Class: DATS 6202– Machine Learning

Date: 4/29/2023

Team 11: Amjad Altuwayjiri

Human Activities with Smartphone Sensors Classification Using Different Machine Learning Network Algorithms

1. Introduction

Human activity recognition using smartphones is a growing area of research with many potential applications, including health monitoring, sleep observing, and fitness tracking (Chawla, Prakash, & Chawla, 2021; Islam et al., 2022). In this project, we used the UCI Human Activity Recognition with Smartphones dataset to train and test machine learning algorithms for accurately classifying different human activities based on accelerometer and gyroscope data collected from a Samsung Galaxy S II smartphone.

Various neural network and machine learning algorithms were applied, including MLP, LVQ, and SVM, to classify the activities and tested their performance using different metrics. One-class SVM novelty detection was conducted too to detect any unusual behaviors in the dataset and identify any subjects with injuries or health problems.

The analysis results indicate that the MLP model achieved the highest accuracy of 94%, while the One-class SVM revealed no anomalies in the data. Thus, we can conclude that human activities can be accurately classified using the current dataset. However, for more comprehensive insights, a more diverse dataset with varying subject demographics would be beneficial.

2. Personal Contribution to Project

This section discusses my personal contributions to the project, which include implementing and evaluating three machine learning models: Multilayer Perceptron (MLP), Learning Vector Quantization (LVQ), and One-class Support Vector Machine (One-class SVM). I tuned the hyperparameters using two different optimization techniques Randomization for LVQ and Bayesian for MLP. I evaluated their performance using accuracy, confusion matrix, precision, recall, F1-score, ROC, and AUC for each model with its suitable method. I plotted the learning curves, confusion matrices, and ROC|AUC of each model if applicable. In addition, I evaluated the data readiness for modeling, such as the number of NA values, feature types, and imbalanced data, and detect outliers by clustering. To assess data imbalance, I plotted the portions of the targeted column, and I plotted the raw dataset for a general demonstration.

2.1. Contribution to Data Preprocessing

Check the feature types, number of NA values, and data balance as in Figure 1.

Figure1

Activity frequency in the dataset

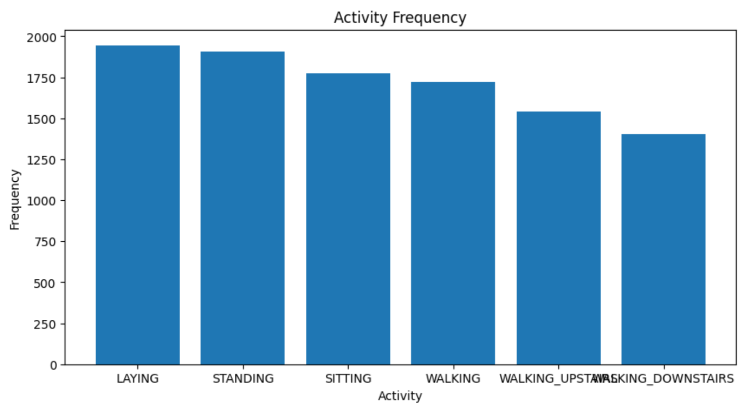

Figure 1: Six activities and their frequency in the dataset

The next step is to detect outliers. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is used to perform clustering on the preprocessed data. The algorithm finds points that are closely packed together and mark them as core samples, and the remaining points as noise or outliers. The DBSCAN algorithm is applied to the preprocessed data, and the number of clusters and outliers are printed. The outliers are then extracted from the data, and a new dataset is created without the outliers for further analysis. Figure 2 shows that there are 28 clusters and 1 outlier cluster. The outlier cluster has 149 rows.

Figure2

Outlier Detection by Clustering

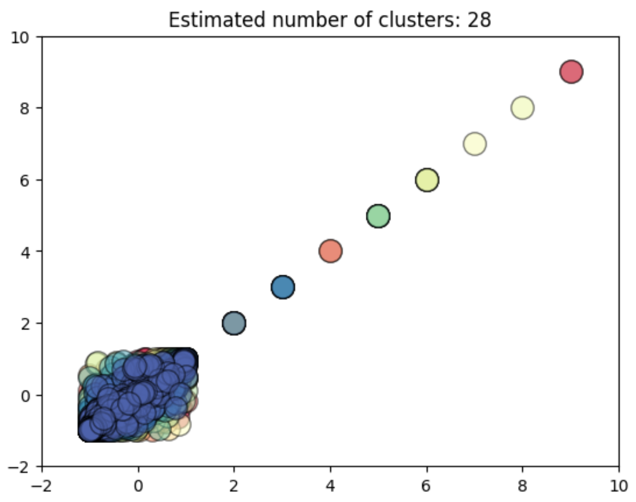

Figure 2: shows 28 clusters by color and most of them cumulative between -1 and 1

Lastly, plotting sample of data as shown in Figure 3.

