	4392	**1191**	108	1760
Walking while talking	433	28	1121	1887	336	**1968**	3811
Talking while standing	4656	201	4383	8092	184	837	**100209**

Table 2: Representative Confusion Matrix for K-Nearest Neighbour Classification Modelling

Comparison of both models

Based on the two classification models, it is revealed that KNN yielded a higher accuracy rate and thus performed better than the decision tree model. KNN performed better with the use of default parameters

(72.46%) and performed even better after tuning the parameters (75.43%) (table 3). This shows that using the KNN model, it is able to predict activities more accurately based on the XYZ axes acceleration. However, computer, standing, walking and talking while standing, were better recognised activities than the others (table 1,2). This trend is observed across both models. As the KNN model performed better, it is recommended that this model is preferred for predicting activities based on the axes acceleration.

	Default parameters	Tuned parameters
Decision Tree	65.93%	70.82%
K-Nearest Neighbours	**72.46%**	**75.43%**

Table 3: Comparison Table of Accuracy of classifiers with default parameters and tuned parameters

DISCUSSION

This study focuses on predicting and recognising the different activities through the different XYZ axes of acceleration, with the use of two classification models: decision tree and K-nearest neighbour. Decision tree is an organised classification model that can be used to assist in making the best decision for a given objective. The process of decision making for an objective can be explicitly represented visually through the use of a decision tree, hence its popularity in machine learning (6). In this study, the target variable and feature variables were decided as activity and XYZ axes acceleration, respectively. The dataset was split 80/20 into a training set and testing set. From default parameters, mentioned in the methodology, the model was able to achieve an accuracy rate of 65.93% (table 3). This was used as a baseline so that accuracy rates due to further tuning in parameters could be compared to. Tuning parameters is an action that can be taken to optimise the decision tree performance. This can be done by choosing a criterion (gini, entropy); maximum depth of the tree and split strategies such as minimum samples for a node split, minimum samples for a terminal node, maximum number of terminal nodes and maximum features to consider for split (6).

The criterion is a parameter that measures the impurity of a node. The two different measures include 'gini' which uses the Gini index, a probability of a random sample being incorrectly classified when a label is randomly picked from a splitting branch. While entropy is a measure of information gain which is used to decide which feature to split at each step in the decision tree. Many studies have concluded that both gini and entropy tend to yield similar results, while entropy tend to be slightly more computationally expensive. In this study, the decision tree was tuned to using the entropy criterion as it enabled the model to yield a slightly higher accuracy (5).

The maximum depth is the maximum vertical depth of the tree where it stops splitting the nodes. When the value of maximum depth increases, it causes over-fitting, while a lower value causes under-fitting. When tuning this parameter, it was increased in increments of 5, and it was established that the maximum depth of 15 yielded the best accuracy score for the model (6).

The minimum samples for a node split is the minimum number of observations required in a node to be considered for splitting. This parameter is used to control over-fitting, as higher values prevents the model from learning relations, while very high values can cause under-fitting. This parameter was tuned to 10 as it was able to yield a high accuracy rate. It was observed that as the value increases, the accuracy rate didn't differ too significantly and so a lower value was more justifiable as it prevents under and over-fitting (6).

The minimum samples for a terminal node is the minimum number of observations required in a terminal node. This parameter is similar to that of minimum samples for a node split. It is suggested that a lower value is more useful for imbalance class problems as the regions will be very small where minority classes are a majority. Due to the imbalance in the dataset, it was decided that a value of 10 is appropriate for this parameter (6).

Maximum number of terminal nodes refers to the maximum number of terminal nodes in the tree, where this parameter was left as default as restricting this parameter led to a decrease in the accuracy of the

model (6).
Lastly, the maximum features for a split is the number of features considered for a split in the decision tree. As a general rule of thumb, the best value for this parameter is to take the square root of the total number of features (6), and hence 'auto' was chosen for this parameter. Due to tuning of the parameters, the decision tree model was able to increase in accuracy by 6.51%, yielding an accuracy rate of 72.44% (table 3). Comparing the confusion matrix for both decision trees, it can be seen that the latter model is able to recognise activities more accurately with a decrease in error rate (table 1). However, in both models, activities such as computer, standing, walking and talking while standing, were better recognised activities than the others. This can be explained as the dataset provided contained an imbalance in the frequency of data for each activity. It can be assumed that those activities with higher frequencies tend to be better recognised by the system. Therefore, due to these imbalances it was important for the features to be rescaled using the MinMax scaler. This was able to provide normality to a certain degree in the dataset.

The K-nearest neighbour (KNN) is a classification model that stores all available data and classifies new input data based on a similarity measure. This means that individual data values are considered to belong to a class if it shares the most number of common values, in terms of distance. This classification algorithm is one of the most fundamental yet simple and hence is gravitated towards in machine learning (9). From default parameters, two rescaling features were used: MinMax scaler and standard scaler. Minmax scaler as per its name, uses the minimum and maximum values from the features, while standard scaler rescales features to ensure the features are approximately standard normally distributed. Comparing these two rescaling features, standard scaler compared to Minmax scaler was able to achieve a higher accuracy rate from default parameters at 73.46% and 72.44%, respectively (table 3). This is expected as there is an imbalance in the distribution and so using the standard scaler ensures that the imbalance reaches a degree of normality.

The three parameters in KNN are number of neighbours, weights and p. Number of neighbours is the core deciding factor of this model and refers to the number of nearest neighbours. For example, if the number of neighbours is 3, this indicates that the three nearest neighbours are considered for computation. The parameter, weights is the weight function used in prediction. Lastly, p is the power parameter for the Minkowski metric (7).

