CPSC-5616EL
Machine Learning/Deep Learning

Exercise Classification

Name: Leyon Ibn Kamal
ID: 0441985

Introduction

In this project, we used data from a mobile phone accelerometer to classify exercise movement. The initial data we have are raw data generated from the Android application called Sensor Logger, which gives us movement data from different phone sensors. We have data for three different exercises: squats, lunges, and jumping jacks. Our goal is to process the raw data and prepare the data to be ready for use with machine learning and deep learning algorithms. We will be testing classification methods using SVM, Random Forest, CNN1D, and GRU.

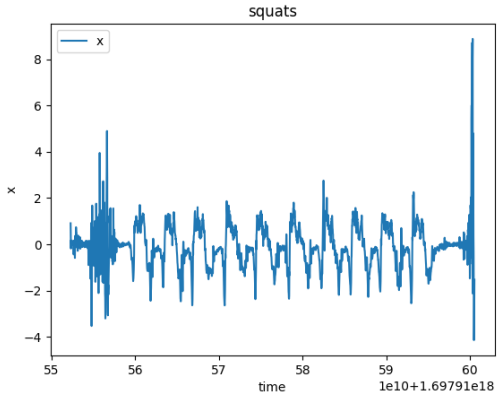
Data Preparation

The initial raw data contains a time, seconds_elapsed, x, y, and z data.

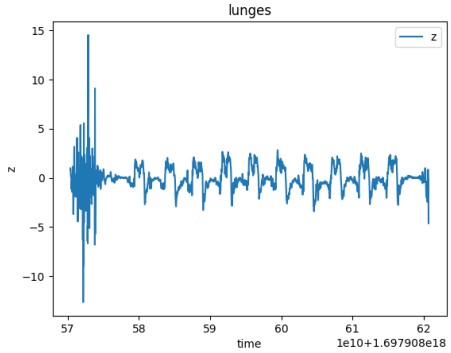
```
1 squat_data.head()
```

	time	seconds_elapsed	z	y	x
0	1697910552273722400	0.126722	0.298795	-0.050724	0.907530
1	1697910552292690000	0.145690	-0.102528	0.207921	0.244633
2	1697910552311657700	0.164658	-0.574512	0.063103	-0.183742
3	1697910552330625300	0.183625	1.251686	0.108361	0.168333
4	1697910552349593000	0.202593	-0.105359	0.113360	0.054346

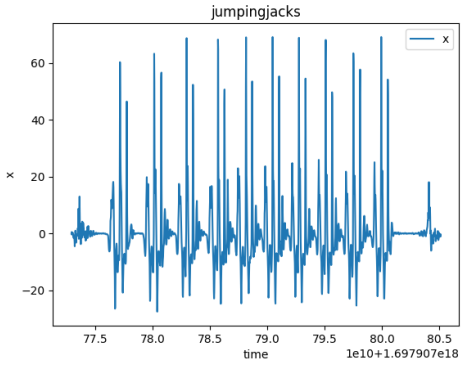
If we create graphs of our data, we notice that different exercises have different graph patterns. For each of the exercises, if we plot the x/y/z values vs the time, we get different patterns for different axes, and the graph pattern for the exercise is more visible on certain axes indicating that the accelerometer was moving more along that axis.
For squats, we see that the x-axis vs time gives a clearer pattern.



For lunges, we see that the z-axis vs time gives a clearer pattern.