Figure3

Two Samples of the Raw Accelerometer and Gyroscope

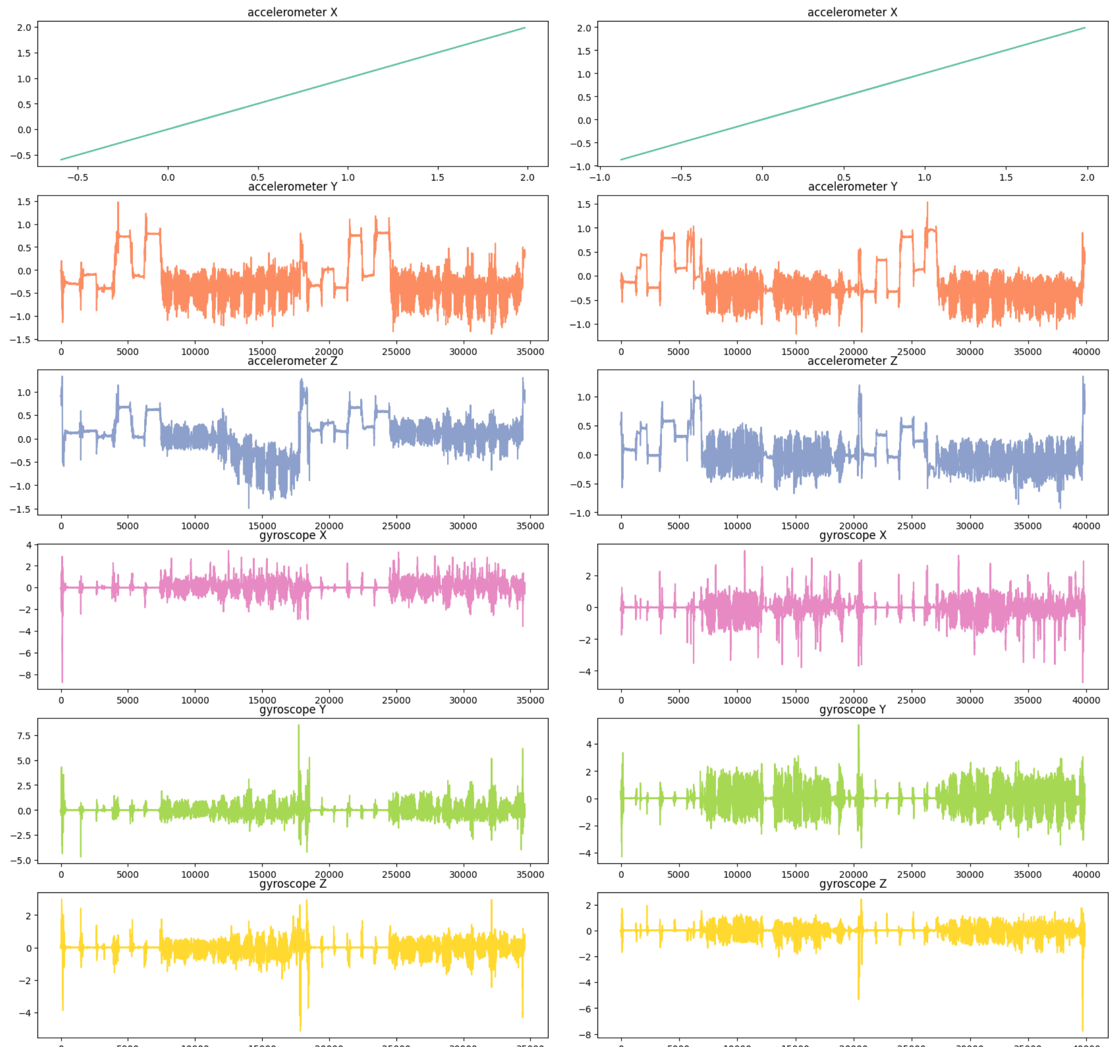

Figure 3: Accelerometer (xyz) and gyroscope (xyz) of two subjects show all six activities

2.2. MLP

The hyperparameters of the MLP classifier are tuned using Bayesian optimization, which tries to find the combination of hyperparameters that maximizes the performance of the model on the training set.