In order to improve the model, the parameters were tuned accordingly. The number of neighbours were increased by increments of 10, and it was found that the most ideal number is 40. With each increase in increments of 10, the accuracy of the model increases. However, as the number of neighbour reaches 50, the accuracy rate remains constant. This indicates that the value of 40 is ideal as it produces the best accuracy rate compared to lower values, and the same accuracy rate as higher values. When the number of neighbours increases in value, it becomes increasingly computationally expensive.
The weights parameters include uniform and distance. Uniform allows all the data points to be weighted equally in each neighbourhood, while distance weighs the data points by the inverse of their distance, such that closer neighbours have a greater influence than those further away (7). Due to the significant presence of outliers in the dataset, the uniform value was preferred in this parameter so that all data points can be equally weighted. Coincidentally, when both uniform and distance were used and compared in the model, uniform yielded a higher accuracy rate.
Lastly, the distance metric where p = 1 was used, which is used to find Manhattan distance, the distance between two data points. It is noted that when distance metric is changed to a value of 2, which calculated Euclidean distance, the distance between two data points in a plane (10), the rate of accuracy dropped. Due to this, a lower distance metric value enabled the model to yield a higher accuracy rate and thus, Manhattan distance (p=1) was used.
From tuning the parameters, the KNN model was able to increase in accuracy by 2.97%, yielding an accuracy rate of 75.43% (table 3).

Comparing the two classification models used in this study, it is evident that KNN is preferred over the decision tree as it was able to yield a higher accuracy score using default and tuned parameters, leading to a better performance. Furthermore, the confusion matrix can be used to distinguish which activity is better recognised in both models. A confusion matrix aims to provide information regarding the actual and predicted classifications, which displays the predictions for each activity and the prediction errors. From table 1,2, it can be seen that various activities are recognised better than others. Activities such as working at a computer and talking while standing were the best recognised activities, while going up and down stairs was difficult to recognise. Furthermore, standing, walking up and down stairs is often confused with walking. This is the same for going up and down stairs. Walking while talking is also often confused with talking while standing. Based on these results, KNN is the recommended classifier for recognising activity based on XYZ axes acceleration.

CONCLUSION

From this dataset, using a classification model was deemed a good idea, by drawing conclusions from new input data provided for training, and predict the classes for the new input data. Often it is useful to evaluate and compare multiple models to determine which model is able to provide a better sense of accuracy. Due to this, in this study, two classification models, decision tree and K-nearest neighbour, were used. The results from both models shows that in general, activities such as standing and walking up down stairs, going up down stairs, walking while talking are comparatively harder to recognise with the use of a tri-axial accelerometer mounted on the chest. While, activities such as working at a computer and talking while standing are well recognised. However, comparing both model has shown that K-nearest neighbour outperformed the decision model, yielding an accuracy rate of 75.43%.

An interesting extension of this study could include allowing participants to perform the same activities but have the tri-axial accelerator mounted on a body part that is more likely to reflect movement acceleration such as, the pelvis. This may provide more useful information and allow better model training, due to the significant difference in the data values for each axis of acceleration as it will be expected to be impacted more compared to a chest mounted accelerator. Perhaps this will help classify the harder to recognise activities.

REFERENCES

1. Khan, A., Hammerla, N., Mellor, S. and Plötz, T., 2016. Optimising sampling rates for accelerometer-based human activity recognition. *Pattern Recognition Letters*, 73, pp.33-40.

2. Jobanputra, C., Bavishi, J. and Doshi, N., 2019. Human Activity Recognition: A Survey. *Procedia Computer Science*, 155, pp.698-703.

3. Archive.ics.uci.edu. 2020. *UCI Machine Learning Repository: Activity Recognition From Single Chest-Mounted Accelerometer Data Set*. [online] Available at: <https://archive.ics.uci.edu/ml/datasets/Activity+Recognition+from+Single+Chest-Mounted+Accelerometer> [Accessed 21 May 2020].

4. Brownlee, J., 2020. *A Gentle Introduction To Imbalanced Classification*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/what-is-imbalanced-classification/> [Accessed 27 May 2020].

5. Martulandi, A., 2019. *Increase 10% Accuracy With Re-Scaling Features In K-Nearest Neighbors + Python Code*. [online] Medium. Available at: <https://medium.com/datadriveninvestor/increase-10-accuracy-with-re-scaling-features-in-k-nearest-neighbors-python-code-677d28032a45> [Accessed 27 May 2020].

6. Kumar, D., 2019. *Machine Learning Techniques For Improved Business Analytics*. 1st ed. Pennsylvania: IGI Global, pp.11-12.

7. Scikit-learn.org. 2020. *Sklearn.Neighbors.Kneighborsclassifier — Scikit-Learn 0.23.1 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html> [Accessed 31 May 2020].

8. Navlani, A., 2019. *Decision Tree Classification In Python*. [online] DataCamp Community. Available at: <https://www.datacamp.com/community/tutorials/decision-tree-classification-python> [Accessed 25 May 2020].

9. Li, S., 2017. *Solving A Simple Classification Problem With Python — Fruits Lovers' Edition*. [online] Medium. Available at: <https://towardsdatascience.com/solving-a-simple-classification-problem-with-python-fruits-lovers-edition-d20ab6b071d2> [Accessed 24 May 2020].

10. Sharma, N., 2019. *Importance Of Distance Metrics In Machine Learning Modelling*. [online] Medium. Available at: <https://towardsdatascience.com/importance-of-distance-metrics-in-machine-learning-modelling-e51395ffe60d> [Accessed 27 May 2020].