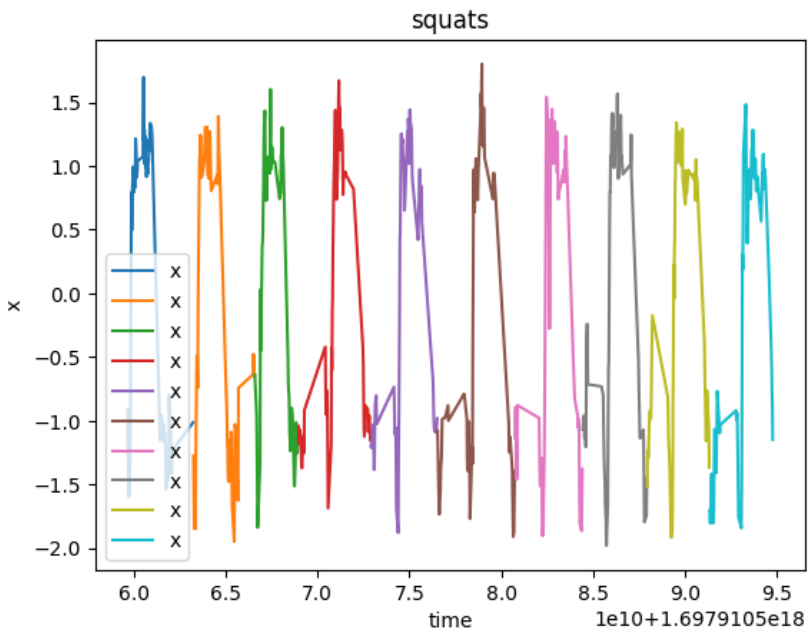
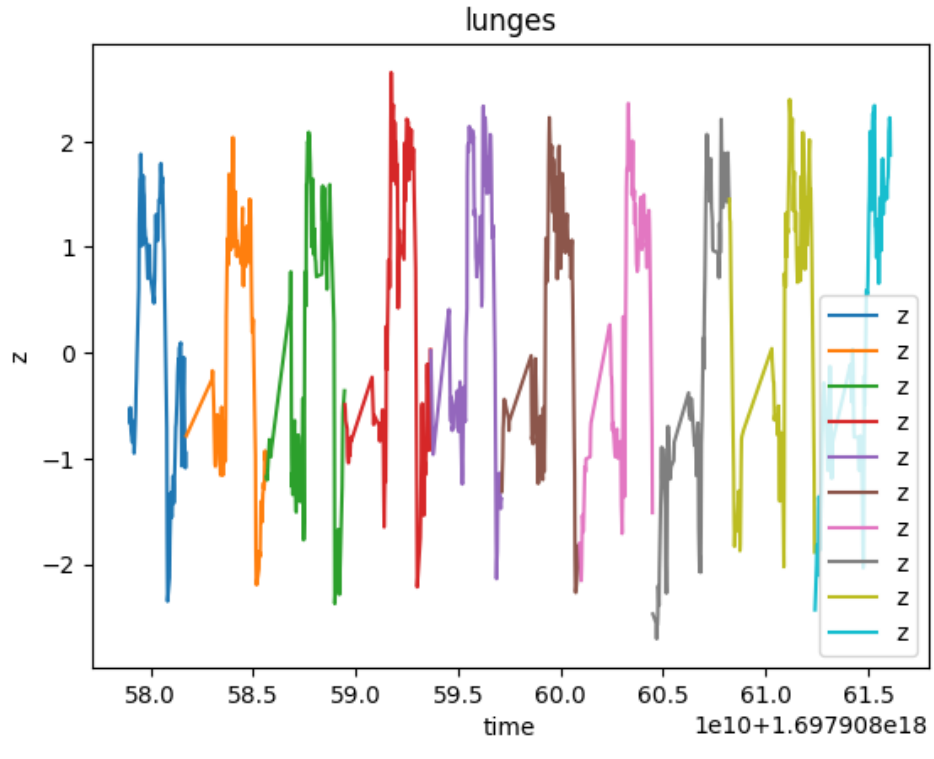


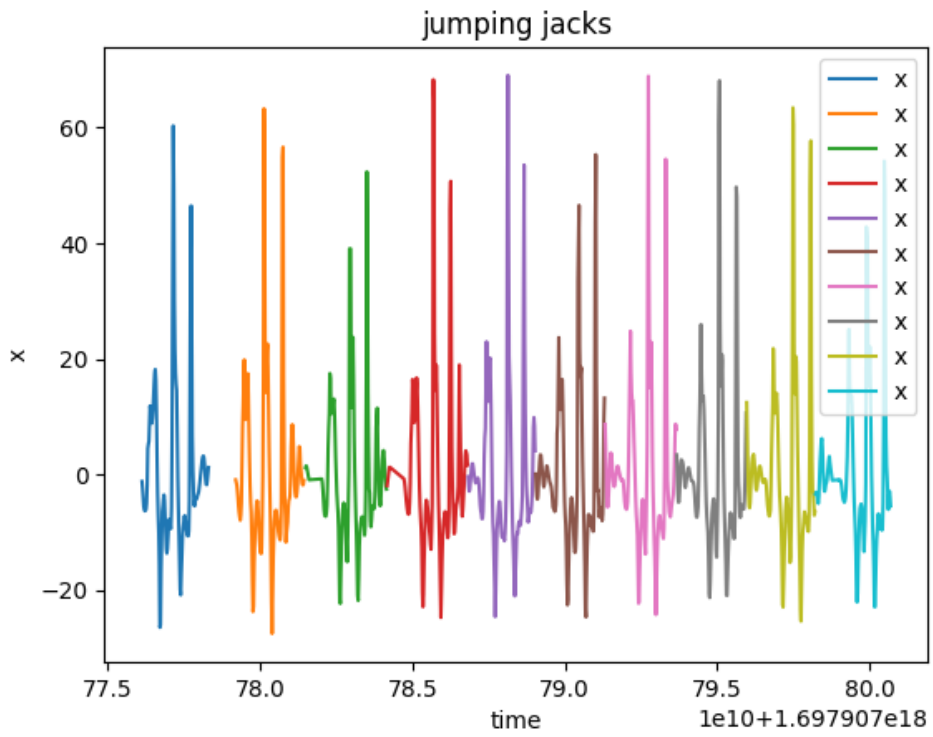
For lunges, we see that the x-axis vs time gives a clearer pattern.