The hyperparameters include the number of hidden layers, the activation function, the solver algorithm, the L2 regularization strength, and the initial learning rate. The hyperparameter search space was defined as follows: the number of hidden layers was sampled uniformly from the range of 1 to 200, the activation function was chosen from a categorical distribution of either ReLU or Tanh, the solver algorithm was also chosen from a categorical distribution of either Adam or L-BFGS, the L2 regularization strength was sampled from a log-uniform distribution ranging from 1e-5 to 1e-3, and the initial learning rate was sampled from a log-uniform distribution ranging from 0.0001 to 0.1.

The best model obtained from the search is (activation = tanh, alpha = 4.950547942793197e-05, hidden_layer_sizes = 144, learning_rate_init = 0.00033522328070073447, solver = adam) and it was evaluated on the test set. I report the accuracy of the model, which is the proportion of correct predictions made by the model. Furthermore, we analyze the performance of the model by computing the confusion matrix, which shows the number of correct and incorrect predictions for each class. Precision, recall, and F1-score are also calculated to provide further insights into the model's performance. Finally, learning curves are plotted to analyze the model's performance with different training set sizes, and the mean and standard deviation of the training and test scores are calculated.

2.3. LVQ

The LVQ model is trained and evaluated using several techniques. First, a set of hyperparameters is defined for the model, and a random search is used to find the best combination of hyperparameters for the model by maximizing the performance on the training set.

The parameter distributions were defined to search over a range of possible values, including the number of prototypes per class, the random state, and the beta value. The prototypes_per_class hyperparameter was varied between 1 and 10, while the random_state was set to range between 0 and 9. The beta hyperparameter was varied over a set of discrete values, including 0, 25, 5, 75, and 1.

The best model that is (random_state = 3, prototypes_per_class = 3, beta = 75) evaluated on the test set to calculate the accuracy, which measures the proportion of correct predictions. Additionally, the confusion matrix is computed to show the number of correct and incorrect predictions for each class. The precision, recall, and F1-score are also calculated to provide further insights into the model's performance. Finally, learning curves are plotted to analyze the model's performance with different training set sizes, and the mean and standard deviation of the training and test scores are calculated.

2.4. One-class SVM

The One-class SVM model is trained and evaluated using various techniques. First, the nu hyperparameter is defined, which controls the proportion of data considered anomalous. In this case, nu is set to 0.1, which means that 10% of the data is considered anomalous. The model is trained on the training set and used to predict anomalies on the testing set. The labels for the testing set are converted to binary, with 1 indicating normal data and -1 indicating anomalous data. The confusion matrix is computed to show the number of true and false positives and negatives. The accuracy of the model is calculated as the proportion of correct predictions made by the model. Finally, the decision function output for the test set is obtained, and the ROC curve and ROC area are computed for each class.

3. Result

3.1. Classification Results

In this study, I evaluated the performance of two machine learning algorithms: MLP and LVQ in terms of their accuracy, precision, recall, and F1-score. The results are presented in Table 1.

Table 1

Evaluation Results of MLP and LVQ

Algorithm	Accuracy	Precision	Recall	F1-score
MLP	94%	0.94	0.94	0.94
LVQ	92%	0.92	0.92	0.92

Table 1: Performance Metrics of the Machine Learning Algorithms

As shown in Table 1, the MLP algorithm performed the best, achieving % accuracy, precision, recall, and F1-score. LVQ also performed well, with accuracy above 99%. According to the MLP confusion matrix Figure 4, most of the classes are predicted correctly but 59 samples of class 1 that is \`Standing\` are predicted to be \`Sitting\`, and 48 samples of class 2 that is \`Sitting\` is predicted to be \`Standing\`.

Figure4

MLP Confusion Matrix

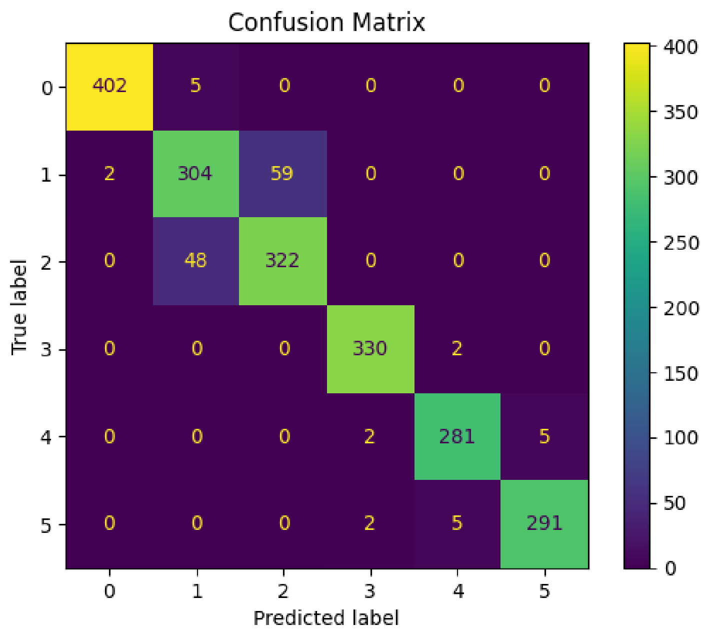

Figure4: Confusion matrix of true and predicted labels of MLP model

The same as MLP, the LVQ confusion matrix Figure 5 shows the same result but with more errors in predicting `Standing` and `Sitting` that it predicted 69 true `Standing` as `Sitting` and 53 true `Sitting` as `Standing`.

Figure5

LVQ Confusion Matrix

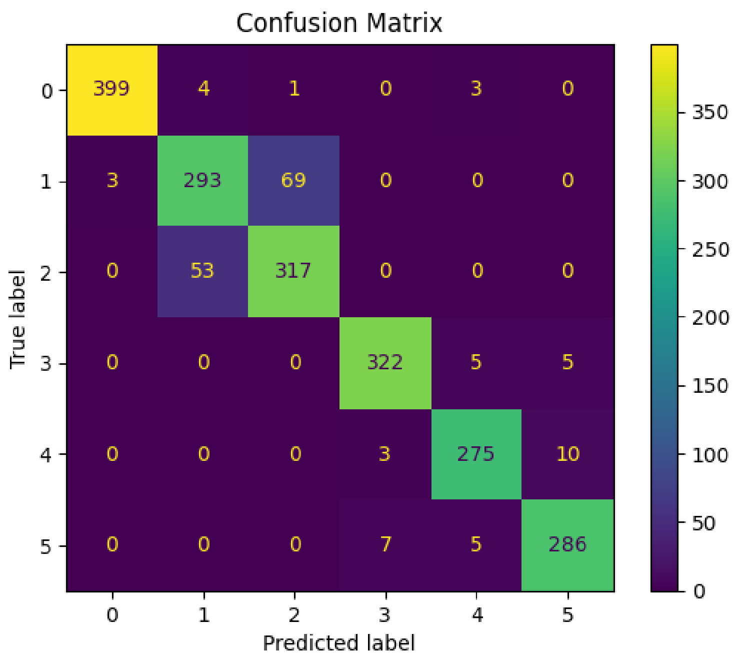

Figure5: Confusion matrix of true and predicted labels of LVQ model

By plotting both models' learning curves, we see that the MLP training curve in Figure 6 is rising and not decreasing, it suggests that the model is overfitting to the training data. This means that the model is becoming too complex and is fitting the noise in the training data instead of the underlying patterns. To address this issue, the model's complexity can be reduced by adjusting its hyperparameters or choosing a simpler model architecture.

Figure6

MLP Learning Curve

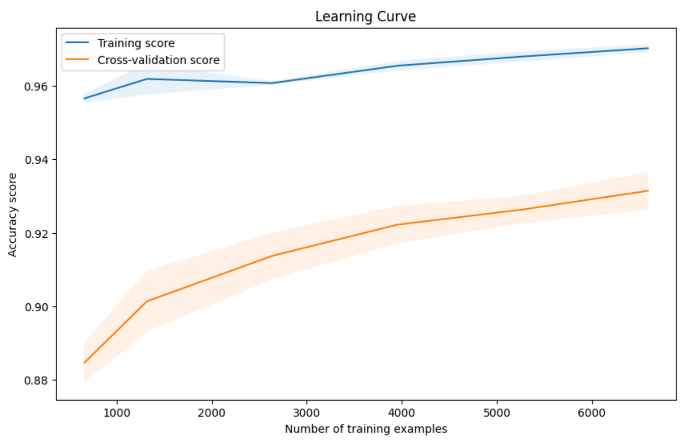

Figure6: MLP training learning curve is raising means getting overfitting

LVQ Figure 7 shows that the training accuracy or loss is decreasing, while the validation accuracy or loss is increasing. This indicates that the model is learning from the training data but are not overfitting or memorizing the training data and are able to generalize to new data. When comparing the two models' learning curves, LVQ has a steeper learning curve than MLP, which means that it is learning better.