So, then to process the data, we calculated a magnitude value using the formula magnitude=sqrt(x**2 + y**2 + z**2).
Then we try to figure out how to separate each exercise repetition into one data row for our final dataset. For this, we first removed noise from our data using the magnitude. We set a minimum and maximum threshold value where we considered magnitude value below the minimum threshold or values above the threshold to be noise unrelated to the exercise which may be caused by the phone shaking or phone making sudden movements. So the data was marked as active or inactive, and the inactive data was removed. Then we trimmed the data to remove unwanted data at the beginning or the end, which may have been caused by getting the phone in position before starting the

set or out of position after the set. Then by trial and error, we found out the number of data points that belong to each repetition of movement.

Squat	
Lunges	

Jumping Jacks	

So to prepare the final dataset, we placed the magnitude values of each repetition in one row along with the total time that we calculated for each repetition.
Then since each repetition took a different time, we had a different number of columns of data. To fix this, we padded our data with leading zeroes. Then we labeled the data as strings, which were later converted to integers using the label encoder of scikit-learn. Jumping jacks were labeled as 0, lunges had 1, and squats had 3.

Machine Learning and Deep Learning Implementation

SVM

A supervised learning model called a Support Vector Machine (SVM) is used in machine learning for regression analysis and classification. SVMs work by mapping training examples to points in space so as to maximize the width of the gap between two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Random Forest

An ensemble learning technique called Random Forest is applied to both regression and classification problems. During training, it builds a large number of decision trees, and it provides the mean prediction for regression tasks or the mode for classification tasks based on the individual trees. Every single tree within the ensemble has been taught on a random part of the training data and employs a random subset of characteristics for decision-making, which accounts for the "random" in Random Forest.

Conv1D

One kind of convolutional layer that works with 1D data is called Conv1D. It is frequently utilized in models, including time series analysis and natural language processing, that deal with temporal data or sequences. While traditional convolutional layers are commonly used for image data with two spatial dimensions (width and height), Conv1D is designed for sequence data with one spatial dimension, such as time-series data or sequences of text.

GRU

"GRU" stands for "Gated Recurrent Unit," which is a type of recurrent neural network (RNN) architecture. Gated Recurrent Unit is a type of recurrent neural network architecture that uses a gating mechanism to control the flow of information within the network. It is designed to capture and propagate relevant information over long sequences.

Results

The results for our 4 models are as follows.

1. **SVM Model:** For the SVM model, we get an accuracy of 66%. And here we can see from the confusion matrix that it failed to predict jumping jacks totally and misclassified them as mostly a lunge or a squat, whereas squats and lunges were classified with full accuracy.

```
ACCURACY OF THE MODEL:  0.6666666666666666
Confusion Matrix:
 [[0 5 3]
 [0 8 0]
 [0 0 8]]
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         8
           1       0.62      1.00      0.76         8
           2       0.73      1.00      0.84         8

    accuracy                           0.67        24
   macro avg       0.45      0.67      0.53        24
weighted avg       0.45      0.67      0.53        24
```

2. **Random Forest:** For the random forest model, we had an accuracy of 100% and every exercise was classified correctly.

```
Test Accuracy:  1.0
Confusion Matrix:
 [[8 0 0]
 [0 8 0]
 [0 0 8]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         8
           1       1.00      1.00      1.00         8
           2       1.00      1.00      1.00         8

    accuracy                           1.00        24
   macro avg       1.00      1.00      1.00        24
weighted avg       1.00      1.00      1.00        24
```

3. **Conv1D:** For the Conv1D model, we had an accuracy of 83%. Conv1D model had multiple layers, with 3 of them being Conv1D and others being ReLu, MaxPooling, Linear, and Softmax output layers. The Conv1D model classified

some of the data from all the classes accurately.

```
 Test Accuracy: 75.00%
Confusion Matrix:
[[2 6 0]
 [0 8 0]
 [0 0 8]]
Classification Report:
              precision    recall  f1-score   support

 jumpingjack       1.00      0.25      0.40         8
       lunge       0.57      1.00      0.73         8
       squat       1.00      1.00      1.00         8

    accuracy                           0.75        24
   macro avg       0.86      0.75      0.71        24
weighted avg       0.86      0.75      0.71        24
```

4. **GRU:**For the GRU model we had a 66% accuracy. And here we can see that it was misclassifying the jumping jack data as a lunge.

```
Test Accuracy: 66.67%
Confusion Matrix:
[[0 6 2]
 [0 8 0]
 [0 0 8]]
Classification Report:
              precision    recall  f1-score   support

 jumpingjack       0.00      0.00      0.00         8
       lunge       0.57      1.00      0.73         8
       squat       0.80      1.00      0.89         8

    accuracy                           0.67        24
   macro avg       0.46      0.67      0.54        24
weighted avg       0.46      0.67      0.54        24
```

Conclusion

Our dataset was very small and so we only had so much data to train and test our models on. We found the Random Forest model to perform the best, followed by Conv1D, then the SVM and GRU models. One of the reasons why deep learning

models didn’t perform very well could be because these algorithms require large amounts of data to find patterns in. Random Forest performed very well with such a low amount of data, which indicates that the Random Forest model is perfect for such projects since we will require less data but get better results. For further work, other than adding more data, it would be interesting to try to figure out why the models were misclassifying jumping jacks as lunges.