Figure7

LVQ Learning Curve

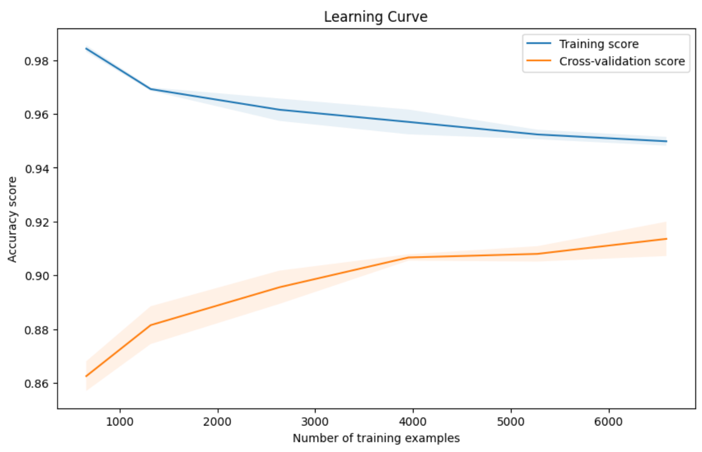

Figure7: LVQ learning curve is steady and steep and not crossing

In LVQ, each neuron in the grid has a weight vector, which is adjusted during training to become similar to the input data points that are closest to that neuron (Hagan et al., 2016). This way, the SOM learns to group similar data points together and to create a meaningful representation of the input data. A heatmap Figure 8 shows the average distance between each neuron and its neighboring neurons in the grid.

Figure8

SOM Heatmap

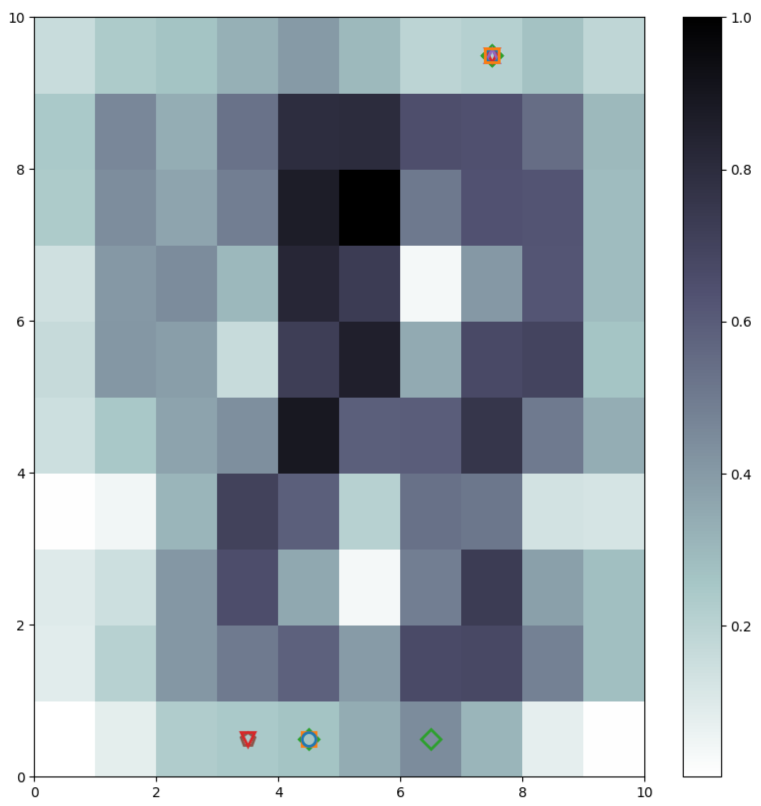

Figure8: Heatmap of the LVQ wights that are colored based on the distance of other weights

3.2 Novelty Detection

One-class SVM was used to identify any unusual behaviors in the dataset. The results in Table 2 showed that there were no anomalous instances in the dataset, indicating that all the recorded activities were within the normal range.

Table 2 *One-class SVM Confusion Matrix*

True positives:	0
False positives	3685
True negatives:	435
False negatives:	0

Table2: there are 3685 false positives and 435 true negatives

The confusion matrix shows that there are no true positives and false negatives, meaning that the algorithm did not classify any of the samples as part of the target class. In addition, in Figure 9 the ROC|AUC of the One-class SVM result is 1.0, which means that the model has correctly classified all the positive examples (in this case, the anomalies) and none of the negative examples (the normal data) as positive. This is a perfect score and indicates that the model is doing an excellent job of identifying anomalies in the data which in this case are 0.

Figure9

One-class SVM ROC|AUC

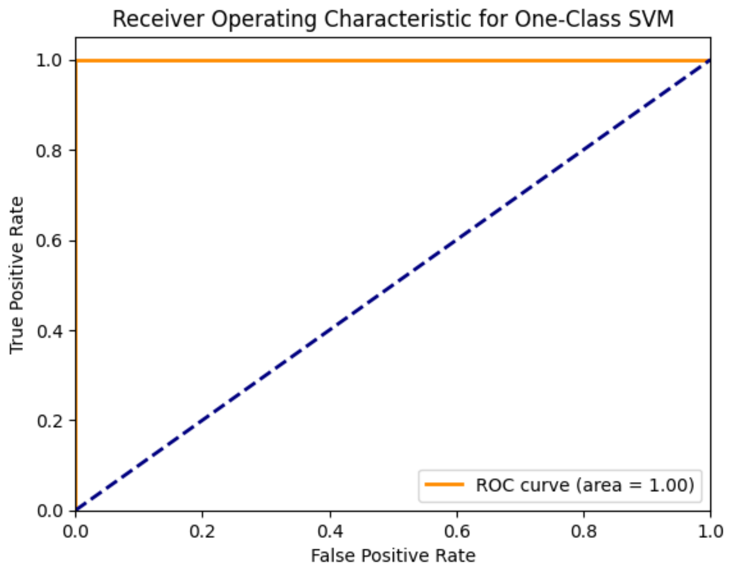

Figure9: An AUC score of 1.0 and a ROC curve on number one

4. Conclusion

Machine learning network algorithms such as MLP and LVQ can be used to accurately classify human activities by 94% - 92% based on accelerometer and gyroscope data that was collected from a smartphone. The application of these models in real-world scenarios will elevate the health sector, such as using them for health monitoring, sleep observing, and fitness tracking.

One limitation of the dataset used in this study is its small size, as it only includes data from 30 subjects. Additionally, there is a lack of information about the health history of the subjects, such as physical injuries or obesity, which may limit the generalizability of the findings. Despite these limitations, our results demonstrate that the One-class SVM model for Novelty detection was able to accurately identify and classify normal subjects. We argue that with a larger and more diverse dataset, this approach has the potential to be an effective tool for detecting abnormalities in various populations. Therefore, future studies should focus on expanding the dataset to include a wider range of individuals and health records to further validate the effectiveness of this approach.

Reference

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. https://doi.org/10.1145/130385.130401

Chawla, K., Prakash, C., & Chawla, A. (2021). Comparative study of computational techniques for smartphone based human activity recognition. *Lecture Notes in Electrical Engineering*, 427–439. https://doi.org/10.1007/978-981-16-3067-5_32

Hagan, M. T., Demuth, H. B., Beale, M. H., & Jesús Orlando De. (2016). *Neural network design*. s. n.

Islam, M. M., Nooruddin, S., Karray, F., & Muhammad, G. (2022). Human activity recognition using tools of convolutional neural networks: A State of the art review, data sets, challenges, and future prospects. *Computers in Biology and Medicine*, *149*, 106060. https://doi.org/10.1016/j.compbiomed.2022.106060

Ivakhnenko, A. G., & Lapa, V. G. (1966). Cybernetic Predicting Devices. *Cybernetics and Systems Analysis*, *1*(1). https://doi.org/chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://core.ac.uk/download/pdf/41822837.pdf

Kohonen, T. (1986). *Learning vector quantization for pattern recognition*. Helsinki University of Technology.

Majidzadeh Gorjani, O., Byrtus, R., Dohnal, J., Bilik, P., Koziorek, J., & Martinek, R. (2021). Human activity classification using Multilayer Perceptron. *Sensors*, *21*(18), 6207. https://doi.org/10.3390/s21186207

OpenAI. (2023). GPT-3.5 Language Model. [Computer software]. Retrieved from https://openai.com/gpt-3/

Reyes-Ortiz, J.-L., Oneto, L., Samà, A., Parra, X., & Anguita, D. (2016). Transition-aware human activity recognition using smartphones. *Neurocomputing*, *171*, 754–767. https://doi.org/10.1016/j.neucom.2015.07.085

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, *13*(7), 1443–1471. https://doi.org/10.1162/089976601750264965